ΔΙΑΤΜΗΜΑΤΙΚΟ ΠΡΟΓΡΑΜΜΑ ΜΕΤΑΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ ΣΤΑ ΠΛΗΡΟΦΟΡΙΑΚΑ ΣΥΣΤΗΜΑΤΑ

Διπλωματική Εργασία

ΔΗΜΙΟΥΡΓΙΑ ΕΚΠΑΙΔΕΥΤΙΚΟΥ ΥΛΙΚΟΥ ΓΙΑ ΤΗΝ ΥΠΟΛΟΓΙΣΤΙΚΗ ΣΚΕΨΗ

Του

ΝΙΚΟΛΑΟΥ ΚΟΝΣΟΥΛΙΔΗ

ΕΠΙΒΛΕΠΩΝ ΚΑΘΗΓΗΤΗΣ:
ΘΡΑΣΥΒΟΥΛΟΣ – ΚΩΝΣΤΑΝΤΙΝΟΣ ΤΣΙΑΤΣΟΣ, ΕΠΙΚΟΥΡΟΣ ΚΑΘΗΓΗΤΗΣ

Υποβλήθηκε ως απαιτούμενο για την απόκτηση του Μεταπτυχιακού Διπλώματος Ειδίκευσης στα Πληροφοριακά Συστήματα

Φεβρουάριος 2019

Αφιερώσεις

Στην Οικογένεια και τους

Φίλους που με Υποστηρίζουν

Ευχαριστίες

Αρχικά θα ήθελα να ευχαριστήσω όλους τους διδάσκοντες και το προσωπικό του Μεταπτυχιακού Προγράμματος για τη βοήθεια και καθοδήγησή τους κατά τη διάρκεια της φοίτησής μου.

Ιδιαίτερα θα ήθελα να ευχαριστήσω τον επιβλέποντα καθηγητή κ. Θρασύβουλο Τσιάτσο για τη συμβολή του στην ολοκλήρωση της διπλωματικής εργασίας μου με την καθοδήγηση, υποστήριξη και συνεργασία που μου παρείχε.

Στη συνέχεια θα ήθελα να ευχαριστήσω την οικογένειά μου για τη στήριξη και τη συμπαράσταση, σε όλη τη διάρκεια της φοίτησής μου, καθώς χωρίς τη βοήθειά της θα ήταν ανέφικτη η ολοκλήρωση του συγκεκριμένου σταδίου των σπουδών μου.

Τέλος θα ήθελα να ευχαριστήσω τους φίλους και συμφοιτητές για την έμπνευση και την υποστήριξή τους.

Περίληψη

Η Υπολογιστική Σκέψη, που αποτελεί βασική έννοια της Υπολογιστικής Επιστήμης, τα τελευταία χρόνια έχει προσελκύσει ξανά το ενδιαφέρον της επιστημονικής κοινότητας. Μέσα από την εφαρμογή των ιδιαίτερων χαρακτηριστικών της διευκολύνεται, μεταξύ άλλων, η διαχείριση της πολυπλοκότητας και της επίλυσης προβλημάτων. Η δεξιότητα της Υπολογιστικής Σκέψης αποτελεί απαραίτητο εργαλείο της καθημερινής ζωής καθώς έχει εφαρμογή όχι μόνο στην Επιστήμη των Υπολογιστών και την ευρύτερη Ακαδημαϊκή Κοινότητα, αλλά και σε ποικίλους άλλους τομείς. Για τους λόγους αυτούς κρίνεται απαραίτητη η ενσωμάτωση της εκμάθησής της σε όλες τις βαθμίδες εκπαίδευσης. Σκοπός της παρούσας εργασίας ήταν η δημιουργία εκπαιδευτικού υλικού που θα συνέβαλε στην ανάπτυξη των δεξιοτήτων της Υπολογιστικής Σκέψης. Προκειμένου να δημιουργηθεί το υλικό αυτό προηγήθηκε η μελέτη και διατύπωση των ορισμών και των χαρακτηριστικών της Υπολογιστικής Σκέψης, εξετάστηκαν οι τρόποι ενσωμάτωσής της σε προγράμματα σπουδών και αναζητήθηκαν τα διαθέσιμα εργαλεία και πλαίσια αξιολόγησής της. Στην παρούσα εργασία, η ενσωμάτωση της Υπολογιστικής Σκέψης στο προπτυχιακό μάθημα «Εισαγωγή στην Πληροφορική» του Τμήματος Πληροφορικής του Α.Π.Θ. πραγματοποιήθηκε με τη χρήση της γλώσσας προγραμματισμού Python. Το εκπαιδευτικό υλικό που δημιουργήθηκε είχε τη μορφή υποθετικών σεναρίων και βασίστηκε στην διδακτέα ύλη του μαθήματος. Η συμμετοχή των εκπαιδευόμενων στις εργασίες ήταν προαιρετική. Από τη στατιστική ανάλυση των δεδομένων προέκυψε ότι οι φοιτητές που παρέδωσαν τις προαιρετικές εργασίες φάνηκε να ενίσχυσαν τις δεξιότητες της Υπολογιστικής τους Σκέψης, σε σύγκριση με τους φοιτητές που δεν τις παρέδωσαν, όπως αξιολογήθηκε από τη βαθμολογία που έλαβαν στην τελική γραπτή εξέταση. Το εύρημα αυτό φαίνεται να επιβεβαιώνει την εκπλήρωση του σκοπού για τον οποίο δημιουργήθηκε το εκπαιδευτικό υλικό. Η αξιολόγηση των δεξιοτήτων των φοιτητών πραγματοποιήθηκε και με τη χορήγηση ενός τεστ Υπολογιστικής Σκέψης, αλλά εξαιτίας της μικρής συμμετοχής των φοιτητών δεν κατέστη δυνατή η στατιστική τους επεξεργασία για την εξαγωγή συμπερασμάτων.

Abstract

Computational Thinking, a fundamental concept of Computational Science, has lately gathered increasing interest from the scientific community. The application of its special features helps manage complexity and problem-solving. Computational Thinking is considered an indispensable tool of everyday life and is applied not only in Computational Science and the Academic Community, but also in a variety of fields. For these reasons, it is necessary to incorporate its learning at all levels of education. The aim of the present study was the creation of educational material that would contribute to the development of Computational Thinking skills. The creation of this material was preceded by the study and formulation of the definitions and characteristics of Computational Thinking, the ways of incorporating it in curricula and the available tools and evaluation frameworks. In the present study, Computational Thinking was incorporated in the Undergraduate Studies "Introduction to Informatics" of the School of Informatics of the Aristotle University of Thessaloniki with the use of Python programming language. The educational material created was in the form of hypothetical scenarios and was based on the course curriculum. Trainees' participation in the exercises was optional. The statistical analysis of the data revealed that the students who participated in the optional assignment seemed to enhance their Computational Thinking skills, as they were assessed by the grade they received in the final written exams, compared to the students who did not deliver them. This finding seems to verify that the educational material created fulfilled its purpose. The assessment of students' skills was carried out with the additional administration of the Computational Thinking Test (CTt), but due to low participation the statistical analysis of the data was not possible.

Πίνακας περιεχομένων

Κατάλογος Πινάκων

Κατάλογος Εικόνων

Κατάλογος Διαγραμμάτων

1 Εισαγωγή

Η Υπολογιστική Σκέψη θεωρείται σημαντική δεξιότητα του 21ου αιώνα καθώς αναπτύσσει και εξελίσσει υψηλές νοητικές διεργασίες. Η εφαρμογή της βασίζεται στις τεχνικές της Επιστήμης των Υπολογιστών και αποτελεί μέθοδο προσέγγισης, κατανόησης και επίλυσης πολύπλοκων προβλημάτων. Το συνεχώς αυξανόμενο ενδιαφέρον της επιστημονικής κοινότητας πηγάζει από την εφαρμογή της Υπολογιστικής Σκέψης σε ένα ευρύ φάσμα επιστημονικών τομέων ενώ επεκτείνεται και στην καθημερινή ζωή. Παρά τη σημαντικότητά της παραμένει ένα θέμα ανοιχτό προς διερεύνηση. Ο συνδυασμός των παραπάνω αποτέλεσε και το έναυσμα για την εκπόνηση της συγκεκριμένης εργασίας. Σκοπός της ήταν ο σχεδιασμός, η υλοποίηση, η ενσωμάτωση και η αξιολόγηση εκπαιδευτικού υλικού με στόχο την ανάπτυξη της Υπολογιστικής Σκέψης. Για την εκπλήρωση του σκοπού και των στόχων διερευνήθηκαν και αποτυπώθηκαν οι ορισμοί, τα χαρακτηριστικά, ο τρόπος ενσωμάτωσης σε διαφορετικά προγράμματα σπουδών και οι τρόποι αξιολόγησης που προέκυψαν από τη βιβλιογραφική επισκόπηση. Όλα τα παραπάνω αποτέλεσαν και το θεωρητικό υπόβαθρο για την ενσωμάτωση και την αξιολόγηση του διδακτικού υλικού που τελικά εφαρμόστηκε πιλοτικά στα πλαίσια του υποχρεωτικού μαθήματος «Εισαγωγή στην Πληροφορική» του Τμήματος Πληροφορικής του Αριστοτελείου Πανεπιστημίου Θεσσαλονίκης. Πιο συγκεκριμένα στα τρία πρώτα κεφάλαια καλύπτεται το θεωρητικό υπόβαθρο, ενώ στο τέταρτο κεφάλαιο αναλύεται η μεθοδολογία και ακολουθούν τα αποτελέσματα και τα συμπεράσματα.

1.1 Ιστορική αναδρομή του όρου Υπολογιστική Σκέψη

Ο αλματώδης ρυθμός ανάπτυξης που συντελέστηκε σε όλους τους επιστημονικούς τομείς είχε ως επακόλουθο την ευρύτερη εξέλιξη των τρόπων κατανόησης και επίλυσης των προβλημάτων που προέκυπταν, μέσω της εφαρμογής Τεχνολογιών Πληροφορίας και Επικοινωνιών. Το γεγονός αυτό οδήγησε με τη σειρά του στο μετασχηματισμό των απαιτήσεων της κοινωνίας του 21^{ου} αιώνα και ιδιαίτερα των δεξιοτήτων που κρίνονται απαραίτητες τόσο για τον επιστημονικό όσο και για τον εργασιακό τομέα. Μια τέτοια δεξιότητα είναι και η ικανότητα επίλυσης προβλημάτων (Wikipedia), η οποία προτού αναγνωριστεί ως βασικό στοιχείο της Υπολογιστικής Σκέψης, είχε προεκτάσεις και εφαρμογές σε ποικίλους επιστημονικούς τομείς.

Σύμφωνα με τη βιβλιογραφία, η αρχική ιδέα της επίλυσης προβλημάτων εξελίχθηκε με το πέρασμα των δεκαετιών και μετασχηματίστηκε στην έννοια της Υπολογιστικής Σκέψης. Συγκεκριμένα, το 1940 ο John von Neumann ανάφερε ότι η υπολογιστική (Computing) δεν θα είχε απλά υποστηρικτικό ρόλο, αλλά θα ήταν ένας ξεχωριστός επιστημονικός τομέας (Denning P. J., 2009)**;** (Ozcinar, Wong, & Ozturk, 2018). Το 1945 ο George Polya αναφέρθηκε στους τρόπους επίλυσης μαθηματικών προβλημάτων με την εφαρμογή τεχνικών όπως η αποσύνθεση και ο μετασχηματισμός προβλημάτων (Denning, 2017b); (Polya, 1973).

Τις δεκαετίες του 1950 και 1960 προσδιορίζεται με την έννοια Αλγοριθμική Σκέψη (Ozcinar, Wong, & Ozturk, 2018). Επίσης, το 1960 ο Alan Perlis αναφέρθηκε στη διαδικασία εκμάθησης προγραμματισμού, τόνισε τη σημασία των νοητικών διεργασιών που απαιτούνται για το σχεδιασμό αλγορίθμων (algorithmizing), έδωσε έμφαση στον τρόπο σκέψης που αυτός προϋποθέτει και προέβλεψε την αξιοποίησή του από όλες τις επιστήμες. (Tedre & Denning, 2016) (Ozcinar, Wong, & Ozturk, 2018). Το 1974 ο Donald Knuth υποστήριξε ότι η μοναδικότητα της υπολογιστικής και ο διαχωρισμός της από τα υπόλοιπα επιστημονικά πεδία οφείλεται στην Αλγοριθμική Σκέψη, μέσω της οποίας διευκολύνεται η εμβάθυνση στα προβλήματα (Denning, 2017b). Το 1979 ο Edsger Dijkstra και ο Knuth έδωσαν ιδιαίτερη έμφαση στην έννοια Αλγοριθμική Σκέψη και επισήμαναν μεταξύ άλλων τη σπουδαιότητα της αφαίρεσης και του διαχωρισμού των προβλημάτων κατά τη διαδικασία του προγραμματισμού (Denning, 2017b); (Yasar, 2018).

Ο Seymour Papert θεωρείται ότι ήταν ο πρώτος που το 1980 χρησιμοποίησε τον όρο Υπολογιστική Σκέψη, εισήγαγε τη γλώσσα προγραμματισμού LOGO και υποστήριξε ότι μέσω αυτής αναπτύσσονται οι δεξιότητες της διαδικαστικής σκέψης και κατ'επέκταση της επίλυσης προβλημάτων (Tedre & Denning, 2016). Αναφέρθηκε επίσης στη θεωρία του κατασκευαστικού εποικοδομισμού (constructionism), σύμφωνα με την οποία η μάθηση επιτυγχάνεται μέσω της επαναληπτικής διαδικασίας της απτής εμπειρίας, του αναστοχασμού και της κατανόησης (Δημητριάδης, 2015).

Στον τομέα των επιστημών, ο συνδυασμός της υπολογιστικής μοντελοποίησης και της προσομοίωσης ήταν ένας καινούργιος και ιδιαίτερος τρόπος προσέγγισης (έρευνας), ο οποίος προστέθηκε στους ήδη υπάρχοντες, το θεωρητικό και τον πειραματικό (Tedre & Denning, 2016). Ο τρόπος αυτός χρησιμοποιήθηκε από τον Kenneth Wilson (Νόμπελ φυσικής 1982) και ονομάστηκε Υπολογιστική Επιστήμη (Computational Science). Κύριο χαρακτηριστικό της ήταν η χρήση της υπολογιστικής σε κάθε επιστημονικό πεδίο. Στο συγκεκριμένο τρόπο προσέγγισης, η νοητική διεργασία που απαιτείται για το σχεδιασμό, τον έλεγχο και την υπολογιστική μοντελοποίηση ονομάστηκε Υπολογιστική Σκέψη και αποτελεί τμήμα της Υπολογιστικής Επιστήμης (Denning, 2017b).

Το 2006 η Jeannette Wing επανέφερε και πάλι την έννοια της Υπολογιστική Σκέψης και υποστήριξε ότι αποτελεί βασική δεξιότητα γιατί προάγει «*την επίλυση προβλημάτων, το σχεδιασμό συστημάτων και την κατανόηση της ανθρώπινης συμπεριφοράς σύμφωνα με τις βασικές έννοιες της επιστήμης των υπολογιστών*» (Wing, 2006). Τα κυριότερα χαρακτηριστικά της Υπολογιστική Σκέψης είναι η σκέψη σε πολλαπλά επίπεδα αφαίρεσης και η αποσύνθεση των προβλημάτων. Το 2011 η Wing προσπάθησε να δώσει μια πληρέστερη περιγραφή της έννοιας, ορίζοντάς την ως: «*οι νοητικές διεργασίες που εμπλέκονται στο μετασχηματισμό των προβλημάτων και της επίλυσής τους με τρόπο που να επιτρέπει την αποτελεσματική διεκπεραίωσή τους από ένα μέσο (agent) επεξεργασίας πληροφοριών*» (Wing, 2011). Το 2012 ο Alfred Aho διαφοροποίησε την περιγραφή της Υπολογιστικής Σκέψης που έδωσε η Wing ως εξής: «*Υπολογιστική Σκέψη είναι οι νοητικές διεργασίες που εμπλέκονται στο μετασχηματισμό των προβλημάτων και των λύσεών τους ούτως ώστε οι λύσεις να παρουσιάζονται με υπολογιστικά βήματα και αλγοριθμικές διαδικασίες*» (Aho, 2012). Ιδιαίτερη έμφαση έδωσε στην αφαίρεση, την οποία θεωρούσε υπολογιστική μοντελοποίηση και βασική έννοια της Υπολογιστικής Σκέψης, ενώ υποστήριξε ότι τα μοντέλα αυτά δεν αποτελούν μόνο τμήμα της επιστήμης των υπολογιστών, αλλά και κομμάτι όλων των επιστημών (Denning, 2017b).

Αν παρατηρήσει κανείς την εννοιολογική εξέλιξη του όρου Υπολογιστική Σκέψη μπορεί να σχηματίσει μια πρώτη εικόνα για τα γενικά χαρακτηριστικά που περιλαμβάνει, καθώς επίσης και τον τρόπο με τον οποίο συνδέεται με την Υπολογιστική Επιστήμη μέσω του βασικού χαρακτηριστικού της υπολογιστικής μοντελοποίησης (Ψυχάρης, Κοτζαμπασάκη, & Καλοβρέκτης, 2018). Το χαρακτηριστικό αυτό αποτέλεσε και το έναυσμα για τον διαχωρισμό της υπολογιστικής επιστήμης από τους υπόλοιπους επιστημονικούς κλάδους και κυρίως των μαθηματικών και της μηχανικής (Denning P. J., 2017a). Παρόλα αυτά, τα κοινά στοιχεία με τους προαναφερθέντες τομείς είναι εμφανή, τόσο ως προς τη συλλογιστική όσο και ως προς την εφαρμογή των τεχνικών που χρησιμοποιούνται (Wing, 2006).

Αξίζει επίσης να σημειωθεί ότι από τη δεκαετία του 1990 φαίνεται να φθίνει η ενασχόληση της επιστημονικής κοινότητας με την Υπολογιστική Σκέψη, ενώ από το 2006 και έπειτα το ενδιαφέρον εντείνεται ξανά. Η παρατήρηση αυτή φαίνεται να επιβεβαιώνεται και από τις συνεχείς προσπάθειες τόσο των ερευνητών όσο και των οργανισμών/φορέων να αποσαφηνίσουν τον όρο της Υπολογιστικής σκέψης και τα χαρακτηριστικά της γνωρίσματα (Ilic, Haseski, & Tugtekin, 2018). Αρωγοί στην προσπάθεια αυτή και στην ένταξη της Υπολογιστικής Σκέψης στον τομέα της Επιστήμης, Τεχνολογίας, Μηχανικής και Μαθηματικών (STEM) ήταν το Carnegie-Mellon University καθώς και οι ακόλουθοι οργανισμοί: Διεθνής Κοινότητα για την Τεχνολογία στην Εκπαίδευση (ISTE), Σύλλογος Καθηγητών Πληροφορικής (CSTA), Εθνικό Ίδρυμα Επιστημών (NSF) στις ΗΠΑ και Εθνική Ακαδημία Επιστημών (THE ROYAL SOCIETY), Computing At School στο Ηνωμένο (Μαυρουδή, Πέτρου, & Φεσάκης, 2014).

Πίνακας 1: Ιστορική αναδρομή Υπολογιστικής Σκέψης

Ημερομηνία	Όνομα	Περιγραφή
1940	John von Neumann	Η Υπολογιστική είναι τρόπος δράσης της ίδιας της επιστήμης
1945	George Polya	Τρόπος επίλυσης μαθηματικών προβλημάτων, μεταξύ άλλων είναι οι τεχνικές Αποσύνθεση και Μετασχηματισμός προβλημάτων
1960	Alan Perlis	Νοητικές διεργασίες για τον σχεδιασμό αλγορίθμων μπορούν να αξιοποιηθούν από

		όλες τις επιστήμες
1974	Donald Knuth	Στην Αλγοριθμική Σκέψη οφείλεται η μοναδικότητα της Υπολογιστικής και ο διαχωρισμός της από τα υπόλοιπα επιστημονικά πεδία
1979	Edsger Dijkstra	Αφαίρεση και Διαχωρισμός προβλημάτων είναι σημαντική διαδικασία του προγραμματισμού και της Αλγοριθμικής Σκέψης
1980	Seymour Papert	Εισήγαγε τον όροΥπολογιστική Σκέψη, τη γλώσσα προγραμματισμού LOGO για ανάπτυξη διαδικαστικής σκέψης και επίλυσης προβλημάτων και υποστήριξε την θεωρία του κατασκευαστικού εποικοδομισμού (constructionism)
1982	Kenneth Wilson	Ο συνδυασμός της υπολογιστικής μοντελοποίησης και προσομοίωσης ονομάστηκε Υπολογιστική Επιστήμη με δυνατότητα χρήσης σε κάθε επιστημονικό πεδίο. Η διεργασία που απαιτείται για τον σχεδιασμό και τον έλεγχο του, Υπολογιστική Σκέψη
2006	Jeannette Wing	Η Υπολογιστική Σκέψη είναι βασική δεξιότητα της Υπολογιστικής Επιστήμης γιατί προάγει την επίλυση προβλημάτων το σχεδιασμό συστημάτων και την κατανόηση της ανθρώπινης συμπεριφοράς
2011	Jeannette Wing	Οι νοητικές διεργασίες που εμπλέκονται στο μετασχηματισμό των προβλημάτων και της λύσης τους με τρόπο που να επιτρέπει αποτελεσματική διεκπεραίωσή τους από ένα μέσο (agent) επεξεργασίας πληροφοριών
2012	Al Aho	Υπολογιστική Σκέψη είναι οι διαδικασίες σκέψης που εμπλέκονται στον μετασχηματισμό των προβλημάτων και των λύσεών τους, έτσι ώστε οι λύσεις να

		παρουσιάζονται με υπολογιστικά βήματα και αλγοριθμικές διαδικασίες

1.2 Ορισμοί Υπολογιστικής Σκέψης

Παρά το συνεχώς αυξανόμενο ενδιαφέρον της επιστημονικής κοινότητας για την Υπολογιστική Σκέψη, από το 2006 και έπειτα δεν έχει δοθεί κάποιος επίσημος ορισμός (Yadav, Stephenson, & Hong, 2017); (Selby, 2015). Στη συνέχεια παρατίθενται οι απόπειρες ορισμού της Υπολογιστικής Σκέψης που συναντά κανείς κατά την ανασκόπηση της βιβλιογραφίας:

- *«Η Υπολογιστική Σκέψη περιλαμβάνει την επίλυση προβλημάτων, το σχεδιασμό συστημάτων και την κατανόηση της ανθρώπινης συμπεριφοράς σύμφωνα με τις βασικές έννοιες της επιστήμης των υπολογιστών»* (Wing, 2006)

- *«Εν συντομία τα κύρια σημεία της Υπολογιστικής Σκέψης είναι:*
  1. *Τρόπος επίλυσης προβλημάτων και σχεδιασμού συστημάτων που βασίζεται στις κύριες έννοιες της επιστήμης των υπολογιστών*
  2. *Δημιουργία και εφαρμογή ποικίλων επιπέδων αφαίρεσης με στόχο την αποτελεσματικότερη κατανόηση και επίλυση προβλημάτων*
  3. *Εφαρμογή αλγοριθμικής σκέψης και μαθηματικών εννοιών για την ανάπτυξη περισσότερο αποτελεσματικών, δίκαιων και ασφαλών λύσεων*
  4. *Κατανόηση των επιπτώσεων της κλίμακας, τόσο για λόγους αποδοτικότητας όσο και για οικονομικούς και κοινωνικούς λόγους» (Lu & Fletcher, 2009)*

- *«Υπολογιστική Σκέψη είναι ο νοητικός προσανατολισμός στη διατύπωση προβλημάτων με τη μορφή της μετατροπής ορισμένων εισροών σε εκροές και αναζήτηση αλγορίθμων για την εκτέλεση αυτών των μετατροπών. Ο όρος έχει επεκταθεί και συμπεριλαμβάνει τη διεργασία σε πολλαπλά επίπεδα αφαίρεσης, την εφαρμογή των μαθηματικών για την ανάπτυξη αλγορίθμων και τη διερεύνηση μιας ευέλικτης λύσης στα προβλήματα διαφορετικών μεγεθών»* (Denning P. J., 2009)

- *«Η Υπολογιστική Σκέψη είναι κριτική σκέψη και επίλυση προβλημάτων με ένα πρόσθετο υπολογιστικό συστατικό, το οποίο είναι ιδιαίτερα περίπλοκο ή κουραστικό*

για να διεκπεραιωθεί χωρίς τη χρήση υπολογιστικού μέσου» (Ater-Kranov, Bryant, Orr, Wallace, & Zhang, 2010)

- *«Υπολογιστική Σκέψη είναι οι νοητικές διεργασίες που εμπλέκονται στο μετασχηματισμό των προβλημάτων και της επίλυσής τους με τρόπο που να επιτρέπει την αποτελεσματική διεκπεραίωσή τους από ένα μέσο (agent) επεξεργασίας πληροφοριών»* (Wing, 2011)

- *«Υπολογιστική Σκέψη είναι μια διαδικασία επίλυσης προβλημάτων που περιλαμβάνει τα ακόλουθα χαρακτηριστικά, αλλά δεν περιορίζεται σε αυτά:*
  1. *Διατύπωση προβλημάτων με τρόπο που να επιτρέπει τη χρήση υπολογιστικών μέσων και άλλων εργαλείων για την επίλυσή τους*
  2. *Λογική οργάνωση και ανάλυση δεδομένων*
  3. *Αναπαράσταση δεδομένων, μέσω αφαιρέσεων, όπως είναι η μοντελοποίηση και η προσομοίωση*
  4. *Αυτοματοποιημένες λύσεις μέσω της εφαρμογής της Αλγοριθμικής Σκέψης (μιας σειράς διαδοχικών βημάτων)*
  5. *Προσδιορισμός, ανάλυση και εφαρμογή πιθανών λύσεων με σκοπό την επίτευξη του πιο αποδοτικού και αποτελεσματικού συνδυασμού βημάτων και πόρων*
  6. *Γενίκευση και μεταφορά αυτής της διαδικασίας επίλυσης προβλημάτων σε ένα εύρος θεμάτων (προβλημάτων)»* (CSTA & ISTE, 2011)
- *«Υπολογιστική Σκέψη είναι μια διαδικασία επίλυσης προβλημάτων που περιλαμβάνει χαρακτηριστικά όπως η λογική οργάνωση, η ανάλυση δεδομένων, η δημιουργία λύσεων με την εφαρμογή διαδοχικών βημάτων (αλγόριθμοι) και στάσεων όπως είναι η ικανότητα διαχείρισης πολυπλοκότητας και ανοιχτών προβλημάτων, με βεβαιότητα. Η Υπολογιστική Σκέψη είναι απαραίτητη για την ανάπτυξη εφαρμογών πληροφορικής, ενώ μπορεί να αξιοποιηθεί και στην υποστήριξη επίλυσης προβλημάτων σε όλους τους κλάδους, συμπεριλαμβανομένων των μαθηματικών, των επιστημών και των ανθρωπιστικών επιστημών»* (Google for Education: Computational Thinking, 2011)

- *«Υπολογιστική Σκέψη είναι οι διεργασίες σκέψης που εμπλέκονται στο μετασχηματισμό των προβλημάτων και των λύσεών τους, έτσι ώστε οι λύσεις να παρουσιάζονται με υπολογιστικά βήματα και αλγοριθμικές διαδικασίες»* (Aho, 2012)

- *«Υπολογιστική Σκέψη είναι η διαδικασία αναγνώρισης πτυχών της υπολογιστικής στον κόσμο που μας περιβάλλει και η εφαρμογή εργαλείων και τεχνικών της*

Επιστήμης των Υπολογιστών για την κατανόηση και αιτιολόγηση τόσο των φυσικών όσο και των τεχνικών συστημάτων και διαδικασιών» (The Royal Society, 2012)

- *«Η Υπολογιστική Σκέψη περιλαμβάνει τρείς κύριες διαστάσεις: υπολογιστικές έννοιες (οι έννοιες που οι σχεδιαστές χρησιμοποιούν καθώς προγραμματίζουν), υπολογιστικές πρακτικές (οι πρακτικές που οι σχεδιαστές αναπτύσσουν καθώς προγραμματίζουν), και υπολογιστικές προοπτικές (οι προοπτικές που οι σχεδιαστές διαμορφώνουν για τον κόσμο γύρω τους και για τον εαυτό τους)»* (Brennan & Resnick, 2012)

- *«Υπολογιστική Σκέψη είναι μια εγκεφαλική δραστηριότητα που διευκολύνει την επίλυση προβλημάτων, την καλύτερη κατανόηση των καταστάσεων καθώς και την καλύτερη έκφραση των αξιών μέσω της συστηματικής εφαρμογής της αφαίρεσης, της αποσύνθεσης, του αλγοριθμικού σχεδιασμού, της γενίκευσης και της αξιολόγησης με σκοπό τη δημιουργία ενός αυτοματισμού που μπορεί να υλοποιηθεί από μια ψηφιακή ή ανθρώπινη υπολογιστική συσκευή»* (Selby & Woollard, 2014)

- *«Υπολογιστική Σκέψη είναι μια νοητική λειτουργία ή διεργασία σκέψης που περιλαμβάνει τη συλλογιστική μέσω της οποίας επιλύονται προβλήματα και πεδία (artefacts), διαδικασίες και συστήματα, κατανοούνται καλύτερα. Περιλαμβάνει:*
  1. *την ικανότητα Αλγοριθμικής Σκέψης*
  2. *την ικανότητα σκέψης με όρους αποσύνθεσης*
  3. *την ικανότητα σκέψης με γενικεύσεις, αναγνώριση και χρήση μοτίβων*
  4. *την ικανότητα σκέψης με αφαιρέσεις, επιλέγοντας σωστές αναπαραστάσεις και*
  5. *την ικανότητα σκέψης με όρους αξιολόγησης[…]*

  Η Υπολογιστική Σκέψη μπορεί να εφαρμοστεί σε ένα εύρος πεδίων (artefacts) και περιλαμβάνει: συστήματα, διαδικασίες, αντικείμενα, αλγόριθμους, προβλήματα, λύσεις, αφαιρέσεις, συλλογές δεδομένων ή πληροφοριών» (Csizmadia, et al., 2015)

- *«Είναι σημαντικό να δοθεί έμφαση στην ιδέα της Υπολογιστικής Σκέψης σαν εφαρμογή της διεργασίας αφαίρεσης υψηλού επιπέδου και της αλγοριθμικής προσέγγισης για την επίλυση κάθε είδους προβλήματος»* (García-Peñalvo, 2016)

- *«Πιστεύουμε ότι η Υπολογιστική Σκέψη είναι μια μέθοδος επίλυσης προβλημάτων που επεκτείνει τη σφαίρα της Επιστήμης των Υπολογιστών σε όλους τους κλάδους και αποτελεί ένα ιδιαίτερο μέσο ανάλυσης και ανάπτυξης λύσεων σε προβλήματα που μπορούν να επιλυθούν υπολογιστικά. Η Υπολογιστική Σκέψη αποτελεί βασικό*

στοιχείο του ευρύτερου τομέα της Επιστήμης των Υπολογιστών και δίνει έμφαση στην αφαίρεση, την αυτοματοποίηση και την ανάλυση». (CSTA, 2016)

Οι ορισμοί της Υπολογιστικής Σκέψης που απαντώνται στη βιβλιογραφία μπορούν να διαχωριστούν σε τρεις κατηγορίες με βάση το βαθμό της λεπτομέρειας, τη χρήση για εκπαιδευτικούς σκοπούς ή την ενσωμάτωση σε κάποιο εκπαιδευτικό πρόγραμμα σπουδών (Roman-Gonzalez, Perez-Gonzalez, & Jimenez-Fernandez, 2017). Η πρώτη κατηγορία περιλαμβάνει τους γενικούς ορισμούς. Η δεύτερη κατηγορία τους λειτουργικούς ορισμούς και διαφοροποιείται από την πρώτη γιατί περιέχει πιο λεπτομερείς ορισμούς της έννοιας της Υπολογιστικής Σκέψης. Η τρίτη κατηγορία περιλαμβάνει τους εκπαιδευτικούς ορισμούς, που είτε έχουν αποτελέσει το θεωρητικό πλαίσιο κάποιου προγράμματος σπουδών είτε προορίζονται για εφαρμογή σε άλλα εκπαιδευτικά περιβάλλοντα (πίνακας 2).

Πίνακας 2: Ταξινόμηση των ορισμών της Υπολογιστικής Σκέψης σε κατηγορίες

Χρονολογία	Γενικοί Ορισμοί	Λειτουργικοί Ορισμοί	Εκπαιδευτικοί Ορισμοί
2006	Wing		
2009		Lu & Fletcher	
2009	Denning		
2010	Ater – Kranov		
2011	Wing		
2011		ISTE & CSTA	
2011			Google Exploring CT
2012	Aho		
2012	Royal Society		
2012			Brennan & Resnick
2014		Selby & Woollard	
2015			CAS
2016	García-Peñalvo		
2016	CSTA		

1.3 Έννοιες και στοιχεία της Υπολογιστικής Σκέψης

Η επιστημονική κοινότητα δεν έχει καταλήξει σε έναν επίσημο ορισμό της Υπολογιστικής Σκέψης, οπότε προκειμένου να καταστεί πιο κατανοητή η έννοια θεωρείται προτιμότερο να οριστεί μέσω της εξέτασης των χαρακτηριστικών που περιλαμβάνει. Επειδή και σε αυτό το εγχείρημα δεν ταυτίζονται οι απόψεις των ερευνητών και τα χαρακτηριστικά που έχουν χρησιμοποιηθεί για να περιγράψουν την Υπολογιστική Σκέψη είναι ποικίλα, θα αναφερθούμε στα στοιχεία που απαντώνται συχνότερα στη βιβλιογραφία και αναφέρονται από τους περισσότερους ερευνητές

(CSTA & ISTE, 2011); (Curzon, Dorling, Ng, Selby, & Woollard, 2014); (Csizmadia, et al., 2015); (Selby, 2015). Αυτά είναι τα ακόλουθα:

1) η αφαίρεση, αφορά στην απόκρυψη των περιττών λεπτομερειών και αποτελεί από τα βασικά γνωρίσματα της Υπολογιστικής Σκέψης καθώς αναφέρεται από την πλειοψηφία των ερευνητών ή φορέων
2) η αποσύνθεση, αναφέρεται στον επιμερισμό του προβλήματος σε μικρότερα τμήματα και επισημαίνεται στις περισσότερες ερευνητικές αναφορές
3) ο αλγόριθμός, δηλαδή μια σειρά πεπερασμένων βημάτων
4) η αναγνώριση προτύπων, στη βιβλιογραφία συναντάται και σαν ένα είδος γενίκευσης και αφορά στην παρατήρηση ομοιοτήτων
5) η γενίκευση, αναφέρεται στην αναγνώριση ομοιοτήτων και στην εφαρμογή τους σε ένα ευρύ φάσμα ενδιαφερόντων
6) και η αξιολόγηση, τόσο με την έννοια της εύρεσης σωστής λύσης όσο και της αξιοποίησης των πόρων και των βημάτων.

Εκτός από τα παραπάνω χαρακτηριστικά της Υπολογιστικής Σκέψης που συναντά κανείς πιο συχνά στη βιβλιογραφία, έχουν αποτυπωθεί επίσης και τα εξής:

7) η συλλογιστική, αναφέρεται στην αντίληψη των γεγονότων και στη λήψη αποφάσεων μέσα από ακριβείς συλλογισμούς (Csizmadia, et al., 2015)
8) η αποσφαλμάτωση, δηλαδή η εύρεση σφαλμάτων και η διόρθωσή τους (Angeli, et al., 2016)
9) και τέλος η αυτοματοποίηση, αφορά στη χρήση υπολογιστικών μέσων για πολύπλοκες ή χρονοβόρες εργασίες καθώς και στη λογική οργάνωση και ανάλυση δεδομένων (CSTA & ISTE, 2011).

Από την παράθεση των χαρακτηριστικών της Υπολογιστικής Σκέψης διαφαίνεται η ομαδοποίηση κάποιων χαρακτηριστικών. Ορισμένοι ερευνητές ή οργανισμοί που αναφέρονται στη βιβλιογραφία θεωρούν τα χαρακτηριστικά που συγχωνεύονται ως άρρηκτα συνδεδεμένα μεταξύ τους. Τέτοια παραδείγματα είναι η αναφορά στην αφαίρεση και τη γενίκευση ως ενιαία χαρακτηριστικά (Grover & Pea, 2018) καθώς και στη γενίκευση με την αναγνώριση προτύπων (Csizmadia, et al., 2015). Υπάρχουν και οι περιπτώσεις όπου η αναγνώριση προτύπων θεωρείται ως ένα είδος γενίκευσης και για το λόγο αυτό δεν αναφέρεται ξεχωριστά (Selby, 2015).

Παρότι η Υπολογιστική Σκέψη περιλαμβάνει ποικίλα και διαφορετικά χαρακτηριστικά, εντούτοις κάποια από αυτά θεωρούνται οι «ακρογωνιαίοι λίθοι» της και είναι η

Αφαίρεση, η Αποσύνθεση, η Αναγνώριση Προτύπων και οι Αλγόριθμοι (Google for Education: Computational Thinking, 2011); (Brackmann, Barone, Casali, Boucinha, & Muñoz-Hernandez, 2016); (Gretter & Yadav, 2016); (Tabesh, 2017). Στη συνέχεια θα αναφερθούν εκτενώς καθώς και η Γενίκευση και η Αξιολόγηση που επίσης απαντώνται συχνά.

1.3.1 Αφαίρεση

Η έννοια της αφαίρεσης δεν χρησιμοποιείται αποκλειστικά στην Επιστήμη των Υπολογιστών, καθώς έχει εισαχθεί από την επιστήμη των μαθηματικών αλλά έχει εφαρμογή σε ένα ευρύ φάσμα τομέων. Θεωρείται ένα από τα κυριότερα εργαλεία της Υπολογιστικής Σκέψης. Είναι απαραίτητη για την επίλυση προβλημάτων, εστιάζει στο επιθυμητό επίπεδο λεπτομερειών, οδηγεί σε απλοποιημένη εκδοχή της κύριας ιδέας ενός προβλήματος, μέσω της απόκρυψης στοιχείων δευτερεύουσας σημασίας ή επουσιωδών λεπτομερειών. Εστιάζει στα προς εξέταση βασικά συστατικά, ενώ το πρόβλημα διατηρεί την αρχική του έννοια χωρίς να αλλοιώνεται (Wing, 2014). Αποτελεί, δηλαδή, στην ουσία ένα εργαλείο διαχείρισης της πολυπλοκότητας των προβλημάτων. Με την εφαρμογή της τεχνικής αυτής οι αρχικές έννοιες αναλύονται ή απλοποιούνται σε δύο τουλάχιστον (ή περισσότερα) επίπεδα, διατηρώντας παράλληλα τη δυνατότητα εναλλαγής της εστίασης μεταξύ των επιπέδων αυτών (Grover & Pea, 2018). Συμπερασματικά, όταν εφαρμόζεται η τεχνική της αφαίρεσης θεωρείται πολύ σημαντική η τήρηση των συσχετίσεων μεταξύ των επιπέδων έτσι ώστε να διατηρείται η αρχική έννοια.

1.3.2 Αποσύνθεση

Αφορά στην τμηματοποίηση ή στο διαχωρισμό μιας σύνθετης έννοιας στα επιμέρους στοιχεία που την απαρτίζουν. Όπως είναι εμφανές από την ιστορική αναδρομή του όρου της Υπολογιστικής Σκέψης και η έννοια της αποσύνθεσης είναι δανεισμένη από την επιστήμη των μαθηματικών. Στην Υπολογιστική Σκέψη η τεχνική της αποσύνθεσης, όπως και της αφαίρεσης, θεωρείται κύριας σημασίας και χρησιμοποιείται για την διαχείριση πολύπλοκων εννοιών ή προβλημάτων. Σκοπός της είναι η αρχική σύνθετη έννοια ή πρόβλημα να διαχωριστεί σε μικρότερες ενότητες, τις οποίες μπορεί κανείς να διαχειριστεί ή να επιλύσει πιο εύκολα με απώτερο στόχο κάθε ενότητα να μπορεί να αντιμετωπιστεί μεμονωμένα, στοιχείο πολύ σημαντικό για την παράλληλη επεξεργασία (Grover & Pea, 2018). Κατά την εφαρμογή της συγκεκριμένης τεχνικής είναι απαραίτητο η σύνθεση των υποπροβλημάτων να οδηγούν στο σχηματισμό της αρχικής έννοιας.

1.3.3 Αναγνώριση Προτύπων

Η αναγνώριση προτύπων αναφέρεται στην παρατήρηση και αναγνώριση επαναλαμβανόμενων στοιχείων, που χρησιμεύουν στον προσδιορισμό των ομοιοτήτων και την εφαρμογή τους στην επίτευξη της γενίκευσης μιας λύσης. Στην Υπολογιστική Σκέψη η αναγνώριση προτύπων θεωρείται τεχνική για την επίλυση προβλημάτων ή το σχεδιασμό συστημάτων. Χρησιμοποιείται για την ομαδοποίηση ή την ταξινόμηση δεδομένων σε κατηγορίες ή την αναγνώριση μιας στρατηγικής επίλυσης προβλημάτων με δοκιμασμένη χρήση και αποτέλεσμα, καθώς και εφαρμογή στην επίλυση παρόμοιου προβλήματος (Grover & Pea, 2018). Στον τομέα του προγραμματισμού ή της σχεδίασης αλγορίθμων, η αναγνώριση προτύπων έχει εφαρμογή τόσο σε απλές περιπτώσεις, όπως είναι η αναγνώριση της καταλληλότητας εφαρμογής μιας επαναληπτικής διαδικασίας όσο και σε πιο σύνθετες περιπτώσεις, όπως είναι η χρήση της τεχνικής της αναδρομής.

1.3.4 Αλγόριθμος

Οι αλγόριθμοί είναι μία σειρά διαδοχικών πεπερασμένων βημάτων ή διαδικασιών για την επίτευξη ενός συγκεκριμένου σκοπού. Χρησιμοποιούνται για την προσέγγιση προβλημάτων προς επίλυση και το σχεδιασμό της κατάλληλης λύσης. Καθώς οι αλγόριθμοι δεν απαντώνται μόνο στην επιστήμη των υπολογιστών, δεν είναι απαραίτητη η χρήση υπολογιστικών μέσων για το σχεδιασμό ή την εφαρμογή τους. Η αναπαράστασή τους μπορεί να πραγματοποιηθεί με τη χρήση κειμένου, διαγραμμάτων ροής ή/και ψευδοκώδικα. Στον τομέα του προγραμματισμού η εφαρμογή των αλγορίθμων πραγματοποιείται με τη χρήση κάποιας γλώσσας προγραμματισμού και τη δημιουργία και εφαρμογή εντολών (Grover & Pea, 2018). Στην Υπολογιστική Σκέψη η χρησιμοποίηση αλγορίθμων ή αλγοριθμικής σκέψης κρίνεται απαραίτητη για τη διατύπωση της λύσης ενός προβλήματος ή κατά το σχεδιασμό ενός συστήματος και όπως διαφάνηκε από την ιστορική αναδρομή του όρου, η διαδικασία εύρεσης των κατάλληλων αλγορίθμων θεωρείται πολύ σημαντική νοητική διεργασία και αποτελεί ισχυρό εργαλείο.

1.3.5 Γενίκευση

Η γενίκευση είναι μια τεχνική διαχείρισης της πολυπλοκότητας και χρησιμοποιείται για την απλοποίηση μιας σύνθετης διαδικασίας ή ενός προβλήματος που χρήζει επίλυσης. Σκοπός είναι τα κοινά χαρακτηριστικά που παρατηρούνται να ομαδοποιηθούν, ούτως ώστε να οδηγήσουν στη μείωση της πολυπλοκότητας ενός προβλήματος ή μιας διαδικασίας. Ένας συνηθισμένος τρόπος εφαρμογής της γενίκευσης είναι η εύρεση και

η διατύπωση μιας λύσης, που να μπορεί να εφαρμοστεί σε περισσότερες από μία περιπτώσεις, η επαναχρησιμοποίηση μιας ήδη υπάρχουσας λύσης ή μέρους αυτής, σε ένα ανεπίλυτο πρόβλημα (Angeli, et al., 2016). Στην Υπολογιστική Σκέψη, ο επαγωγικός αυτός τρόπος, δηλαδή η μετακίνηση από το ειδικό στο γενικό θεωρείται σημαντικός στον τομέα του προγραμματισμού ή της σχεδίασης αλγορίθμων, ενώ συνήθως παράλληλα εφαρμόζεται και η τεχνική της αφαίρεσης.

1.3.6 Αξιολόγηση

Κατά τη διαδικασία επίλυσης ενός προβλήματος, η αξιολόγηση εφαρμόζεται είτε κατά τον σχεδιασμό μιας λύσης είτε έπεται χρονικά και θεωρείται σημαντικό κομμάτι της Υπολογιστικής Σκέψης. Η αξιολόγηση της λύσης αναφέρεται τόσο στην επίτευξη ή μη του σκοπού για τον οποίο δημιουργήθηκε, όσο και στην επιλογή της λύσης που θα χρησιμοποιηθεί, σε περίπτωση που υπάρχουν παραπάνω από μία επιλογές. Επίσης, ένα ακόμα γνώρισμα της αξιολόγησης είναι ο έλεγχος για την αποτελεσματικότητα και την αποδοτικότητα των πόρων που χρησιμοποιήθηκαν (Csizmadia, et al., 2015). Στον τομέα του προγραμματισμού, η αξιολόγηση μιας λύσης δεν είναι μια τυποποιημένη διαδικασία. Κάθε φορά θα πρέπει να λαμβάνονται υπόψη διαφορετικά κριτήρια, συνήθως αλληλένδετα μεταξύ τους, ώστε να εντοπίζεται ο κατάλληλος συνδυασμός που ταιριάζει στη συγκεκριμένη περίπτωση.

2 Εργαλεία και Πλαίσια Αξιολόγησης

Η αξιολόγηση αποτελεί σημαντικό στάδιο στην ανάπτυξη της Υπολογιστικής Σκέψης, καθώς προσφέρει ανατροφοδότηση του επιπέδου γνώσης των συμμετεχόντων, του διδακτικού υλικού που χρησιμοποιείται και της διδακτικής προσέγγισης που ακολουθείται. Παρότι έχει αναγνωριστεί η συμβολή της στην επιτυχή ολοκλήρωση της εκπαιδευτικής διαδικασίας, εν τούτοις η διαδικασία καθορισμού ενός συγκεκριμένου τρόπου αξιολόγησης βρίσκεται ακόμα σε εξέλιξη (Bocconi, et al., 2016); (Lockwood & Mooney, 2018).

Η αξιολόγηση μπορεί να πραγματοποιηθεί είτε κατά τη διάρκεια της εκπαιδευτικής διαδικασίας, με σκοπό την άμεση ανατροφοδότηση για τον επανασχεδιασμό της διδακτικής προσέγγισης (Grover, Basu, Bienkowski, Eagle, Diana, & Stamper, 2017), είτε μετά την ολοκλήρωσή της για το σχηματισμό μιας πληρέστερης εικόνας (Román-González, Pérez-González, Moreno-León, & Robles, 2018). Η διαδικασία αξιολόγησης που απαντάται συχνότερα στη βιβλιογραφία αφορά στην αποτίμηση του τελικού αποτελέσματος της εργασίας που έφεραν εις πέρας οι εκπαιδευόμενοι (Chen, Shen, Barth-Cohen, Jiang, Huang, & Eltoukhy, 2017). Σκοπός του συγκεκριμένου κεφαλαίου είναι η διερεύνηση των διαθέσιμων εργαλείων μέτρησης και πλαισίων αξιολόγησης.

2.1 Πλαίσιο FACT

Το συγκεκριμένο «σύστημα αξιολόγησης» (Grover, 2017) χρησιμοποιήθηκε σε ένα μάθημα μέσης εκπαίδευσης με τίτλο FACT (Foundations for Advancing Computational Thinking) με σκοπό την εξοικείωση των συμμετεχόντων με τις έννοιες της Επιστήμης των Υπολογιστών και του προγραμματισμού καθώς και την ανάπτυξη δεξιοτήτων Υπολογιστικής Σκέψης.

Για την αξιολόγηση της επίδοσης των εκπαιδευόμενων πραγματοποιήθηκε ποσοτική και ποιοτική ανάλυση των δεδομένων που αντλήθηκαν κατά τη διάρκεια της εκπαιδευτικής περιόδου. Αρχικά διερευνήθηκε η προηγούμενη εμπειρία τους σε προγραμματιστικά περιβάλλοντα. Στη συνέχεια, με τη χρήση ερωτήσεων κατανόησης κώδικα και αποσφαλμάτωσης, πραγματοποιήθηκε η αρχική μέτρηση συγκεκριμένων δεξιοτήτων της Υπολογιστικής Σκέψης, όπως αυτές είχαν οριστεί από το FACT. Μετά την ολοκλήρωση του προγράμματος πραγματοποιήθηκε επανάληψη των μετρήσεων προκειμένου να γίνει η σύγκρισή με τις αρχικές.

Κατά τη διάρκεια του εκπαιδευτικού προγράμματος οι συμμετέχοντες κλήθηκαν να απαντήσουν σε γραπτές δοκιμασίες πολλαπλής επιλογής ή σε ερωτήσεις ανοιχτού τύπου, με σκοπό να κατανοήσουν καλύτερα και ταχύτερα συγκεκριμένες έννοιες και δομές της Επιστήμης των Υπολογιστών. Βασικό στοιχείο των ασκήσεων αυτών ήταν η αυτόματη βαθμολόγησή και η ανατροφοδότηση που πρόσφεραν στους συμμετέχοντες, με αποτέλεσμα τον άμεσο εντοπισμό των δυσκολιών που αντιμετώπιζαν. Οι πληροφορίες αυτές διευκόλυναν το διδάσκοντα στον επανασχεδιασμό του μαθήματος και τους συμμετέχοντες στη βελτίωση της επίδοσής. Το αποτέλεσμα ήταν ένα πιο ευέλικτο πλαίσιο μάθησης.

Ένα άλλο μέσο αξιολόγησης ήταν οι κλίμακες διαβαθμισμένων κριτηρίων (rubrics) σχετικών με την ευρηματικότητα. Σκοπός ήταν η αξιολόγηση των προγραμματιστικών εργασιών που διεκπεραίωναν οι εκπαιδευόμενοι κατά τη διάρκεια του προγράμματος με χρήση οπτικής γλώσσας προγραμματισμού (Scratch), μέσω της εκτέλεσης υπολογιστικών εννοιών όπως Δόμηση, Αλγοριθμική και Υπολογιστική Σκέψη.

Η τελική εργασία που παρέδωσαν οι εκπαιδευόμενοι αξιολογήθηκε με τη χρήση κλιμάκων διαβαθμισμένων κριτηρίων σχετικών με το σχεδιασμό παιχνιδιών και προγραμματισμού. Διενεργήθηκαν και συμπληρωματικές συνεντεύξεις με τους συμμετέχοντες αναφορικά με το περιεχόμενο της εργασίας που παρέδωσαν προκειμένου να ελεγχθεί ο βαθμός κατανόησης των εννοιών της Υπολογιστικής Σκέψης. Το θέμα της τελικής εργασίας ήταν ελεύθερης επιλογής, που οι συμμετέχοντες παρουσίασαν στους συμφοιτητές τους και ακολουθούσε συζήτηση με αφορμή τα σχόλια και τις παρατηρήσεις τους.

Το τελευταίο εργαλείο του συστήματος αξιολόγησης ήταν το PFL (Preparation for Future Learning) τεστ, με το οποίο αντλήθηκαν δεδομένα σχετικά με την ικανότητα των εκπαιδευόμενων να μεταφέρουν στη συγγραφή κώδικα κειμένου, τις έννοιες που διδάχθηκαν και απέκτησαν κατά την ενασχόλησή τους με τον προγραμματισμό δομημένων μπλοκ.

Τα διαφορετικά εργαλεία μέτρησης που σχεδιάστηκαν και χρησιμοποιήθηκαν στο FACT αξιολογούσαν τους διαφορετικούς τρόπους μάθησης των εκπαιδευόμενων και παρείχαν ποικίλες ευκαιρίες αξιοποίησης των γνώσεων που είχαν αποκτήσει.

Η χρήση μεμονωμένων εργαλείων αξιολόγησης δεν θα αποτύπωνε τις διάφορες πτυχές γνώσης που καλούνταν να αποκτήσουν οι εκπαιδευόμενοι, καθώς μεμονωμένα, δεν

παρείχαν την ευκαιρία στους εκπαιδευόμενους να αναδείξουν τις γνώσεις που απέκτησαν. Κάθε εργαλείο επιτελούσε έναν σκοπό και συνδυαστικά πρόσφεραν καλύτερη κατανόηση του τρόπου μάθησης

2.2 Εργαλείο Αξιολόγησης Υπολογιστικής Σκέψης σε Πραγματικό Χρόνο

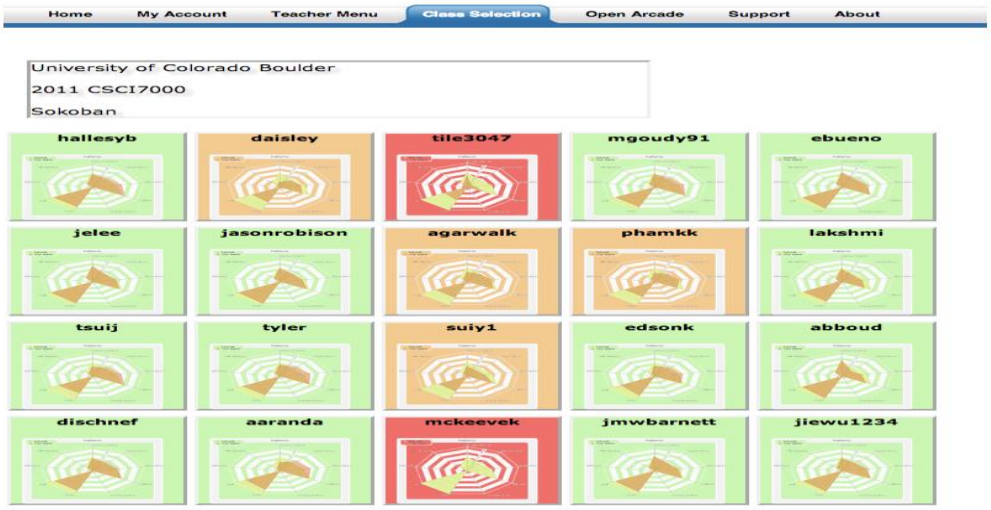
Το συγκεκριμένο εργαλείο είναι γνωστό ως REACT (Real Time Evaluation and Assessment of Computational Thinking) (Koh, Basawapatna, Nickerson, & Repenning, 2014a). Το βασικό του πλεονέκτημα είναι ότι προσφέρει αξιολόγηση των εκπαιδευόμενων σε πραγματικό χρόνο καθώς δημιουργούν παιχνίδια ή επιστημονικές προσομοιώσεις. Το χαρακτηριστικό αυτό το διαφοροποιεί από άλλα εργαλεία, που έχουν δημιουργηθεί για τον ίδιο σκοπό, αλλά αξιολογούν μόνο το τελικό αποτέλεσμα.

Σκοπός του REACT είναι να έχει ο διδάσκων εικόνα του επιπέδου κατανόησης και των δυσκολιών που ενδεχομένως αντιμετωπίζουν οι εκπαιδευόμενοι με τις έννοιες της Υπολογιστικής Σκέψης. Από τεχνικής άποψης είναι ένα διαδικτυακό εργαλείο που:

- χρησιμοποιείται σε διάφορα υπολογιστικά περιβάλλοντα και υποστηρίζει το διδάσκοντα στην αποτελεσματική επίτευξη των μαθησιακών στόχων λόγω του πλεονεκτήματος που προαναφέρθηκε
- αποτυπώνει την ατομική, αλλά και τη συνολική επίδοση των εκπαιδευόμενων, καθώς και την εξοικείωσή τους με το περιεχόμενο του μαθήματος, με τη σχετική ενημέρωση του διδάσκοντα
- παρέχει οπτικοποίηση των πληροφοριών που προαναφέρθηκαν, ώστε οι διδάσκοντες να μπορούν να προσαρμόσουν ανάλογα και σε σύντομο χρονικό διάστημα τη διδακτική τους προσέγγιση.

Το REACT συλλέγει, αποθηκεύει και συγκρίνει σε πραγματικό χρόνο πληροφορίες που συγκεντρώνει από τις προγραμματιστικές εργασίες των συμμετεχόντων, με τη χρήση Ανάλυσης Μοτίβων Υπολογιστικής Σκέψης (Computational Thinking Pattern Analysis). Τα αποτελέσματα που προκύπτουν από τα μοτίβα, παρουσιάζονται με πολλαπλούς τρόπους απεικόνισης είτε με μορφή γραφημάτων εξέλιξης για κάθε εκπαιδευόμενο είτε με αποτύπωση της συνολικής απόδοσής. Τα συγκεκριμένα μοτίβα είναι αφαιρετικές προγραμματιστικές αναπαραστάσεις που επιτρέπουν τη διαχείριση αντικειμένων του παιχνιδιού ή των προσομοιώσεων (Koh, Nickerson, Basawapatna, & Repenning, 2014b). Στη συνολική οπτική απεικόνιση της επίδοσης των

εκπαιδευόμενων χρησιμοποιείται χρωματική σήμανση προς διευκόλυνση του διδάσκοντα. Πιο συγκεκριμένα, με πράσινο χρώμα επισημαίνονται οι εκπαιδευόμενοι που δεν αντιμετωπίζουν προβλήματα στις προγραμματιστικές εργασίες, με πορτοκαλί όσοι ενδεχομένως χρειάζονται υποστήριξη και με κόκκινο οι συμμετέχοντες που αντιμετωπίζουν δυσκολίες και χρειάζονται άμεση υποστήριξη (βλ.εικόνα1).



Εικόνα 1: Παράδειγμα συνολικής οπτικής απεικόνισης (Koh, Basawapatna, Nickerson, & Repenning, 2014a)

2.3 Εργαλείο Αξιολόγησης Συνολικού Πλαισίου

Σύμφωνα με το προτεινόμενο πλαίσιο αξιολόγησης (Basso, Fronza, Colombi, & Pahl, 2018), η Υπολογιστική Σκέψη θα πρέπει να εξετάζεται μέσα από ένα εκτεταμένο φάσμα παραγόντων καθώς η μερική μελέτη των δεξιοτήτων που την αποτελούν δεν αποδίδει πάντα αξιόπιστα αποτελέσματα. Συνεπώς για την αξιολόγησή της θα πρέπει να χρησιμοποιείται συνδυασμός προγραμματισμού, εκμάθησης και δεξιοτήτων, απαραίτητων στην καθημερινή ζωή. Εχει παρατηρηθεί ότι καθ' όλη τη διάρκεια της εκπαιδευτικής διαδικασίας υπάρχει διακύμανση επίδοσης των εκπαιδευόμενων στον προγραμματισμό καθώς και στα συναισθήματα που βιώνουν. Έχει επίσης διαπιστωθεί, ότι όσο πιο ευρύ είναι το φάσμα εξάσκησης, τόσο διευρύνεται και η εμπειρία μάθησης, ενώ όσο περισσότερες εμπειρίες αποκομίζει ο εκπαιδευόμενος, τόσο αναπτύσσονται οι γνωστικές και κοινωνικές του δεξιότητες. Σύμφωνα με το συγκεκριμένο πλαίσιο, η χρονική στιγμή της αξιολόγησης διαφέρει ανάλογα με τον υπό εξέταση παράγοντα (βλ.είκονα 2).

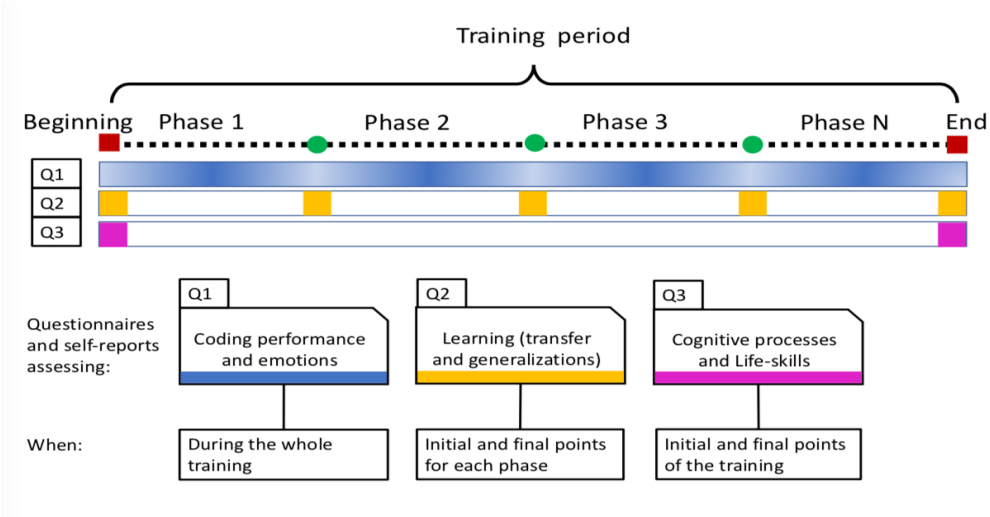
Η πρώτη κατηγορία παραγόντων, δηλαδή ο προγραμματισμός και η συναισθηματική αντίδραση, θα πρέπει να καταγράφεται την ίδια χρονική στιγμή και με διάφορες μεθόδους αξιολόγησης λόγω της μεταβλητής τους φύσης. Για την επαλήθευση της ορθότητας του τελικού αποτελέσματος, προτείνεται μια και μόνο καταγραφή στο τέλος της εκπαιδευτικής περιόδου, με τη χρήση ερωτηματολογίου.

Το δεύτερο μέρος του πλαισίου αποτελεί και το εκπαιδευτικό τμήμα της διαδικασίας και χωρίζεται σε φάσεις, ανάλογα με τις εκάστοτε ανάγκες. Η αξιολόγηση γίνεται με τη χορήγηση ερωτηματολογίου και με αναφορές αυτοαξιολόγησης στην αρχή και στο τέλος της κάθε φάσης. Η επανάληψη αξιολόγησης της επίδοσης των συμμετεχόντων θα πρέπει να περιλαμβάνει τα ίδια στοιχεία, διαφορετικά εκφρασμένα για αποφυγή μηχανικού τρόπου απάντησης. Επιπρόσθετα εφαρμόζονται και εξειδικευμένοι τρόποι αξιολόγησης, ανάλογα με τον στόχο επίτευξης της κάθε φάσης.

Το τρίτο μέρος του πλαισίου πραγματοποιείται με τη λήψη δύο μετρήσεων: η πρώτη λειτουργεί ως σημείο αναφοράς και η δεύτερη ως τελικό αποτέλεσμα. Στόχος είναι η συγκέντρωση και αξιολόγηση δεδομένων σχετικών με τις γνωστικές δυνατότητες των εκπαιδευόμενων και δεξιότητες, που είναι απαραίτητες στην καθημερινότητά τους, οι οποίες χρειάζονται χρόνο για να καλλιεργηθούν και να εδραιωθούν. Συνεπώς, ενώ η καθημερινή μέτρηση είναι εφικτή, εν τούτοις η καταγραφή σε τόσο σύντομα χρονικά διαστήματα θεωρείται αναποτελεσματική.

Με την εφαρμογή του παραπάνω συνδυαστικού πλαισίου, επιτυγχάνεται βραχυπρόθεσμα και μακροπρόθεσμα η αξιολόγηση ποιοτικών και ποσοτικών δεδομένων της αποτελεσματικότητας εκμάθησης της Υπολογιστικής Σκέψης. Παράλληλα συλλέγονται και σημαντικές πληροφορίες που συμβάλλουν στη βελτίωση μεθόδων εκμάθησης. Αναφορικά με την αξιολόγηση αποτελεσματικότητας των εκπαιδευτικών φάσεων λαμβάνονται υπόψη οι αρχικές και τελικές μετρήσεις των συνθηκών που επηρέασαν την επίδοση. Η συλλογή όλων των προαναφερθέντων δεδομένων βοηθάει στον εντοπισμό στοιχείων που ενισχύουν τις δεξιότητες εκμάθησης.

Συμπερασματικά, το συγκεκριμένο πλαίσιο θα πρέπει να συνδυάζεται με την αξιολόγηση υπολογιστικών τεχνικών και εννοιών, προκειμένου να σχηματιστεί μια πιο ολοκληρωμένη εικόνα και κατ’ επέκταση βελτίωση της εκπαιδευτικής διαδικασίας.



Εικόνα 2: Παράδειγμα συνδυαστικού πλαισίου αξιολόγησης (Basso, Fronza, Colombi, & Pahl, 2018)

2.4 Τεστ Υπολογιστικής Σκέψης

Το Computational Thinking Test (CTt) (Roman-Gonzalez, Perez-Gonzalez, & Jimenez-Fernandez, 2017) είναι ένα διαδικτυακό εργαλείο αξιολόγησης της Υπολογιστικής Σκέψης με μέγιστο χρόνο συμπλήρωσης τα 45 λεπτά. Αρχικά αποτελούνταν από 40 ερωτήσεις πολλαπλής επιλογής, αλλά μετά τη διαδικασία αξιολόγησης της εγκυρότητας του περιεχομένου του και μετά από πρόταση είκοσι ειδικών, έλαβε την τελική του μορφή και πλέον αποτελείται από 28 ερωτήσεις πολλαπλής επιλογής. Η οπτική αναπαράσταση των απαντήσεών του τεστ πραγματοποιείται με τη χρήση «λαβυρίνθου» (Maze) ή «καμβά» (βλ. εικόνα 3) (Canvas) (Roman-Gonzalez, Moreno-Leon, & Robles, 2017).

Η εγκυρότητα κριτηρίου του τεστ επιβεβαιώθηκε μέσα από την ταυτόχρονη χορήγηση παρόμοιων τεστ, όπως είναι το Τεστ Μέτρησης Βασικών Προγραμματιστικών Ικανοτήτων (Test for Measuring Basic Programming Abilities) καθώς και το Τεστ Αντιμεταθετικής Αξιολόγησης (Commutative Assessment), το οποίο αξιολογεί το επίπεδο κατανόησης βασικών υπολογιστικών εννοιών των συμμετεχόντων, με τη χρήση είτε μπλοκ προγραμματισμού είτε κώδικα κειμένου.

Το CTt σταθμίστηκε με τη χρήση των ακόλουθων δυο αναγνωρισμένων εργαλείων. Το Primary Mental Abilities (PMA battery) που παρέχει ακριβείς μετρήσεις για την προφορική, αριθμητική, λογική και χωροταξική ικανότητα των συμμετεχόντων και το

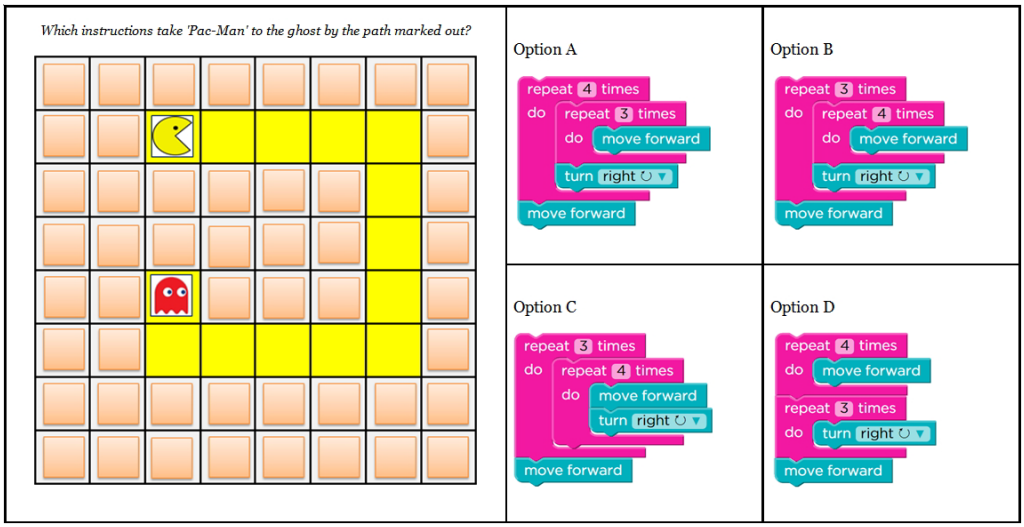
RP30 problem-solving test που καταγράφει την ταχύτητα και την ευελιξία κατά την επίλυση προβλημάτων.

Στόχος του τεστ είναι η αξιολόγηση της Υπολογιστικής Σκέψης και πιο συγκεκριμένα της ικανότητας δημιουργίας και λύσης προβλημάτων με τη χρήση οπτικής γλώσσας προγραμματισμού. Για την απάντηση των ερωτήσεων πολλαπλής επιλογής οι συμμετέχοντες καλούνται να επιλέξουν τη μια από τις τέσσερεις προσφερόμενες απαντήσεις.

Οι υπολογιστικές έννοιες τις οποίες αξιολογεί το τεστ είναι: «*βασικές κατευθύνσεις και αλληλουχίες*» (Basic directions and sequences), «*βρόγχοι – αριθμός επαναλήψεων*» (Loops - repeat times), «*βρόγχοι – επανάλαβε μέχρι*» (Loops - repeat until), *«δομή απόφασης If»* (If - simple conditional), *«σύνθετη δομή απόφασης If/else»* (If/else - complex conditional), *«δομή επανάληψης While»* (While conditional) και *«απλές συναρτήσεις»* (Simple functions). Οι ασκήσεις είναι αυξανόμενης δυσκολίας και διαφοροποιούνται ανάλογα με τις δεξιότητες της Υπολογιστικής Σκέψης που εφαρμόζονται, όπως είναι η αλληλουχία, η ολοκλήρωση και η αποσφαλμάτωση. Η οπτικοποίηση των απαντήσεων πραγματοποιείται με οπτικά βέλη και μπλοκ προγραμματισμού (Roman-Gonzalez, Moreno-Leon, & Robles, 2017).

Το Τεστ Υπολογιστικής Σκέψης χορηγήθηκε σε 1251 συμμετέχοντες και διαπιστώθηκε ότι βοηθάει στην διαχωρισμό των συμμετεχόντων ανάλογα με το επίπεδο των δεξιοτήτων τους στην Υπολογιστική Σκέψη. Η στατιστική επεξεργασία των δεδομένων που προκύπτουν από τη χορήγηση του τεστ σε δύο φάσεις, συμβάλλει στη βελτίωση της διαδικασίας εκμάθησης δεξιοτήτων της Υπολογιστικής Σκέψης. Επίσης, από την αρχική χορήγηση του τεστ είναι εύκολο να διαπιστωθεί το επίπεδο Υπολογιστικής Σκέψης των συμμετεχόντων χωρίς προγραμματιστική εμπειρία (Roman-Gonzalez, Perez-Gonzalez, & Jimenez-Fernandez, 2017).

Ένας από τους βασικούς περιορισμούς του τεστ είναι ότι δεν προσφέρει έναν ολοκληρωμένο τρόπο αξιολόγησης της Υπολογιστικής Σκέψης, όπως συμβαίνει και με παρόμοια εργαλεία. Οι Roman - Gonzalez et al σε μία προσπάθεια βελτίωσής του, υιοθέτησαν την ιδέα ενός πλαισίου αξιολόγησης με τα εργαλεία Bebras Tasks και Dr. Scratch για μελέτη διαφορετικών διαστάσεων των εννοιών της Υπολογιστικής Σκέψης (Roman-Gonzalez, Moreno-Leon, & Robles, 2017).



Εικόνα 3: Παράδειγμα τεστ Υπολογιστικής Σκέψης (Roman-Gonzalez, Moreno-Leon, & Robles, 2017)

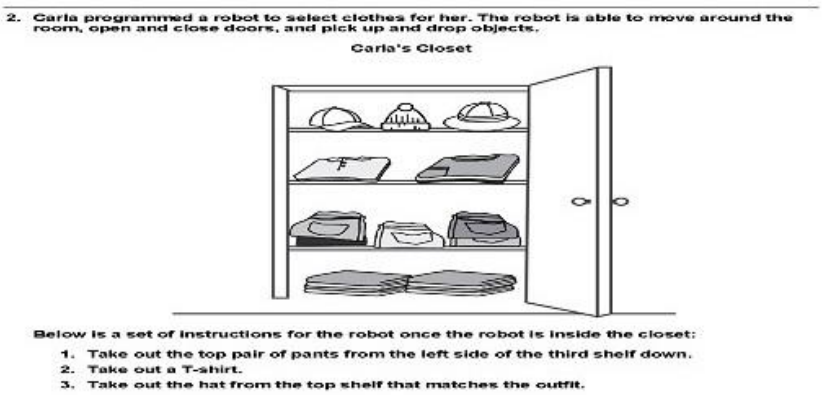
2.5 Αξιολόγηση Βασισμένη σε Αρχές

Το συγκεκριμένο μοντέλο αξιολόγησης εφαρμόστηκε στο πρόγραμμα σπουδών ECS (Exploring Computer Science) της δευτεροβάθμιας εκπαίδευσης, που έχει ως αντικείμενο την Επιστήμη των Υπολογιστών. Αποτελείται από τέσσερις θεματικές ενότητες με στόχο την εκμάθηση υπολογιστικών εννοιών και δεξιοτήτων Υπολογιστικής Σκέψης (Snow, Rutstein, Bienkowski, & Xu, 2017).

Η αξιολόγηση που εφαρμόστηκε βασίστηκε στο ECD (Evidence-Centered Design), το οποίο συνδέει συγκεκριμένα στοιχεία ενός έργου (task), με την επίδοση των συμμετεχόντων σε αυτό και τις γνώσεις και δεξιότητες που απαιτούνται για την διεκπεραίωσή του. Ο σχεδιασμός του πλαισίου περιλαμβάνει τρία στάδια. Πρώτον, την ανάλυση πεδίου (Domain Analysis,) κατά την οποία καθορίζεται το εν λόγω πεδίο και συγκεντρώνονται βασικές πληροφορίες από διάφορες πηγές. Δεύτερον, τη μοντελοποίηση πεδίου (Domain Modeling), κατά την οποία οι πληροφορίες που συλλέχθηκαν οργανώνονται, προκειμένου να δημιουργηθεί ένα θεωρητικό προσχέδιο που καθορίζει τις γνώσεις, τις δεξιότητες και τις συνθήκες που απαιτούνται, για να προκύψει το επιθυμητό αποτέλεσμα. Και τέλος, το εννοιολογικό πλαίσιο αξιολόγησης (Conceptual Assessment Framework), όπου ορίζεται με ακρίβεια ο τρόπος σχεδιασμού των ασκήσεων, των αξιολογήσεων και των εργαλείων μέτρησης και βαθμολόγησης, προκειμένου να είναι εφικτή η πρακτική εφαρμογή του (Snow, Tate, Rutstein, & Bienkowski, 2017).

Στην περίπτωση του προγράμματος ECS, αντλήθηκαν πληροφορίες από προηγούμενες έρευνες πάνω στις πρακτικές της Υπολογιστικής Σκέψης όπως και από μαθησιακούς στόχους και δραστηριότητες που περιλαμβάνονταν στο διδακτικό πρόγραμμα. Στη συνέχεια καθορίστηκαν οι γνώσεις και οι δεξιότητες που έπρεπε να αναλυθούν, ενώ παράλληλα εντοπίστηκαν και επιλέχθηκαν μετρήσιμα στοιχεία απαραίτητα στο σχεδιασμό της θεωρητικής δομής. Πάνω σε αυτή τη δομή βασίστηκε η δημιουργία των ασκήσεων. Κάθε ενότητα του προγράμματος ECS περιελάμβανε οκτώ (8) έως δέκα (10) ασκήσεις με τη μορφή υποθετικών σεναρίων. Κάθε σενάριο εστίαζε σε συγκεκριμένες γνώσεις και δεξιότητες της Υπολογιστικής Σκέψης και αποτελούνταν από μια (1) έως έξι (6) ερωτήσεις. Για τη βαθμολόγηση των απαντήσεων των συμμετεχόντων αναπτύχθηκε μια κλίμακα διαβαθμισμένων κριτηρίων, από έναν έως τρεις βαθμούς, ανάλογα με την πολυπλοκότητα της απάντησης. Με την παραπάνω διαδικασία ολοκληρώνονταν η αξιολόγηση των συμμετεχόντων στην κάθε ενότητα.

Το συγκεκριμένο μοντέλο αξιολόγησης εφαρμόστηκε πιλοτικά για δύο χρόνια, από το 2014 έως το 2016. Από τα δεδομένα που συλλέγονταν κατά τον πρώτο χρόνο διαπιστώνονταν αν οι ασκήσεις και οι αντίστοιχες αξιολογήσεις επιτύγχαναν το σκοπό για τον οποίο είχαν δημιουργηθεί ή έπρεπε να αναθεωρηθούν. Το δεύτερο χρόνο, στις τέσσερις υπάρχουσες αξιολογήσεις προστέθηκε και μια πέμπτη για τη συνολική αξιολόγηση του προγράμματος. Με τον τρόπο αυτό διαμορφώθηκε η τελική μορφή του μοντέλου αξιολόγησης και επιβεβαιώθηκε η αποτελεσματικότητά του. Εν κατακλείδι, το μοντέλο αυτό φαίνεται να βελτιώνει τον τρόπο αξιολόγησης της Υπολογιστικής Σκέψης, αν και απαιτείται περαιτέρω μελέτη για τη γενίκευση της εφαρμογής του (Snow, Rutstein, Bienkowski, & Xu, 2017).



Εικόνα 4: Παράδειγμα άσκησης (Snow, Rutstein, Bienkowski, & Xu, 2017)

3 Περιπτώσεις Ενσωμάτωσης Υπολογιστικής Σκέψης

Στον παρόν κεφάλαιο θα γίνει αναφορά στους τρόπους με τους οποίους τα εκπαιδευτικά ιδρύματα εισάγουν την έννοια της Υπολογιστικής Σκέψης στα προπτυχιακά προγράμματα σπουδών τους. Στη συνέχεια, θα περιγραφεί η διαδικασία με την οποία η Υπολογιστική Σκέψη ενσωματώνεται στα εκπαιδευτικά προγράμματα, μέσω της εφαρμογής εκπαιδευτικού υλικού που αποσκοπεί στην ανάπτυξη των δεξιοτήτων του συγκεκριμένου τρόπου σκέψης. Οι δυο τρόποι που χρησιμοποιούνται πιο συχνά για την ενσωμάτωσή της είναι η εφαρμογή συγκεκριμένης γλώσσας προγραμματισμού (Cooper & Dann, 2015) και η εκτέλεση δραστηριοτήτων που δεν προϋποθέτουν τη χρήση υπολογιστικών συσκευών (Hunsaker, 2016); (Tsarava, Moeller, Pinkwart, Butz, Trautwein, & Ninaus, 2017). Ακολούθως παρατίθεται πίνακας με τα πλεονεκτήματα και τα μειονεκτήματα της κάθε περίπτωσης.

3.1 Η περίπτωση του Κολεγίου Colby, Ηνωμένες Πολιτείες Αμερικής (Maxwell & Taylor, 2017)

Το Κολλέγιο Colby των Ηνωμένων Πολιτειών προσφέρει από το 2008 ένα εισαγωγικό μάθημα πληροφορικής Οπτικών Μέσων CS151 με αντικείμενο τη σχεδίαση σύνθετων εικόνων και σκηνικών μέσω προγραμματισμού (Visual Media).

Το 2015 το πρόγραμμα σπουδών εμπλουτίστηκε με την προσθήκη μίας νέας εκδοχής του μαθήματος που εστιάζει στις Επιστημονικές Εφαρμογές CS151s, με αντικείμενο τη διαχείριση επιστημονικών δεδομένων μέσω προγραμματισμού (Science Applications).

Σκοπός και των δύο εκδοχών του μαθήματος ήταν να κατανοήσουν οι εκπαιδευόμενοι, μέσω διαφόρων ασκήσεων, τις βασικές εισαγωγικές αρχές της πληροφορικής και να εξοικειωθούν με την έννοια της Υπολογιστικής Σκέψης με εφαρμογή δεξιοτήτων, όπως η αφαίρεση, η αποσύνθεση, οι αλγόριθμοι, και η γενίκευση, στην επίλυση σύνθετων προβλημάτων. Επίσης, χρησιμοποιήθηκαν συγκεκριμένοι τύποι προγραμματιστικών εργασιών σε βασικές έννοιες της Επιστήμης των Υπολογιστών, προκειμένου να αποκτήσουν οι συμμετέχοντες πρακτική εμπειρία.

Κατά το χειμερινό ακαδημαϊκό έτος του 2015, στο Τμήμα Οπτικών Μέσων συμμετείχαν 37 εκπαιδευόμενοι, ενώ στο Τμήμα των Επιστημονικών Εφαρμογών 44. Και στις δυο εκδοχές του μαθήματος διδάχθηκε η γλώσσα προγραμματισμού Python. Το εβδομαδιαίο πρόγραμμα περιλάμβανε τρεις διαλέξεις διάρκειας 50 λεπτών, ένα

εργαστήριο διάρκειας 80 λεπτών, μια σύντομη εργασία για το σπίτι, ένα σύντομο κουίζ και μία ολοκληρωμένη προγραμματιστική εργασία (project). Το τμήμα απασχολούσε δυο εισηγητές, έναν για τη διδασκαλία του θεωρητικού μέρους και έναν για το εργαστηριακό μέρος.

Αναφορικά με το υλικό του μαθήματος, αυτό ήταν κοινό για τις εργασίες στο σπίτι, τα τεστ ερωτήσεων και τις τελικές εξετάσεις, ενώ ανάλογα με την εκδοχή του προγράμματος διαφοροποιούνταν οι προγραμματιστικές εργασίες που δίνονταν στους εκπαιδευόμενους κάθε εβδομάδα. Στο τέλος του μαθήματος εξετάστηκαν μέσω ερωτήσεων στην κατανόηση τμημάτων κώδικα Python.

Αναλυτικά στην πρώτη εκδοχή του μαθήματος των Οπτικών Μέσων η ύλη χωριζόταν σε έντεκα εργασίες, δομημένες σε τρεις ομάδες προκειμένου οι εκπαιδευόμενοι να εξοικειωθούν με τις ακόλουθες έννοιες:

Μεταβλητές, Συναρτήσεις, Δομές Επανάληψης, Δομές Απόφασης, Λίστες, Στοίβες, Κλάσεις, Κληρονομικότητα, Πολυμορφισμό, Αναδρομή, Αναφορές, «*Προχωρημένα Μοντέλα Ανάλυσης Ακολουθιών που βασίζονταν στην ιεραρχία των Γραμματικών*» (grammars) (Μπάγκος, 2015) και Διερμηνευτών (interpretation).

Για τις τρεις πρώτες εργασίες οι συμμετέχοντες έκαναν χρήση των γραφικών της βιβλιοθήκης turtle και δημιουργούσαν σχέδια αυξανόμενης πολυπλοκότητας μέσα από συναρτήσεις.

Οι επόμενες τρεις εργασίες απαιτούσαν τη χρήση άλλου πακέτου γραφικών διαχείρισης εικόνων. Όπως και παραπάνω, οι πρώτες δυο ασκήσεις αφορούσαν στην επεξεργασία εικόνων με τη διαχείριση εικονοστοιχείων (pixels) και η τρίτη στη δημιουργία ενός σεναρίου σχεδιοκίνησης.

Στις τελευταίες πέντε εργασίες εφαρμόστηκαν τα μαθηματικά μοντέλα (L-system) και η βιβλιοθήκη turtle για τη δημιουργία ιδιαίτερα σύνθετων εικόνων που ολοκληρώνονταν με τη χρήση ενός τρισδιάστατου πακέτου γραφικών. (CS 151: Computational Thinking: Visual Media)

Η δεύτερη εκδοχή του μαθήματος CS151s, με ίδιο αριθμό εργασιών, αφορούσε στις επιστημονικές εφαρμογές των ίδιων βασικών εννοιών και περιελάμβανε ανάλυση αρχείων (file parsing), αρχεία τιμών διαχωρισμένων με κόμματα (CSV), ένα πιο

προχωρημένο σύνολο εντολών Unix (όπως είναι η grep, η cut και η curl), με διαφοροποίηση ως προς τις έννοιες της Γραμματικής Ιεραρχίας και των Διερμηνευτών.

Οι πρώτες τρείς εργασίες αφορούσαν στη στατιστική επεξεργασία δεδομένων πραγματικού χρόνου, τα οποία συλλέγονταν από έναν σημαντήρα που βρισκόταν εγκαταστημένος στην τοπική λίμνη.

Οι επόμενες τέσσερις εργασίες αποτελούνταν από υποθετικά σενάρια για τον έλεγχο και την υπολογιστική μοντελοποίηση του πληθυσμού πιγκουίνων και ελεφάντων με χρήση προγραμματισμού αυξανόμενης δυσκολίας.

Η τελευταία ομάδα, αποτελούνταν από τέσσερις ασκήσεις, περιλάμβανε το σχεδιασμό δισδιάστατης μοντελοποίησης, βαρύτητας, κρούσης, περιστροφής και αλληλεπίδρασης με τη χρήση του ίδιου πακέτου γραφικών. (CS 151: Computational Thinking: Science Applications)

Προκειμένου να ενισχυθεί η αξιοπιστία του τρόπου αξιολόγησης του εγχειρήματος, οι εκπαιδευόμενοι χωρίστηκαν σε δυο ομάδες και για τη βαθμολογία τους στα εβδομαδιαία τεστ χρησιμοποιήθηκαν, εναλλάξ ανά τρίμηνο, η εικοσαβάθμια κλίμακα και κλίμακες διαβαθμισμένων κριτηρίων.

Τα αποτελέσματα της αξιολόγησης προέκυψαν από τη σύγκριση δυο δεικτών, δηλαδή του μέσου όρου της τελικής εξέτασης και της συνολικής επίδοσης του κάθε τμήματος.

Οι επιδόσεις των συμμετεχόντων και των δυο τμημάτων ήταν υψηλές ενώ δεν παρατηρήθηκε στατιστικώς σημαντική διαφορά μεταξύ τους. Με τον τρόπο αυτό επιβεβαιώθηκε ότι παρά τις διαφορές στο περιεχόμενο των μαθημάτων, δεν υπήρξαν διαφορές στο μαθησιακό αποτέλεσμα. Το γενικό συμπέρασμα της αξιολόγησης ήταν ότι η λειτουργία και των δυο τμημάτων ήταν επιτυχής και επιτεύχθηκε ο σκοπός για τον οποίο συστάθηκαν.

3.2 Η περίπτωση του πανεπιστημίου Guarda, Πορτογαλία (Figueiredo & García-Peñalvo, 2017)

Το Πολυτεχνείο Guarda στην Πορτογαλία πρόσφερε στους εκπαιδευόμενους μηχανικούς υπολογιστών ένα προαιρετικό προπαρασκευαστικό μάθημα διευκόλυνσης των συμμετεχόντων, στα μαθήματα πληροφορικής του προγράμματος σπουδών. Στόχος του μαθήματος ήταν η δημιουργία κινήτρου για την ενασχόληση με τον

προγραμματισμό, η εξοικείωση με τις βασικές έννοιες καθώς και η ενίσχυση της Υπολογιστικής Σκέψης μέσω της επίλυσης προβλημάτων. Έμφαση δινόταν στα χαρακτηριστικά της αφαίρεσης, της αποσύνθεσης, της αναγνώρισης προτύπων και αλγορίθμων, με την ανάθεση εργασιών οι οποίες δεν απαιτούσαν τη χρήση κάποιας γλώσσας προγραμματισμού.

Οι εργασίες αυτές διαχωρίζονταν με βάση το περιεχόμενο και το σκοπό τους σε πέντε ομάδες:

Η πρώτη ομάδα εργασιών με θεματική *«Ακολούθησε και Δώσε οδηγίες»* (Follow and Give instructions), αφορούσε εκφώνηση λεπτομερών οδηγιών, για το σχεδιασμό και την αποτύπωση σχήματος ή εικόνας. Ο εκπαιδευόμενος εκτελούσε βήμα προς βήμα τις οδηγίες για εξάσκηση στη συλλογιστική και στην οπτική απεικόνιση, καθώς και στη μεθοδική προετοιμασία εκμάθησης προγραμματισμού.

Η δεύτερη ομάδα εργασιών με τίτλο *«Σχεδιασμός Χάρτη»* (Map Design) ζητούσε από τον εκπαιδευόμενο να δημιουργήσει μια λεπτομερή διαδρομή μεταξύ δυο καθορισμένων σημείων πάνω σε ένα χάρτη δυο διαστάσεων, να τη σχεδιάσει και να την περιγράψει με σαφήνεια και με τον καλύτερο δυνατό τρόπο. Στόχος της συγκεκριμένης ομάδας ασκήσεων ήταν να εισαγάγει βαθμιαία τον εκπαιδευόμενο στην έννοια του προγραμματισμού.

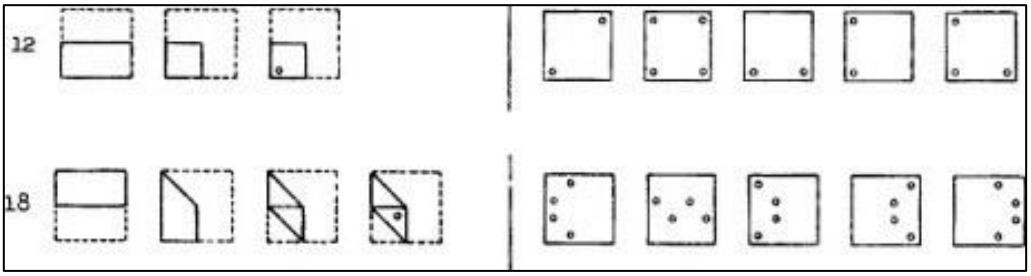
Η τρίτη ομάδα εργασιών με θέμα *«Αναδίπλωση Χαρτιού και Οριγκάμι»* (Paper Folding and Origami) περιλάμβανε μια σειρά βημάτων για τη μεταφορά τρισδιάστατων κατασκευών από χαρτί σε δυο διαστάσεις, με αναδίπλωση και διάτρηση σε συγκεκριμένο σημείο. Ο εκπαιδευόμενος καλούνταν να αντιστρέψει νοερά τη διαδικασία αναδίπλωσης και να επιλέξει τη σωστή από τις προτεινόμενες επιλογές, ανάλογα με τις θέσεις των σημείων διάτρησης (βλ. εικόνα 5). Στόχος των εργασιών της ομάδας αυτής ήταν η ανάπτυξη της ικανότητας οπτικής απεικόνισης.

Η τέταρτη ομάδα εργασιών με τίτλο *«Γλώσσα Μεταφοράς Μνήμης»* (Memory Transfer Language) αποτελούνταν από δυο μέρη: έναν ψευδοκώδικα και έναν πίνακα που αναπαρίστανε τη μνήμη τυχαίας προσπέλασης (RAM). Στις ασκήσεις αυτές δίνονταν τιμές σε ορισμένες μεταβλητές του ψευδοκώδικα, ο εκπαιδευόμενος καλούνταν να συμπληρώσει στον πίνακα τις τιμές αυτές και τις αλλαγές, στην κατάσταση των μεταβλητών. Σκοπός των εργασιών της ομάδας αυτής ήταν να κατανοήσει ο

εκπαιδευόμενος την αφηρημένη έννοια των μεταβλητών και προοδευτικά να εξοικειωθεί με τον προγραμματισμό. (Quitério Figueiredo, 2017)

Η πέμπτη και τελευταία ομάδα εργασιών ήταν τα *«Προβλήματα Parson»* (Parson Problems), τα οποία αποτελούνταν από τρία σκέλη. Το πρώτο σκέλος ήταν η εκφώνηση, όπου περιγράφονταν το ζητούμενο αποτέλεσμα του κώδικα, το δεύτερο τα μεμονωμένα τμήματα του κώδικα και το τρίτο ένας κενός πίνακας. Ο εκπαιδευόμενος καλούνταν να συμπληρώσει τον κενό πίνακα με τα μεμονωμένα τμήματα του κώδικα, ώστε το πρόγραμμα να έχει τη σωστή ροή και σύνταξη. Στόχος της συγκεκριμένης ομάδας ασκήσεων ήταν η ομαλή μετάβαση και εξοικείωση των εκπαιδευόμενων με τη σύνταξη που χρησιμοποιείται στον προγραμματισμό.

Για την αξιολόγηση των δραστηριοτήτων χρησιμοποιήθηκαν δύο ερωτηματολόγια. Το πρώτο αποτελούνταν από 20 ερωτήσεις σχετικές με το περιεχόμενο της τρίτης ομάδας εργασιών *«Αναδίπλωση Χαρτιού και Οριγκάμι»*, ενώ το δεύτερο ερωτηματολόγιο αποτελούνταν από πέντε δραστηριότητες σχετικές με την πρώτη και την δεύτερη ομάδα ασκήσεων *«Ακολούθησε και Δώσε Οδηγίες»* και *«Σχεδιασμός Χάρτη»*. Τα αποτελέσματα που προέκυψαν από το δείγμα των 46 εκπαιδευόμενων έδειξαν, ότι η επιτυχία του μαθήματος ήταν άμεσα συνυφασμένη με τις εργασίες που ενίσχυαν την Υπολογιστική Σκέψη.



Εικόνα 5: Παράδειγμα αναδίπλωσης χαρτιού (Jaeger, 2015)

3.3 Η περίπτωση του πανεπιστημίου Virginia Tech, Ηνωμένες Πολιτείες Αμερικής

Στη συνέχεια περιγράφεται η περίπτωση του Πανεπιστημίου Virginia Tech (Kafura, Bart, & Chowdhury, 2015), το οποίο πρόσθεσε στο εκπαιδευτικό του πρόγραμμα ένα μάθημα Υπολογιστικής Σκέψης το φθινόπωρο του 2014 και μια επανασχεδιασμένη εκδοχή του το 2018 (Kafura, Bart, & Chowdhury, 2018). Στόχος και των δυο εκδοχών ήταν να διδαχθούν οι εκπαιδευόμενοι της έννοιες της Υπολογιστικής Σκέψης μέσα από διαφορετική μαθησιακή προσέγγιση. Βασικά στοιχεία της προσέγγισης αυτής ήταν: α)

η επαναλαμβανόμενη εφαρμογή των βασικών εννοιών της Υπολογιστικής Σκέψης με διαφορετικούς τρόπους και μέσα, β) ο σχηματισμός των εκπαιδευόμενων σε ομάδες που απαρτίζονταν από άτομα που προέρχονταν από διαφορετικούς επιστημονικούς τομείς, γ) η χρήση μεγάλων δεδομένων (Big Data) και δ) η ομαλή μετάβαση των εκπαιδευόμενων από τον προγραμματισμό με δομημένα μπλοκ στη χρήση γλώσσας προγραμματισμού Python.

Αναλυτικότερα, οι μαθησιακοί στόχοι ήταν να αποκτήσουν οι εκπαιδευόμενοι εμπειρία στη διατύπωση προβλημάτων, να εξασκηθούν στην επίλυσή τους με τις τεχνικές της Υπολογιστικής Σκέψης, να μπορούν να δίνουν παραδείγματα Υπολογιστικής Σκέψης και να διακρίνουν τη σημασία των τεχνικών της, σε διάφορους επιστημονικούς τομείς. Οι συμμετέχοντες, εκπαιδεύονταν στην απλοποίηση των τρόπων μοντελοποίησης και στη χρήση τους, για την ερμηνεία σύνθετων ή ευρείας κλίμακας φαινομένων, με τη βοήθεια της υπολογιστικής καθώς και στην αξιοποίηση των τεχνικών της υπολογιστικής στην καθημερινή ζωή. Ήταν χωρισμένοι σε ομάδες, οι οποίες διατηρούνταν καθ' όλη τη διάρκεια της εκπαίδευσης των 15 εβδομάδων, ενώ ο εκπαιδευόμενος επέλεγε τα δεδομένα ανάλογα με τον επιστημονικό του τομέα. Οι θεματικές ενότητες του μαθήματος χωρίζονταν με τον ακόλουθο τρόπο:

Οι πρώτες τέσσερις αφορούσαν υπολογιστική μοντελοποίηση και προσομοίωση με τη χρήση του προγραμματιστικού περιβάλλοντος NetLogo. Επιπλέον η παραμετροποίηση και ο αντίκτυπός της σε συνδυασμό με την οπτικοποίηση βοηθούσε στην καλύτερη κατανόηση της αφαίρεσης και των στοιχείων επιρροής της συμπεριφοράς ενός μοντέλου.

Στόχος των επόμενων δύο ήταν η εκμάθηση των βασικών εννοιών του προγραμματισμού με τη χρήση δομημένων μπλοκ (Blocky), σε συνδυασμό με μεγάλα δεδομένα. Ένα σημαντικό πλεονέκτημα της συγκεκριμένης γλώσσας προγραμματισμού είναι η δυνατότητα εμφάνισης των δομημένων μπλοκ με τον τρόπο σύνταξης της Python, γεγονός που διευκολύνει την ομαλότερη μετάβαση σε αυτήν.

Οι επόμενες επτά ήταν αφιερωμένες στη δημιουργία μιας ολοκληρωμένης εργασίας η οποία συνδύαζε, τη χρήση αλγορίθμων εκφρασμένων με τη σύνταξη της python, τα μεγάλα δεδομένα στα οποία έπρεπε να υπάρξει επεξεργασία και τη χρήση συγκεκριμένης βιβλιοθήκης για οπτικοποίηση του αποτελέσματος σε μορφή διαγραμμάτων.

Οι τελευταίες δυο περιλάμβαναν την ενημέρωση των εκπαιδευόμενων για τον τρόπο εφαρμογής της υπολογιστικής σε κοινωνικά θέματα.

Βασικός σκοπός του μαθήματος ήταν να κατανοήσουν οι εκπαιδευόμενοι τα χαρακτηριστικά της Υπολογιστικής Σκέψης και ιδίως την αφαίρεση, την αποσύνθεση, τη γενίκευση – αναγνώριση προτύπων και τους αλγόριθμους, μέσω της επαναλαμβανόμενης εφαρμογής των συγκεκριμένων εννοιών με διαφορετικούς τρόπους.

Τα προκαταρκτικά συμπεράσματα έδειξαν υψηλό κίνητρο για την εκμάθηση της Υπολογιστικής Σκέψης, ο σχηματισμός σε ομάδες αποδείχθηκε εξαιρετικά βοηθητικός στην κατανόηση του αντικειμένου, οι βαθύτερες προεκτάσεις της υπολογιστικής στους διάφορους επιστημονικούς τομείς αξιολογήθηκαν θετικά και τέλος οι συμμετέχοντες, φαίνεται ότι προτίμησαν τη χρήση της Blocky και της Python σε σχέση με τη NetLogo.

Στην αναθεωρημένη εκδοχή του μαθήματος (Kafura, Bart, & Chowdhury, 2018) διατηρήθηκε η ίδια διδακτική προσέγγιση, περιλάμβανε τις επαναλαμβανόμενες έννοιες, τη συνεργατική μάθηση, την πρόσβαση και χρήση μεγάλων δεδομένων και τη μετάβαση από δομημένα μπλοκ προγραμματισμού, στον προγραμματισμό με σύνταξη της Python σε μορφή κειμένου. Ιδιαίτερη έμφαση δόθηκε στην εξάσκηση των χαρακτηριστικών της Υπολογιστικής Σκέψης, αφαίρεση σε πολλαπλά επίπεδα και αλγόριθμοι. Το μάθημα περιλάμβανε τις ακόλουθες επτά ενότητες:

Στην πρώτη ενότητα οι εκπαιδευόμενοι παρέδωσαν την πρώτη τους ολοκληρωμένη εργασία σχετικά με τη χρήση μεγάλων δεδομένων και εμβάθυναν στην εφαρμογή της αφαίρεσης μέσω της οπτικοποίησης των δεδομένων με τη χρήση διαγραμμάτων.

Η δεύτερη ενότητα αφορούσε στην εκμάθηση τόσο των βασικών αρχών του προγραμματισμού όσο και απλών αλγορίθμων με τη βοήθεια της οπτικής γλώσσας προγραμματισμού BlockPy.

Στην τρίτη ενότητα με τη βοήθεια οπτικοποίησης και διαγραμμάτων, εμβάθυναν στους αλγόριθμους, τα μεγάλα δεδομένα, στην αξιοποίηση σύνθετων δεδομένων και παρέδωσαν τη δεύτερη εργασία τους.

Η τέταρτη ενότητα εστίαζε στην αναγνώριση πολλαπλών επίπεδων αφαίρεσης που προέκυπταν από την επεξεργασία δεδομένων, με τη δημιουργία αλγορίθμων στη γλώσσα προγραμματισμού Python και την οπτικοποίηση του αποτελέσματος.

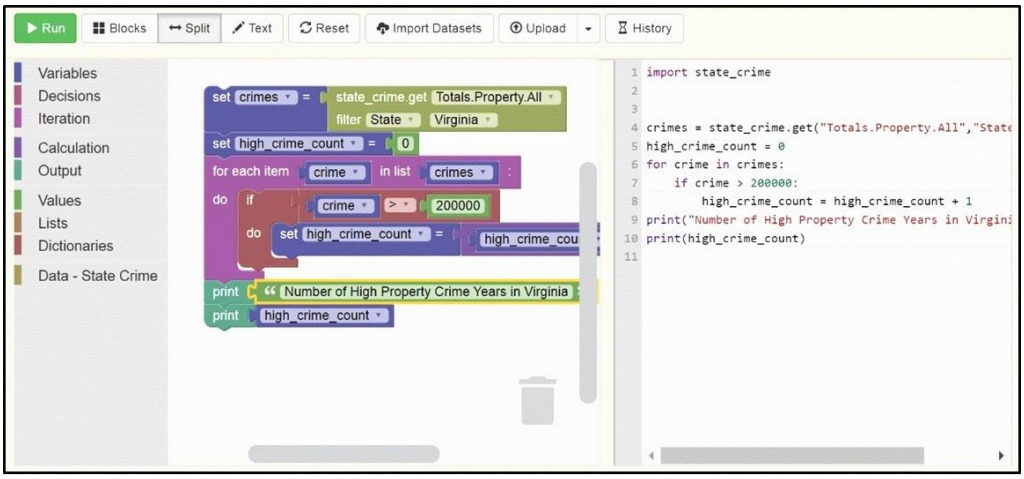
Στην πέμπτη ενότητα οι εκπαιδευόμενοι παρέδωσαν την τρίτη εργασία, η οποία ήταν ομαδική, και αφορούσε προχωρημένες δομές της Python σε συνδυασμό με τα μεγάλα δεδομένα και την οπτικοποίηση του αποτελέσματος.

Η έκτη ενότητα περιλάμβανε την παράδοση της τέταρτης και τελευταίας ατομικής εργασίας. Τα στοιχεία της Python που χρησιμοποιήθηκαν ήταν παρόμοια με της προηγούμενης άσκησης, ενώ σκοπός της εργασίας ήταν η οπτικοποίηση των επιλεγμένων δεδομένων.

Στην έβδομη ενότητα αναλύθηκε ο κώδικας που είχε χρησιμοποιηθεί στην τελευταία εργασία και ακολούθησε μια συνολική επισκόπηση του μαθήματος.

Η διαφοροποίηση των δύο εκδοχών του μαθήματος βρισκόταν στο περιεχόμενό τους και στο προγραμματιστικό περιβάλλον που χρησιμοποιήθηκε για την μετάβαση από τις οπτικές γλώσσες προγραμματισμού σε κώδικα κειμένου.

Το μάθημα δεν περιλάμβανε τελικές εξετάσεις, η αξιολόγηση των συμμετεχόντων πραγματοποιήθηκε με τη χρήση περιγραφικών κλιμάκων διαβαθμισμένων κριτηρίων ποιότητας (rubrics) και τη βαθμολογία τους από τις τέσσερις ολοκληρωμένες εργασίες.



Εικόνα 6: Παράδειγμα μετάβασης από μπλοκ προγραμματισμού σε Python (Kafura, Bart, & Chowdhury, 2018)

3.4 Η Περίπτωση του Πανεπιστημίου Montana Tech, Ηνωμένες Πολιτείες Αμερικής

Η ακόλουθη μελέτη περίπτωσης (Van Dyne & Braun, 2014) αφορά στο μάθημα με τίτλο «Υπολογιστική Σκέψη», το οποίο από το 2010 διδασκόταν στο Πανεπιστήμιο Montanta Tech. Το μάθημα αρχικά προοριζόταν ως προπαρασκευαστικό για τους

εκπαιδευόμενους της επιστήμης των υπολογιστών και τους μηχανικούς λογισμικού, με στόχο την ανάπτυξη αναλυτικού και κριτικού τρόπου σκέψης καθώς και την εξοικείωση με τη διαδικασία επίλυσης προβλημάτων. Η απόκτηση των συγκεκριμένων δεξιοτήτων κρίθηκε απαραίτητη προκειμένου να αποκτηθεί κοινό μαθησιακό υπόβαθρο και να γίνει πιο ομαλή η μετάβασή στα επόμενα μαθήματα του εξαμήνου. Αφού διαπιστώθηκε ότι επιτυγχάνονταν οι εκπαιδευτικοί στόχοι του μαθήματος, το 2013 συμπεριλήφθηκε στο πρόγραμμα σπουδών σαν μάθημα γενικής κατεύθυνσης.

Το μάθημα διεξαγόταν σε εβδομαδιαία βάση με τη μορφή διαλέξεων και εργαστηρίων διάρκειας δύο και τριών ωρών αντίστοιχα, ενώ περιλάμβανε και την παράδοση εβδομαδιαίων εργασιών, για καλύτερη κατανόηση των διδασκόμενων εννοιών.

Η διδακτέα ύλη του μαθήματος περιλάμβανε τις έννοιες του παραλληλισμού, της ανάλυσης τάσεων και προτύπων, της συλλογιστικής, των μαθηματικών λεκτικών προβλημάτων και την κριτική αξιολόγηση των πληροφοριών. Περιλάμβανε επίσης και έννοιες της υπολογιστικής, όπως είναι η δημιουργία αλγορίθμων, οι τύποι και οι δομές δεδομένων, η αφαίρεση, η αποσύνθεση, ο μετασχηματισμός, η προσομοίωση, η επανάληψη και η αναδρομή.

Η διδασκαλία του μαθήματος γινόταν με σύντομη διατύπωση οδηγιών και συνεργατική μάθηση, στην επίλυση ασκήσεων σχετικών με τις έννοιες της Υπολογιστικής Σκέψης. Τα εργαστήρια του μαθήματος αποσκοπούσαν στην καλύτερη κατανόηση του θεωρητικού μέρους, με τη χρήση πακέτου ρομποτικής Lego Mindstorms NXT. Οι εκπαιδευτικοί στόχοι των εργαστηρίων ήταν, κατανόηση εννοιών όπως αλγόριθμοι, μεταβλητές, δεδομένα, συναρτήσεις, δομές επανάληψης και ταξινόμηση. Η διδασκαλία των εννοιών αυτών γινόταν, σε πρώτη φάση με τη δημιουργία και τον προγραμματισμό ρομπότ μέσω προγραμματιστικού περιβάλλοντος με γραφικά και σε δεύτερη φάση με τη χρήση γλώσσας προγραμματισμού Java και τη λήψη βοήθειας από συγκεκριμένη βιβλιοθήκη (LeJos). Ο λόγος που επιλέχθηκε η εκπαιδευτική ρομποτική ως διδακτική προσέγγιση είναι η δυνατότητα άμεσου εντοπισμού σφαλμάτων από τους εκπαιδευόμενους.

Με τη χρήση του εργαλείου WASI (Whimbey Analytical Skill Inventory) πραγματοποιούνταν η αξιολόγηση της επίδοσης και των δεξιοτήτων κατανόησης και επίλυσης προβλημάτων των εκπαιδευόμενων. Ένα πρώτο τεστ που περιλάμβανε 38 ερωτήσεις, αποτελούσε το σημείο αναφοράς και ένα δεύτερο παρόμοιου περιεχομένου 37 ερωτήσεων, καθόριζε το βαθμό βελτίωσης σε σχέση με την

αρχική βαθμολογία. Πιο συγκεκριμένα, το τεστ αξιολογούσε δεξιότητες όπως η συλλογιστική, ο αναλυτικός τρόπος σκέψης και κυρίως η παρατηρητικότητα, ενώ δεν χρησιμοποιούσε ιδιαίτερες μαθηματικές γνώσεις παρά μόνο βασικές αρχές της άλγεβρας.

Τα αποτελέσματα από την ανάλυση της βαθμολογίας ήταν πολύ ενθαρρυντικά. Οι εκπαιδευόμενοι φαίνεται ότι επωφελήθηκαν από το πρόγραμμα εκπαίδευσης στην Υπολογιστική Σκέψη καθώς βελτιώθηκε η αναλυτική τους δεξιότητα επίλυσης προβλημάτων.

3.5 Η περίπτωση του Πανεπιστημίου Purdue Ηνωμένες Πολιτείες Αμερικής

Η παρακάτω περίπτωση (Vieira, Yan, & Magana, 2015) αφορά σε ένα εισαγωγικό μάθημα πληροφορικής του Πανεπιστημίου Purdue των Ηνωμένων Πολιτειών της Αμερικής, στα πλαίσια του προγράμματος σπουδών του Τμήματος πληροφορικής και τεχνολογίας των πληροφοριών.

Σκοπός του συγκεκριμένου μαθήματος ήταν η εκμάθηση προγραμματισμού και σχεδίασης αλγορίθμων με τη χρήση γλώσσας προγραμματισμού C#. Η εβδομαδιαία διδασκαλία αποτελούνταν από διαλέξεις και εργαστήρια. Στόχευαν στην εξοικείωση με τις τεχνικές της Υπολογιστικής Σκέψης όπως είναι, η αφαίρεση, η αποσύνθεση, η αναγνώριση προτύπων και η γενίκευση. Σε τρία από τα εργαστηριακά μαθήματα οι εφαρμογές γινόταν με τη χρήση λυμένων παραδειγμάτων (Worked Examples) βαθμιαίας δυσκολίας, σχεδιασμένα από έμπειρους προγραμματιστές, με αντικείμενο τις δομές επανάληψης και τις δυναμικές λίστες (ArrayLists). Στα συγκεκριμένα εργαστηριακά μαθήματα οι εκπαιδευόμενοι είχαν ως πρότυπο τα λυμένα παραδείγματα και καλούνταν να επιλύσουν παρόμοια προβλήματα.

Κάθε εργαστηριακό μάθημα, περιλάμβανε δύο λυμένα παραδείγματα, τα οποία αποτελούνταν από διαφορετικές απεικονίσεις. Η πρώτη, αφορούσε στη λύση με μορφή κώδικα κειμένου, ο οποίος συμπεριλάμβανε διαχωρισμό και επεξήγησή του με σχόλια. Η δεύτερη, οπτικοποιούσε ανάλυση του κώδικα με αντίστοιχο διάγραμμα ροής και η Τρίτη, περιέγραφε τη λεκτική αντιστοίχιση των δύο προηγούμενων τρόπων απεικόνισης (βλ. εικόνα 7).

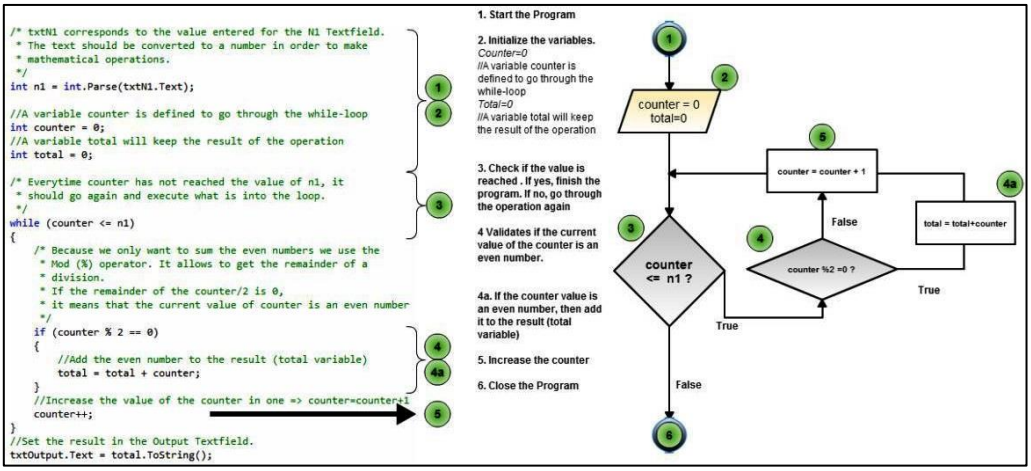
Κατά τη διάρκεια της πειραματικής περιόδου διδασκαλίας, ακολουθήθηκε διαδικασία τεσσάρων σταδίων. Στο πρώτο στάδιο οι εκπαιδευόμενοι συμπλήρωσαν ένα αρχικό τεστ, το οποίο αποτέλεσε το σημείο αναφοράς. Το δεύτερο στάδιο περιλάμβανε δύο λυμένα παραδείγματα με μικρές διαφοροποιήσεις ως προς τα σχόλια. Το πρώτο λυμένο παράδειγμα συμπεριλάμβανε σχολιασμό, ο οποίος αφαιρούνταν στο δεύτερο και το ζητούμενο ήταν να συμπληρωθεί κατ' αναλογία. Με τον τρόπο αυτό είχαν τη δυνατότητα να παρατηρήσουν τις τεχνικές επίλυσης των λυμένων παραδειγμάτων, προκειμένου να μπορούν να τις εφαρμόσουν στις τρεις επόμενες προγραμματιστικές εργασίες, που καλούνταν να επιλύσουν κατά το τρίτο στάδιο. Η διαδικασία ολοκληρωνόταν με το τέταρτο στάδιο και τη συμπλήρωση του τελικού τεστ, με απάντηση ερωτήσεων κατανόησης του περιεχομένου του μαθήματος καθώς και ερωτήσεις ανοιχτού τύπου, αναφορικά με την αποτελεσματικότητα των λυμένων παραδειγμάτων. Μέσα από την ολοκλήρωση της εβδομαδιαίας αυτής διαδικασίας προέκυπτε η ανατροφοδότηση και ο σχεδιασμός των επόμενων λυμένων παραδειγμάτων.

Για την αξιολόγηση του τρόπου διδασκαλίας του μαθήματος συμμετείχαν 35 εκπαιδευόμενοι που χωρίστηκαν σε δύο ομάδες, την πειραματική και την ομάδα ελέγχου. Και στις δύο ομάδες συμμετείχαν άτομα με και χωρίς προηγούμενη προγραμματιστική εμπειρία. Το εκπαιδευτικό υλικό (λυμένα παραδείγματα) ήταν κοινό κατά την πρώτη εβδομάδα του μαθήματος, ενώ τη δεύτερη και τρίτη, η ομάδα ελέγχου παρακολούθησε τον παραδοσιακό τρόπο διδασκαλίας, ενώ η πειραματική τη διδακτική προσέγγιση των λυμένων παραδειγμάτων. Οι δύο ομάδες συγκρίθηκαν με βάση τα συλλεχθέντα δεδομένα αναφορικά με το αρχικό τεστ, το τελικό τεστ, τις ερωτήσεις κατανόησης, το σχολιασμό του κώδικα και τις προγραμματιστικές εργασίες.

Από την ανάλυση των δεδομένων της πρώτης εβδομάδας δεν βρέθηκε στατιστικώς σημαντική διαφορά μεταξύ των δύο ομάδων. Παρατηρήθηκε όμως ότι όσοι διέθεταν προγραμματιστική εμπειρία, διέφεραν ως προς την επίδοσή τους στις ερωτήσεις κατανόησης του μαθήματος. Από την ανάλυση των δεδομένων της δεύτερης εβδομάδας, δεν προέκυψαν στατιστικώς σημαντικές διαφορές μεταξύ των δύο ομάδων, ούτε στις απαντήσεις ερωτήσεων κατανόησης του μαθήματος. Κατά την τρίτη εβδομάδα του μαθήματος, πραγματοποιήθηκε αλλαγή ρόλων στον ερευνητικό σχεδιασμό, με την πειραματική ομάδα να παρακολουθεί το πρόγραμμα της ομάδας ελέγχου και το αντίστροφο. Από τα δεδομένα που προέκυψαν, βρέθηκε στατιστικώς σημαντική διαφορά ως προς την επίδοση, στο αρχικό και τελικό τεστ, με την

πειραματική ομάδα να υπερέχει βαθμολογικά. Το ίδιο αποτέλεσμα παρατηρήθηκε και στους συμμετέχοντες που δεν διέθεταν προηγούμενη προγραμματιστική εμπειρία.

Στους περιορισμούς της συγκεκριμένης έρευνας αναφέρονται το σύντομο χρονικό διάστημα της πειραματικής περιόδου, το περιορισμένο δείγμα των εκπαιδευόμενων και ο περιορισμένος χρόνος εξοικείωσης, συγκριτικά με την παραδοσιακή μέθοδο διδασκαλίας.



Εικόνα 7: Παράδειγμα λυμένου παραδείγματος (Vieira, Yan, & Magana, 2015)

3.6 Η Περίπτωση του LOOP (Learning Object-Oriented Programming)

Η ακόλουθη περίπτωση (Krugel & Hubwieser, 2017) δεν αφορά διδασκαλία προπτυχιακού μαθήματος πανεπιστημίου παρά περιγράφει το σχεδιασμό και τη δημιουργία Μαζικού Ανοικτού Διαδικτυακού Μαθήματος (MOOC).

Το μάθημα δημιουργήθηκε από το Τεχνικό Πανεπιστήμιο του Μονάχου, λειτούργησε δοκιμαστικά το 2016, με σκοπό την εξοικείωση ενδιαφερόμενων, σχετικών με την επιστήμη της πληροφορικής ή άλλων επιστημονικών τομέων, στον αντικειμενοστρεφή προγραμματισμό για τη δημιουργία ενιαίου υποβάθρου.

Σκοπός του μαθήματος ήταν η ομαλή εισαγωγή των συμμετεχόντων στον προγραμματισμό με τη χρήση της γλώσσας Java μέσω εξοικείωσης με τις έννοιες της Υπολογιστικής Σκέψης και της αντικειμενοστρέφειας. Η δημιουργία του διδακτικού υλικού και η διδασκαλία γινόταν με τη χρήση της πλατφόρμας edX Edge. Η διάρκειά του ήταν πέντε εβδομάδες και περιλάμβανε τις ενότητες, αντικειμενοστρεφή μοντελοποίηση, αλγόριθμοι, αντικειμενοστρεφή προγραμματισμό, εφαρμογή αλγορίθμων και δυναμικών λιστών όπως και συσχετίσεις και αναφορές.

Η διδασκαλία του μαθήματος γινόταν με τη χρήση οπτικοακουστικού υλικού και αποτελούνταν από 24 βίντεο σύντομης διάρκειας. Ο διδάσκων περιέγραφε το μαθησιακό περιεχόμενο της αντίστοιχης θεματικής ενότητας με τη χρήση διαφανειών και επεσήμαινε τα σημαντικά σημεία. Οι ποικίλες αναπαραστάσεις του περιεχομένου ανταποκρίνονταν στο διαφορετικό τρόπο αντίληψης των συμμετεχόντων. Προκειμένου να ελεγχθεί ο βαθμός κατανόησης των εννοιών που παρουσιάζονταν στα βίντεο, οι εκπαιδευόμενοι συμπλήρωναν τεστ (διαμορφωτική αξιολόγηση), τα οποία εντόπιζαν τα ασαφή σημεία του μαθήματος που έπρεπε να επαναληφθούν.

Το μάθημα περιλάμβανε, διαδραστικές και προγραμματιστικές εργασίες για καλύτερη κατανόηση των εννοιών του αντικειμενοστρεφούς προγραμματισμού και της Υπολογιστικής Σκέψης, με τη βοήθεια διαδικτυακών εργαλείων (SVG-edit, trinket, SVG.JS, CindyJs, Blockly-Games,UmpleOnline, Java-Tutor, codeboard, java reflection, jack).

Οι διαδραστικές ασκήσεις βοηθούσαν στην κατανόηση των βασικών εννοιών του αντικειμενοστρεφούς προγραμματισμού, όπως είναι οι κλάσεις και τα αντικείμενα, για

τη δημιουργία γραφικών, μέσω σύνθεσης και σχεδίασης γεωμετρικών σχημάτων, δηλαδή κύκλων, παραλληλόγραμμων και γραμμών.

Ζητούμενο της πρώτης διαδραστικής άσκησης ήταν να δημιουργηθεί, με συνδυασμό και προσαρμογή των κατάλληλων διαδικτυακών εργαλείων, εικόνα με γραφικά αντικείμενα και κλήσεις μεθόδων, με τη χρήση ψευδοκώδικα και βασικών εντολών. Με τη βοήθεια της Ενοποιημένης Γλώσσας Σχεδίασης Προτύπων (UML) και συγκεκριμένα την εμφάνιση του διαγράμματος αντικειμένων, η άσκηση αυτή αποσκοπούσε στην εκμάθηση εννοιών, σχετικών με την κατάσταση των αντικειμένων, και τις εναλλαγές των καταστάσεων που πραγματοποιούνταν με την κλήση μεθόδων. Οι συμμετέχοντες χρησιμοποιώντας ένα αναπαραστατικό διαδικτυακό εργαλείο εξοικειώνονταν με τους αλγόριθμους, μέσω της οπτικοποίησης των βημάτων και της αναγνώρισης του αποτελέσματος διαφορετικών δεδομένων εισόδου.

Στην επόμενη διαδραστική άσκηση και μέσω της παιχνιδοποίησης, οι εκπαιδευόμενοι κατανοούσαν καλύτερα τη δομή των αλγορίθμων. Η έμφαση δινόταν στις δομές απόφασης και επανάληψης, με τη χρήση ενός διαδικτυακού εργαλείου και γλώσσας προγραμματισμού με τη μορφή μπλοκ.

Για την σταδιακή εξοικείωση των εκπαιδευόμενων με τη σύνταξη της γλώσσας προγραμματισμού Java, χρησιμοποιήθηκε ένα ακόμη διαδικτυακό εργαλείο, το οποίο επέτρεπε αλλαγή και παρακολούθηση αλληλεπιδράσεων, μεταξύ του διαγράμματος κλάσεων της UML και της σύνταξης της γλώσσας.

Χρησιμοποιήθηκε επίσης μια διαδικτυακή εφαρμογή για την κατανόηση της σειριακής εκτέλεσης του κώδικα ενός προγράμματος, με ταυτόχρονη οπτικοποίηση αποτελεσμάτων και αλλαγών των μεταβλητών στις θέσεις μνήμης.

Η εφαρμογή αυτοματοποιημένης ανίχνευσης λαθών και βαθμολόγησης, επεκτεινόταν στις προγραμματιστικές εργασίες, με τη χρήση διαδικτυακών εργαλείων και αποτέλεσμα την πληρέστερη ανατροφοδότηση των εκπαιδευόμενων.

Αναφορικά με την απήχηση που είχε το πρόγραμμα, αρχικά υπήρξε μεγάλος αριθμός εγγραφών, ενώ το ολοκλήρωσαν 40 συμμετέχοντες. Τα προκαταρκτικά αποτελέσματα της ανατροφοδότησης αφορούσαν κυρίως στη βελτίωση και στον επανασχεδιασμό του περιεχομένου του μαθήματος, το οποίο είχε θετική ανταπόκριση. Οι εκπαιδευτικοί στόχοι φάνηκε να επιτυγχάνονται, για την επιβεβαίωση, κρίθηκε απαραίτητη περαιτέρω διερεύνηση.

Πίνακας 3: Πλεονεκτήματα και Μειονεκτήματα περιπτώσεων

Περίπτωση	Τρόπος Ενσωμάτωσης	Πλεονεκτήματα	Μειονεκτήματα
περίπτωση κολεγίου Colby	Python + βιβλιοθήκες γραφικών	• Εκμάθηση υπολογιστικών εννοιών με έμμεσο τρόπο (πχ πολυμέσα) • Άμεσα ορατό διαδραστικό αποτέλεσμα (πχ pixel σε εικόνα)	• Άμεση εξάρτηση της εκμάθησης από το ενδιαφέρον για τη θεματική ενότητα • Ο χρόνος που χρειάζεται να αφιερωθεί στις προγραμματιστικές ασκήσεις
περίπτωση πανεπιστημίου Guarda	Δραστηριότητες + χρήση ψευδογλώσσας	• Η μη χρησιμοποίηση υπολογιστικών συσκευών • Η μη επιβάρυνση με εκμάθηση σύνταξης γλώσσας προγραμματισμού	• Μπορεί να χρησιμοποιηθεί μόνο σε προπαρασκευαστικά μαθήματα • Άμεση εξάρτηση από τη σαφήνεια και το συντονισμό οδηγιών σε κάποιες ασκήσεις
περίπτωση πανεπιστημίου Virginia Tech	NetLogo + Blocky + BlockPy (αναθεωρημένη έκδοση) + Python	• Η σταδιακή μετάβαση σε γλώσσα προγραμματισμού • Η Δημιουργία και χρήση ομάδων (συνεργατική μάθηση) • Η χρήση Big Data για προσθήκη ρεαλισμού	• Η μη εφαρμογή της σε βραχυπρόθεσμες περιπτώσεις • Στις ομάδες όλα τα μέλη μπορεί να μην συνεισφέρουν εξίσου
περίπτωση πανεπιστημίου Montana Tech	Lego Mindstorms NXT + Java + Βιβλιοθήκη LeJos	• Αύξηση ενδιαφέροντος λόγω της διαδραστικότητας με την ρομποτική • Άμεση κατανόηση σφαλμάτων	• Άμεση εξάρτηση του μαθήματος από τα εργαλεία (ρομπότ) • Το υψηλό κόστος για τα ρομπότ
περίπτωση Πανεπιστημίου Purdue	C#	• Πολλαπλές απεικονίσεις επίλυσης με αντιστοίχιση	• Απαιτείται η άμεση ενασχόληση και κατανόηση των

		• Η αναλογική επίλυση προβλημάτων	λυμένων παραδειγμάτων για επίτευξη μαθησιακών αποτελεσμάτων • Ο διαρκής επανασχεδιασμός των λυμένων παραδειγμάτων σύμφωνα με τις ανάγκες του μαθήματος
περίπτωση LOOP	Χρήση διαδικτυακών εργαλείων + Java	• Δεν απαιτείται η εγκατάσταση λογισμικού • Η παρακολούθηση των μαθημάτων μπορεί να γίνει οποιαδήποτε ώρα και σε οποιοδήποτε μέρος	• Χαμηλό ποσοστό συμμετεχόντων που ολοκληρώνουν το μάθημα • Δεν υπάρχει άμεση αλληλεπίδραση μεταξύ εκπαιδευτή - εκπαιδευόμενου

3.7 Υπολογιστική Σκέψη στο Ελληνικό Πρόγραμμα Σπουδών

Σύμφωνα με τα όσα περιγράφηκαν παραπάνω, το αυξανόμενο ενδιαφέρον των ερευνητών για την Υπολογιστική Σκέψη και τους πιθανούς τρόπους ανάπτυξής των δεξιοτήτων της, έχει οδηγήσει σε σημαντικές πρωτοβουλίες για την αξιοποίησή της στην εκπαίδευση και σε ολοκληρωμένα προγράμματα σπουδών. Προκειμένου όμως να είναι συστηματική και βιώσιμη η ένταξη της Υπολογιστικής Σκέψης στην τυπική εκπαίδευση, είναι σημαντικό η διαδικασία αυτή να ξεκινήσει από την πρωτοβάθμια και να συνεχιστεί μέχρι και την τριτοβάθμια εκπαίδευση.

Σύμφωνα με το ελληνικό πρόγραμμα σπουδών για το μάθημα της Πληροφορικής στο Γυμνάσιο (ebooks.edu.gr, 2013), ο όρος Υπολογιστική Σκέψη δεν αναφέρεται ρητά στο σχεδιασμό του, αλλά γίνεται εκτενής αναφορά στην έννοια του πληροφορικού γραμματισμού. Το πρόγραμμα καλύπτει μεγάλο μέρος των εννοιών της Υπολογιστικής Σκέψης, ενώ δίνεται μεγαλύτερο βάρος στις Τεχνολογίες της Πληροφορικής και των Επικοινωνιών σε σχέση με τις έννοιες της επιστήμης της Πληροφορικής (Μαυρουδή, Πέτρου, & Φεσάκης, 2014).

Στο Πρόγραμμα σπουδών του μαθήματος «Πληροφορική» της Γ' τάξης του Λυκείου, η πρώτη θεματική ενότητα είναι η «Εισαγωγή στην Υπολογιστική σκέψη» και περιλαμβάνει τα χαρακτηριστικά της και την εφαρμογή της στην επίλυση προβλημάτων (Νόμος 4310, ΦΕΚ 258/8−12−2014). Πιο συγκεκριμένα, οι μαθητές πρέπει να μάθουν να περιγράφουν τις έννοιες που αποτελούν την Υπολογιστική Σκέψη, να εξηγούν τη σημασία της στην επίλυση προβλημάτων της καθημερινής ζωής και να αναλύουν ένα υπολογιστικό πρόβλημα, περιγράφοντας τις βασικές διαδικασίες με τις οποίες αντιμετωπίζεται, όπως είναι η διάσπαση, η αναγνώριση προτύπων, η γενίκευση και η σχεδίαση αλγορίθμου.

Τέλος, στην τριτοβάθμια εκπαίδευση, η έννοια της Υπολογιστικής Σκέψης έχει ενταχθεί στο αναλυτικό πρόγραμμα μεταπτυχιακών σπουδών, όπως για παράδειγμα στο Δ.Π.Μ.Σ. «Δίκαιο και Πληροφορική» στο μάθημα "Υπολογιστική Σκέψη και Λογισμικό". Στους εκπαιδευτικούς στόχους του μαθήματος μεταξύ άλλων περιλαμβάνονται η εισαγωγή στην έννοια της Υπολογιστικής Σκέψης, η ανάλυση προβλήματος, η εισαγωγή στον αλγοριθμικό τρόπο σκέψης, η εκμάθηση βασικών αλγοριθμικών δομών, οι δομές δεδομένων, οι αλγόριθμοι αναζήτησης και οι αλγόριθμοι ταξινόμησης (Απόφαση 72, ΦΕΚ 4273/27−09−2018).

Παρόλο που οι προσπάθειες ένταξης της διδασκαλίας των εννοιών της Υπολογιστικής Σκέψης βρίσκονται ακόμα σε πρώιμο στάδιο, εντούτοις είναι εμφανής η προσπάθεια σταδιακής εξοικείωσης των μαθητών και φοιτητών με τα χαρακτηριστικά της καθώς και με τις πρακτικές εφαρμογές της. Προκειμένου όμως να υπάρξει το βέλτιστο μαθησιακό αποτέλεσμα και η γενίκευση των δεξιοτήτων αυτών και σε άλλους επιστημονικούς τομείς, κρίνεται απαραίτητη η ύπαρξη ενός οργανωμένου και συστηματικού σχεδιασμού για την ένταξη της διδασκαλίας της σε όλες τις βαθμίδες εκπαίδευσης.

Ερευνητικές Υποθέσεις

Τα ερευνητικά ερωτήματα που διατυπώθηκαν κατά τον αρχικό σχεδιασμό της έρευνας και επιχειρήθηκε να απαντηθούν ήταν τα εξής:

1. Με ποιο τρόπο μπορούν να ενισχυθούν οι δεξιότητες Υπολογιστικής Σκέψης των φοιτητών μέσα από την εκτέλεση ασκήσεων με τη χρήση της γλώσσας προγραμματισμού Python;

2. Ποιος είναι ο πιο κατάλληλος τρόπος να αξιολογηθούν οι δεξιότητες Υπολογιστικής Σκέψης των εκπαιδευόμενων;

3. Παρατηρήθηκε διαφορά στις δεξιότητες Υπολογιστικής Σκέψης των εκπαιδευόμενων που παρέδωσαν τις προαιρετικές εργασίες, σε σύγκριση με εκείνους που δεν τις συμπλήρωσαν, όπως αυτές αξιολογήθηκαν από την τελική γραπτή εξέταση του μαθήματος;

4 Μεθοδολογία

Σκοπός της παρούσας έρευνας ήταν η δημιουργία εκπαιδευτικού υλικού για την ενίσχυση της Υπολογιστικής Σκέψης των εκπαιδευόμενων, στα πλαίσια του μαθήματος «Εισαγωγή στην Πληροφορική». Το μάθημα είναι υποχρεωτικό για τους φοιτητές του Τμήματος Πληροφορικής του Αριστοτελείου Πανεπιστημίου Θεσσαλονίκης και διδάσκεται στο πρώτο εξάμηνο.

Βασικός στόχος του μαθήματος είναι να αποκτήσουν όλοι οι πρωτοετείς φοιτητές κοινό υπόβαθρο στη χρήση υπολογιστικών συστημάτων και στη διαχείριση της πληροφορίας με την εφαρμογή μεθόδων και συστημάτων της Επιστήμης των Υπολογιστών. Μετά την ολοκλήρωση του μαθήματος οι φοιτητές θα έχουν μάθει να αναζητούν, να αναλύουν και να συνθέτουν δεδομένα και πληροφορίες, κάνοντας χρήση των απαραίτητων τεχνολογιών.

Το μάθημα διεξάγεται σε 13 τρίωρες εβδομαδιαίες διαλέξεις, εκ των οποίων οι πέντε αφιερώνονται στη διδασκαλία της γλώσσας προγραμματισμού Python. Το εκπαιδευτικό υλικό που χρησιμοποιείται για την επίτευξη των εκπαιδευτικών στόχων είναι διαφάνειες παρουσίασης, διαδραστικές ασκήσεις και μελέτη βιβλιογραφίας, ενώ χρησιμοποιούνται και τα ηλεκτρονικά παιχνίδια για την κινητροδότηση των φοιτητών. Κατά τη διάρκεια του μαθήματος, η αξιολόγηση των φοιτητών πραγματοποιείται με τη συμμετοχή τους σε γραπτές εξετάσεις με ερωτήσεις σύντομης απάντησης και με επίλυση προβλημάτων, ενώ η τελική αξιολόγηση της επίδοσης τους πραγματοποιείται με γραπτές εξετάσεις σε ερωτήσεις πολλαπλής επιλογής.

Σύμφωνα με τη βιβλιογραφία, η πιο συνηθισμένη μέθοδος εκμάθησης δεξιοτήτων Υπολογιστικής Σκέψης είναι μέσω υπολογιστικών ασκήσεων, που βασίζονται ως επί το

πλείστον σε διαφορετικά είδη προγραμματιστικών εργασιών (π.χ. Python) (Brackmann, Román-González, Robles, Moreno-León, Casali, & Barone, 2017). Επίσης, αναφέρεται συχνά ότι οι ασκήσεις σχεδιασμού αλγορίθμων και προγραμματισμού αποτελούν αναπόσπαστο κομμάτι της κατανόησης και απόκτησης δεξιοτήτων Υπολογιστικής Σκέψης (Vieira, Yan, & Magana, 2015).

Η συλλογιστική αυτή αποτέλεσε και τη βάση για το σχεδιασμό της παρούσας έρευνας. Για την επίτευξη των σκοπών της έρευνας αρχικά σχεδιάστηκε το εκπαιδευτικό υλικό για την ενίσχυση της Υπολογιστικής Σκέψης. Οι ασκήσεις αυτές ανατέθηκαν σταδιακά στους εκπαιδευόμενους κατά τη διάρκεια του εξαμήνου. Μετά την ολοκλήρωση του μαθήματος, οι φοιτητές συμμετείχαν στις τελικές γραπτές εξετάσεις, που περιλάμβαναν και ερωτήσεις πολλαπλής επιλογής που αξιολογούσαν τις θεωρητικές και πρακτικές τους γνώσεις για τις έννοιες της Υπολογιστικής Σκέψης. Στόχος της έρευνας ήταν να διερευνηθεί αν οι ασκήσεις αυτές συνέβαλαν στην ενίσχυση της Υπολογιστικής Σκέψης των φοιτητών, όπως αυτή αποτυπώθηκε από τη βαθμολογία που έλαβαν στην τελική γραπτή εξέταση και στη συνέχεια αν υπήρξε κάποια συσχέτιση, ανάμεσα στην επίδοση των φοιτητών στις δυο αυτές δοκιμασίες. Τέλος, χορηγήθηκε στους εκπαιδευόμενους και το Τεστ Υπολογιστικής Σκέψης, το οποίο αποτελεί μετάφραση και προσαρμογή του Computational Thinking Test (CTt), ως ένα δεύτερο μέσο συλλογής δεδομένων αναφορικά με την κατανόηση των εννοιών της Υπολογιστικής Σκέψης, προκειμένου να διαπιστωθεί αν υπήρξε συνάφεια με τη βαθμολογία που έλαβαν οι φοιτητές στην τελική γραπτή εξέταση. Η παρούσα πιλοτική χορήγηση του τεστ είχε ως σκοπό τη μελλοντική αξιοποίησή του για την αξιολόγηση της Υπολογιστικής Σκέψης των φοιτητών σε δυο χρονικές στιγμές, πριν και μετά τη διδασκαλία του μαθήματος.

Στη συνέχεια γίνεται αναλυτική περιγραφή του τρόπου σχεδιασμού των εργασιών που δόθηκαν κατά τη διάρκεια του εξαμήνου στους φοιτητές για την ενίσχυση της Υπολογιστικής Σκέψης καθώς και η μετάφραση και προσαρμογή του Τεστ Υπολογιστικής Σκέψης που χορηγήθηκε μετά την ολοκλήρωση του μαθήματος.

4.1 Εκπαιδευτικό υλικό ενίσχυσης της Υπολογιστικής Σκέψης (προαιρετικές ασκήσεις)

Το θεωρητικό υπόβαθρο στο οποίο βασίστηκε ο σχεδιασμός των προαιρετικών ασκήσεων ήταν κοινό με τις παρακάτω θεματικές ενότητες που περιλαμβάνονταν στις πέντε διαλέξεις του μαθήματος και αφιερώθηκαν στη διδασκαλία της γλώσσας

προγραμματισμού Python: Δομές ελέγχου ροής - δομή απόφασης και επαναληπτική δομή while, Επαναληπτική δομή for, δομές δεδομένων - λίστες, δομές δεδομένων - λεξικά και Συναρτήσεις. Παράλληλα εισάγονταν από τον διδάσκοντα η έννοια της Υπολογιστικής Σκέψης και επισημαίνονταν τα χαρακτηριστικά της, όταν ανέκυπταν, στο πλαίσιο παραδειγμάτων εφαρμογής.

Αρχικά μελετήθηκε το εκπαιδευτικό υλικό του μαθήματος και στη συνέχεια δημιουργήθηκαν δέκα υποθετικά σενάρια (ασκήσεις) κλιμακούμενης δυσκολίας, τα οποία αποσκοπούσαν τόσο στην εξοικείωση των φοιτητών με τον προγραμματισμό όσο και στην ενίσχυση των δεξιοτήτων Υπολογιστικής Σκέψης. Στις προαναφερθείσες θεματικές ενότητες αντιστοιχούσαν δυο ασκήσεις, οι οποίες δίνονταν στους εκπαιδευόμενους. Η συμμετοχή των φοιτητών στις εργασίες ήταν προαιρετική, αλλά η παράδοση των εργασιών ισοδυναμούσε και με την αντίστοιχη επιδότηση βαθμού.

Στη συνέχεια θα γίνει ξεχωριστή αναφορά σε κάθε θεματική ενότητα και στη συλλογιστική της δημιουργίας των αντίστοιχων υποθετικών σεναρίων.

4.1.1 Δομές ελέγχου ροής (δομή απόφασης και επαναληπτική δομή while)

Οι βασικότερες δομές με τις οποίες πραγματοποιείται ο έλεγχος ενός προγράμματος είναι η δομή ακολουθίας εντολών, η δομή απόφασης, και η δομή της επανάληψης. Οι δομές αυτές ελέγχουν την σειρά εκτέλεσης ενός προγράμματος και καλούνται να δρομολογήσουν την ροή του προγράμματος σύμφωνα με μία ή περισσότερες ισχύουσες συνθήκες. Η δομή if αποτελεί την απλούστερη μορφή απόφασης, όπου η ροή του προγράμματος διαχωρίζεται σε δύο αποκλίνουσες κατευθύνσεις και απαιτείται να ακολουθηθεί η μία εκ των δύο. Μπορεί επίσης να χρησιμοποιηθεί και στην πιο σύνθετη μορφή της και για πολλαπλές αποφάσεις. Η επαναληπτική δομή while αποτελεί έναν βρόγχο που συνεχίζει να εκτελείται όσο μια συνθήκη παραμένει αληθής (Μανής , 2015).

Στόχος των δυο ασκήσεων αυτής της θεματικής ενότητας ήταν να μάθουν οι φοιτητές να αποδομούν ένα πρόβλημα προς επίλυση, να εστιάζουν σε συγκεκριμένα χαρακτηριστικά του προβλήματος και να αναγνωρίζουν πρότυπα με στόχο την αλγοριθμική επίλυση. Επίσης, κλήθηκαν να εξοικειωθούν με τη γλώσσα προγραμματισμού Python και συγκεκριμένα με τη χρήση μεταβλητών, δομών απόφασης (if/elif/else) και επανάληψης (while).

Το πρώτο υποθετικό σενάριο *«Άσκηση 1: while και if»* (βλ. Παράρτημα Ι) αφορούσε σε ένα σύστημα διαχείρισης αποθήκης, το οποίο ενημέρωνε το χρήστη για τον αριθμό τεμαχίων εισαγωγής-εξαγωγής καθώς και τη συνολική διαθέσιμη ποσότητα. Το ζητούμενο ήταν να υλοποιηθεί ένα πρόγραμμα στη γλώσσα προγραμματισμού Python, όπου για την επίλυσή του απαιτούνταν η ανάλυση και η χρήση των παραπάνω δομών, εμφάνιζε μήνυμα υποδοχής, δεχόταν εισαγωγή δεδομένων με ένα συγκεκριμένο μοτίβο, πραγματοποιούσε έλεγχο εισαγόμενης τιμής ως προς το μέγεθος της καταχώρησης, εμφανίζοντας το αντίστοιχο μήνυμα και παρείχε δυνατότητα εξόδου από το πρόγραμμα.

Στο δεύτερο σενάριο *«Άσκηση 2: while και if»* (βλ. Παράρτημα Ι) περιγράφονταν η περίπτωση μιας ανεγερθείσας οικοδομής, στην οποία οι ιδιοκτήτες αποφασίζαν να διαθέσουν ορισμένα από τα διαμερίσματα προς πώληση, άλλα προς ενοικίαση και κάποια να κρατηθούν για ιδιοκατοίκηση. Δόθηκε η λίστα με τις σχετικές πληροφορίες για κάθε διαμέρισμα και ζητήθηκε να υλοποιηθεί πρόγραμμα, όπου για την επίλυσή του απαιτούνταν η ανάλυση και η χρήση των παραπάνω δομών, εμφάνιζε συγκεκριμένο μήνυμα εισόδου-εξόδου από το σύστημα, δεχόταν εισαγωγή δεδομένων με τη μορφή αριθμού διαμερίσματος, πραγματοποιούνταν έλεγχος ορίων ως προς την τιμή με εμφάνιση αντίστοιχου μηνύματος και τέλος εμφάνιζε εάν το διαμέρισμα ήταν διαθέσιμο προς ενοικίαση ή προς πώληση με την αντίστοιχη τιμή, καθώς και αν προοριζόταν για ιδιοκατοίκηση.

4.1.2 Επαναληπτική δομή for

Μία από τις δομές επανάληψης που αναφέρθηκαν παραπάνω είναι και η δομή for. Η δομή for αποτελεί έναν επαναληπτικό βρόγχο για προκαθορισμένο αριθμό επαναλήψεων, σταματάει την εκτέλεση εντολών όταν η μεταβλητή επανάληψης φτάσει στον προαποφασισμένο αριθμό. Η μεταβλητή αυξάνεται ή μειώνεται με συγκεκριμένη τιμή, που προκαθορίζεται και ονομάζεται βήμα (Μανής , 2015).

Σκοπός των ασκήσεων της παρούσας θεματικής ενότητας ήταν να εξασκηθούν οι φοιτητές στα βήματα της Υπολογιστικής Σκέψης με τη χρήση δομής απόφασης και εμβάθυνση, στις δομές επανάληψης και τους βρόγχους, και πιο συγκεκριμένα τη δομή επανάληψης (for), που αποτελεί επαναληπτική διαδικασία για συγκεκριμένο αριθμό βημάτων.

Το τρίτο υποθετικό σενάριο *«Άσκηση 3: for»* (βλ. Παράρτημα Ι) αφορούσε μια κατασκευαστική εταιρεία, η οποία αναλάμβανε να διεκπεραιώσει την κατασκευή ενός

δρόμου ορισμένου μήκους σε προκαθορισμένο χρονικό διάστημα. Λόγω καιρικών συνθηκών και για τήρηση του χρονοδιαγράμματος κάποιες ημέρες απαιτούνταν υπερωριακή εργασία. Το ζητούμενο ήταν η υλοποίηση προγράμματος στη γλώσσα προγραμματισμού Python, όπου δέχονταν είσοδο αριθμού ημερών απασχόλησης με έλεγχο τιμής, υπολόγιζε και εμφάνιζε αριθμό εργάσιμων ημερών και υπερωριακής εργασίας, για τήρηση του χρονικού πλαισίου και υπολογισμό αντίστοιχης ατομικής αμοιβής καθώς και συνολικό κόστος αμοιβής των εργαζομένων με αντίστοιχο έλεγχο τιμής. Για την επίλυση, απαιτούνταν η ανάλυσή του καθώς και η χρήση των προαναφερόμενων δομών.

Το τέταρτο υποθετικό σενάριο *«Άσκηση 4: for»* (βλ. Παράρτημα Ι) αφορούσε ένα υποθετικό παιχνίδι ερωτήσεων, στο οποίο διαγωνίζονταν δυο αντίπαλοι και σε κάθε ερώτηση αντιστοιχούσαν έξι πόντοι, με συγκεκριμένο τρόπο καταμερισμού, ανάλογα με την απάντηση. Με χρήση αυτοματοποιημένης ή μη διαδικασίας υπολογίζονταν το σύνολο των πόντων κάθε παίκτη. Το ζητούμενο ήταν η υλοποίηση προγράμματος στη γλώσσα προγραμματισμού Python, για τον καθορισμό του συνολικού αριθμού ερωτήσεων, την ερώτηση έναρξης της αυτοματοποιημένης καταμέτρησης, που αν ήταν διάφορη της πρώτης, ζητούνταν χειροκίνητη εισαγωγή βαθμολογίας των δύο παικτών. Τέλος ζητούνταν να υπολογιστεί το σύνολο των πόντων των δύο παικτών, για 60 ερωτήσεις, με βάση ένα συγκεκριμένο μοτίβο.

4.1.3 Αποθηκευτικές δομές (λίστες)

Οι λίστες αποτελούν ένα πολύ σημαντικό, δυναμικό και ευέλικτο εργαλείο της Python καθώς προσφέρουν τη δυνατότητα αποθήκευσης και προσπέλασης διαφορετικού τύπου δεδομένων και τη δυνατότητα αύξησης ή μείωσης αυτών, κατά τη διάρκεια εκτέλεσης του προγράμματος (Μανής , 2015).

Οι μαθησιακοί στόχοι των ασκήσεων σε αυτή τη θεματική ενότητα αφορούσαν στην εξοικείωση των φοιτητών με τις δομές ελέγχου και επανάληψης, σε συνδυασμό με μια πιο σύνθετη και ευέλικτη δομή δεδομένων, τις λίστες, που χρησιμοποιούνται για αποθήκευση και προσπέλαση δεδομένων, με σκοπό την αλγοριθμική επίλυση του προβλήματος και την εφαρμογή των χαρακτηριστικών της Υπολογιστικής Σκέψης.

Το πέμπτο υποθετικό σενάριο *«Άσκηση 5: list»* (βλ. Παράρτημα Ι) αναφέρονταν σε έναν πανεπιστήμιο που διέθετε μεταπτυχιακά προγράμματα τριών διαφορετικών σχολών με κοινή γραμματεία. Κατά τη διάρκεια των εγγραφών, η γραμματεία παρείχε

στους φοιτητές ένα όνομα χρήστη και έναν κωδικό για την εισαγωγή τους στο ηλεκτρονικό σύστημα διαχείρισης μαθημάτων. Το ζητούμενο ήταν, η υλοποίηση ενός προγράμματος, στη γλώσσα προγραμματισμού Python, για την εισαγωγή του αριθμού των εισακτέων προηγούμενων ετών, του τρέχοντος έτους ανά τμήμα, τη δημιουργία ονόματος χρήστη και κωδικού πρόσβασης σύμφωνα με συγκεκριμένα κριτήρια, την αντιστοίχιση με το ονοματεπώνυμο που εισήγαγε ο χρήστης και τέλος την εμφάνιση της συνολικής λίστας όπως αυτή διαμορφώνονταν.

Το έκτο υποθετικό σενάριο *«Άσκηση 6: list»* (βλ. Παράρτημα Ι) αφορούσε ένα τραπεζικό ίδρυμα, το οποίο ανακοίνωνε καθημερινά επιτόκια προθεσμιακών καταθέσεων για καθορισμένα χρονικά διαστήματα. Το ζητούμενο ήταν η υλοποίηση ενός προγράμματος στη γλώσσα προγραμματισμού Python, για εισαγωγή ποσού προθεσμιακής κατάθεσης, δημιουργία πίνακα επιτοκίων με συγκεκριμένα ημερολογιακά κριτήρια, υπολογισμός, επιλογή και εμφάνιση του μεγαλύτερου επιτοκίου βασισμένου σε προκαθορισμένα κριτήρια, υπολογισμός του συνόλου των τόκων που θα απέφερε, και συχνότητα εμφάνισης του μεγαλύτερου ημερήσιου επιτοκίου για συγκεκριμένη χρονική περίοδο.

4.1.4 Αποθηκευτικές δομές (λεξικά)

Μια ακόμα δομή δεδομένων όπως προαναφέρθηκε είναι τα λεξικά που χρησιμοποιούνται για την αποθήκευση και προσπέλαση δεδομένων, θυμίζουν λίστες αλλά λειτουργούν με χαρακτηριστική αντιστοίχιση ενός μοναδικού κλειδιού με μια τιμή (Μανής , 2015).

Μέσω των εργασιών της ενότητας αυτής οι φοιτητές εξοικειώνονταν με τη χρήση των λεξικών, μιας δομής αντιστοίχισης κλειδιού-τιμής, καθώς επίσης και μιας πιο σύνθετης δομής που προέκυπτε από το συνδυασμό λίστας-λεξικών, με τελικό στόχο την αλγοριθμική επίλυση του προβλήματος και με χρήση των συνιστωσών της Υπολογιστικής Σκέψης.

Το έβδομο υποθετικό σενάριο *«Άσκηση 7: dictionary»* (βλ. Παράρτημα Ι) αναφέρονταν στον ιδιοκτήτη ενός καταστήματος πώλησης μεταχειρισμένων αυτοκινήτων, ο οποίος ήθελε να γνωρίζει το είδος και τον αριθμό των αυτοκινήτων στα οποία μπορούσε να προσφέρει έκπτωση, με κριτήριο τα χιλιόμετρά τους καθώς και την οικονομικότερη διαθέσιμη επιλογή. Το ζητούμενο ήταν η υλοποίηση ενός προγράμματος, στη γλώσσα προγραμματισμού Python, για τον ορισμό αριθμού διαθέσιμων αυτοκινήτων στο

κατάστημα, εισαγωγή τύπου αυτοκινήτου από το χρήστη, δημιουργία, αποθήκευση και εμφάνιση χαρακτηριστικών των αυτοκινήτων, με βάση προκαθορισμένα κριτήρια, υπολογισμός μέσου όρου χιλιομέτρων και έκπτωσης με κριτήρια, και τέλος υπολογισμός και εμφάνιση της οικονομικότερης επιλογής.

Το όγδοο υποθετικό σενάριο *«Άσκηση 8: dictionary»* (βλ. Παράρτημα Ι) αφορούσε συμμετέχοντες στις εισαγωγικές εξετάσεις μιας γυμναστικής ακαδημίας, στο άθλημα της σφαιροβολίας και ορίζονταν η βαθμολογία ανάλογα με τις επιδόσεις του κάθε αθλητή σε τρείς προσπάθειες, καθώς και ο τελικός πίνακας κατάταξης. Το ζητούμενο ήταν, η υλοποίηση ενός προγράμματος στη γλώσσα προγραμματισμού Python, για δημιουργία ερώτησης επόμενου αθλητή και εισαγωγή των στοιχείων του, εμφάνιση αριθμού προσπάθειας και καθορισμό εγκυρότητάς της, δημιουργία ρίψης, δημιουργία και εμφάνιση βαθμολογίας ρίψης, με βάση προκαθορισμένα κριτήρια, υπολογισμός, εμφάνιση και έλεγχος ατομικών προσπαθειών σύμφωνα με τα προκαθορισμένα όρια, μέσο όρο συνολικής βαθμολογίας, και τέλος δημιουργία, ταξινόμηση και εμφάνιση πίνακα κατάταξης.

4.1.5 Συναρτήσεις

Οι συναρτήσεις αποτελούν ένα από τα σπουδαιότερα δομικά στοιχεία ενός προγράμματος καθώς επιτρέπουν την ομαδοποίηση εντολών και την επαναχρησιμοποίηση τμημάτων κώδικα. Κάθε συνάρτηση διαθέτει μια ονομασία η οποία χρησιμοποιείται για την κλήση της. Η κλήση των συναρτήσεων μπορεί να πραγματοποιηθεί παραπάνω από μια φορές και σε οποιοδήποτε τμήμα του κώδικα (Μανής , 2015).

Με τις παρακάτω δύο ασκήσεις της τελευταίας θεματικής ενότητας του μαθήματος οι φοιτητές εξοικειώνονταν με τις συναρτήσεις για τη σωστή δόμηση ενός προγράμματος, καθώς και την επαναχρησιμοποίηση τους, με σκοπό την αποφυγή του πλεονασμού και την απλοποίηση του κώδικα. Η χρήση των συναρτήσεων σε συνδυασμό με τις ιδιότητες της Υπολογιστικής Σκέψης αποσκοπούσε στην εξεύρεση της καλύτερης προσέγγισης για την επίλυση ενός προβλήματος.

Το ένατο υποθετικό σενάριο *«Άσκηση 9: functions»* (Βακάλη, και συν., 2015) (βλ. Παράρτημα Ι) αφορούσε έναν μετατροπέα συναλλάγματος ξένων νομισμάτων σε ευρώ, με προκαθορισμένα είδη μετατροπών, που δίνονταν στην εκφώνηση της άσκησης. Το ζητούμενο ήταν η υλοποίηση ενός προγράμματος, στη γλώσσα προγραμματισμού

Python, για τη δημιουργία, εμφάνιση και επιλογή μενού μετατροπών, καθώς και έλεγχο ορίων τιμής, εισαγωγή ποσού μετατροπής, υπολογισμό και εμφάνιση μετατροπής συναλλάγματος, βάσει ισοτιμιών, υπολογισμό και εμφάνιση συναλλάγματος βάσει του μικρότερου αριθμού χαρτονομισμάτων και νομισμάτων, σύμφωνα με τις προκαθορισμένες αξίες και τέλος υπολογισμό και εμφάνιση του συνολικού συναλλάγματος ανά κατηγορία νομίσματος.

Το δέκατο και υψηλότερης δυσκολίας υποθετικό σενάριο *«Άσκηση 10: functions»* (βλ. Παράρτημα Ι) αφορούσε μια μεταφορική εταιρία που ήθελε να εφαρμόσει ένα πιλοτικό πρόγραμμα, για το σχεδιασμό της διαδρομής που θα ακολουθούσε το αυτοκίνητό της, βάσει σημείων παραλαβής και παράδοσης, για τον υπολογισμό κατανάλωσης καυσίμων συγκεκριμένης διαδρομής. Το ζητούμενο ήταν η υλοποίηση ενός προγράμματος, στη γλώσσα προγραμματισμού Python, για καθορισμό αριθμού πελατών προς εξυπηρέτηση, δημιουργία, αποθήκευση και εμφάνιση πελατών με τις αντίστοιχες τοποθεσίες παραλαβής και παράδοσης, δημιουργία και εκτέλεση διαδρομών με βάση την κοντινότερη απόσταση μεταξύ των πελατών και μέχρι την ολοκλήρωση της λίστας, εμφάνιση διαδρομής που διανύθηκε και τέλος υπολογισμός και εμφάνιση της συνολικής απόστασης με την ανάλογη κατανάλωση καυσίμων.

Για όλες τις παραπάνω ασκήσεις δόθηκε στους φοιτητές και ένας κενός πίνακας με τα στοιχεία της Υπολογιστικής Σκέψης που θα έπρεπε να αναγνωρίσουν και να συμπληρώσουν σε κάθε περίπτωση (αφαίρεση, αποσύνθεση, εντολές και πρότυπο) (Steka & Tsiatsos, 2018) (βλ. Παράρτημα Ι). Με τον τρόπο αυτό οι φοιτητές σχεδίαζαν ένα είδος εννοιολογικού χάρτη και χώριζαν τον κώδικα που σχεδίασαν στα αντίστοιχα χαρακτηριστικά της Υπολογιστικής Σκέψης. Ο διδάσκων είχε τη δυνατότητα να διαπιστώσει αν οι φοιτητές είχαν κατανοήσει και εφαρμόσει τις διδαχθείσες έννοιες. Τέλος, μετά την εκφώνηση της κάθε άσκησης δινόταν και ένα παράδειγμα του αναμενόμενου αποτελέσματος.

4.2 Τεστ Υπολογιστικής Σκέψης (Computational Thinking Test – CTt)

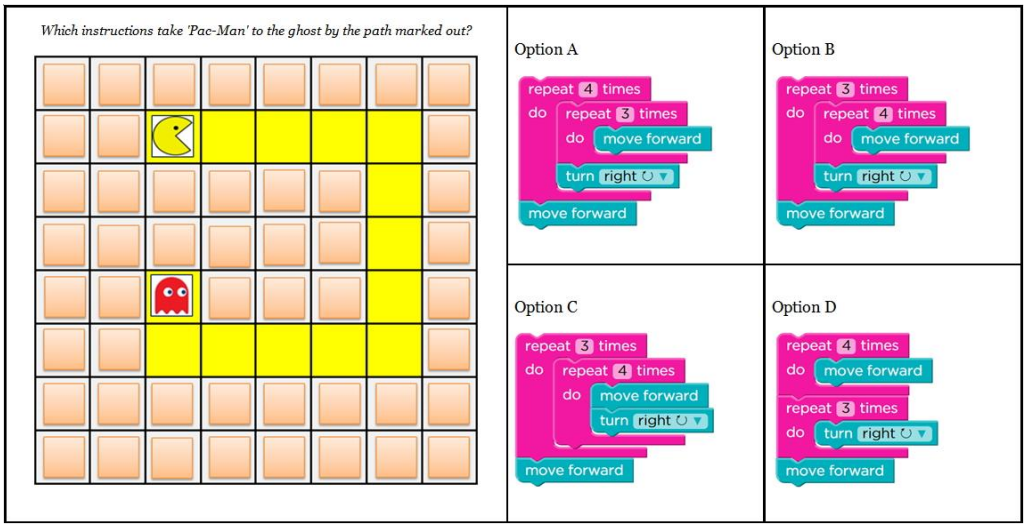
Το CTt σχεδιάστηκε αρχικά για την αξιολόγηση της Υπολογιστικής Σκέψης μαθητών μέσης εκπαίδευσης (Roman-Gonzalez, Perez-Gonzalez, & Jimenez-Fernandez, 2017) σύμφωνα με τις πρακτικές οδηγίες κατασκευής εργαλείων μέτρησης του επιπέδου γνώσης της Επιστήμης των Υπολογιστών. Η τελική του μορφή (έκδοση 2.0, Δεκέμβριος 2014) αποτελείται από 28 ερωτήσεις πολλαπλής επιλογής έπειτα από διαδικασία εκτίμησης της εγκυρότητας του περιεχομένου του, που πραγματοποιήθηκε με την κρίση είκοσι ειδικών. Το CTt μπορεί να χορηγηθεί ομαδικά σε ηλεκτρονική μορφή με τη χρήση φορητών ή/και σταθερών ηλεκτρονικών συσκευών.

Για τους σκοπούς της παρούσας έρευνας το CTt μεταφράστηκε στην ελληνική γλώσσα από τον ερευνητή. Η εκδοχή του Τεστ Υπολογιστικής Σκέψης που προέκυψε από τη διαδικασία αυτή διατήρησε τον ίδιο τρόπο παρουσίασης των ερωτήσεων με το CTt. Το παρόν Τεστ Υπολογιστικής Σκέψης αποτελείται από 28 ερωτήσεις πολλαπλών επιλογών, χωρισμένες σε επτά σελίδες, με την κάθε σελίδα να περιλαμβάνει τέσσερις ερωτήσεις. Οι συμμετέχοντες καλούνται να επιλέξουν μια από τις τέσσερις επιλογές που δίνονται σε κάθε ερώτηση (Α,Β,Γ και Δ), ενώ υπάρχει μόνο μια σωστή απάντηση. Ο μέγιστος χρόνος που απαιτείται για τη συμπλήρωση του τεστ είναι 45 λεπτά. Πριν την παρουσίαση των 28 ερωτήσεων του τεστ υπάρχουν τρία λυμένα παραδείγματα, προκειμένου να εξοικειωθούν οι συμμετέχοντες με το είδος των ερωτήσεων που ακολουθούν, καθώς και με τους χαρακτήρες που εμφανίζονται σε αυτές.

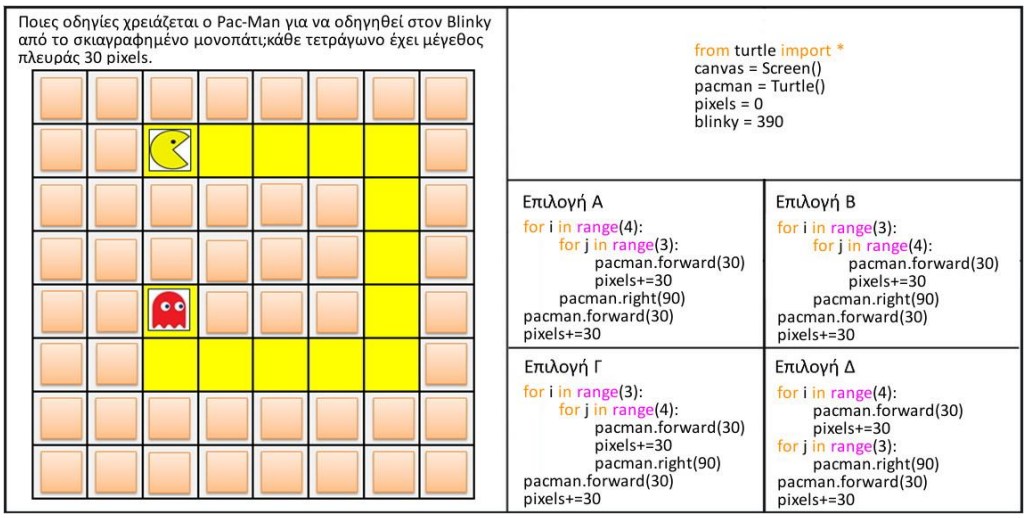
Η έκδοση του τεστ που χορηγήθηκε στην παρούσα έρευνα διαφέρει από το CTt σε τρία σημεία. Από την αρχική έκδοση του τεστ αφαιρέθηκαν οι δύο ερωτήσεις αυτοαξιολόγησης αναφορικά με την επίδοση των συμμετεχόντων στο τεστ και τη σχέση τους με τους υπολογιστές. Επίσης, στην ελληνική έκδοση προστέθηκε μια σελίδα με τα δημογραφικά στοιχεία των φοιτητών. Η μεγαλύτερη αλλαγή όμως που έγινε στο περιεχόμενο του τεστ ήταν ο διαφορετικός τρόπος εκφώνησης και ο μετασχηματισμός από οπτική γλώσσα προγραμματισμού στην γλώσσα προγραμματισμού python (βλ. Παράρτημα ΙΙ), προκειμένου να προσαρμοστεί στο ηλικιακό και εκπαιδευτικό επίπεδο των συμμετεχόντων καθώς και στους μαθησιακούς στόχους του συγκεκριμένου μαθήματος (Βλ. Εικόνες 8 & 9).

Η χορήγησή του τεστ έγινε σε ηλεκτρονική μορφή μέσω της ηλεκτρονικής πλατφόρμας του Αριστοτελείου Πανεπιστημίου Θεσσαλονίκης (elearning.auth.gr). Οι φοιτητές που παρακολούθησαν το μάθημα «Εισαγωγή στην Πληροφορική» έλαβαν το σχετικό σύνδεσμο με το τεστ, ενημερώθηκαν ότι αποτελεί μέρος της διπλωματικής εργασίας φοιτητή και ότι αξιολογεί την Υπολογιστική Σκέψη.

Εικόνα 8: Παράδειγμα αρχικής μορφής του τεστ Υπολογιστικής Σκέψης (Roman-Gonzalez, Moreno-Leon, & Robles, 2017)



Εικόνα 9: Παράδειγμα της τελικής μορφής μετά την τροποποίηση



5 Αποτελέσματα

Τα ευρήματα που παρατίθενται στη συνέχεια προέκυψαν από τη στατιστική ανάλυση των δεδομένων που αντλήθηκαν κατά το ακαδημαϊκό έτος 2017-18 από τη βαθμολογία των φοιτητών στην τελική γραπτή εξέταση του μαθήματος «Εισαγωγή στην Πληροφορική», από τη βαθμολογία στις προαιρετικές εργασίες που δόθηκαν στους φοιτητές κατά τη διάρκεια του εξαμήνου και από τη βαθμολογία τους στο τεστ αξιολόγησης της Υπολογιστικής Σκέψης που χορηγήθηκε μετά τις γραπτές εξετάσεις του μαθήματος. Η στατιστική ανάλυση των δεδομένων πραγματοποιήθηκε με τη χρήση του προγράμματος PASW Statistics 18.0 (WinWrap, 2009).

5.1 Συμμετέχοντες

Στην παρούσα έρευνα έλαβαν μέρος οι φοιτητές του τμήματος Πληροφορικής του Αριστοτελείου Πανεπιστημίου Θεσσαλονίκης, οι οποίοι παρακολούθησαν το μάθημα «Εισαγωγή στην Πληροφορική» κατά το εκπαιδευτικό έτος 2017-18. Από τους 282 εγγεγραμμένους φοιτητές στο μάθημα, οι 186 έλαβαν μέρος στις τελικές γραπτές εξετάσεις, ενώ το 20,4% αυτών (38 άτομα) παρέδωσαν και τις προαιρετικές εργασίες που δόθηκαν κατά τη διάρκεια του εξαμήνου και 16 άτομα συμπλήρωσαν το τεστ αξιολόγησης της Υπολογιστικής Σκέψης.

Στον Πίνακα 4 παρουσιάζονται οι μέσοι όροι, οι τυπικές αποκλίσεις, η ελάχιστη και μέγιστη τιμή της βαθμολογίας που έλαβαν οι φοιτητές του δείγματος, στην τελική εξέταση του μαθήματος και στις προαιρετικές εργασίες, καθώς και του αριθμού των ασκήσεων που έλυσαν.

Πίνακας 4: Βαθμολογία τελικής εξέτασης, προαιρετικές εργασίες, τεστ αξιολόγησης ΥΣ

Βαθμολογία	N	M.O.	SD	Min.	Max.
Τελικές εξετάσεις μαθήματος	186	29,94	7,00	13,0	40,0
Προαιρετικές εργασίες	38	1,01	0,52	0,2	2,0
Αριθμός ασκήσεων που έλυσαν	38	6,50	2,49	2,0	10,0
Τεστ αξιολόγησης Υπολογιστικής Σκέψης	16	20,69	4,56	11,0	27,0

Όπως φαίνεται στον Πίνακα 4, ο μέσος όρος της βαθμολογίας των φοιτητών στο μάθημα ήταν 29,94 (SD = 7,00) με άριστα το 40 και στις προαιρετικές εργασίες ήταν 1,01 (SD = 0,52) με άριστα το 2. Οι φοιτητές που παρέδωσαν τις προαιρετικές εργασίες έλυσαν κατά μέσο όρο 6,50 ασκήσεις (SD = 2,49). Οι δεκαέξι φοιτητές που

συμπλήρωσαν το τεστ αξιολόγησης της Υπολογιστικής Σκέψης είχαν μέσο όρο 20,69 (SD = 4,56) με άριστα το 28. Λόγω της χαμηλής συμμετοχής των φοιτητών στο συγκεκριμένο τεστ, δεν συμπεριλήφθηκε στις περαιτέρω στατιστικές αναλύσεις.

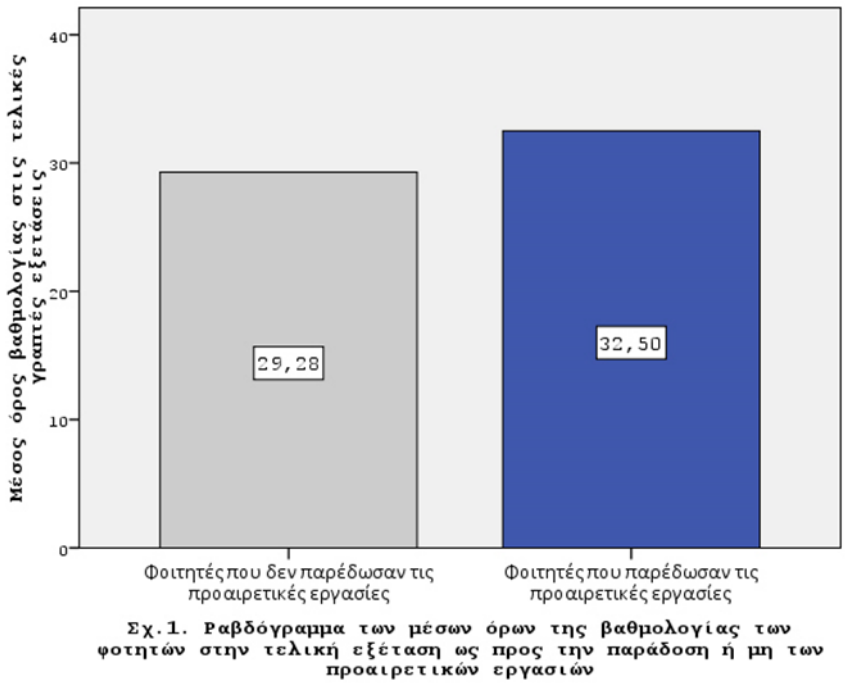
Προκειμένου να διερευνηθεί αν υπήρξε διαφορά μεταξύ των φοιτητών που παρέδωσαν τις προαιρετικές εργασίες και αυτών που δεν τις παρέδωσαν, ως προς τη βαθμολογία που έλαβαν στην τελική εξέταση του μαθήματος, πραγματοποιήθηκε ο έλεγχος των μέσων όρων με το κριτήριο t για ανεξάρτητα δείγματα.

Όπως φαίνεται στον Πίνακα 5, βρέθηκε στατιστικώς σημαντική διαφορά μεταξύ των δύο ομάδων ως προς τη βαθμολογία στην τελική εξέταση, t (68,81) = -2,90, p < 0,01. Πιο συγκεκριμένα, οι φοιτητές που δεν είχαν παραδώσει τις προαιρετικές εργασίες είχαν μέσο όρο βαθμολογίας στις εξετάσεις 29,28 (SD = 7,15), ενώ ο αντίστοιχος μέσος όρος για τους φοιτητές που είχαν παραδώσει τις εργασίες ήταν 32,50 (SD = 5,80). (βλ. Πίνακα 5, Σχήμα 1).

Πίνακας 5: Μέσοι όροι, t τιμές της βαθμολογίας των φοιτητών στην τελική εξέταση

	Μέσος όρος (M) βαθμολογίας φοιτητών που δεν παρέδωσαν προαιρετικές εργασίες	**Μέσος όρος (M) βαθμολογίας φοιτητών που παρέδωσαν προαιρετικές εργασίες**	**t - test**		
			t	**df**	**p**
Βαθμολογία στην τελική εξέταση	29,28	32,50	-2,90	68,81	0,005**

Σημείωση: **p<0,01.



Σχ.1. Ραβδόγραμμα των μέσων όρων της βαθμολογίας των φοτητών στην τελική εξέταση ως προς την παράδοση ή μη των προαιρετικών εργασιών

Σχήμα 1: Διαφορά μέσων όρων των συμμετεχόντων στην τελική εξέταση

Στη συνέχεια από τις 40 ερωτήσεις της τελικής εξέτασης επιλέχθηκαν δέκα (10) ερωτήσεις που αξιολογούν τις θεωρητικές γνώσεις των φοιτητών πάνω στην Υπολογιστική Σκέψη, όπως για παράδειγμα τον ορισμό της και τα χαρακτηριστικά της γνωρίσματα. Οι ερωτήσεις αυτές ήταν οι 21,22,23,24,25,26,27,28,29 και 30. Επιλέχθηκαν επίσης δώδεκα (12) ερωτήσεις που αξιολογούσαν την πρακτική κατανόηση των φοιτητών για την Υπολογιστική Σκέψη, συμπεριλαμβανομένης της ερμηνείας κώδικα. Οι ερωτήσεις αυτές ήταν οι: 14,15,31,32,33,34,35,36,37,38,39 και 40. Οι υπόλοιπες δεκαοκτώ (18) ερωτήσεις αφορούσαν στην υπόλοιπη διδακτέα ύλη του εξεταζόμενου μαθήματος.

Ο μέσος όρος της βαθμολογία των φοιτητών του δείγματος στις δώδεκα (12) πρακτικές ερωτήσεις της τελικής εξέτασης του μαθήματος ήταν 8,73 (SD = 3,03) και στις δέκα (10) θεωρητικές ερωτήσεις ήταν 8,30 (SD = 1,52) (βλ. Πίνακα 6).

Πίνακας 6: Πρακτικές και Θεωρητικές ερωτήσεις της τελικής εξέτασης του μαθήματος

Βαθμολογία	N	M.O.	SD	Min.	Max.
Βαθμολογία στις πρακτικές ερωτήσεις	186	8,73	3,03	2,0	12,0
Βαθμολογία στις θεωρητικές ερωτήσεις	186	8,30	1,52	3,0	10,0

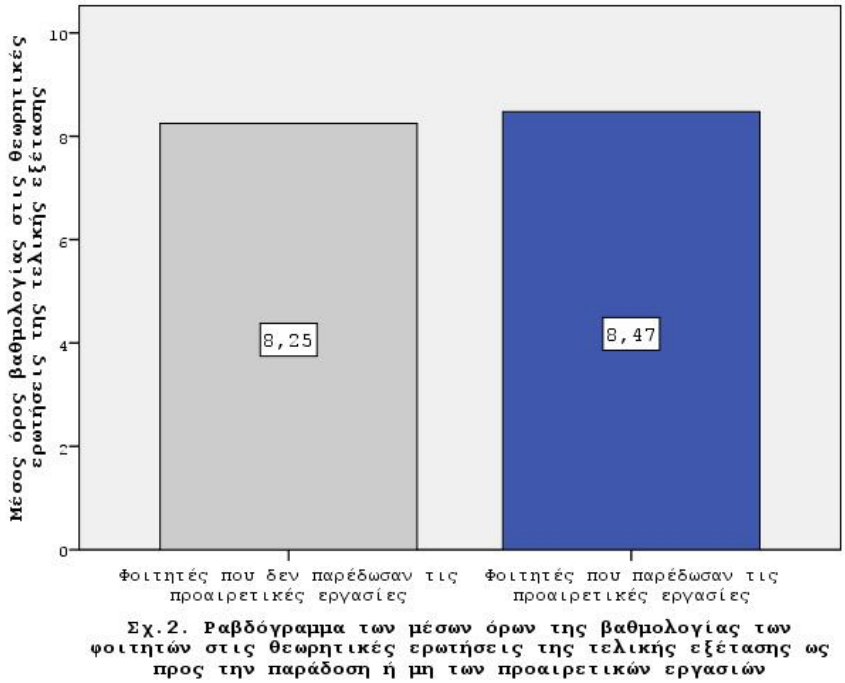
Προκειμένου να διερευνηθεί, αν υπήρξε διαφορά μεταξύ των φοιτητών που παρέδωσαν τις προαιρετικές εργασίες και αυτών που δεν τις παρέδωσαν, ως προς τη βαθμολογία που έλαβαν ξεχωριστά στις θεωρητικές και πρακτικές ερωτήσεις της τελικής εξέτασης

του μαθήματος, πραγματοποιήθηκε ο έλεγχος των μέσων όρων με το κριτήριο t για ανεξάρτητα δείγματα (βλ. Πίνακες 7 & 8, Σχήμα 2 & 3).

Πίνακας 7: Μέσοι όροι, t τιμές βαθμολογίας στις θεωρητικές ερωτήσεις της τελικής εξέτασης

	Μέσος όρος (M) βαθμολογίας φοιτητών που δεν παρέδωσαν προαιρετικές εργασίες	Μέσος όρος (M) βαθμολογίας φοιτητών που παρέδωσαν προαιρετικές εργασίες	t - test		
			t	df	p
Βαθμολογία στις θεωρητικές ερωτήσεις	**8,25**	**8,47**	**-0,81**	**184**	**0,421**

Σημείωση: $^{**}p<0,01$.



Σχ.2. Ραβδόγραμμα των μέσων όρων της βαθμολογίας των φοιτητών στις θεωρητικές ερωτήσεις της τελικής εξέτασης ως προς την παράδοση ή μη των προαιρετικών εργασιών

Σχήμα 2: Διαφορά μέσων όρων των συμμετεχόντων στην τελική εξέταση - θεωρητικές ερωτήσεις

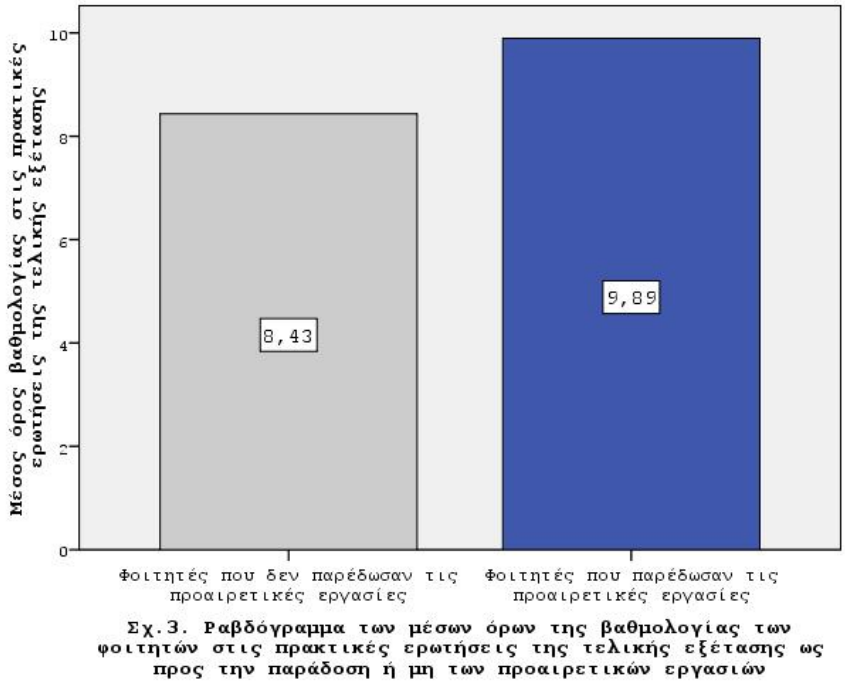
Όπως φαίνεται στον Πίνακα 7, δεν βρέθηκε στατιστικώς σημαντική διαφορά μεταξύ των δυο ομάδων ως προς τη βαθμολογία των φοιτητών στις θεωρητικές ερωτήσεις της τελικής εξέτασης, t (184) = -0,81, p > 0,05. Πιο συγκεκριμένα, οι φοιτητές που δεν είχαν παραδώσει τις προαιρετικές εργασίες είχαν μέσο όρο βαθμολογίας στις θεωρητικές ερωτήσεις της τελικής εξέτασης 8,25 (SD = 1,55), ενώ ο αντίστοιχος μέσος όρος για τους φοιτητές που είχαν παραδώσει τις εργασίες ήταν 8,47 (SD = 1,41).

Όπως φαίνεται στον Πίνακα 8, βρέθηκε στατιστικώς σημαντική διαφορά μεταξύ των δύο ομάδων ως προς τη βαθμολογία των φοιτητών στις πρακτικές ερωτήσεις της τελικής εξέτασης, t (89,40) = -3,49, p < 0,01. Πιο συγκεκριμένα, οι φοιτητές που δεν είχαν παραδώσει τις προαιρετικές εργασίες είχαν μέσο όρο βαθμολογίας στις πρακτικές ερωτήσεις της τελικής εξέτασης 8,43 (SD = 3,17), ενώ ο αντίστοιχος μέσος όρος για τους φοιτητές που είχαν παραδώσει τις εργασίες ήταν 9,89 (SD = 2,02). (βλ. Πίνακα 8).

Πίνακας 8: Μέσοι όροι t τιμές βαθμολογίας στις πρακτικές ερωτήσεις της τελικής εξέτασης

	Μέσος όρος (M) βαθμολογίας φοιτητών που δεν παρέδωσαν προαιρετικές εργασίες	**Μέσος όρος (M) βαθμολογίας φοιτητών που παρέδωσαν προαιρετικές εργασίες**	**t - test**		
			t	**df**	**p**
Βαθμολογία στις πρακτικές ερωτήσεις	**8,43**	**9,89**	**-3,49**	**89,40**	**0,001****

Σημείωση: **p<0,01.



Σχ.3. Ραβδόγραμμα των μέσων όρων της βαθμολογίας των φοιτητών στις πρακτικές ερωτήσεις της τελικής εξέτασης ως προς την παράδοση ή μη των προαιρετικών εργασιών

Σχήμα 3: Διαφορά μέσων όρων των συμμετεχόντων στην τελική εξέταση - πρακτικές ερωτήσεις

Προκειμένου να υπολογιστεί ο βαθμός και η κατεύθυνση της συνάφειας μεταξύ της βαθμολογίας των φοιτητών (N=38) στις προαιρετικές εργασίες, στις πρακτικές ερωτήσεις της τελικής εξέτασης και τις θεωρητικές ερωτήσεις της τελικής εξέτασης, χρησιμοποιήθηκε ο δείκτης συνάφειας Pearson r (βλ. Πίνακα 9). Όλοι οι δείκτες ήταν θετικής κατεύθυνσης και στατιστικώς σημαντικοί (με εξαίρεση της συνάφειας ανάμεσα στη βαθμολογία στις προαιρετικές εργασίες και στις θεωρητικές ερωτήσεις της τελικής εξέτασης, η οποία ήταν αρνητικής κατεύθυνσης και στατιστικώς ασήμαντη). Το μέγεθος των δεικτών κυμάνθηκε από 0,329 (συνάφεια μεταξύ της βαθμολογίας στις προαιρετικές εργασίες και στις πρακτικές ερωτήσεις της τελικής εξέτασης) μέχρι 0,350 (συνάφεια μεταξύ της βαθμολογίας στις πρακτικές ερωτήσεις και στις θεωρητικές ερωτήσεις της τελικής εξέτασης). Δηλαδή, όσο υψηλότερη βαθμολογία είχε ένας φοιτητής στις προαιρετικές εργασίες, τόσο υψηλότερη έτεινε να είναι η βαθμολογία του στις πρακτικές ερωτήσεις της τελικής εξέτασης, αν και το μέγεθος των συσχετίσεων αυτών ήταν χαμηλό. Η θετική συνάφεια που βρέθηκε ανάμεσα στη βαθμολογία στις θεωρητικές και πρακτικές ερωτήσεις ήταν αναμενόμενη καθώς αποτελούν μέρη της ίδιας γραπτής εξέτασης.

Πίνακας 9: Συνάφεια (Pearson r) προαιρετικές εργασίες, πρακτικές - θεωρητικές ερωτήσεις τελικής εξέτασης

	Προαιρετικές εργασίες	Πρακτικές ερωτήσεις	Θεωρητικές ερωτήσεις
Βαθμολογία στις προαιρετικές εργασίες	1,00		
Βαθμολογία τις πρακτικές ερωτήσεις	0,329*	1,00	
Βαθμολογία στις θεωρητικές ερωτήσεις	-0,097	0,350*	1,00

Σημείωση: *p<0,05.

6 Συμπεράσματα – Περιορισμοί - Προτάσεις

Σκοπός της παρούσας έρευνας ήταν η δημιουργία εκπαιδευτικού υλικού για την ενίσχυση της Υπολογιστικής Σκέψης των εκπαιδευόμενων στα πλαίσια του μαθήματος «Εισαγωγή στην Πληροφορική». Αφορμή για το εγχείρημα αυτό στάθηκε το αυξανόμενο ενδιαφέρον που παρατηρείται στη βιβλιογραφία από το 2006 για τα χαρακτηριστικά της Υπολογιστικής Σκέψης και την εφαρμογή της σε ποικίλους επιστημονικούς τομείς (Ilic, Haseski, & Tugtekin, 2018).

Για τις ανάγκες της έρευνας σχεδιάστηκαν δέκα προαιρετικές ασκήσεις κλιμακούμενης δυσκολίας, οι οποίες είχαν ως στόχο την εξοικείωση των φοιτητών με τις βασικές έννοιες της Υπολογιστικής Σκέψης, όπως είναι η αφαίρεση, η αποσύνθεση, η αναγνώριση προτύπων, οι αλγόριθμοι και ένα μέρος της αξιολόγησης. Η εξάσκηση των δεξιοτήτων της Υπολογιστικής Σκέψης πραγματοποιήθηκε με τη χρήση της γλώσσας προγραμματισμού Python στις ακόλουθες θεματικές: δομές ελέγχου ροής, επαναληπτική δομή for, αποθηκευτικές δομές και πιο συγκεκριμένα λίστες και λεξικά και τέλος συναρτήσεις. Για την τελική αξιολόγηση των δεξιοτήτων Υπολογιστικής Σκέψης των εκπαιδευόμενων μεταφράστηκε, προσαρμόστηκε και χορηγήθηκε ένα Τεστ Υπολογιστικής Σκέψης, το οποίο αποτελείται από ερωτήσεις πολλαπλών επιλογών με μια σωστή απάντηση.

Τα ερευνητικά ερωτήματα που διατυπώθηκαν κατά τον αρχικό σχεδιασμό της έρευνας ήταν τα εξής:

1. Με ποιο τρόπο μπορούν να ενισχυθούν οι δεξιότητες Υπολογιστικής Σκέψης των φοιτητών μέσα από την εκτέλεση ασκήσεων με τη χρήση της γλώσσας προγραμματισμού Python;

Το εκπαιδευτικό υλικό που δημιουργήθηκε, με τη μορφή δέκα υποθετικών σεναρίων που ζητούσαν από τους φοιτητές να υλοποιήσουν κάθε φορά ένα πρόγραμμα, στη γλώσσα προγραμματισμού Python που να εκτελεί ποικίλες εντολές, βοήθησε τους συμμετέχοντες να εξοικειωθούν σταδιακά με τη συγκεκριμένη γλώσσα προγραμματισμού και να εξασκηθούν στις εντολές που περιλαμβάνονταν στις θεματικές ενότητες που αναφέρθηκαν. Ο σταδιακά αυξανόμενος βαθμός δυσκολίας των ασκήσεων, σε συνδυασμό με τη χορήγηση του λευκού πίνακα με τις έννοιες της Υπολογιστικής Σκέψης, συνέβαλαν στο να μάθουν οι φοιτητές να διακρίνουν τα

χαρακτηριστικά της Υπολογιστικής Σκέψης και να τα εφαρμόζουν στην επίλυση ασκήσεων.

Πιο συγκεκριμένα η διαδικασία που ακολουθείται για την επίτευξη της αλγοριθμικής επίλυσης ασκήσεων και η αποτύπωση της με τη μορφή κώδικα, επέρχεται με τη σταδιακή προσέγγιση της λύσης μέσω εστίασης στα κύρια ζητούμενα της άσκησης, απλοποίησης, αναγνώρισης και ομαδοποίησης επαναλαμβανόμενων στοιχείων, δεξιότητες που αναπτύσσονται, ενισχύονται και διευρύνονται με την επαναλαμβανόμενη εφαρμογή τους.

2. Ποιος είναι ο πιο κατάλληλος τρόπος να αξιολογηθούν οι δεξιότητες Υπολογιστικής Σκέψης των εκπαιδευόμενων;

Προκειμένου να απαντηθεί το ερώτημα αυτό χρησιμοποιήθηκαν δύο διαφορετικά μέσα αξιολόγησης της Υπολογιστικής Σκέψης. Το πρώτο ήταν η βαθμολογία τους, στην τελική γραπτή εξέταση του μαθήματος, σε ερωτήσεις πολλαπλής επιλογής και το δεύτερο η βαθμολογία τους στο Τεστ Υπολογιστικής Σκέψης. Δυστυχώς λόγω του χαμηλού ποσοστού επιστροφής συμπληρωμένων Τεστ Υπολογιστικής Σκέψης δεν στάθηκε εφικτή η διερεύνηση της συσχέτισης ανάμεσα στις δυο βαθμολογίες, γεγονός που αποτέλεσε περιορισμό στη συγκεκριμένη μελέτη. Στόχος της πιλοτικής αυτής χορήγησης του τεστ ήταν, να δίνεται προς συμπλήρωση μελλοντικά στην αρχή και στη λήξη του εξαμήνου, προκειμένου να έχει ο διδάσκων μια πληρέστερη εικόνα των γνώσεων των φοιτητών γύρω από την Υπολογιστική Σκέψη καθώς και των δυσκολιών που συναντούν στη κατανόηση και εφαρμογή των εννοιών της. Επιπρόσθετα για να αυξηθεί η συμμετοχή των εκπαιδευόμενων στο τεστ αυτό, προτείνεται να χορηγείται κατά τη διάρκεια του μαθήματος, ως ένα είδος άσκησης.

3. Παρατηρήθηκε διαφορά στις δεξιότητες Υπολογιστικής Σκέψης των εκπαιδευόμενων που παρέδωσαν τις προαιρετικές εργασίες, σε σύγκριση με εκείνους που δεν τις συμπλήρωσαν, όπως αυτές αξιολογήθηκαν από την τελική γραπτή εξέταση του μαθήματος;

Από τη στατιστική ανάλυση των δεδομένων της έρευνας βρέθηκε ότι οι φοιτητές που παρέδωσαν τις προαιρετικές εργασίες είχαν υψηλότερο μέσο όρο στη βαθμολογία της τελικής γραπτής εξέτασης, σε σύγκριση με όσους δεν ασχολήθηκαν με την επίλυσή τους. Οι δυο αυτές ομάδες διέφεραν επίσης και ως προς το μέσο όρο της βαθμολογίας που έλαβαν στις ερωτήσεις της τελικής εξέτασης που αξιολογούσαν την πρακτική

κατανόηση των όρων της Υπολογιστικής Σκέψης, με τους φοιτητές που παρέδωσαν τις εργασίες να έχουν υψηλότερο βαθμό. Τα ευρήματα αυτά φαίνεται να επιβεβαιώνουν την υπόθεση ότι το εκπαιδευτικό υλικό που σχεδιάστηκε για την ενίσχυση της Υπολογιστικής Σκέψης των φοιτητών είχε το επιθυμητό αποτέλεσμα. Το συμπέρασμα αυτό ενισχύεται και από την εύρεση θετικής συνάφειας, ανάμεσα στη βαθμολογία που έλαβαν οι φοιτητές στις ερωτήσεις που αξιολογούσαν την πρακτική κατανόηση των εννοιών της Υπολογιστικής Σκέψης και στη βαθμολογία των προαιρετικών ασκήσεων. Θα μπορούσαμε, συνεπώς, να υποθέσουμε ότι η εξάσκηση των εκπαιδευόμενων σε ασκήσεις που απαιτούν τη δημιουργία προγράμματος με τη χρήση γλώσσας προγραμματισμού προάγει την κατανόηση και αξιοποίηση των χαρακτηριστικών που απαρτίζουν την Υπολογιστική Σκέψη.

Με δεδομένο το ενδιαφέρον που παρατηρείται τα τελευταία χρόνια για τη βελτίωση των δεξιοτήτων της Υπολογιστικής Σκέψης και την αξιοποίησή της σε διάφορα επιστημονικά πεδία, κρίνεται ότι θα ήταν βοηθητικό να ενσωματωθούν πρακτικές ασκήσεις, που συμβάλλουν στην ενίσχυσή της, στη διδακτέα ύλη αντίστοιχων μαθημάτων. Θα είχε επίσης ενδιαφέρον να συμπληρωθεί το εκπαιδευτικό υλικό με τον κατάλληλο τρόπο ώστε να μπορεί να ενσωματωθεί και να διεξαχθεί με τη μορφή διαδικτυακού μαθήματος καθώς επίσης και τα αποτελέσματα που θα προέκυπταν, να συγκριθούν με τον τρόπο που ακολουθήθηκε στην παρούσα εργασία. Τέλος, για την καλύτερη αξιολόγηση των εκπαιδευτικών αποτελεσμάτων, θα ήταν βοηθητική η χορήγηση σε δυο φάσεις, ενός σταθμισμένου εργαλείου μέτρησης της Υπολογιστικής Σκέψης, το οποίο να παρέχει πληροφόρηση στους εκπαιδευόμενους για τις δυσκολίες που αντιμετωπίζουν, στην κατανόηση και εφαρμογή των εννοιών και να βοηθά τους διδάσκοντες να τους καθοδηγούν καλύτερα.

Βιβλιογραφία

Άρθρα – Πρακτικά Συσκέψεων - Εκθέσεις

Aho, A. V. (2012). Computation and Computational Thinking. *The Computer Journal, 55*(7), pp. 832-835.

Angeli, C., Voogt, J., Fluck, A., Webb, M., Cox, M., Malyn-Smith, J., et al. (2016, January). A K-6 Computational Thinking Curriculum Framework: Implications for Teacher Knowledge. *Educational Technology & Society, 19*, pp. 47-57.

Ater-Kranov, A., Bryant, R., Orr, G., Wallace, S., & Zhang, M. (2010). Developing a community definition and teaching modules for computational thinking: accomplishments and challenges. *Proceedings of the 2010 ACM conference on Information technology education* (pp. 143-148). Midland, Michigan, USA: ACM.

Basso, D., Fronza, I., Colombi, A., & Pahl, C. (2018). Improving Assessment of Computational Thinking Through a Comprehensive Framework. *Koli Calling '18 Proceedings of the 18th Koli Calling International Conference on Computing Education Research.* Koli, Finland: ACM.

Bocconi, S., Chioccariello, A., Dettori, G., Ferrari, A., Engelhardt, K., Kampylis, P., et al. (2016). Developing Computational Thinking in Compulsory Education. Implications for policy and practice. *EUR - Scientific and Technical Research Reports*, pp. 1-68.

Brackmann, C., Barone, D., Casali, A., Boucinha, R., & Muñoz-Hernandez, S. (2016). Computational thinking: Panorama of the Americas. *2016 International Symposium on Computers in Education (SIIE)*, (pp. 1-6). Salamanca.

Brackmann, C., Román-González, M., Robles, G., Moreno-León, J., Casali, A., & Barone, D. (2017). Development of Computational Thinking Skills through Unplugged Activities in Primary School. *WiPSCE '17 Proceedings of the 12th Workshop on Primary and Secondary Computing Education* (pp. 65-72). Nijmegen, Netherlands: ACM.

Brennan, K., & Resnick, M. (2012). New frameworks for studying and assessing the development of computational thinking. *American Educational Research Association*, (pp. 1-25). Vancouver, Canada.

Chen, G., Shen, J., Barth-Cohen, L., Jiang, S., Huang, X., & Eltoukhy, M. (2017, June). Assessing elementary students computational thinking in everyday reasoning and robotics programming. *Computers & Education, 109*, pp. 162-175.

Cooper, S., & Dann, W. (2015, March). Programming: A Key Component of Computational Thinking in CS Courses for Non-majors. *ACM Inroads, 6*(1), pp. 50-54.

CSTA. (2016). *[Interim] CSTA K-12 Computer Science Standards Revised 2016 CSTA STANDARDS TASK FORCE.* New York: ACM.

Curzon, P., Dorling, M., Ng, T., Selby, C., & Woollard, J. (2014). Developing computational thinking in the classroom: a framework. *Computing At School*, pp. 1-15.

Denning, P. J. (2009, June). The Profession of IT: Beyond Computational Thinking. *Communications of the ACM, 52*(6), pp. 28-30.

Denning, P. J. (2017a, January-February). Computational Thinking in Science. *American Scientist, 105*(1), pp. 13-17.

Denning, P. J. (2017b). Remaining Trouble Spots with Computational Thinking. *COMMUNICATIONS OF THE ACM, 60*(6), pp. 33-39.

Figueiredo, J., & García-Peñalvo, F. (2017). Improving Computational Thinking Using Follow and Give Instructions. *TEEM 2017 Proceedings of the 5th International Conference on Technological Ecosystems for Enhancing Multiculturality Article No. 3* (pp. 1-7). Cádiz, Spain: ACM.

García-Peñalvo, F. (2016). What Computational Thinking Is. *Journal of Information Technology Research, 9*(3), pp. 5-8.

Gretter, S., & Yadav, A. (2016, September). Computational Thinking and Media & Information Literacy: An Integrated Approach to Teaching Twenty-First Century Skills. *TechTrends, 60*(5), pp. 510-516.

Grover, S., Basu, S., Bienkowski, M., Eagle, M., Diana, N., & Stamper, J. (2017, August). A Framework for Using Hypothesis-Driven Approaches to Support Data-Driven Learning Analytics in Measuring Computational Thinking in Block-Based Programming Environments. *ACM Transactions on Computing Education, 17*(3), p. Article 14.

Ilic, U., Haseski, H., & Tugtekin, U. (2018). Publication Trends Over 10 Years of Computational Thinking Research. *CONTEMPORARY EDUCATIONAL TECHNOLOGY, 9*(2), pp. 131-153.

Jaeger, A. J. (2015). What Does the Punched Holes Task Measure? Chicago, Illinois: University of Illinois at Chicago.

Kafura, D., Bart, A., & Chowdhury, B. (2018, December). A Computational Thinking Course Accessible to Non-stem Majors. *Journal of Computing Sciences in Colleges, 34*(2), pp. 157-163.

Kafura, D., Bart, A., & Chowdhury, B. (2015). Design and Preliminary Results From a Computational Thinking Course. *ITiCSE '15 Proceedings of the 2015 ACM Conference on Innovation and Technology in Computer Science Education* (pp. 63-68). Vilnius, Lithuania: ACM.

Koh, K., Basawapatna, A., Nickerson, H., & Repenning, A. (2014a). Real Time Assessment of Computational Thinking. *2014 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*, (pp. 49-52). Melbourne.

Koh, K., Nickerson, H., Basawapatna, A., & Repenning, A. (2014b). Early Validation of Computational Thinking Pattern Analysis. *ITiCSE '14 Proceedings of the 2014*

conference on Innovation & technology in computer science education (pp. 213-218). Uppsala, Sweden: ACM.

Krugel, J., & Hubwieser, P. (2017). Computational thinking as springboard for learning object-oriented programming in an interactive MOOC. *2017 IEEE Global Engineering Education Conference (EDUCON)* (pp. 1709-1712). Athens, Greece: IEEE.

Lockwood, J., & Mooney, A. (2018, January). Computational Thinking in Secondary Education: Where does it fit? A systematic literary review. *International Journal of Computer Science Education in Schools, 2*(1), pp. 41-60.

Lu, J., & Fletcher, G. (2009). Thinking about Computational Thinking. *ACM Sigcse Bulletin* (pp. 260-264). Chattanooga, Tennessee, USA: ACM.

Maxwell, B., & Taylor, S. (2017). Comparing Outcomes Across Different Contexts in CS1. *SIGCSE '17 Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education* (pp. 399-403). Seattle, Washington, USA: ACM.

Quitério Figueiredo, J. (2017). How to Improve Computational Thinking: a Case Study. *Education in the Knowledge Society, 18*(4), pp. 35-51.

Roman-Gonzalez, M., Moreno-Leon, J., & Robles, G. (2017). Complementary Tools for Computational Thinking Assessment. In S.-c. Kong, J. Sheldon, & R.-y. Li (Ed.), *Conference Proceedings of International Conference on Computational Thinking Education 2017* (pp. 154-159). Hong Kong: The Education University of Hong Kong.

Roman-Gonzalez, M., Perez-Gonzalez, J.-C., & Jimenez-Fernandez, C. (2017). Which cognitive abilities underlie computational thinking? Criterion validity of the Computational Thinking Test. *Computers in Human Behavior, 72*, pp. 678-691.

Román-González, M., Pérez-González, J.-C., Moreno-León, J., & Robles, G. (2018, November). Can computational talent be detected? Predictive validity of the Computational Thinking Test. *International Journal of Child-Computer Interaction*(18), pp. 47-58.

Selby, C. (2015). Relationships: Computational Thinking, Pedagogy of Programming, and Bloom's Taxonomy. *WiPSCE '15 Proceedings of the Workshop in Primary and Secondary Computing Education* (pp. 80-87). London, United Kingdom: ACM.

Selby, C., & Woollard, J. (2014). Refining an understanding of computational thinking. pp. 1-23.

Snow, E., Rutstein, D., Bienkowski, M., & Xu, Y. (2017). Principled Assessment of Student Learning in High School Computer Science. *ICER '17 Proceedings of the 2017 ACM Conference on International Computing Education* (pp. 209-216). Tacoma, Washington, USA: ACM.

Snow, E., Tate, C., Rutstein, D., & Bienkowski, M. (2017). *Assessment Design Patterns for Computational Thinking Practices in Exploring Computer Science.* Menlo Park, CA: SRI International.

Steka, M., & Tsiatsos, T. (2018). Case Study: Integrating Computational Thinking into the Introductory Course of Computer Science via the Use of the Programming Language

Python. In M. Auer, & T. Tsiatsos (Ed.), *Interactive Mobile Communication Technologies and Learning.IMCL 2017.Advances in Intelligent Systems and Computing. 725*, pp. 531-541. Springer, Cham.

Tabesh, Y. (2017). Computational Thinking:A 21st Century Skill. *Olympiads in Informatics, 11*(Special Issue), pp. 65-70.

Tedre, M., & Denning, P. (2016). The long quest for computational thinking. *Koli Calling '16 Proceedings of the 16th Koli Calling International Conference on Computing Education Research* (pp. 120-129). Koli, Finland: ACM.

Tsarava, K., Moeller, K., Pinkwart, N., Butz, M., Trautwein, U., & Ninaus, M. (2017). Training Computational Thinking: Game-Based Unplugged and Plugged-in Activities in Primary School. *ECGBL 2017 11th European Conference on Game-Based Learning* (pp. 687-695). Graz, Austria: ACPIL.

Van Dyne, M., & Braun, J. (2014). Effectiveness of a Computational Thinking (CS0) Course on Student Analytical Skills. *SIGCSE '14 Proceedings of the 45th ACM technical symposium on Computer science education* (pp. 133-138). Atlanta, Georgia, USA: ACM.

Vieira, C., Yan, J., & Magana, A. (2015, July). Exploring Design Characteristics of Worked Examples to Support Programming and Algorithm Design. *The Journal of Computational Science Education, 6*, pp. 2-15.

Wing, J. (2006, March). Computational Thinking. *Communications of the ACM, 49*(3), pp. 33-35.

Yadav, A., Stephenson, C., & Hong, H. (2017, April). Computational Thinking for Teacher Education. *Communications of the ACM, 60*(4), pp. 55-62.

Yasar, O. (2018). A New Perspective on Computational Thinking. *COMMUNICATIONS OF THE ACM, 61*(7), pp. 33-39.

Απόφαση 72, ΦΕΚ 4273/27−09−2018. (n.d.).

Μαυρουδή, Ε., Πέτρου, Α., & Φεσάκης, Γ. (2014). Υπολογιστική Σκέψη: Εννοιολογική εξέλιξη, διεθνείς πρωτοβουλίες και προγράμματα σπουδών. *7ο Πανελλήνιο Συνέδριο Διδακτικής της Πληροφορικής*, (σσ. 111-120). Ρέθυμνο.

Νόμος 4310, ΦΕΚ 258/8−12−2014. (n.d.).

Ψυχάρης, Σ., Κοτζαμπασάκη, Ε., & Καλοβρέκτης, Κ. (2018, Μάρτιος). Υπολογιστική Σκέψη, Επιστημολογία των Μηχανικών και Υπολογιστική Παιδαγωγική: Μια πρόταση εισαγωγής του STEM στην εκπαίδευση. *ΕΚΠΑΙΔΕΥΣΗ & ΕΠΙΣΤΗΜΕΣ*(1), pp. 1-11.

βιβλία

Grover, S. (2017). Assessing Algorithmic and Computational Thinking in K-12: Lessons from a Middle School Classroom. In P. J. Rich, & C. B. Hodges (Eds.), *Emerging Research,*

Practice, and Policy on Computational Thinking (1st ed. ed., pp. 269-288). Springer Publishing Company.

Grover, S., & Pea, R. (2018). Computational Thinking:A Competency Whose Time Has Come. In S. Sentance, E. Barendsen, & C. Schulte (Eds.), *Computer Science Education: Perspectives on Teaching and Learning in School* (pp. 20-38). Bloomsbury Academic.

Ozcinar, H., Wong, G., & Ozturk, H. T. (2018). *Teaching Computational Thinking in Primary Education.* Hershey PA: IGI Global.

Polya, G. (1973). *How To Solve It.* Princeton, New Jersey: Princeton University Press.

Βακάλη, Α., Γιαννόπουλος, Η., Ιωαννίδης, Ν., Κοίλιας, Χ., Μάλαμας, Κ., Μανωλόπουλος, Ι., και συν. (2015). *Ανάπτυξη Εφαρμογών σε Προγραμματιστικό Περιβάλλον.* ΑΘΗΝΑ: Ι.Τ.Υ.Ε. "ΔΙΟΦΑΝΤΟΣ".

Δημητριάδης, Σ. Ν. (2015). *Θεωρίες Μάθησης & Εκπαιδευτικό Λογισμικό.* Αθήνα: Σύνδεσμος Ελληνικών Ακαδημαϊκών Βιβλιοθηκών.

Μανής , Γ. (2015). *Εισαγωγή στον Προγραμματισμό με Αρωγό τη Γλώσσα Python.* Ανάκτηση 2018, από kallipos.gr: https://repository.kallipos.gr/handle/11419/2745

Μπάγκος, Π. (2015). Υπολογιστικές Γραμματικές. Στο *Βιοπληροφορική* (σσ. 343-359). Αθήνα: Σύνδεσμος Ελληνικών Ακαδημαϊκών Βιβλιοθηκών.

Τοποθεσίες Web

CS 151: Computational Thinking: Science Applications. (n.d.). Retrieved 2018, από Colby Computer Science: http://cs.colby.edu/courses/F15/cs151s/index.php

CS 151: Computational Thinking: Visual Media. (n.d.). Retrieved 2018, από Colby Computer Science: http://cs.colby.edu/courses/F15/cs151/index.php

Csizmadia, A., Curzon, P., Dorling, M., Humphreys, S., Ng, T., Selby, C., et al. (2015). *Computational thinking A guide for teachers*. Retrieved 9 2017, from Computing At School: https://www.computingatschool.org.uk/computationalthinking

CSTA, & ISTE. (2011). *Operational definition of computational thinking for K-12 education*. Retrieved 9 2018, from http://www.iste.org/docs/ct-documents/computational-thinking-operational-definition-flyer.pdf

ebooks.edu.gr. (2013). Retrieved 10 2018, from http://ebooks.edu.gr/new/ps.php

Google for Education: Computational Thinking. (2011). Retrieved 8 2018, from Exploring Computational Thinking: https://edu.google.com/resources/programs/exploring-computational-thinking /

Hunsaker, E. (2016). *K-12 technology integration*. Retrieved 10 2018, from PressBooks: https://k12techintegration.pressbooks.com/chapter/integrating-computational-thinking /

The Royal Society. (2012). *Shut down or restart? The way forward for computing in UK schools pp.24-31.* Retrieved 8 2018, from https://royalsociety.org/~/media/education/computing-in-schools/2012-01-12-computing-in-schools.pdf

Wikipedia. (n.d.). *21st century skills*. Retrieved 10 27, 2018, from Wikipedia, The Free Encyclopedia: https://en.wikipedia.org/w/index.php?title=21st_century_skills&oldid=862681152

Wing, J. (2011, March). *The Link*. Retrieved Σεπτέμβριος 2018, from Carnegie-Mellon University School of Computer Science: https://www.cs.cmu.edu/link/research-notebook-computational-thinking-what-and-why

Wing, J. (2014, January 10). *COMPUTATIONAL THINKING BENEFITS SOCIETY*. Retrieved Σεπτέμβριος 2018, from Social Issues in Computing: http://socialissues.cs.toronto.edu/index.html%3Fp=279.html

Παράρτημα Ι

Προαιρετικές ασκήσεις στην Python

Ακολουθούν 10 προαιρετικές ασκήσεις στην γλώσσα προγραμματισμού Python. Καθεμία συνεισφέρει 2/100 έξτρα βαθμούς bonus στην τελική σας βαθμολογία (προσμετρώνται σε περίπτωση που έχετε λάβει την βάση στην τελική γραπτή εξέταση και στις υποχρεωτικές εργασίες).

Αν οι εργασίες είναι άριστες το συνολικό bonus θα είναι συνολικά 2*10= 20/100 = 2/10 μονάδες.

Σύντομες οδηγίες:

μελετάτε την «Εκφώνηση» και το «Παράδειγμα». Απαντάτε στην άσκηση

(α) συμπληρώνοντας τον πίνακα της παραγράφου "Σκεπτικό Επίλυσης"

(β) παραθέτοντας το πρόγραμμα στην παράγραφο "Απάντηση", όπως ζητείται εκεί.

Υποβάλλετε την εργασία, δηλ. τα (α) και (β) που υλοποιήσατε παραπάνω, στο elearning.auth.gr με ένα αρχείο με όνομα <το ΑΕΜ φοιτητή>ergasia_bonus.zip παράδειγμα ονόματος αρχείου 1245ergasia_bonus.zip), περιέχει:

- Το αρχείο docx (παράδειγμα ονόματος αρχείου 1245ergasia_bonus.docx)
- Τα αρχεία py για τις εργασίες.

Σε περίπτωση που ΔΕΝ απαντήσατε κάποια αργασία στις παραγράφους "Σκεπτικό Επίλυσης" και "Απάντηση" να αναφέρετε «Δεν απαντήθηκε».

Καλή επιτυχία!

Άσκηση 1 while και if

Μαθησιακοί στόχοι

Οι φοιτητές θα μάθουν να χρησιμοποιούν την αποδόμηση ενός προβλήματος προς επίλυση, την εστίαση σε συγκεκριμένα χαρακτηριστικά του προβλήματος και την αναγνώριση προτύπων με στόχο την αλγοριθμική επίλυση. Επίσης, θα εξοικειωθούν με τη γλώσσα προγραμματισμού python και συγκεκριμένα με τη χρήση βασικών συναρτήσεων και μεταβλητών καθώς επίσης και με τις δομές ελέγχου(if/elif/else) και επανάληψης(while).

Λέξεις κλειδιά

Python, δομή επανάληψης, δομή ελέγχου, συνθήκες, τελεστές σύγκρισης, if, if/elif/else, while υπολογιστική σκέψη, αφαίρεση, αποσύνθεση, αναγνώριση προτύπων.

Εκφώνηση

Υποθέτουμε ότι ένα σύστημα διαχείρισης αποθήκης είναι απαραίτητο ώστε ο χρήστης να ενημερώνεται για τον αριθμό των τεμαχίων που εισέρχονται και εξέρχονται από αυτήν καθώς και την συνολική διαθέσιμη ποσότητα.

Να υλοποιηθεί πρόγραμμα στην γλώσσα προγραμματισμού Python το οποίο:

1. Να εμφανίζει μήνυμα στον χρήστη: Σύστημα Διαχείρισης Αποθήκης. «Παρακαλώ πληκτρολογήστε εισαγωγή(Ι) ή εξαγωγή(Ε) τεμαχίων και την ποσότητα, για έξοδο πιέστε: Q»

2. Να δέχεται εισαγωγή δεδομένων από τον χρήστη, με τη μορφή (Χ τιμή) όπου το Χ θα μπορεί να είναι είτε I(Import) είτε E(Export). Για παράδειγμα παραλαβή 500 τεμαχίων εισάγεται στο σύστημα ως I 500 και E 200 αντίστοιχα, παραγγελία 200 τεμαχίων.
3. Να πραγματοποιείται έλεγχος για το μέγεθος της καταχώρισης το οποίο δεν μπορεί να ξεπερνάει τα 2000 τεμάχια είτε πρόκειται για εισαγωγή ή για εξαγωγή. Στην περίπτωση που ο χρήστης εισάγει τιμή που δεν βρίσκεται στα επιτρεπόμενα όρια, να εμφανίζεται μήνυμα σφάλματος. Επίσης μήνυμα σφάλματος να εμφανίζεται και στην περίπτωση που η ποσότητα παραγγελίας υπερβαίνει το σύνολο των διαθέσιμων αποθεμάτων.
4. Σε περίπτωση που ο χρήστης επιλέξει «Q» να πραγματοποιείται έξοδος από το πρόγραμμα και να εμφανίζονται τα συνολικά αποθέματα σε τεμάχια.

Παράδειγμα

Ακολουθεί σχετικό παράδειγμα όπου ο χρήστης:

- εισάγει 400 τεμάχια.
- εισάγει 2100 τεμάχια και εμφανίζεται μήνυμα σφάλματος.
- εισάγει 200 τεμάχια.
- εξάγει από την αποθήκη 650 και εμφανίζεται μήνυμα σφάλματος.
- εξάγει 300 τεμάχια.
- επιλέγει έξοδο όπου εμφανίζεται το συνολικό απόθεμα.

Η **Error! Reference source not found.**10 παρουσιάζει μια ενδεικτική εκτέλεση του προγράμματος

```
                    Σύστημα Διαχείρισης Αποθήκης
Παρακαλώ πληκτρολογήστε εισαγωγή(I) ή εξαγωγή(E) τεμαχίων και ποσότητα,
για έξοδο πιέστε: Q
I 400
Παρακαλώ πληκτρολογήστε εισαγωγή(I) ή εξαγωγή(E) τεμαχίων και ποσότητα,
για έξοδο πιέστε: Q
I 2100
Λάθος τιμή, η διαθέσιμη ποσότητα είναι: 400 τεμάχια και
η ποσότητα εισαγωγής ή εξαγωγης δεν μπορεί να ξεπερνάει τα: 2000 τεμάχια

Παρακαλώ πληκτρολογήστε εισαγωγή(I) ή εξαγωγή(E) τεμαχίων και ποσότητα,
για έξοδο πιέστε: Q
I 200
Παρακαλώ πληκτρολογήστε εισαγωγή(I) ή εξαγωγή(E) τεμαχίων και ποσότητα,
για έξοδο πιέστε: Q
E 650
Λάθος τιμή, η διαθέσιμη ποσότητα είναι: 600 τεμάχια και
η ποσότητα εισαγωγής ή εξαγωγης δεν μπορεί να ξεπερνάει τα: 2000 τεμάχια

Παρακαλώ πληκτρολογήστε εισαγωγή(I) ή εξαγωγή(E) τεμαχίων και ποσότητα,
για έξοδο πιέστε: Q
E 300
Παρακαλώ πληκτρολογήστε εισαγωγή(I) ή εξαγωγή(E) τεμαχίων και ποσότητα,
για έξοδο πιέστε: Q
Q
Η συνολική ποσότητα στην αποθήκη είναι: 300 τεμάχια
>>>
```

Εικόνα 10: Ενδεικτική Εκτέλεση του Προγράμματος Πρώτης Άσκησης

Σκεπτικό Επίλυσης

Συμπληρώστε τον παρακάτω πίνακα με τα βήματα υπολογιστικής σκέψης. Προσθέστε αν χρειαστεί κελιά.

Αφαίρεση	**Αποσύνθεση**	εντολές	**Πρότυπο**	εντολές

Απάντηση

Αναφέρετε το όνομα του αρχείου .py που έχετε επισυνάψει στο .zip αρχείο (π.χ. exercise01.py) και αντιστοιχεί στην λύση της άσκησης. Το αρχείο .py να είναι επαρκώς σχολιασμένο.

Άσκηση 2 while και if:

Μαθησιακοί στόχοι

Οι φοιτητές θα μάθουν να χρησιμοποιούν τις συνιστώσες της υπολογιστικής σκέψης για την προσέγγιση ενός προβλήματος προς επίλυση, με την αφαίρεση, την αποσύνθεση, την αναγνώριση προτύπων και τελικό αποτέλεσμα την αλγοριθμική επίλυση. Επίσης θα εξοικειωθούν με τη γλώσσα προγραμματισμού python και συγκεκριμένα με τη χρήση βασικών συναρτήσεων και μεταβλητών, με τις δομές ελέγχου(if/elif/else) και επανάληψης(while) καθώς και με την έννοια του αμυντικού προγραμματισμού.

Λέξεις κλειδιά

Python, δομή επανάληψης, δομή ελέγχου, συνθήκες, τελεστές σύγκρισης, if, if/elif/else, while υπολογιστική σκέψη, αφαίρεση, αποσύνθεση, αναγνώριση προτύπων.

Εκφώνηση

Υποθέτουμε ότι σε μια ανεγερθείσα οικοδομή οι ιδιοκτήτες αποφασίζουν να διαθέσουν ορισμένα από τα διαμερίσματα προς πώληση, άλλα προς ενοικίαση και κάποια να τα κρατήσουν για ιδιοκατοίκηση. Η οικοδομή αποτελείται από 42 διαμερίσματα και ο αριθμός των ορόφων είναι άρτιος. Μετά από συνεννόηση αποφασίζουν να κρατήσουν τα 4 διαμερίσματα για ιδιοκατοίκηση, τα 13 να τα διαθέσουν προς ενοικίαση, και τα υπόλοιπα να τα διαθέσουν προς πώληση.

Οι λίστες με τον αριθμό των διαμερισμάτων διαμορφώνονται ως εξής:

- Διαμερίσματα προς Πώληση:
  1, 2, 4, 5, 7, 8, 10, 11, 13, 14, 15, 16, 17, 19, 21, 23, 27, 29, 30, 32, 35, 36, 38, 40, 42
- Διαμερίσματα προς Ενοικίαση:
  3, 6, 9, 12, 18, 20, 24, 26, 28, 31, 33, 37, 39
- Διαμερίσματα προς Ιδιοκατοίκηση:
  22, 25, 34, 41

Να υλοποιηθεί πρόγραμμα στην γλώσσα προγραμματισμού Python το οποίο:

1. Να εμφανίζειτο παρακάτω μήνυμα στον χρήστη «Διαχείριση Πολυκατοικίας»:
   «πληκτρολογήστε τον αριθμό διαμερίσματος που θέλετε να ελέγξετε, για έξοδο πληκτρολογήστε Ε».

2. Να δέχεται εισαγωγή δεδομένων από τον χρήστη είτε (α) κάποιον αριθμό διαμερίσματος είτε (β) το γράμμα «Ε» (οπότε και θα πραγματοποιείται έξοδος από το πρόγραμμα).
3. Να πραγματοποιείται έλεγχος ότι η τιμή που αντιστοιχεί στον αριθμό διαμερίσματος βρίσκεται στα επιτρεπόμενα όρια. Σε αντίθετη περίπτωση να εμφανίζεται μήνυμα σφάλματος.
4. Να διεξάγεται αναζήτηση για τον αριθμό διαμερίσματος και να εμφανίζεται εάν το διαμέρισμα είναι διαθέσιμο προς ενοικίαση ή προς πώληση και την αντίστοιχη τιμή ή αν πρόκειται για ιδιοκατοίκηση οπότε να εμφανίζεται και το κατάλληλο μήνυμα.
    - Οι τιμές πώλησης ανά 7 διαμερίσματα είναι:
      75000€, 75000€, 95000€, 95000€, 11500€ και 115000€
    - Οι τιμές ενοικίασης ανά 7 διαμερίσματα είναι:
      400€/μήνα, 400€/μήνα, 450€/μήνα, 450€/μήνα, 500€/μήνα, 500€/μήνα.

Παράδειγμα

Ακολουθεί σχετικό παράδειγμα όπου ο χρήστης:

- εισάγει τον αριθμό διαμερίσματος 6.
- εισάγει τον αριθμό διαμερίσματος 50.
- εισάγει τον αριθμό διαμερίσματος 16.
- εισάγει τον αριθμό διαμερίσματος 22.
- εισάγει τον αριθμό διαμερίσματος 35.
- Επιλέγει έξοδο από το πρόγραμμα.

Η **Error! Reference source not found.**11 παρουσιάζει μια ενδεικτική εκτέλεση του προγράμματος

```
Διαχείριση Πολυκατοικίας
πληκτρολογήστε τον αριθμό διαμερίσματος που θέλετε να ελέγξετε,
για έξοδο πληκτρολογήστε Ε
6
διαμέρισμα προς ενοικίαση με τιμή:400€/μήνα

πληκτρολογήστε τον αριθμό διαμερίσματος που θέλετε να ελέγξετε,
για έξοδο πληκτρολογήστε Ε
50
Ο αριθμός διαμερίσματος είναι λανθασμένος

πληκτρολογήστε τον αριθμό διαμερίσματος που θέλετε να ελέγξετε,
για έξοδο πληκτρολογήστε Ε
16
διαμέρισμα προς πώληση με τιμή:95000€

πληκτρολογήστε τον αριθμό διαμερίσματος που θέλετε να ελέγξετε,
για έξοδο πληκτρολογήστε Ε
22
διαμέρισμα προς ιδιοκατοίκηση
πληκτρολογήστε τον αριθμό διαμερίσματος που θέλετε να ελέγξετε,
για έξοδο πληκτρολογήστε Ε
35
διαμέρισμα προς πώληση με τιμή:115000€

πληκτρολογήστε τον αριθμό διαμερίσματος που θέλετε να ελέγξετε,
για έξοδο πληκτρολογήστε Ε
Ε
>>>
```

Ln: 33 Col: 4

Εικόνα 11: Ενδεικτική Εκτέλεση του Προγράμματος Δεύτερης Άσκησης

Σκεπτικό Επίλυσης

Συμπληρώστε τον παρακάτω πίνακα με τα βήματα υπολογιστικής σκέψης. Προσθέστε αν χρειαστεί κελιά.

Αφαίρεση	**Αποσύνθεση**	εντολές	**Πρότυπο**	εντολές

Απάντηση

Αναφέρετε το όνομα του αρχείου .py που έχετε επισυνάψει στο .zip αρχείο (π.χ. exercise02.py) και αντιστοιχεί στην λύση της άσκησης. Το αρχείο .py να είναι επαρκώς σχολιασμένο.

Άσκηση 3 for:

Μαθησιακοί στόχοι

Οι φοιτητές θα εξασκηθούν στα βήματα της υπολογιστικής σκέψης χρησιμοποιώντας τη δομή ελέγχου και εμβαθύνοντας στις δομές επανάληψης και στους βρόχους, συγκεκριμένα με τη δομή for, που αποτελεί επαναληπτική διαδικασία για συγκεκριμένο αριθμό βημάτων.

Λέξεις κλειδιά

Python, δομή επανάληψης, δομή ελέγχου, συνθήκες, τελεστές σύγκρισης, βήμα επανάληψης, if, if/elif/else, for, range, while υπολογιστική σκέψη, αφαίρεση, αποσύνθεση, αναγνώριση προτύπων.

Εκφώνηση

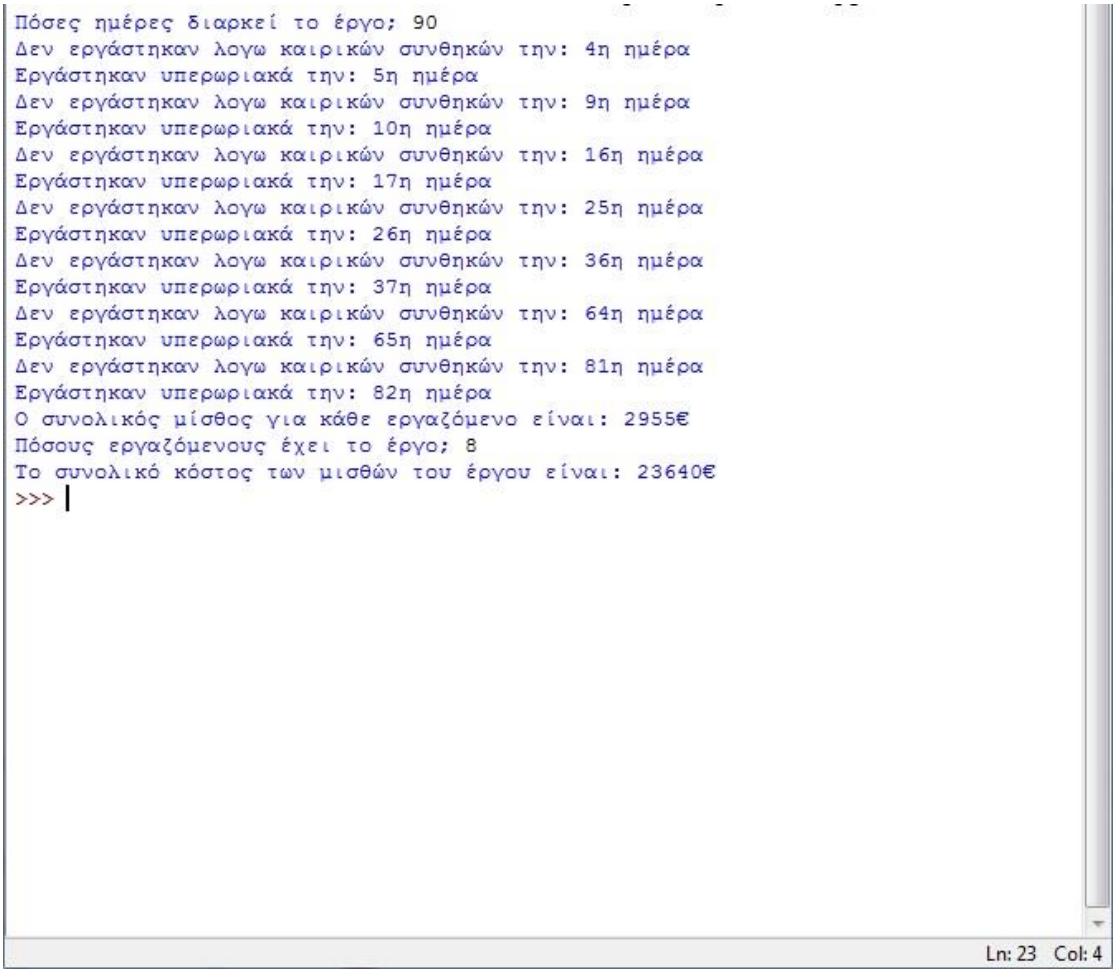
Υποθέτουμε ότι μια κατασκευαστική εταιρία αναλαμβάνει να διεκπεραιώσει την κατασκευή ενός δρόμου 50 χιλιομέτρων σε ένα χρονικό διάστημα Χ ημερών. Εξαιτίας των καιρικών συνθηκών κάποιες ημέρες δεν ήταν δυνατόν να εκτελεστούν εργασίες. Για να τηρηθεί το χρονικό πλαίσιο παράδοσης του έργου απαιτείται υπερωριακή εργασία.

Να υλοποιηθεί πρόγραμμα στην γλώσσα προγραμματισμού Python το οποίο:

1. Να δέχεται τον αριθμό των ημερών με είσοδο των δεδομένων από τον χρήστη. Στην προκειμένη περίπτωση είναι Χ = 90 ημέρες. Σε περίπτωση που το πλήθος των ημερών είναι διαφορετικό από 90 να ξαναεμφανίζεται η ερώτηση διάρκειας του έργου.
2. Να υπολογίζει και να εμφανίζει τις ημέρες που οι εργασίες δεν εκτελέστηκαν λόγω καιρικών συνθηκών, λαμβάνοντας υπόψη ότι διεξάγονται την αμέσως επόμενη ημέρα ώστε να τηρηθεί το χρονικό πλαίσιο. Οι ημέρες που έγιναν οι υπερωρίες είναι: 5η , 10η , 17η , 26η , 37η , 65η , 82η.
3. Να υπολογίζει και να εμφανίζει τη συνολική αμοιβή κάθε εργαζομένου που υπολογίζεται με ημερομίσθιο 30€ για κάθε ημέρα κανονικής απασχόλησης και 45€ για κάθε ημέρα υπερωριακής απασχόλησης. Επίσης να ληφθεί υπόψη ότι η αμοιβή για τις Κυριακές πληρώνεται με ημερομίσθιο 60€ ενώ η πρώτη ημέρα έναρξης των εργασιών ήταν Δευτέρα.
4. Να δέχεται τον αριθμό των εργαζομένων από τον χρήστη, να ελέγχει ότι είναι στα επιτρεπτά όρια και να εμφανίζει το συνολικό κόστος αμοιβής όλων των εργαζομένων. Το συνολικό κόστος αμοιβής δεν θα πρέπει να ξεπερνάει τα 26.595€ και το έργο δεν μπορεί να ολοκληρωθεί έγκαιρα με λιγότερους από 7 εργαζόμενους.

Παράδειγμα

Η **Error! Reference source not found.**2 παρουσιάζει μια ενδεικτική εκτέλεση του προγράμματος



Εικόνα 12: Ενδεικτική Εκτέλεση του Προγράμματος Τρίτης Άσκησης

Σκεπτικό Επίλυσης

Συμπληρώστε τον παρακάτω πίνακα με τα βήματα υπολογιστικής σκέψης. Προσθέστε αν χρειαστεί κελιά.

Αφαίρεση	**Αποσύνθεση**	εντολές	**Πρότυπο**	εντολές

Απάντηση

Αναφέρετε το όνομα του αρχείου .py που έχετε επισυνάψει στο .zip αρχείο (π.χ. exercise03.py) και αντιστοιχεί στην λύση της άσκησης. Το αρχείο .py να είναι επαρκώς σχολιασμένο.

Άσκηση 4 for:

Μαθησιακοί στόχοι

Οι φοιτητές θα εξοικειωθούν περαιτέρω με τις δομές επανάληψης και ελέγχου, τις έννοιες της αφαίρεσης, αποσύνθεσης και αναγνώρισης προτύπων με σκοπό τη δημιουργία αλγορίθμου για την επίλυση συγκεκριμένου προβλήματος.

Λέξεις κλειδιά

Python, δομή επανάληψης, δομή ελέγχου, συνθήκες, τελεστές σύγκρισης, βήμα επανάληψης, if, if/elif/else, for, range while υπολογιστική σκέψη, αφαίρεση, αποσύνθεση, αναγνώριση προτύπων.

Εκφώνηση

Σε ένα υποθετικό παιχνίδι ερωτήσεων διαγωνίζονται δυο αντίπαλοι και σε κάθε ερώτηση αναλογούν 6 πόντοι. Στην προσπάθεια να καταμετρηθούν οι πόντοι των δυο αντιπάλων αυτοματοποιήθηκε η διαδικασία. Οι πόντοι καταμερίζονται ως εξής:

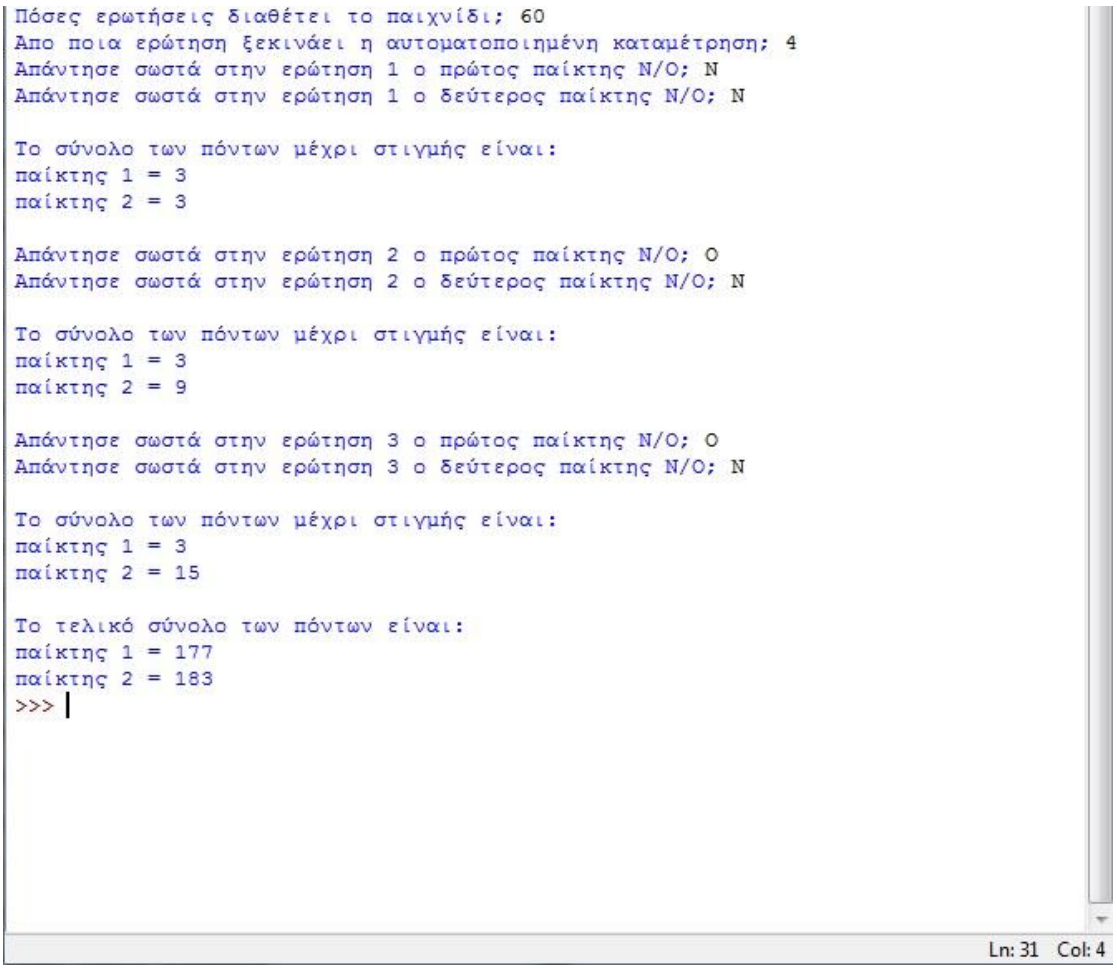
- Στην περίπτωση που ο ένας από τους δυο αντιπάλους απαντήσει σωστά και ο άλλος λάθος και οι 6 πόντοι της ερώτησης προστίθενται στο σύνολό του.
- Στην περίπτωση που και οι δυο αντίπαλοι απαντήσουν σωστά μοιράζονται τους πόντους της ερώτησης(3 για τον παίκτη 1 και 3 για τον παίκτη 2)
- Στην τελευταία περίπτωση που και οι δυο παίκτες απαντήσουν λάθος κανένας από τους παίκτες δεν κερδίζει πόντους και συνεχίζουν στην επόμενη ερώτηση.

Να υλοποιηθεί πρόγραμμα στην γλώσσα προγραμματισμού Python το οποίο:

1. Να εμφανίζει μήνυμα στον χρήστη για το α) πόσες ερωτήσεις θα διαθέτει το παιχνίδι και β) από ποια ερώτηση θα ξεκινάει η αυτοματοποιημένη καταμέτρηση. Και για τις δυο ερωτήσεις να δέχεται εισαγωγή δεδομένων από τον χρήστη.
2. Εάν η αυτοματοποιημένη καταμέτρηση των πόντων δεν ξεκινάει από την πρώτη ερώτηση να εμφανίζει ερώτηση στον χρήστη εάν οι παίκτες απάντησαν σωστά στην προηγούμενη ερώτηση. Ανάλογα με την απάντηση (Ναι/Οχι) να εμφανίζεται το σύνολο των πόντων του κάθε παίκτη.
3. Να υπολογιστούν οι πόντοι των δυο παικτών, για 60 ερωτήσεις εάν α) στην πρώτη ερώτηση απαντήσουν και οι δυο σωστά, β) στη δεύτερη και την τρίτη ερώτηση απαντήσει λάθος ο πρώτος παίκτης και σωστά ο δεύτερος και γ) στις υπόλοιπες ερωτήσεις θεωρούμε ότι ο πρώτος παίκτης απαντάει ότι απάντησε ο δεύτερος παίκτης στην προηγούμενη ερώτηση. Για παράδειγμα εάν ο δεύτερος παίκτης απαντήσει λάθος στην ερώτηση 5 στην επόμενη ερώτηση θα απαντήσει λάθος ο πρώτος παίκτης. Ο δεύτερος παίκτης απαντάει λάθος όταν ο αριθμός της ερώτησης είναι πολλαπλάσιο του 5.
4. Να εμφανίζει το τελικό σύνολο των πόντων του κάθε παίκτη ξεχωριστά.

Παράδειγμα

Η **Error! Reference source not found.**3 παρουσιάζει μια ενδεικτική εκτέλεση του προγράμματος



Εικόνα 13: Ενδεικτική Εκτέλεση του Προγράμματος Τέταρτης Άσκησης

Σκεπτικό Επίλυσης

Συμπληρώστε τον παρακάτω πίνακα με τα βήματα υπολογιστικής σκέψης. Προσθέστε αν χρειαστεί κελιά.

Αφαίρεση	**Αποσύνθεση**	εντολές	**Πρότυπο**	εντολές

Απάντηση

Αναφέρετε το όνομα του αρχείου .py που έχετε επισυνάψει στο .zip αρχείο (π.χ. exercise04.py) και αντιστοιχεί στην λύση της άσκησης. Το αρχείο .py να είναι επαρκώς σχολιασμένο.

Άσκηση 5 list

Μαθησιακοί στόχοι

Οι φοιτητές θα εξοικειωθούν με τις δομές ελέγχου και επανάληψης σε συνδυασμό με μια πιο σύνθετη και ευέλικτη δομή δεδομένων, τις λίστες, που χρησιμοποιούνται για αποθήκευση και προσπέλαση δεδομένων, με σκοπό την αλγοριθμική επίλυση του προβλήματος και με την εφαρμογή των χαρακτηριστικών της υπολογιστικής σκέψης.

Λέξεις κλειδιά

Python, δομή επανάληψης, δομή ελέγχου, συνθήκες, τελεστές σύγκρισης, λίστες, δεικτοδότηση, περιγραφή λίστας, τομή λίστας, if, if/elif/else, for, while υπολογιστική σκέψη, αφαίρεση, αποσύνθεση, αναγνώριση προτύπων.

Εκφώνηση

Υποθέτουμε ότι ένα πανεπιστήμιο διαθέτει μεταπτυχιακά προγράμματα από τρεις διαφορετικές σχολές και έχει κοινή γραμματεία. Κατά τη διάρκεια των εγγραφών στο κάθε τμήμα, η γραμματεία είναι υποχρεωμένη να προμηθεύει τους φοιτητές με ένα όνομα χρήστη(username) και έναν κωδικό(password) για να μπορέσουν να εισέλθουν στο ηλεκτρονικό σύστημα διαχείρισης μαθημάτων.

Να υλοποιηθεί πρόγραμμα στη γλώσσα προγραμματισμού Python το οποίο:

1. Να δέχεται είσοδο δεδομένων από τον χρήστη για τον αριθμό των εισακτέων των προηγούμενων ετών όπως επίσης και τον αριθμό των εισακτέων του τρέχοντος έτους, του κάθε τμήματος, λαμβάνοντας υπόψη ότι για το συγκεκριμένο έτος οι φοιτητές που εγγράφηκαν στα μεταπτυχιακά προγράμματα ήταν: 20 στο τμήμα «Οργάνωση και Διοίκηση Επιχειρήσεων», 20 στο τμήμα «λογιστικής και χρηματοοικονομικής» και 20 στο τμήμα «Εφαρμοσμένης Πληροφορικής», ενώ οι εγγραφές των προηγούμενων ετών ήταν 1000 φοιτητές στο κάθε τμήμα.
2. Να δημιουργεί το όνομα χρήστη που θα αποτελείται από τα αρχικά του τμήματος στο οποίο ανήκει ο κάθε φοιτητής και τον αντίστοιχο αριθμό εισαγωγής. Παράδειγμα ο πρώτος φοιτητής στο τμήμα Οργάνωσης και Διοίκησης Επιχειρήσεων θα έχει όνομα χρήστη ΟΔΕ1001. Αντίστοιχα για τα υπόλοιπα δύο τμήματα θα είναι ΕΠ1001 και ΛΧ1001.
3. Να δημιουργεί τον κωδικό (password) για το κάθε όνομα χρήστη ο οποίος θα αποτελείται από 6 χαρακτήρες, τρία αγγλικά κεφαλαία γράμματα από το A έως το Z και τρείς ψευδοτυχαίους αριθμούς από το 1 έως το 9 κατ' εναλλαγή.
4. Να δέχεται είσοδο δεδομένων από τον χρήστη, τα αρχικά της σχολής και το ονοματεπώνυμο του φοιτητή, χωρισμένα με κόμμα ή το «Ε» για έξοδο από το πρόγραμμα και
5. Τελικά όταν επιλέγεται από τον χρήστη έξοδος από το πρόγραμμα, να εμφανίζει την οριστική λίστα που θα περιλαμβάνει τα ακόλουθα στοιχεία: όνομα χρήστη, ονοματεπώνυμο φοιτητή και κωδικό εισόδου.

Παράδειγμα

Ακολουθεί σχετικό παράδειγμα όπου ο χρήστης:

- Εισάγει τον φοιτητή της ΟΔΕ Πέτρο Πέτρου.
- Εισάγει τον φοιτητή της ΛΧ Γιάννη Γιάννου.
- Εισάγει τον φοιτητή της ΕΠ Χρήστο Νάκα.
- Εισάγει την φοιτήτρια της ΕΠ Χριστίνα Καρρά.
- Εισάγει την φοιτήτρια της ΛΧ Άννα Παπαδοπούλου.
- Εισάγει την φοιτήτρια της ΟΔΕ Μαρία Στρατή.
- Επιλέγει έξοδο από το πρόγραμμα.

Η **Error! Reference source not found.**4 παρουσιάζει μια ενδεικτική εκτέλεση του προγράμματος

```
Πόσοι φοιτητες εγγράφηκαν τα προηγούμενα έτη; 1000
Πόσοι φοιτητες αντιστοιχούν στο κάθε τμήμα; 20
Παρακαλώ εισάγετε Τμήμα,Ονοματεπώνυμο φοιτητή,
για έξοδο πιέστε Ε: ΟΔΕ,Πέτρος Πέτρου
Παρακαλώ εισάγετε Τμήμα,Ονοματεπώνυμο φοιτητή,
για έξοδο πιέστε Ε: ΛΧ,Γιάννης Γιάννου
Παρακαλώ εισάγετε Τμήμα,Ονοματεπώνυμο φοιτητή,
για έξοδο πιέστε Ε: ΕΠ,Χρήστος Νάκας
Παρακαλώ εισάγετε Τμήμα,Ονοματεπώνυμο φοιτητή,
για έξοδο πιέστε Ε: ΕΠ,Χριστίνα Καρρά
Παρακαλώ εισάγετε Τμήμα,Ονοματεπώνυμο φοιτητή,
για έξοδο πιέστε Ε: ΛΧ,Άννα Παπαδοπούλου
Παρακαλώ εισάγετε Τμήμα,Ονοματεπώνυμο φοιτητή,
για έξοδο πιέστε Ε: ΟΔΕ,Μαρία Στρατή
Παρακαλώ εισάγετε Τμήμα,Ονοματεπώνυμο φοιτητή,
για έξοδο πιέστε Ε: Ε
['ΟΔΕ1001,Πέτρος Πέτρου,password: D9C2A2', 'ΛΧ1001,Γιάννης Γιάννου,password: V9H7B9'
, 'ΕΠ1001,Χρήστος Νάκας,password: X7Z9C2', 'ΕΠ1002,Χριστίνα Καρρά,password: V4V0V4',
'ΛΧ1002,Άννα Παπαδοπούλου,password: B5L7A4', 'ΟΔΕ1002,Μαρία Στρατή,password: V3W9R7'
]
>>>
```

Ln: 22 Col: 4

Εικόνα 14: Ενδεικτική Εκτέλεση του Προγράμματος Πέμπτης Άσκησης

Σκεπτικό Επίλυσης

Συμπληρώστε τον παρακάτω πίνακα με τα βήματα υπολογιστικής σκέψης. Προσθέστε αν χρειαστεί κελιά.

Αφαίρεση	**Αποσύνθεση**	εντολές	**Πρότυπο**	εντολές

Απάντηση

Αναφέρετε το όνομα του αρχείου .py που έχετε επισυνάψει στο .zip αρχείο (π.χ. exercise05.py) και αντιστοιχεί στην λύση της άσκησης. Το αρχείο .py να είναι επαρκώς σχολιασμένο.

Άσκηση 6 list

Μαθησιακοί στόχοι

Οι φοιτητές θα εμβαθύνουν στη χρήση λιστών και ιδιαίτερα στη δημιουργία λίστας με περιγραφή λίστας και λίστας δύο διαστάσεων(ή λίστα λιστών). Για την επίτευξη του τελικού σκοπού δηλαδή της αλγοριθμικής επίλυσης, θα συνδυαστούν με τις δομές επανάληψης και ελέγχου καθώς επίσης και με τα βήματα της υπολογιστικής σκέψης.

Λέξεις κλειδιά

Python, δομή επανάληψης, δομή ελέγχου, συνθήκες, τελεστές σύγκρισης, λίστες, λίστες λιστών, δεικτοδότηση, περιγραφή λίστας, τομή λίστας, if, if/elif/else, for, while υπολογιστική σκέψη, αφαίρεση, αποσύνθεση, αναγνώριση προτύπων.

Εκφώνηση

Οι καταθέσεις προθεσμίας αφορούν κλειστές καταθέσεις χρημάτων για συγκεκριμένο χρονικό διάστημα με σκοπό την αύξηση του αρχικού κεφαλαίου κατά το ποσό της απόδοσης. Υποθέτουμε ότι ένα τραπεζικό ίδρυμα ανακοινώνει καθημερινά τα επιτόκια για τις προθεσμιακές καταθέσεις για τα εξής χρονικά διαστήματα: 1 μήνας, 3 μήνες, 6 μήνες, 12 μήνες.

Να υλοποιηθεί πρόγραμμα στη γλώσσα προγραμματισμού Python το οποίο:

1. Να δέχεται είσοδο δεδομένων από τον χρήστη για το ποσό της προθεσμιακής κατάθεσης.
2. Να δημιουργεί και να εμφανίζει τον πίνακα με τα επιτόκια, όπου στην πρώτη στήλη θα ορίζεται η ημέρα της εβδομάδας και στις υπόλοιπες τέσσερις τα επιτόκια, για τα χρονικά διαστήματα που αναφέρθηκαν παραπάνω, λαμβάνοντας υπόψη ότι τα επιτόκια θα είναι ψευδοτυχαίοι δεκαδικοί αριθμοί από το 0,03 έως το 0,09.
3. Να υπολογίζει το μεγαλύτερο επιτόκιο από τον πίνακα και σε περίπτωση που υπάρχει το μέγιστο επιτόκιο για διαφορετικά χρονικά διαστήματα την ίδια ημέρα να επιλέγει το μεγαλύτερο χρονικό διάστημα. Σε περίπτωση που υπάρχει το μέγιστο επιτόκιο σε διαφορετικές ημέρες να επιλέγει την ημέρα που βρίσκεται προς το τέλος της εβδομάδας χωρίς να λαμβάνει υπόψη το χρονικό διάστημα.
4. Να εμφανίζει το μεγαλύτερο επιτόκιο καθώς και την ημέρα και το χρονικό διάστημα που αυτό αντιστοιχεί.
5. Να υπολογίζει και να εμφανίζει το σύνολο των τόκων που θα αποφέρει με βάση το ποσό. Ο υπολογισμός του ποσού γίνεται: Ποσό x Επιτόκιο x χρονικό διάστημα. Για παράδειγμα ποσό 20000 με επιτόκιο 0,09 για 3 μήνες υπολογίζεται (20000 x 0.09) x 3/12=450.
6. Να υπολογίζει και να εμφανίζει για ένα διάστημα τεσσάρων εβδομάδων τις ημέρες που είχαν το μεγαλύτερο επιτόκιο με βάση τον ορισμό του ερωτήματος 3.

Παράδειγμα

Η **Error! Reference source not found.**5 παρουσιάζει μια ενδεικτική εκτέλεση του προγράμματος

```
Επιτόκια Προθεσμικών καταθέσεων
Πληκτρολογηστε το χρηματικό ποσό: 20000
επιτόκια :1 μήν 3 μήν 6 μήν 12 μήν
Δευτέρα  :[0.04, 0.06, 0.08, 0.08]
Τρίτη    :[0.06, 0.08, 0.08, 0.04]
Τετάρτη  :[0.09, 0.05, 0.04, 0.03]
Πέμπτη   :[0.09, 0.09, 0.05, 0.04]
Παρασκευή:[0.06, 0.08, 0.07, 0.03]

Το μέγιστο επιτόκιο είναι:0.09 και η προθεσμιακή κατάθεση θα ξεκινήσει:Πέμπτη
και θα διαρκέσει για 3 μήνες
Το σύνολο των τόκων που θα επιστραφεί είναι: 450.0

Δευτέρα  :[0.03, 0.08, 0.07, 0.05]
Τρίτη    :[0.07, 0.09, 0.05, 0.08]
Τετάρτη  :[0.07, 0.06, 0.06, 0.04]
Πέμπτη   :[0.09, 0.08, 0.05, 0.06]
Παρασκευή:[0.06, 0.05, 0.08, 0.04]

Το μέγιστο επιτόκιο είναι:0.09 και η προθεσμιακή κατάθεση θα ξεκινήσει:Πέμπτη
και θα διαρκέσει για 1 μήνας
Το σύνολο των τόκων που θα επιστραφεί είναι: 150.0

Δευτέρα  :[0.05, 0.06, 0.03, 0.05]
Τρίτη    :[0.06, 0.04, 0.06, 0.06]
Τετάρτη  :[0.09, 0.06, 0.05, 0.08]
Πέμπτη   :[0.07, 0.09, 0.03, 0.06]
Παρασκευή:[0.05, 0.08, 0.04, 0.07]

Το μέγιστο επιτόκιο είναι:0.09 και η προθεσμιακή κατάθεση θα ξεκινήσει:Πέμπτη
και θα διαρκέσει για 3 μήνες
Το σύνολο των τόκων που θα επιστραφεί είναι: 450.0

Δευτέρα  :[0.09, 0.08, 0.08, 0.07]
Τρίτη    :[0.03, 0.08, 0.07, 0.06]
Τετάρτη  :[0.04, 0.06, 0.07, 0.08]
Πέμπτη   :[0.05, 0.06, 0.05, 0.08]
Παρασκευή:[0.07, 0.04, 0.04, 0.04]

Το μέγιστο επιτόκιο είναι:0.09 και η προθεσμιακή κατάθεση θα ξεκινήσει:Δευτέρα
και θα διαρκέσει για 1 μήνας
Το σύνολο των τόκων που θα επιστραφεί είναι: 150.0

Οι ημέρες με τα μεγαλύτερα επιτόκια σε διάστημα 4 εβδομάδων ήταν:
 Δ  Τ  Τ  Π  Π
[1, 0, 0, 3, 0]
>>>
                                                                  Ln: 51  Col: 4
```

Εικόνα 15: Ενδεικτική Εκτέλεση του Προγράμματος Έκτης Άσκησης

Σκεπτικό Επίλυσης

Συμπληρώστε τον παρακάτω πίνακα με τα βήματα υπολογιστικής σκέψης. Προσθέστε αν χρειαστεί κελιά.

Αφαίρεση	**Αποσύνθεση**	εντολές	**Πρότυπο**	εντολές

Απάντηση

Αναφέρετε το όνομα του αρχείου .py που έχετε επισυνάψει στο .zip αρχείο (π.χ. exercise06.py) και αντιστοιχεί στην λύση της άσκησης. Το αρχείο .py να είναι επαρκώς σχολιασμένο.

Άσκηση 7 dictionary

Μαθησιακοί στόχοι

Οι φοιτητές θα μάθουν να χρησιμοποιούν τα λεξικά που αποτελούν μια δομή αντιστοίχισης κλειδιού-τιμής καθώς επίσης και τη χρησιμοποίηση μιας πιο σύνθετης δομής που προκύπτει από τον συνδυασμό λίστας και λεξικών με τελικό στόχο την αλγοριθμική επίλυση του προβλήματος χρησιμοποιώντας τις συνιστώσες της υπολογιστικής σκέψης.

Λέξεις κλειδιά

Python, δομή επανάληψης, δομή ελέγχου, δομή αντιστοίχισης, συνθήκες, τελεστές σύγκρισης, λίστες, περιγραφή λίστας, λεξικό, κλειδί – τιμή, λίστες λεξικών, if , for, while υπολογιστική σκέψη, αφαίρεση, αποσύνθεση, αναγνώριση προτύπων.

Εκφώνηση

Υποθέτουμε ότι ο ιδιοκτήτης ενός καταστήματος που διαθέτει προς πώληση μεταχειρισμένα αυτοκίνητα θα ήθελε να γνωρίζει σε πόσα και σε ποία αυτοκίνητα μπορεί να προσφέρει έκπτωση με κριτήριο τα χιλιόμετρα καθώς επίσης και ποια είναι η οικονομικότερη διαθέσιμη επιλογή.

Να υλοποιηθεί πρόγραμμα στη γλώσσα προγραμματισμού Python το οποίο:

1. Να δέχεται είσοδο δεδομένων από το χρήστη για το πλήθος των αυτοκινήτων που θα διαθέτει το κατάστημα, θεωρώντας ότι η τιμή που εισάγει ο χρήστης είναι σωστή.
2. Να αποθηκεύει τα εξής χαρακτηριστικά για κάθε αυτοκίνητο:
   - Την μάρκα και μοντέλο τα οποία θα εισάγονται από το χρήστη, θεωρώντας ότι τα στοιχεία που εισάγει ο χρήστης είναι σωστά.
   - Τα χιλιόμετρα που θα έχει το αυτοκίνητο, που θα είναι ψευδοτυχαίοι αριθμοί μεταξύ 10000 και 60000.
   - Την τιμή του αυτοκινήτου που θα είναι ψευδοτυχαίοι αριθμοί μεταξύ 5000 και 15000.
   - Τις πινακίδες του αυτοκινήτου, οι οποίες θα αποτελούνται από τρία γράμματα την παύλα(-) και τέσσερις ψευδοτυχαίους αριθμούς. Τα γράμματα της πινακίδας θα πρέπει να ανήκουν στην εξής λίστα:A, B, E, Z, H, I, K, M, N, O, P, T, Y, X.
3. Να εμφανίζει τα διαθέσιμα αυτοκίνητα ένα σε κάθε γραμμή με τα παραπάνω χαρακτηριστικά.
4. Να υπολογίζει και να εμφανίζει τον μέσο όρο των χιλιομέτρων των αυτοκινήτων.
5. Να υπολογίζει και να εμφανίζει έκπτωση της τάξης του 10% σε κάθε αυτοκίνητο που τα χιλιόμετρά του είναι παραπάνω από τον μέσο όρο.
6. Να υπολογίζει και να εμφανίζει το αυτοκίνητο με την χαμηλότερη τιμή και την αντίστοιχη πινακίδα του.

Παράδειγμα

Ακολουθεί σχετικό παράδειγμα όπου ο χρήστης:

- Εισάγει το μοντέλο Ford Focus.
- Εισάγει το μοντέλο Ford Fiesta.
- Εισάγει το μοντέλο Opel Astra.
- Εισάγει το μοντέλο Opel Corsa.

- Εισάγει το μοντέλο Volkswagen Polo.
- Εισάγει το μοντέλο Volkswagen Golf.
- Εισάγει το μοντέλο Peugeot 208.
- Εισάγει το μοντέλο Peugeot 308.
- Εισάγει το μοντέλο Toyota Auris.
- Εισάγει το μοντέλο Toyota Yaris.

Η **Error! Reference source not found.**6 παρουσιάζει μια ενδεικτική εκτέλεση του προγράμματος

```
Πόσα αυτοκίνητα διαθέτει το κατάστημα; 10
Παρακαλώ εισάγετε την μάρκα και το μοντέλο: Ford Focus
Παρακαλώ εισάγετε την μάρκα και το μοντέλο: Ford Fiesta
Παρακαλώ εισάγετε την μάρκα και το μοντέλο: Opel Astra
Παρακαλώ εισάγετε την μάρκα και το μοντέλο: Opel Corsa
Παρακαλώ εισάγετε την μάρκα και το μοντέλο: Volkswagen Polo
Παρακαλώ εισάγετε την μάρκα και το μοντέλο: Volkswagen Golf
Παρακαλώ εισάγετε την μάρκα και το μοντέλο: Peugeot 208
Παρακαλώ εισάγετε την μάρκα και το μοντέλο: Peugeot 308
Παρακαλώ εισάγετε την μάρκα και το μοντέλο: Toyota Auris
Παρακαλώ εισάγετε την μάρκα και το μοντέλο: Toyota Yaris

Μάρκα:Ford Focus     , Χιλιόμετρα:50946, Τιμή:11692, Πινακίδα:ETE-1291, Έκπτωση:Όχι
Μάρκα:Ford Fiesta    , Χιλιόμετρα:28446, Τιμή:10019, Πινακίδα:MAZ-1318, Έκπτωση:Όχι
Μάρκα:Opel Astra     , Χιλιόμετρα:41263, Τιμή: 7283, Πινακίδα:EHM-7938, Έκπτωση:Όχι
Μάρκα:Opel Corsa     , Χιλιόμετρα:43552, Τιμή:12661, Πινακίδα:IMY-5601, Έκπτωση:Όχι
Μάρκα:Volkswagen Polo, Χιλιόμετρα:28341, Τιμή: 7898, Πινακίδα:HYO-8600, Έκπτωση:Όχι
Μάρκα:Volkswagen Golf, Χιλιόμετρα:54368, Τιμή: 6281, Πινακίδα:ATK-1872, Έκπτωση:Όχι
Μάρκα:Peugeot 208    , Χιλιόμετρα:44835, Τιμή:13155, Πινακίδα:BYT-7646, Έκπτωση:Όχι
Μάρκα:Peugeot 308    , Χιλιόμετρα:18082, Τιμή: 8662, Πινακίδα:BNT-7241, Έκπτωση:Όχι
Μάρκα:Toyota Auris   , Χιλιόμετρα:26161, Τιμή:14697, Πινακίδα:KTE-9404, Έκπτωση:Όχι
Μάρκα:Toyota Yaris   , Χιλιόμετρα:27446, Τιμή:12317, Πινακίδα:KZY-8970, Έκπτωση:Όχι

Ο μέσος όρος των χιλιομέτρων είναι: 36344.0

Μπορείτε να κάνετε έκπτωση στο Ford Focus με πινακίδα ETE-1291, η καινούργια τιμή είναι:10522.8
Μάρκα:Ford Focus, Χιλιόμετρα:50946, Τιμή:10522.8, Πινακίδα:ETE-1291, Έκπτωση:10%


Μπορείτε να κάνετε έκπτωση στο Opel Astra με πινακίδα EHM-7938, η καινούργια τιμή είναι:6554.7
Μάρκα:Opel Astra, Χιλιόμετρα:41263, Τιμή:6554.7, Πινακίδα:EHM-7938, Έκπτωση:10%


Μπορείτε να κάνετε έκπτωση στο Opel Corsa με πινακίδα IMY-5601, η καινούργια τιμή είναι:11394.9
Μάρκα:Opel Corsa, Χιλιόμετρα:43552, Τιμή:11394.9, Πινακίδα:IMY-5601, Έκπτωση:10%


Μπορείτε να κάνετε έκπτωση στο Volkswagen Golf με πινακίδα ATK-1872, η καινούργια τιμή είναι:5652.9
Μάρκα:Volkswagen Golf, Χιλιόμετρα:54368, Τιμή:5652.9, Πινακίδα:ATK-1872, Έκπτωση:10%


Μπορείτε να κάνετε έκπτωση στο Peugeot 208 με πινακίδα BYT-7646, η καινούργια τιμή είναι:11839.5
Μάρκα:Peugeot 208, Χιλιόμετρα:44835, Τιμή:11839.5, Πινακίδα:BYT-7646, Έκπτωση:10%

Η οικονομικότερη επιλογή με βαση τα διαθέσιμα αυτοκίνητα είναι το:Volkswagen Golf με τιμή 5652.9 και
πινακίδα ATK-1872
>>>
```

Ln: 50 Col: 4

Εικόνα 16: Ενδεικτική Εκτέλεση του Προγράμματος Έβδομης Άσκησης

Σκεπτικό Επίλυσης

Συμπληρώστε τον παρακάτω πίνακα με τα βήματα υπολογιστικής σκέψης. Προσθέστε αν χρειαστεί κελιά.

Αφαίρεση	**Αποσύνθεση**	εντολές	**Πρότυπο**	εντολές

Απάντηση

Αναφέρετε το όνομα του αρχείου .py που έχετε επισυνάψει στο .zip αρχείο (π.χ. exercise07.py) και αντιστοιχεί στην λύση της άσκησης. Το αρχείο .py να είναι επαρκώς σχολιασμένο.

Άσκηση 8 dictionary

Μαθησιακοί στόχοι

Οι φοιτητές θα εξοικειωθούν περαιτέρω με τη χρήση της δομής αντιστοίχισης ζευγών (κλειδιού-τιμής) καθώς επίσης και με τις συναρτήσεις και μεθόδους που τα λεξικά διαθέτουν. Τελικός στόχος ορίζεται η επίλυση του προβλήματος με αλγοριθμικό τρόπο χρησιμοποιώντας τα συστατικά της υπολογιστικής σκέψης.

Λέξεις κλειδιά

Python, δομή επανάληψης, δομή ελέγχου, δομή αντιστοίχισης, συνθήκες, τελεστές σύγκρισης, λίστες, περιγραφή λίστας, λεξικό, κλειδί – τιμή, λίστες λεξικών, πλειάδες, if , for, while υπολογιστική σκέψη, αφαίρεση, αποσύνθεση, αναγνώριση προτύπων.

Εκφώνηση

Υποθέτουμε ότι για τις εισαγωγικές εξετάσεις μιας γυμναστικής ακαδημίας οι ενδιαφερόμενοι θα πρέπει να δώσουν εξετάσεις στο άθλημα της σφαιροβολίας και ανάλογα με τις επιδόσεις του κάθε αθλητή να οριστεί η βαθμολογία καθώς και ο τελικός πίνακας κατάταξης.

Να υλοποιηθεί πρόγραμμα στη γλώσσα προγραμματισμού Python το οποίο:

1. Να εμφανίζει ερώτηση για το εάν υπάρχει επόμενος αθλητής και με enter να συνεχίζει ενώ εάν ο χρήστης εισάγει «Ε» να πραγματοποιείται έξοδος από το πρόγραμμα.
2. Να δέχεται είσοδο δεδομένων από το χρήστη για το ονοματεπώνυμο του αθλητή.
3. Να εμφανίζει τον αριθμό της προσπάθειας και να δέχεται είσοδο δεδομένων από το χρήστη για το εάν η προσπάθεια που πραγματοποιήθηκε ήταν έγκυρη, λαμβάνοντας υπόψη ότι κάθε αθλητής δικαιούται τρείς προσπάθειες. Στην περίπτωση που η προσπάθεια είναι έγκυρη να δημιουργείται η ρίψη από ψευδοτυχαίους αριθμούς μεταξύ 3,00 και 15,00 και να εμφανίζεται.
4. Να υπολογίζει και να εμφανίζει τη ρίψη και τη βαθμολογία της καλύτερης προσπάθειας σύμφωνα με τα εξής όρια:
   - Βαθμός 0 για ρίψη μέχρι 6,39μ.
   - Βαθμός 1 για ρίψη μέχρι 7,19μ.
   - Βαθμός 2 για ρίψη μέχρι 7,99μ.
   - Βαθμός 3 για ρίψη μέχρι 8,79μ.
   - Βαθμός 4 για ρίψη μέχρι 9,59μ.
   - Βαθμός 5 για ρίψη μέχρι 10,39μ.
   - Βαθμός 6 για ρίψη μέχρι 11,19μ.
   - Βαθμός 7 για ρίψη μέχρι 11,99μ.
   - Βαθμός 8 για ρίψη μέχρι 12,79μ.
   - Βαθμός 9 για ρίψη μέχρι 13,59μ.
   - Βαθμός 10 για ρίψη μέχρι 15,00μ.
5. Σε περίπτωση που ο χρήστης επιλέξει έξοδο από το πρόγραμμα, να υπολογίζει και να εμφανίζει για κάθε αθλητή εάν η μεγαλύτερη από τις τρεις προσπάθειες ήταν πάνω από τη βάση που σε αυτήν την περίπτωση είναι το πέντε καθώς επίσης και τον μέσο όρο της

βαθμολογίας των αθλητών λαμβάνοντας υπόψη την καλύτερη προσπάθεια του κάθε αθλητή.

6. Τέλος να εμφανίζει τον πίνακα κατάταξης των αθλητών, που αποτελείται από το ονοματεπώνυμο του αθλητή και τη βαθμολογία της καλύτερης προσπάθειάς του, σε φθίνουσα σειρά με βάση το σκόρ.

Παράδειγμα

Ακολουθεί σχετικό παράδειγμα όπου ο χρήστης:

- Εισάγει τον αθλητή Παπαδόπουλο Γιάννη
- Προσπάθειες: έγκυρη, έγκυρη, άκυρη
- Εισάγει τον αθλητή Νάκα Γιώργο
- Προσπάθειες: άκυρη, έγκυρη, έγκυρη
- Εισάγει τον αθλητή Πέτρου Πέτρο
- Προσπάθειες: άκυρη, έγκυρη, έγκυρη
- Εισάγει τον αθλητή Κωνσταντινίδη Κυριάκο
- Προσπάθειες: έγκυρη, έγκυρη, άκυρη
- Επιλέγει έξοδο από το πρόγραμμα

Η **Error! Reference source not found.**7 παρουσιάζει μια ενδεικτική εκτέλεση του προγράμματος

```
πρόγραμμα βαθμολογίας
Υπάρχει επόμενος αθλητής;
Εισάγετε ονοματεπώνυμο αθλητή: Παπαδόπουλος Γιάννης
1η προσπάθεια
Ήταν έγκυρη η προσπάθεια;Ν/Ο: Ν
Η ρίψη ήταν 5.11
2η προσπάθεια
Ήταν έγκυρη η προσπάθεια;Ν/Ο: Ν
Η ρίψη ήταν 7.65
3η προσπάθεια
Ήταν έγκυρη η προσπάθεια;Ν/Ο: Ο
Η καλύτερη ρίψη ήταν 7.65 και αντιστοιχεί στον βαθμό 2
Υπάρχει επόμενος αθλητής;
Εισάγετε ονοματεπώνυμο αθλητή: Νάκας Γιώργος
1η προσπάθεια
Ήταν έγκυρη η προσπάθεια;Ν/Ο: Ο
2η προσπάθεια
Ήταν έγκυρη η προσπάθεια;Ν/Ο: Ν
Η ρίψη ήταν 3.85
3η προσπάθεια
Ήταν έγκυρη η προσπάθεια;Ν/Ο: Ν
Η ρίψη ήταν 7.82
Η καλύτερη ρίψη ήταν 7.82 και αντιστοιχεί στον βαθμό 2
Υπάρχει επόμενος αθλητής;
Εισάγετε ονοματεπώνυμο αθλητή: Πέτρου Πέτρος
1η προσπάθεια
Ήταν έγκυρη η προσπάθεια;Ν/Ο: Ο
2η προσπάθεια
Ήταν έγκυρη η προσπάθεια;Ν/Ο: Ν
Η ρίψη ήταν 9.7
3η προσπάθεια
Ήταν έγκυρη η προσπάθεια;Ν/Ο: Ν
Η ρίψη ήταν 10.86
Η καλύτερη ρίψη ήταν 10.86 και αντιστοιχεί στον βαθμό 6
Υπάρχει επόμενος αθλητής;
Εισάγετε ονοματεπώνυμο αθλητή: Κωνσταντινίδης Κυριάκος
1η προσπάθεια
Ήταν έγκυρη η προσπάθεια;Ν/Ο: Ν
Η ρίψη ήταν 4.29
2η προσπάθεια
Ήταν έγκυρη η προσπάθεια;Ν/Ο: Ν
Η ρίψη ήταν 4.91
3η προσπάθεια
Ήταν έγκυρη η προσπάθεια;Ν/Ο: Ο
Η καλύτερη ρίψη ήταν 4.91 και αντιστοιχεί στον βαθμό 0
Υπάρχει επόμενος αθλητής; Ε
Ο αθλητής Παπαδόπουλος Γιάννης δεν πέρασε με score:2
Ο αθλητής Νάκας Γιώργος  δεν πέρασε με score:2
Ο αθλητής Πέτρου Πέτρος πέρασε με score:6
Ο αθλητής Κωνσταντινίδης Κυριάκος δεν πέρασε με score:0
Ο μέσος όρος των αθλητών ειναι: 2.5


Πίνακας κατάταξης:
Πέτρου Πέτρος          :6
Παπαδόπουλος Γιάννης   :2
Νάκας Γιώργος          :2
Κωνσταντινίδης Κυριάκος:0
>>>
                                                    Ln: 63  Col: 4
```

Εικόνα 17: Ενδεικτική Εκτέλεση του Προγράμματος Όγδοης Άσκησης

Σκεπτικό Επίλυσης

Συμπληρώστε τον παρακάτω πίνακα με τα βήματα υπολογιστικής σκέψης. Προσθέστε αν χρειαστεί κελιά.

Αφαίρεση	**Αποσύνθεση**	εντολές	**Πρότυπο**	εντολές

Απάντηση

Αναφέρετε το όνομα του αρχείου .py που έχετε επισυνάψει στο .zip αρχείο (π.χ. exercise08.py) και αντιστοιχεί στην λύση της άσκησης. Το αρχείο .py να είναι επαρκώς σχολιασμένο.

Άσκηση 9 functions

Μαθησιακοί στόχοι

Οι φοιτητές θα μάθουν να χρησιμοποιούν τις συναρτήσεις για τη σωστή δόμηση ενός προγράμματος καθώς και την επαναχρησιμοποίηση αυτών με σκοπό την αποφυγή του πλεονασμού και την απλοποίηση του κώδικα. Η χρήση των συναρτήσεων σε συνδυασμό με τις ιδιότητες της υπολογιστικής σκέψης αποσκοπεί στην εξεύρεση καλύτερης προσέγγισης για την επίλυση ενός προβλήματος.

Λέξεις κλειδιά

Python, δομή επανάληψης, δομή ελέγχου, συνθήκες, τελεστές σύγκρισης, if, if/elif/else , for, while,συναρτήσεις, επαναχρησιμοποίηση κώδικα, παράμετροι, ορίσματα, εμβέλεια μεταβλητών, υπολογιστική σκέψη, αφαίρεση, αποσύνθεση, αναγνώριση προτύπων.

Εκφώνηση

Υποθέτουμε ότι ένας μετατροπέας συναλλάγματος από ξένα νομίσματα σε ευρώ, προσφέρει τις εξής συναλλαγές:

- Στερλίνες σε Ευρώ.
- Δολάρια σε Ευρώ.
- Ελβετικό φράγκο σε Ευρώ.
- Καναδικά Δολάρια σε Ευρώ.

Να υλοποιηθεί πρόγραμμα στη γλώσσα προγραμματισμού Python το οποίο:

1. Να εμφανίζει αριθμημένη λίστα επιλογής (μενού) από το ένα έως το τέσσερα προσφέροντας τις παραπάνω συναλλαγές και να δέχεται την επιλογή με είσοδο δεδομένων από το χρήστη διασφαλίζοντας την εγκυρότητα της τιμής.
2. Να δέχεται είσοδο δεδομένων από το χρήστη για το ποσό προς μετατροπή.
3. Να υπολογίζει και να εμφανίζει την επιλεγόμενη μετατροπή συναλλάγματος σε Ευρώ με τις εξής ισοτιμίες:
    - 1 Στερλίνα =1,13 Ευρώ
    - 1 Δολάριο =0,86 Ευρώ
    - 1 Ελβετικό φράγκο =0,86 Ευρώ
    - 1 Καναδικά Δολάριο =0,68 Ευρώ
4. Να υπολογίζει και να εμφανίζει τα προτεινόμενα χαρτονομίσματα και νομίσματα με βάση το μικρότερο δυνατό αριθμό χαρτονομισμάτων και νομισμάτων σε Ευρώ. Το συνάλλαγμα σε Ευρώ θα υπολογίζεται με τις εξής αξίες:
    - Χαρτονομίσματα αξίας:200€, 100€, 50€, 20€, 10€, 5€
    - Νομίσματα αξίας:2€, 1€, 0.50€, 0.20€, 0.10€, 0.05€, 0.02€, 0.01€

5. Να υπολογίζει και να εμφανίζει το σύνολο του συναλλάγματος ανά κατηγορία νομίσματος.
6. Να δέχεται είσοδο δεδομένων από το χρήστη για το εάν ο χρήστης επιθυμεί να πραγματοποιήσει άλλη μετατροπή.

Παράδειγμα

Ακολουθεί σχετικό παράδειγμα όπου ο χρήστης:

- Επιλέγει τη μετατροπή από δολάριο σε ευρώ
- Εισάγει το ποσό 3525
- Επιλέγει επανάληψη μετατροπής
- Επιλέγει τη μετατροπή από στερλίνα σε ευρώ
- Εισάγει το ποσό 2740
- Επιλέγει έξοδο από το πρόγραμμα

Η **Error! Reference source not found.**8 παρουσιάζει μια ενδεικτική εκτέλεση του προγράμματος

```
Μετατροπέας συναλλάγματος σε Ευρώ
Παρακαλώ επιλέξτε ένα απο τα παρακάτω:
1: GBP σε EUR
2: USD σε EUR
3: CHF σε EUR
4: CAD σε EUR
Παρακαλώ εισάγετε την επιλογή σας: 2

Παρακαλώ εισάγετε το ποσό για μετατροπή: 3525
Έχετε επιλέξει μετατροπή συναλλάγματος απο Δολάρια(USD) σε Ευρώ(EURO)
Το ποσό που αντιστοιχεί είναι: 3031.5 Ευρώ

Προτεινόμενα χαρτονομίσματα και νομίσματα:
€200.0: 15
€ 20.0: 1
€ 10.0: 1
€  1.0: 1
€  0.5: 1

Το σύνολο του συναλλάγματος είναι:
EURO:3031.5
GBP:0
USD:3525.0
CHF:0
CAD:0

θέλετε να πραγματοποιήσετε άλλο Υπολογισμό; N/O: N

Μετατροπέας συναλλάγματος σε Ευρώ
Παρακαλώ επιλέξτε ένα απο τα παρακάτω:
1: GBP σε EUR
2: USD σε EUR
3: CHF σε EUR
4: CAD σε EUR
Παρακαλώ εισάγετε την επιλογή σας: 1

Παρακαλώ εισάγετε το ποσό για μετατροπή: 2740
Έχετε επιλέξει μετατροπή συναλλάγματος απο Στερλίνες(GBP) σε Ευρώ(EURO)
Το ποσό που αντιστοιχεί είναι: 3096.2 Ευρώ

Προτεινόμενα χαρτονομίσματα και νομίσματα:
€200.0: 15
€ 50.0: 1
€ 20.0: 2
€  5.0: 1
€  1.0: 1
€  0.2: 1

Το σύνολο του συναλλάγματος είναι:
EURO:6127.7
GBP:2740.0
USD:3525.0
CHF:0
CAD:0

θέλετε να πραγματοποιήσετε άλλο Υπολογισμό; N/O: O

>>>
                                                        Ln: 62  Col: 4
```

Εικόνα 18: Ενδεικτική Εκτέλεση του Προγράμματος Ένατης Άσκησης

Σκεπτικό Επίλυσης

Συμπληρώστε τον παρακάτω πίνακα με τα βήματα υπολογιστικής σκέψης. Προσθέστε αν χρειαστεί κελιά.

Αφαίρεση	**Αποσύνθεση**	εντολές	**Πρότυπο**	εντολές

Απάντηση

Αναφέρετε το όνομα του αρχείου .py που έχετε επισυνάψει στο .zip αρχείο (π.χ. exercise09.py) και αντιστοιχεί στην λύση της άσκησης. Το αρχείο .py να είναι επαρκώς σχολιασμένο.

Άσκηση 10 functions

Μαθησιακοί στόχοι

Οι φοιτητές θα εξοικειωθούν περαιτέρω με την ομαδοποίηση των εντολών κάνοντας χρήση των συναρτήσεων, με παραμέτρους ή χωρίς, για αποφυγή της επανάληψης του κώδικα, που σε συνδυασμό με τα χαρακτηριστικά της υπολογιστικής σκέψης αποσκοπούν στην εξεύρεση βελτιωμένης προσέγγισης επίλυσης προβλημάτων.

Λέξεις κλειδιά

Python, δομή επανάληψης, δομή ελέγχου, συνθήκες, τελεστές σύγκρισης, πλειάδες, if, if/elif/else , for, while,συναρτήσεις, επαναχρησιμοποίηση κώδικα, παράμετροι, ορίσματα, εμβέλεια μεταβλητών, υπολογιστική σκέψη, αφαίρεση, αποσύνθεση, αναγνώριση προτύπων .

Εκφώνηση

Υποθέτουμε ότι μια μεταφορική εταιρία θέλει να εφαρμόσει ένα πιλοτικό πρόγραμμα για την εξυπηρέτηση των αναγκών των πελατών της και για τον λόγο αυτό ξεκινάει την εφαρμογή του προγράμματος με ένα επαγγελματικό αυτοκίνητο, όπου στόχος του προγράμματος είναι ο σχεδιασμός της διαδρομής που θα ακολουθήσει το αυτοκίνητο με βάση τα σημεία παραλαβής και παράδοσης καθώς επίσης και ο υπολογισμός της κατανάλωσης καυσίμων για τη συγκεκριμένη διαδρομή. Το υποθετικό σενάριο υλοποιείται σε ένα καρτεσιανό σύστημα συντεταγμένων με αφετηρία το σημείο 0,0.

Να υλοποιηθεί πρόγραμμα στη γλώσσα προγραμματισμού Python το οποίο:

1. Να εμφανίζει ερώτηση για το πόσοι πελάτες θα πρέπει να εξυπηρετηθούν και να δέχεται είσοδο δεδομένων από το χρήστη για την επιλογή.
2. Να δημιουργεί και να αποθηκεύει τον αύξοντα αριθμό των πελατών και τις αντίστοιχες τοποθεσίες παραλαβής και παράδοσης, οι οποίες θα είναι συντεταγμένες του καρτεσιανού συστήματος και θα αποτελούνται από ψευδοτυχαίους αριθμούς μεταξύ ένα και δεκαπέντε. Για παράδειγμα ο πρώτος πελάτης θα αποθηκεύεται με τη μορφή (1,5,4,3,8) όπου 1 είναι ο αύξοντας αριθμός του πελάτη ενώ οι συντεταγμένες 5,4 αντιστοιχούν στο σημείο παραλαβής και 3,8 στο σημείο παράδοσης. Μετά τη δημιουργία των πελατών να εμφανίζει τη λίστα.
3. Να δημιουργεί τη διαδρομή με τον εξής τρόπο:
   - Ξεκινώντας από την αφετηρία (σημείο 0,0) για την εύρεση του πρώτου πελάτη να υπολογίζει την κοντινότερη απόσταση από το σύνολο των πελατών με βάση τον τύπο της ευκλείδειας απόστασης $\sqrt{(\chi 2-\chi 1)^2+(\psi 2-\psi 1)^2}$ που θα αποτελεί και την τοποθεσία παραλαβής. Στην συνέχεια να υπολογίζει την απόσταση μέχρι την τοποθεσία παράδοσης και να εμφανίζει μήνυμα ότι ο πελάτης εξυπηρετήθηκε και να εμφανίζει τη λίστα των μη εξυπηρετηθέντων πελατών. Αυτή η διαδικασία επαναλαμβάνεται από το σημείο παράδοσης για να βρεθεί η επόμενη κοντινότερη απόσταση παραλαβής, μέχρι να εξυπηρετηθεί και ο τελευταίος πελάτης περίπτωση στην οποία θα εμφανίζεται μήνυμα ότι όλοι οι πελάτες εξυπηρετήθηκαν.

4. Να εμφανίζει τη συνολική διαδρομή που ακολουθήθηκε σε συντεταγμένες, λαμβάνοντας υπόψη ότι από το τελευταίο σημείο παράδοσης πρέπει να πραγματοποιηθεί η επιστροφή στην αφετηρία (σημείο 0,0).
5. Να υπολογίζει και να εμφανίζει τη συνολική απόσταση που διανύθηκε καθώς επίσης και τα συνολικά λίτρα των καυσίμων που καταναλώθηκαν με βάση την μέση κατανάλωση που ορίζεται σε 28,6λίτρα ανά 100 χιλιόμετρα.

Παράδειγμα

Η **Error! Reference source not found.**9 παρουσιάζει μια ενδεικτική εκτέλεση του προγράμματος

```
Διαχείριση πελατών
Πόσοι πελάτες πρέπει να εξυπηρετηθούν; 5

Λίστα Πελατών:
(1, 7, 10, 8, 3)
(2, 2, 5, 8, 8)
(3, 8, 9, 9, 13)
(4, 4, 1, 7, 14)
(5, 9, 4, 6, 10)

Ο πελάτης 4 εξυπηρετήθηκε

Λίστα Πελατών:
(1, 7, 10, 8, 3)
(2, 2, 5, 8, 8)
(3, 8, 9, 9, 13)
(5, 9, 4, 6, 10)

Ο πελάτης 1 εξυπηρετήθηκε

Λίστα Πελατών:
(2, 2, 5, 8, 8)
(3, 8, 9, 9, 13)
(5, 9, 4, 6, 10)

Ο πελάτης 5 εξυπηρετήθηκε

Λίστα Πελατών:
(2, 2, 5, 8, 8)
(3, 8, 9, 9, 13)

Ο πελάτης 3 εξυπηρετήθηκε

Λίστα Πελατών:
(2, 2, 5, 8, 8)

Ο πελάτης 2 εξυπηρετήθηκε

Λίστα Πελατών:
Όλοι οι πελάτες εξυπηρετήθηκαν

Η διαδρομή που ακολουθήθηκε ήταν:
[(0, 0), (4, 1), (7, 14), (7, 10), (8, 3), (9, 4), (6, 10), (8, 9), (9, 13),
(2, 5), (8, 8), (0, 0)]

Η συνολική απόσταση που διανύθηκε ήταν 71.66 και τα καύσιμα που καταναλώθηκαν
ήταν 20.49 λίτρα
>>>
```

Ln: 50 Col: 4

Εικόνα 19: Ενδεικτική Εκτέλεση του Προγράμματος Δέκατης Άσκησης

Σκεπτικό Επίλυσης

Συμπληρώστε τον παρακάτω πίνακα με τα βήματα υπολογιστικής σκέψης. Προσθέστε αν χρειαστεί κελιά.

Αφαίρεση	**Αποσύνθεση**	εντολές	**Πρότυπο**	εντολές

Απάντηση

Αναφέρετε το όνομα του αρχείου .py που έχετε επισυνάψει στο .zip αρχείο (π.χ. exercise10.py) και αντιστοιχεί στην λύση της άσκησης. Το αρχείο .py να είναι επαρκώς σχολιασμένο.

Παράρτημα ΙΙ

[1]Τεστ Υπολογιστικής Σκέψης (Roman-Gonzalez, Moreno-Leon, & Robles, 2017)

Καλωσορίσατε στο Τέστ της Υπολογιστικής Σκέψης

* Απαιτείται

Φόρμα συμπλήρωσης στοιχείων

1. Επώνυμο *

_____

2. Όνομα *

_____

3. Αριθμός Μητρώου *

_____

4. Ημερομηνία *

_____

Παράδειγμα: 15 Δεκεμβρίου 2012

5. Φύλλο *

Να επισημαίνεται μόνο μία έλλειψη.

○ Άνδρας

◉ Γυναίκα

[1] Το τεστ στην αρχική του μορφή βρίσκεται στη διεύθυνση: http://goo.gl/IYEKMB (Spanish version)

Οδηγίες

Το τέστ υπολογιστικής σκέψης αποτελείται από 28 ερωτήσεις , χωρισμένο σε 7 σελίδες με 4 ερωτήσεις η κάθε σελίδα. Όλες οι ερωτήσεις έχουν 4 διαθέσιμες επιλογές για απάντηση (Α,Β,Γ,Δ) από τις οποίες μόνο μια είναι σωστή. Από την έναρξη του τέστ έχετε 45 λεπτά στη διάθεσή σας για να απαντήσετε σε όσες περισσότερες ερωτήσεις μπορείτε. Δεν κρίνεται απαραίτητο να απαντήσετε σε όλες τις ερωτήσεις. Πριν ξεκινήσουν οι ερωτήσεις του τεστ ,υπάρχουν 3 λυμένα παραδείγματα για την εξοικείωση με το είδος των ερωτήσεων που ακολουθούν καθώς επίσης και με τους χαρακτήρες που εμφανίζονται.

Χαρακτήρες

Pac man

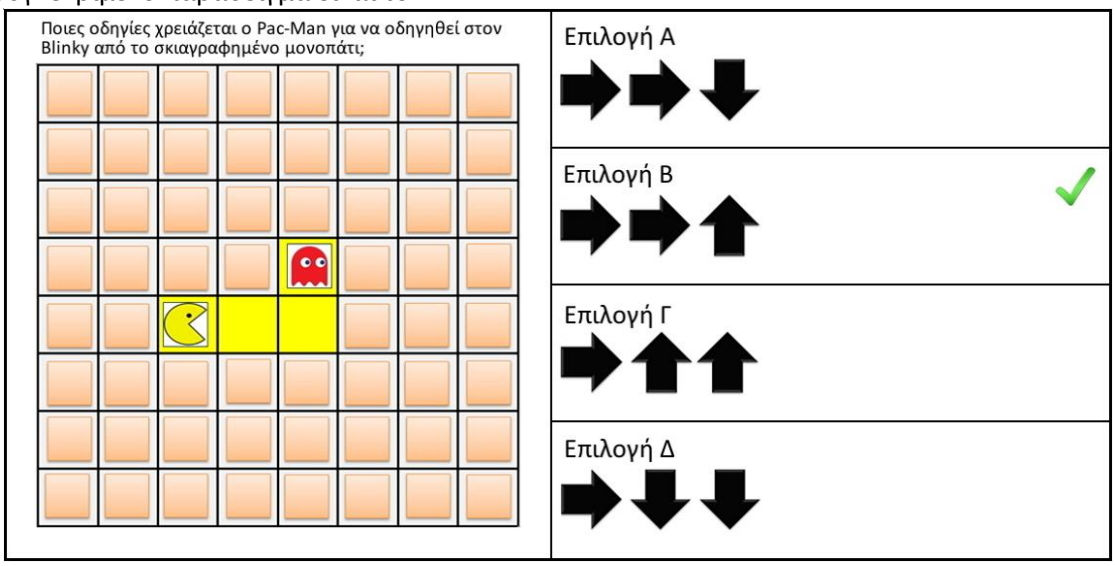
Blinky

Artist

[2]Παραδείγματα 1-3

Παράδειγμα 1

Στο πρώτο παράδειγμα ζητούνται οι οδηγίες που χρειάζεται ο Pac-Man για να οδηγηθεί στον Blinky από το σκιαγραφημένο μονοπάτι. Αυτό σημαίνει να οδηγηθεί ο Pac-Man ακριβώς στο τετράγωνο που βρίσκεται ο Blinky (δίχως να παραμείνει στο προηγούμενο ή να μεταβεί στο επόμενο τετράγωνο) ακολουθώντας το σκιαγραφημένο κίτρινο μονοπάτι (χωρίς να εφάπτεται στους τοίχους που αναπαρίστανται με τα πορτοκαλί τετράγωνα). Η σωστή απάντηση στο συγκεκριμένο παράδειγμα είναι το Β.



Επιλέξτε τη σωστή απάντηση (σε αυτό το παράδειγμα η σωστή απάντηση είναι το Β):
Να επισημαίνεται μόνο μία έλλειψη.

Επιλογή Α Επιλογή Β

Επιλογή Γ Επιλογή Δ

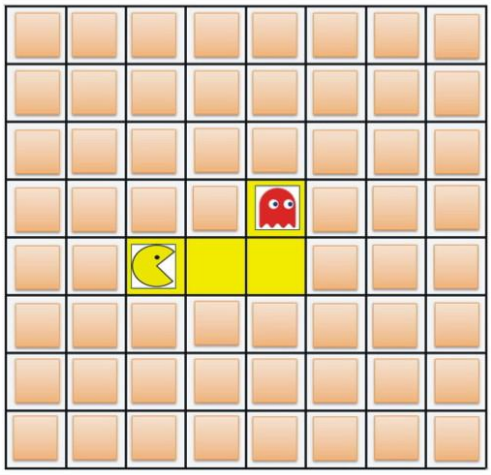
Παράδειγμα 2

Στο δεύτερο παράδειγμα ζητούνται πάλι οι οδηγίες που χρειάζεται ο Pac-Man για να οδηγηθεί στον Blinky από το σκιαγραφημένο μονοπάτι. Ωστόσο σε αυτήν την
ερώτηση οι απαντήσεις εμφανίζονται σαν ομάδες από μπλοκ εντολών και όχι σαν βέλη. Υπενθυμίζεται ότι ζητείται να οδηγηθεί ο Pac-Man ακριβώς στο τετράγωνο που βρίσκεται ο Blinky (δίχως να παραμείνει στο προηγούμενο ή να μεταβεί στο επόμενο τετράγωνο) ακολουθώντας το σκιαγραφημένο κίτρινο μονοπάτι (χωρίς να εφάπτεται στους τοίχους που αναπαρίστανται με τα πορτοκαλί τετράγωνα). Η σωστή απάντηση στο συγκεκριμένο

[2] Το τεστ στην αρχική του μορφή βρίσκεται στη διεύθυνση: http://goo.gl/IYEKMB (Spanish version)

παράδειγμα είναι το Γ.

Ποιες οδηγίες χρειάζεται ο Pac-Man για να οδηγηθεί στον Blinky από το σκιαγραφημένο μονοπάτι; Κάθε τετράγωνο έχει μέγεθος πλευράς 30 pixels.



```
from turtle import *
canvas = Screen()
pacman = Turtle()
```

Επιλογή Α
```
pacman.forward(30)
pacman.left(90)
pacman.forward(30)
pacman.forward(30)
```

Επιλογή Β
```
pacman.forward(30)
pacman.right(90)
pacman.forward(30)
pacman.forward(30)
```

Επιλογή Γ ✓
```
pacman.forward(30)
pacman.forward(30)
pacman.left(90)
pacman.forward(30)
```

Επιλογή Δ
```
pacman.forward(30)
pacman.forward(30)
pacman.right(90)
pacman.forward(30)
```

Επιλέξτε τη σωστή απάντηση (σε αυτό το παράδειγμα η σωστή απάντηση είναι το Γ):

Να επισημαίνεται μόνο μία έλλειψη.

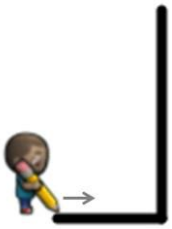
Επιλογή Α Επιλογή Β

Επιλογή Γ Επιλογή Δ

Παράδειγμα 3

Στο τρίτο παράδειγμα ζητούνται οι οδηγίες που χρειάζεται να ακολουθήσει ο καλλιτέχνης για να σχεδιάσει το σχήμα της εικόνας. Αυτό σημαίνει πως θα μετακινηθεί το μολύβι για να σχεδιαστεί το σχήμα. Με την οδηγία κινήσου το μολύβι σχεδιάζει ενώ με την οδηγία πήδηξε ο Καλλιτέχνης αλλάζει θέση χωρίς να σχεδιάζει. Το γκρι βέλος υποδεικνύει την αρχική κατεύθυνση του μολυβιού. Η σωστή απάντηση στο συγκεκριμένο παράδειγμα είναι το Α.

Ποιες οδηγίες χρειάζεται να ακολουθήσει ο καλλιτέχνης για να σχεδιάσει το σχήμα; Η μικρή πλευρά έχει μέγεθος 50 pixels και η μεγάλη 100 pixels.



```
from turtle import *
canvas = Screen()
artist = Turtle()
```

Επιλογή Α ✓
```
artist.forward(50)
artist.left(90)
artist.forward(100)
```

Επιλογή Β
```
artist.forward(50)
artist.right(90)
artist.forward(100)
```

Επιλογή Γ
```
artist.forward(100)
artist.left(90)
artist.forward(50)
```

Επιλογή Δ
```
artist.forward(100)
artist.right(90)
artist.forward(50)
```

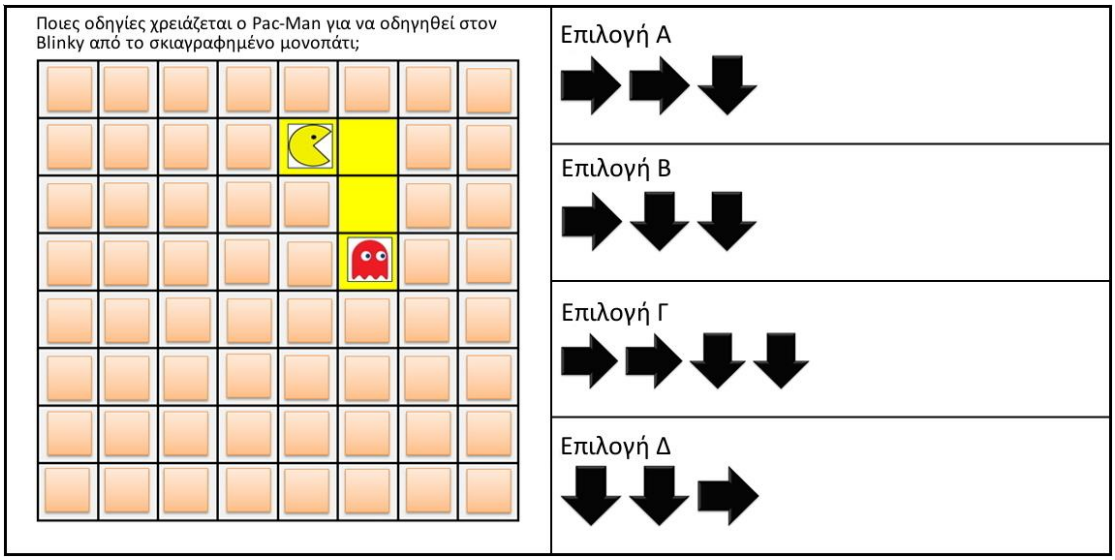
Επιλέξτε τη σωστή απάντηση (σε αυτό το παράδειγμα η σωστή απάντηση είναι το Α):
Να επισημαίνεται μόνο μία έλλειψη.

Επιλογή Α Επιλογή Β

Επιλογή Γ Επιλογή Δ

[3]Ερωτήσεις 1-4

Ερώτηση 1

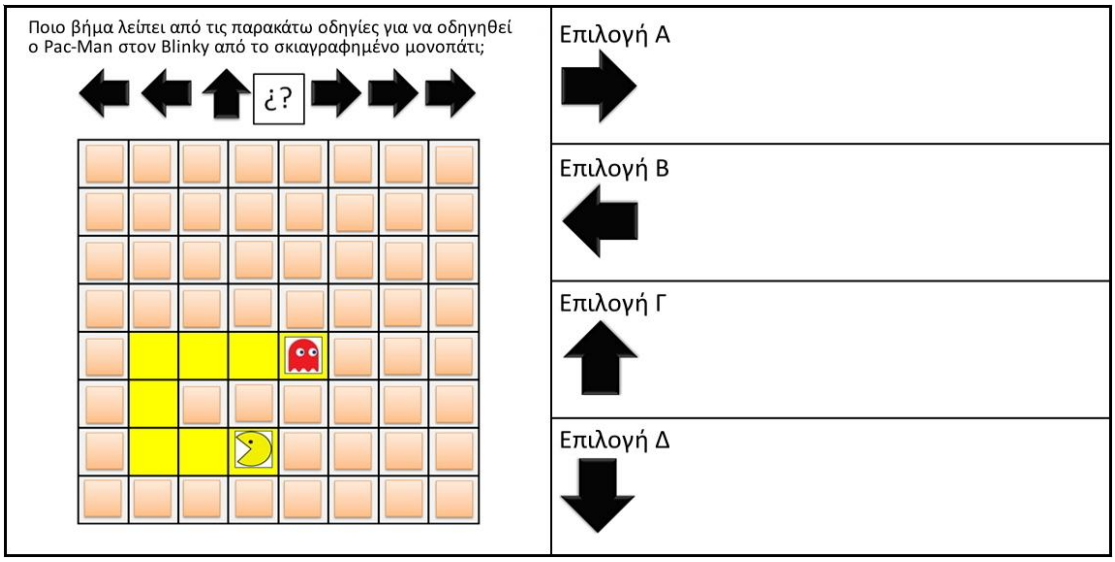


Επιλέξτε τη σωστή απάντηση:
Να επισημαίνεται μόνο μία έλλειψη.

Επιλογή Α Επιλογή Β

Επιλογή Γ Επιλογή Δ

Ερώτηση 2



Επιλέξτε τη σωστή απάντηση:
Να επισημαίνεται μόνο μία έλλειψη.

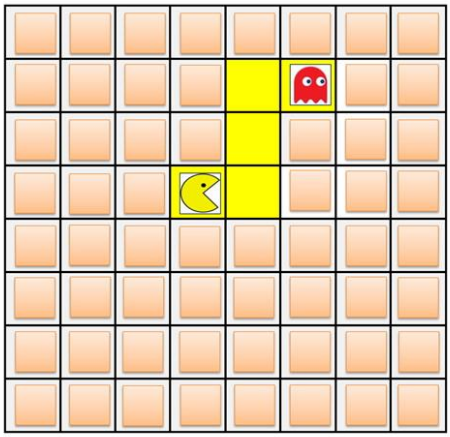
Επιλογή Α Επιλογή Β

[3] Το τεστ στην αρχική του μορφή βρίσκεται στη διεύθυνση: http://goo.gl/IYEKMB (Spanish version)

Επιλογή Γ Επιλογή Δ

Ερώτηση 3

Παρακάτω αναφέρονται οι οδηγίες που πρέπει να ακολουθήσει ο Pac-Man για να φτάσει στον Blinky από το σκιαγραφημένο μονοπάτι. Σε ποιο από τα βήματα των οδηγιών υπάρχει **λάθος**; Κάθε τετράγωνο έχει μέγεθος πλευράς 30 pixels.



```
from turtle import *
canvas = Screen()
pacman = Turtle()
```

```
pacman.forward(30)   → Βήμα Α
pacman.left(90)      → Βήμα Β
pacman.forward(30)
pacman.forward(30)   → Βήμα Γ
pacman.left(90)      → Βήμα Δ
pacman.forward(30)
```

Επιλέξτε τη σωστή απάντηση:

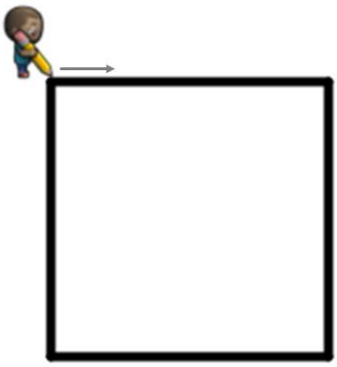
Να επισημαίνεται μόνο μία έλλειψη.

Επιλογή Α Επιλογή Β

Επιλογή Γ Επιλογή Δ

Ερώτηση 4

Ποιες οδηγίες χρειάζεται να ακολουθήσει ο καλλιτέχνης για να σχεδιάσει το τετράγωνο με μέγεθος πλευράς 100 pixels;



```
from turtle import *
canvas = Screen()
artist = Turtle()
```

Επιλογή Α

```
artist.forward(100)
artist.right(90)
artist.forward(100)
artist.left(90)
artist.forward(100)
artist.right(90)
artist.forward(100)
```

Επιλογή Β

```
artist.forward(25)
artist.right(90)
artist.forward(25)
artist.left(90)
artist.forward(25)
artist.right(90)
artist.forward(25)
```

Επιλογή Γ

```
artist.forward(50)
artist.right(90)
artist.forward(50)
artist.right(90)
artist.forward(50)
artist.right(90)
artist.forward(50)
```

Επιλογή Δ

```
artist.forward(100)
artist.right(90)
artist.forward(100)
artist.right(90)
artist.forward(100)
artist.right(90)
artist.forward(100)
```

Επιλέξτε τη σωστή απάντηση:

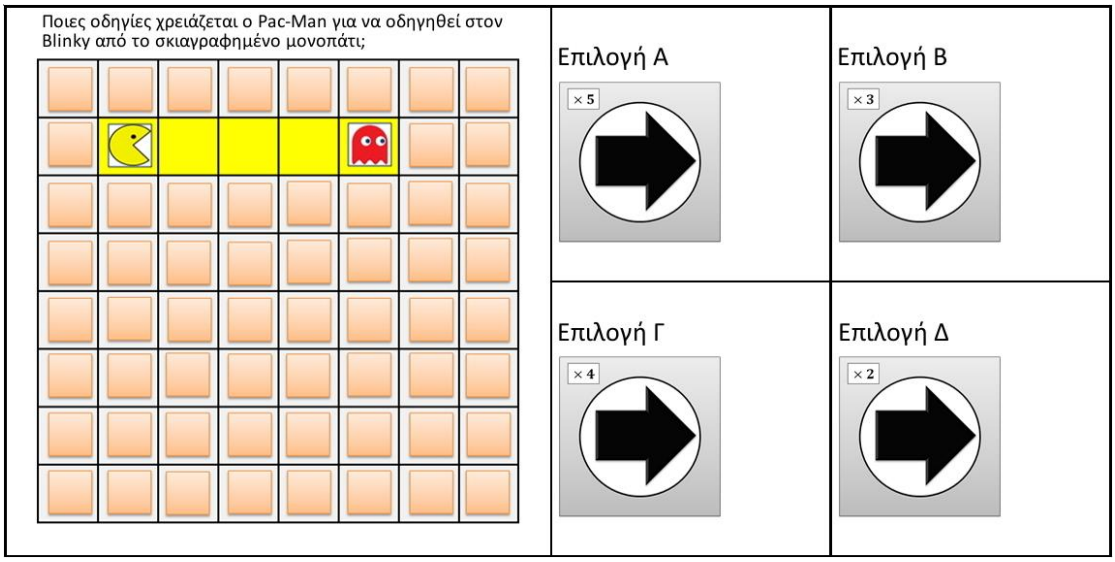
Να επισημαίνεται μόνο μία έλλειψη.

Επιλογή Α Επιλογή Β

Επιλογή Γ Επιλογή Δ

[4]Ερωτήσεις 5-8

Ερώτηση 5



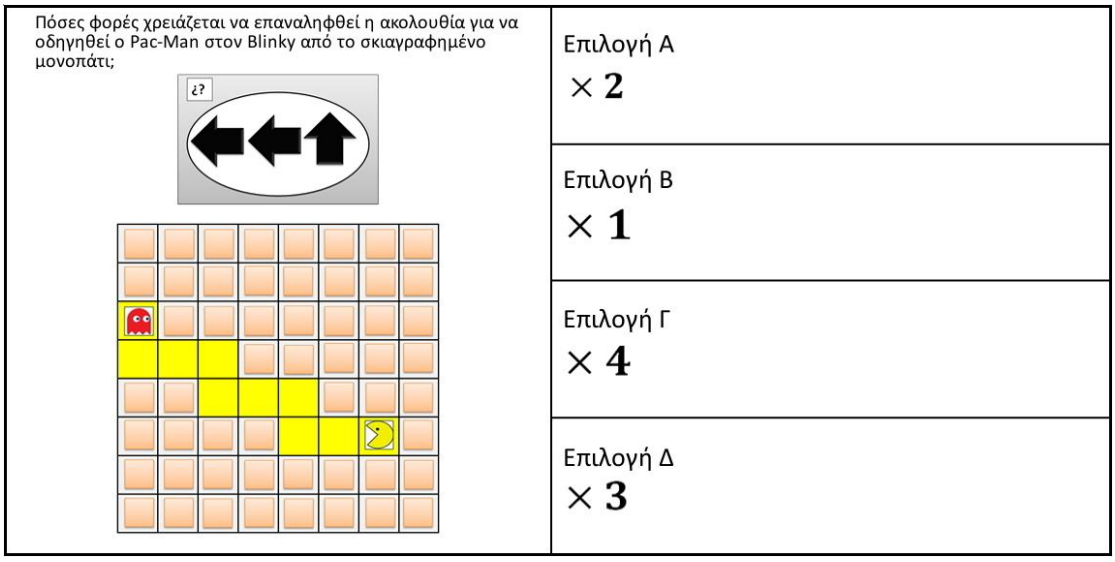
Επιλέξτε τη σωστή απάντηση:

Να επισημαίνεται μόνο μία έλλειψη.

◯ Επιλογή Α ◯ Επιλογή Β

◯ Επιλογή Γ ◯ Επιλογή Δ

Ερώτηση 6



Επιλέξτε τη σωστή απάντηση:

Να επισημαίνεται μόνο μία έλλειψη.

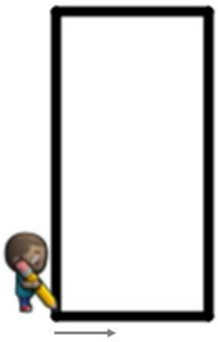
◯ Επιλογή Α ◯ Επιλογή Β

◯ Επιλογή Γ ◯ Επιλογή Δ

[4] Το τεστ στην αρχική του μορφή βρίσκεται στη διεύθυνση: http://goo.gl/IYEKMB (Spanish version)

Ερώτηση 7

Ποιες οδηγίες χρειάζεται να ακολουθήσει ο καλλιτέχνης για να σχεδιάσει το παραλληλόγραμμο **μία φορά** (50 pixels πλάτος και 100 pixels ύψος). Σε ποιο από τα βήματα των οδηγιών υπάρχει **λάθος**;



```
from turtle import *
canvas = Screen()
artist = Turtle()
```

Βήμα Α

```
for i in range(4):
    artist.forward(50)
    artist.left(90)        → Βήμα Β
    artist.forward(100)    → Βήμα Γ
    artist.left(90)        → Βήμα Δ
```

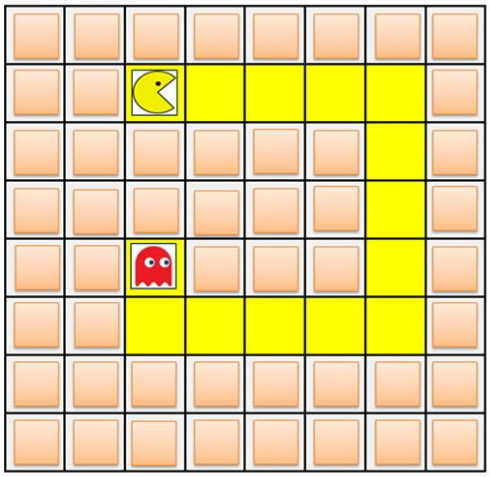
Επιλέξτε τη σωστή απάντηση:

Να επισημαίνεται μόνο μία έλλειψη.

- Επιλογή Α
- Επιλογή Β
- Επιλογή Γ
- Επιλογή Δ

Ερώτηση 8

Ποιες οδηγίες χρειάζεται ο Pac-Man για να οδηγηθεί στον Blinky από το σκιαγραφημένο μονοπάτι; Κάθε τετράγωνο έχει μέγεθος πλευράς 30 pixels.



```
from turtle import *
canvas = Screen()
pacman = Turtle()
pixels = 0
blinky = 390
```

Επιλογή Α

```
for i in range(4):
    for j in range(3):
        pacman.forward(30)
        pixels+=30
    pacman.right(90)
pacman.forward(30)
pixels+=30
```

Επιλογή Β

```
for i in range(3):
    for j in range(4):
        pacman.forward(30)
        pixels+=30
    pacman.right(90)
pacman.forward(30)
pixels+=30
```

Επιλογή Γ

```
for i in range(3):
    for j in range(4):
        pacman.forward(30)
        pixels+=30
        pacman.right(90)
pacman.forward(30)
pixels+=30
```

Επιλογή Δ

```
for i in range(4):
    pacman.forward(30)
    pixels+=30
for j in range(3):
    pacman.right(90)
pacman.forward(30)
pixels+=30
```

Επιλέξτε τη σωστή απάντηση:

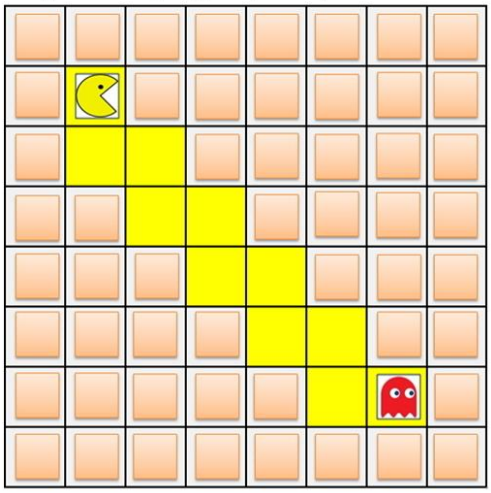
Να επισημαίνεται μόνο μία έλλειψη.

- Επιλογή Α
- Επιλογή Β
- Επιλογή Γ
- Επιλογή Δ

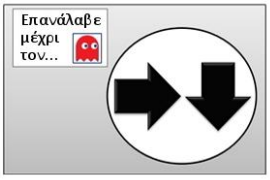
[5]Ερωτήσεις 9-12

Ερώτηση 9

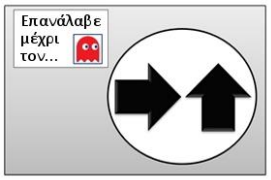
Ποιες οδηγίες χρειάζεται ο Pac-Man για να οδηγηθεί στον Blinky από το σκιαγραφημένο μονοπάτι;



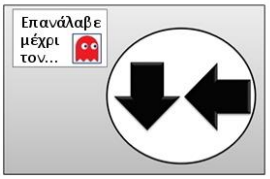
Επιλογή Α



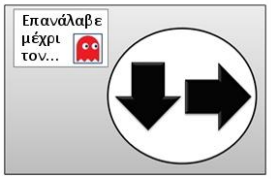
Επιλογή Β



Επιλογή Γ



Επιλογή Δ



Επιλέξτε τη σωστή απάντηση:

Να επισημαίνεται μόνο μία έλλειψη.

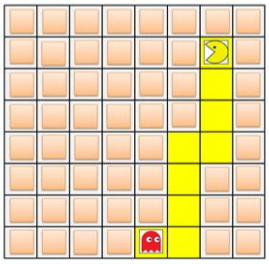
◯ Επιλογή Α ◯ Επιλογή Β

◯ Επιλογή Γ ◯ Επιλογή Δ

Ερώτηση 10

Ποιό βήμα λείπει από τις παρακάτω οδηγίες που χρειάζεται ο Pac-Man για να οδηγηθεί στον Blinky από το σκιαγραφημένο μονοπάτι; Κάθε τετράγωνο έχει μέγεθος πλευράς 30 pixels.

```
while pixels!=blinky:
    pacman.left(90)
    pacman.forward(30)
    pixels+=30
    ????????
    pacman.forward(30)
    pixels+=30
    pacman.right(90)
    pacman.forward(30)
    pixels+=30
```



```
from turtle import *
canvas = Screen()
pacman = Turtle()
pixels = 0
blinky = 240
pacman.left(180)
```

Επιλογή Α

```
pacman.left(90)
```

Επιλογή Β

```
pacman.right(90)
```

Επιλογή Γ

```
pacman.forward(30)
pixels+=30
```

Επιλογή Δ

Δεν λείπει κανένα βήμα

Επιλέξτε τη σωστή απάντηση:

Να επισημαίνεται μόνο μία έλλειψη.

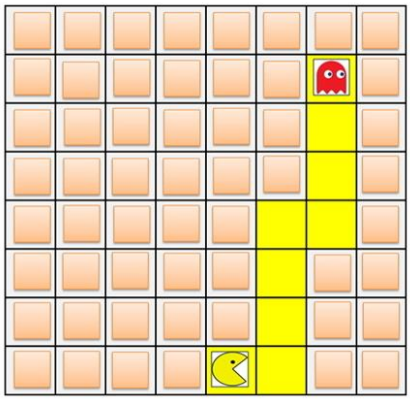
◯ Επιλογή Α ◯ Επιλογή Β

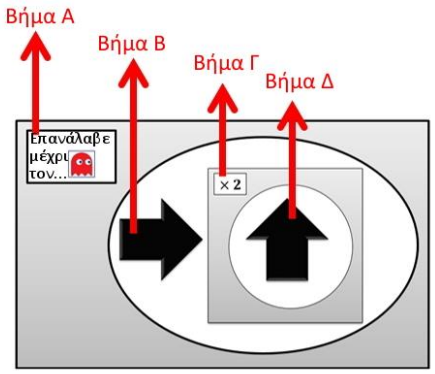
◯ Επιλογή Γ ◯ Επιλογή Δ

[5] Το τεστ στην αρχική του μορφή βρίσκεται στη διεύθυνση: http://goo.gl/IYEKMB (Spanish version)

Ερώτηση 11

Παρακάτω αναφέρονται οι οδηγίες που πρέπει να ακολουθήσει ο Pac-Man για να φτάσει στον Blinky από το σκιαγραφημένο μονοπάτι. Σε ποιο από τα βήματα των οδηγιών υπάρχει λάθος;





Επιλέξτε τη σωστή απάντηση:

Να επισημαίνεται μόνο μία έλλειψη.

Επιλογή Α Επιλογή Β

Επιλογή Γ Επιλογή Δ

Ερώτηση 12

Ποιες οδηγίες χρειάζεται να ακολουθήσει ο καλλιτέχνης για να σχεδιάσει τη σκάλα που φτάνει στο λουλούδι; Κάθε σκαλοπάτι έχει μέγεθος 30 pixels.

```
from turtle import *
canvas = Screen()
artist = Turtle()
pixels = 0
flower = 840
artist.left(90)
```

Επιλογή Α

```
while pixels!=flower:
    for i in range(4):
        artist.forward(30)
        pixels+=30
        artist.right(90)
    penup()
    artist.forward(30)
    pendown()
```

Επιλογή Β

```
while pixels!=flower:
    for i in range(4):
        artist.forward(120)
        pixels+=120
        artist.right(90)
    penup()
    artist.forward(30)
    pendown()
```

Επιλογή Γ

```
while pixels!=flower:
    for i in range(4):
        artist.forward(30)
        pixels+=30
        artist.right(90)
    penup()
    artist.forward(210)
    pendown()
```

Επιλογή Δ

```
while pixels!=flower:
    for i in range(7):
        artist.forward(30)
        pixels+=30
        artist.right(90)
    penup()
    artist.forward(30)
    pendown()
```

Επιλέξτε τη σωστή απάντηση:

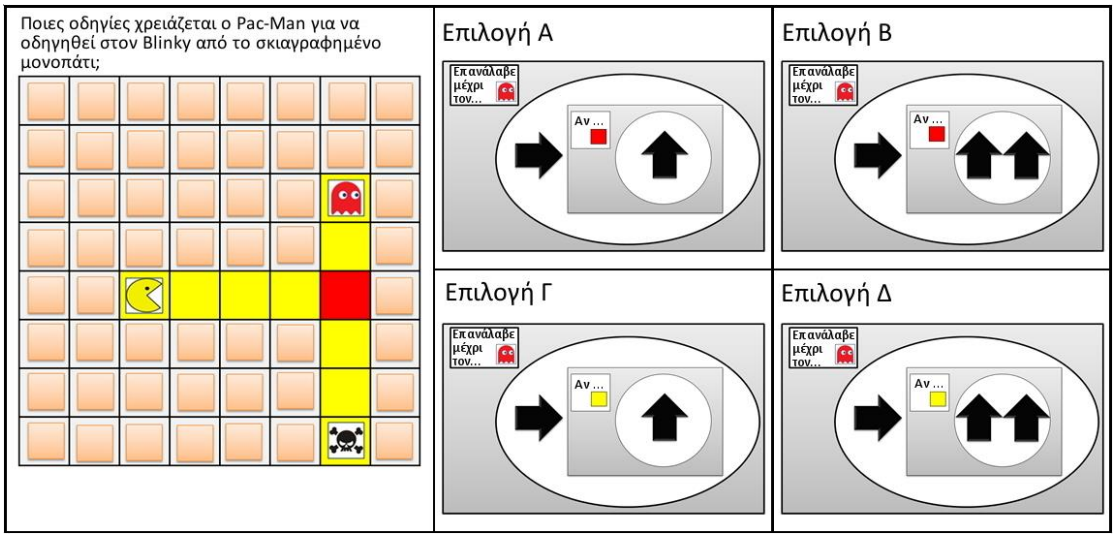
Να επισημαίνεται μόνο μία έλλειψη.

Επιλογή Α Επιλογή Β

Επιλογή Γ Επιλογή Δ

[6]Ερωτήσεις 13-16

Ερώτηση 13



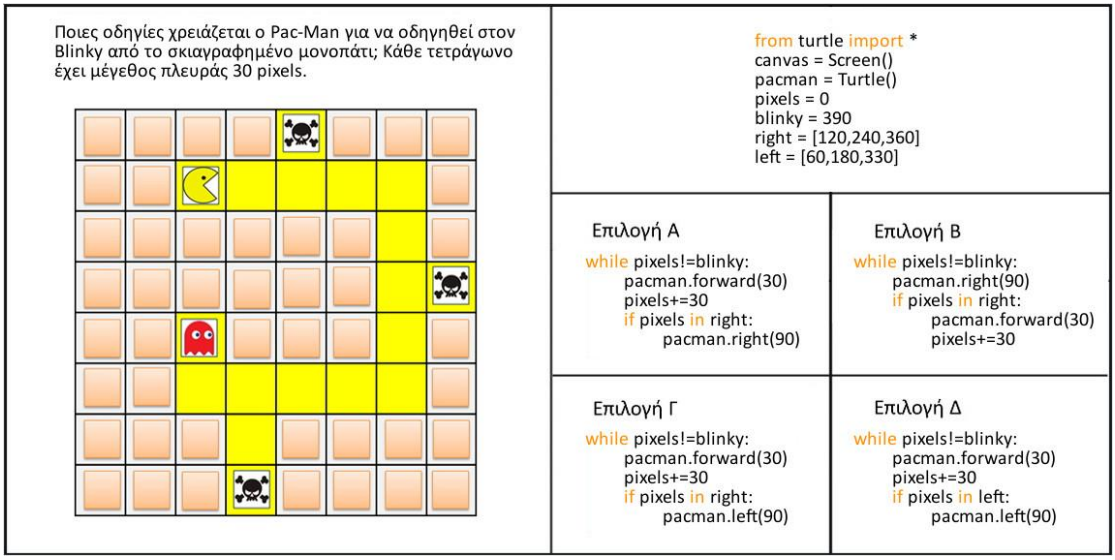
Επιλέξτε τη σωστή απάντηση:

Να επισημαίνεται μόνο μία έλλειψη.

◯ Επιλογή Α ◯ Επιλογή Β

◯ Επιλογή Γ ◯ Επιλογή Δ

Ερώτηση 14



Επιλέξτε τη σωστή απάντηση:

Να επισημαίνεται μόνο μία έλλειψη.

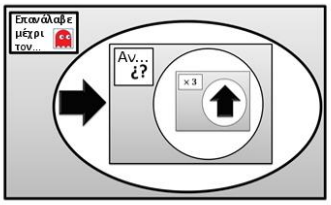
◯ Επιλογή Α ◯ Επιλογή Β

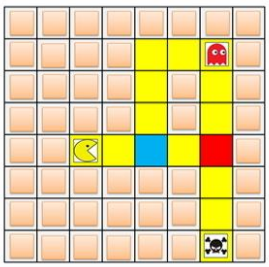
◯ Επιλογή Γ ◯ Επιλογή Δ

[6] Το τεστ στην αρχική του μορφή βρίσκεται στη διεύθυνση: http://goo.gl/IYEKMB (Spanish version)

Ερώτηση 15

Τι λείπει από τις παρακάτω οδηγίες ώστε ο Pac-Man να οδηγηθεί στον Blinky από το σκιαγραφημένο μονοπάτι;





Επιλογή Α

Επιλογή Β

Επιλογή Γ

Επιλογή Δ

Οι Επιλογές Α και Γ είναι σωστές

Επιλέξτε τη σωστή απάντηση:

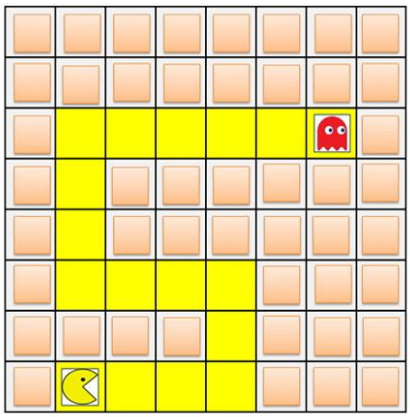
Να επισημαίνεται μόνο μία έλλειψη.

◯ Επιλογή Α ◯ Επιλογή Β

◯ Επιλογή Γ ◯ Επιλογή Δ

Ερώτηση 16

Παρακάτω αναφέρονται οι οδηγίες που πρέπει να ακολουθήσει ο Pac-Man για να φτάσει στον Blinky από το σκιαγραφημένο μονοπάτι. Σε ποιο από τα βήματα των οδηγιών υπάρχει **λάθος**; Κάθε τετράγωνο έχει μέγεθος πλευράς 30 pixels.



```
from turtle import *
canvas = Screen()
pacman = Turtle()
pixels = 0
blinky = 480
left = [90,150]
right = [240,330]
```

```
while pixels!=blinky:
    pacman.forward(30)
    pixels+=30
    if pixels in left:          → Βήμα Α
        pacman.left(90)         → Βήμα Β
    if pixels in right:         → Βήμα Γ
        pacman.forward(30)      → Βήμα Δ
```

Επιλέξτε τη σωστή απάντηση:

Να επισημαίνεται μόνο μία έλλειψη.

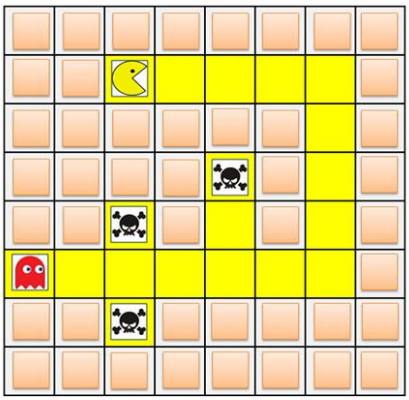
◯ Επιλογή Α ◯ Επιλογή Β

◯ Επιλογή Γ ◯ Επιλογή Δ

[7]Ερωτήσεις 17-20

Ερώτηση 17

Ποιες οδηγίες χρειάζεται ο Pac-Man για να οδηγηθεί στον Blinky από το σκιαγραφημένο μονοπάτι; Κάθε τετράγωνο έχει μέγεθος πλευράς 30 pixels.



```
from turtle import *
canvas = Screen()
pacman = Turtle()
pixels =0
blinky =420
intersections =[120,240,300,360]
```

Επιλογή Α

```
while pixels!=blinky:
    if pixels not in intersections\
    or pixels>250:
        pacman.forward(30)
        pixels+=30
    else:
        pacman.left(90)
        pixels+=30
```

Επιλογή Β

```
while pixels!=blinky:
    if pixels not in intersections\
    or pixels>250:
        pacman.forward(30)
        pixels+=30
    else:
        pacman.right(90)
        pixels+=30
```

Επιλογή Γ

```
while pixels!=blinky:
    if pixels in intersections:
        pacman.right(90)
        pixels+=30
    else:
        pacman.forward(30)
        pixels+=30
```

Επιλογή Δ

```
while pixels!=blinky:
    if pixels in intersections:
        pacman.left(90)
        pixels+=30
    else:
        pacman.forward(30)
        pixels+=30
```

Επιλέξτε τη σωστή απάντηση:

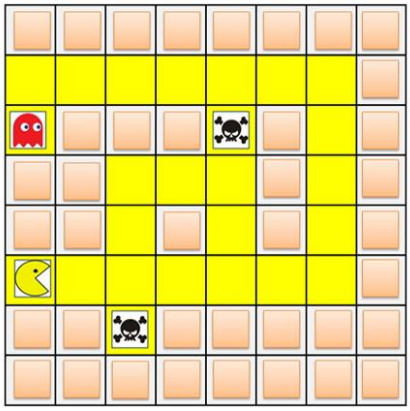
Να επισημαίνεται μόνο μία έλλειψη.

◯ Επιλογή Α ◯ Επιλογή Β

◯ Επιλογή Γ ◯ Επιλογή Δ

Ερώτηση 18

Ποιες οδηγίες χρειάζεται ο Pac-Man για να οδηγηθεί στον Blinky από το σκιαγραφημένο μονοπάτι; Κάθε τετράγωνο έχει μέγεθος πλευράς 30 pixels.



```
from turtle import *
canvas = Screen()
pacman = Turtle()
pixels = 0
blinky = 510
intersections = [60,120,180,300,360,480]
```

Επιλογή Α

```
while pixels!=blinky:
    if pixels not in intersections\
    or pixels<130 or pixels==360:
        pacman.forward(30)
        pixels+=30
    else:
        pacman.left(90)
        pixels+=30
```

Επιλογή Β

```
while pixels!=blinky:
    if pixels not in intersections\
    or pixels<130 or pixels==360:
        pacman.forward(30)
        pixels+=30
    else:
        pacman.right(90)
        pixels+=30
```

Επιλογή Γ

```
while pixels!=blinky:
    if pixels in intersections:
        pacman.right(90)
        pixels+=30
    else:
        pacman.forward(30)
        pixels+=30
```

Επιλογή Δ

```
while pixels!=blinky:
    if pixels in intersections:
        pacman.left(90)
        pixels+=30
    else:
        pacman.forward(30)
        pixels+=30
```

Επιλέξτε τη σωστή απάντηση:

Να επισημαίνεται μόνο μία έλλειψη.

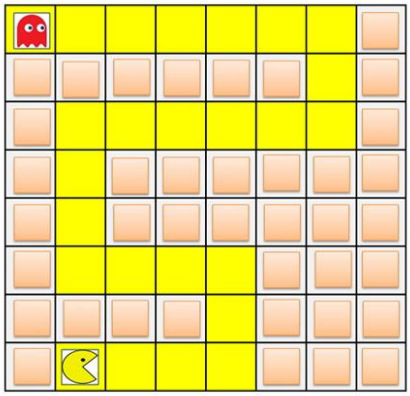
◯ Επιλογή Α ◯ Επιλογή Β

◯ Επιλογή Γ ◯ Επιλογή Δ

[7] Το τεστ στην αρχική του μορφή βρίσκεται στη διεύθυνση: http://goo.gl/IYEKMB (Spanish version)

Ερώτηση 19

Παρακάτω αναφέρονται οι οδηγίες που πρέπει να ακολουθήσει ο Pac-Man για να φτάσει στον Blinky από το σκιαγραφημένο μονοπάτι. Σε ποιο από τα βήματα των οδηγιών υπάρχει **λάθος**; Κάθε τετράγωνο έχει μέγεθος πλευράς 30 pixels.



```
from turtle import *
canvas = Screen()
pacman = Turtle()
pixels = 0
blinky = 720
intersections = [90,150,240,330,480,540]
right = [240,330]
```

```
while pixels!=blinky:
    if pixels not in intersections:
        pacman.forward(30)        → Βήμα Α
        pixels+=30
    else:
        if pixels in right:       → Βήμα Β
            pacman.left(90)       → Βήμα Γ
            pixels+=30
        else:
            pacman.right(90)      → Βήμα Δ
            pixels+=30
```

Επιλέξτε τη σωστή απάντηση:

Να επισημαίνεται μόνο μία έλλειψη.

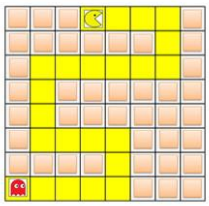
Επιλογή Α Επιλογή Β

Επιλογή Γ Επιλογή Δ

Ερώτηση 20

Ποιό βήμα λείπει από τις παρακάτω οδηγίες που χρειάζεται ο Pac-Man για να οδηγηθεί στον Blinky από το σκιαγραφημένο μονοπάτι; Κάθε τετράγωνο έχει μέγεθος πλευράς 30 pixels.

```
while pixels!=blinky:
    if pixels not in intersections:
        pacman.forward(30)
        pixels+=30
    else:
        if pixels<=150 or pixels>=480:
            pacman.right(90)
            pixels+=30
        else:
            ????????
```



```
from turtle import *
canvas = Screen()
pacman = Turtle()
pixels = 0
blinky =660
intersections = [90,150,300,390,480,540]
```

Επιλογή Α

```
pacman.forward(30)
pixels+=30
```

Επιλογή Β

```
pacman.right(90)
pixels+=30
```

Επιλογή Γ

```
pacman.left(90)
pixels+=30
```

Επιλογή Δ

Δεν λείπει κανένα βήμα

Επιλέξτε τη σωστή απάντηση:

Να επισημαίνεται μόνο μία έλλειψη.

Επιλογή Α Επιλογή Β

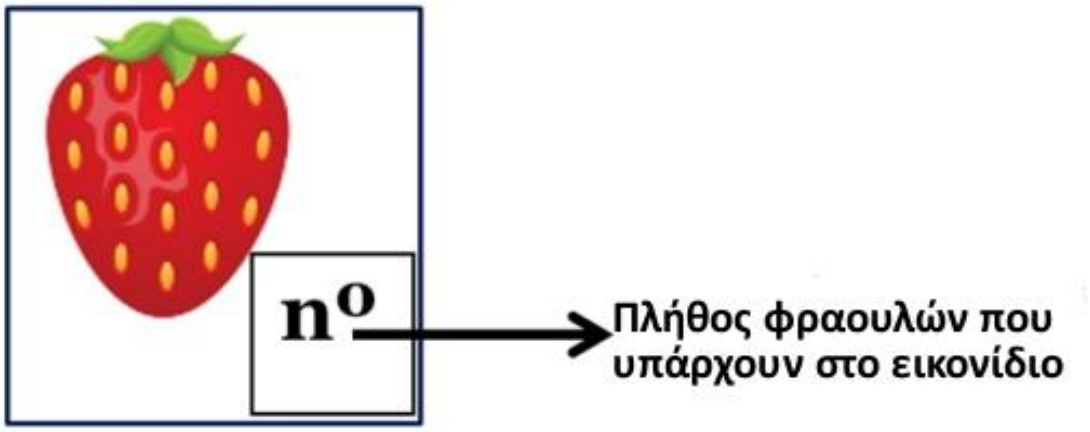
Επιλογή Γ Επιλογή Δ

[8]Ερωτήσεις 21-24

ΣΗΜΑΝΤΙΚΟ:ΔΙΑΒΑΣΤΕ ΠΡΟΣΕΚΤΙΚΑ

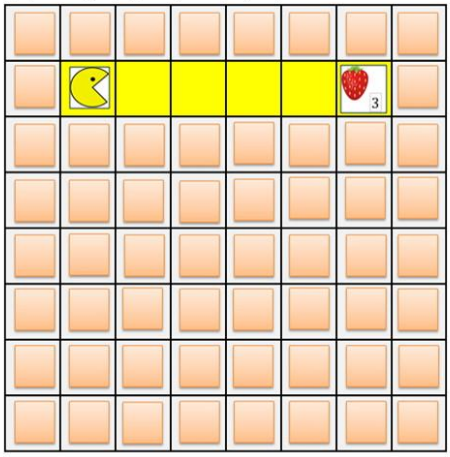
Σε αυτή την ομάδα ερωτήσεων το εικονίδιο της φράουλας εμφανίζεται σε κάποια τετράγωνα. Ο αριθμός κάτω δεξιά στο εικονίδιο υποδηλώνει το πλήθος των φραουλών του τετραγώνου.

[8] Το τεστ στην αρχική του μορφή βρίσκεται στη διεύθυνση: http://goo.gl/IYEKMB (Spanish version)



Ερώτηση 21

Ποιες οδηγίες χρειάζεται ο Pac-Man για να οδηγηθεί από το σκιαγραφημένο μονοπάτι στις φράουλες ώστε να καταναλώσει όσες φράουλες υπάρχουν στο εικονίδιο; Κάθε τετράγωνο έχει μέγεθος πλευράς 30 pixels.




```
from turtle import *
canvas = Screen()
pacman = Turtle()
pixels = 0
target = 150
strawberries = 3
```

Επιλογή Α

```
while pixels!=target:
    pacman.forward(30)
    pixels+=30
for i in range(3):
    strawberries-=1
```

Επιλογή Β

```
while pixels!=target:
    pacman.forward(30)
    pixels+=30
for i in range(4):
    strawberries-=1
```

Επιλογή Γ

```
while pixels!=target:
    pacman.forward(30)
    pixels+=30
for i in range(5):
    strawberries-=1
```

Επιλογή Δ

```
while pixels!=target:
    pacman.forward(30)
    pixels+=30
    for i in range(3):
        strawberries-=1
```

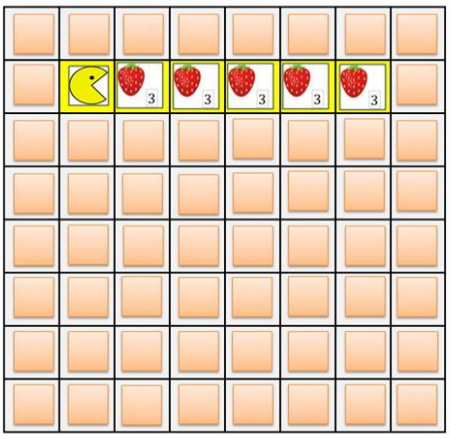
Επιλέξτε τη σωστή απάντηση:

Να επισημαίνεται μόνο μία έλλειψη.

- ◯ Επιλογή Α
- ◯ Επιλογή Β
- ◯ Επιλογή Γ
- ◯ Επιλογή Δ

Ερώτηση 22

Ποιες οδηγίες χρειάζεται ο Pac-Man για να οδηγηθεί από το σκιαγραφημένο μονοπάτι στις φράουλες ώστε να καταναλώσει όσες φράουλες υπάρχουν στο εικονίδιο; Κάθε τετράγωνο έχει μέγεθος πλευράς 30 pixels.




```
from turtle import *
canvas = Screen()
pacman = Turtle()
pixels = 0
target =150
strawberries = 15
```

Επιλογή Α

```
while pixels!=target:
    for i in range(5):
        pacman.forward(30)
        pixels+=30
    for i in range(3):
        strawberries-=1
```

Επιλογή Β

```
while pixels!=target:
    pacman.forward(30)
    pixels+=30
    for i in range(3):
        strawberries-=1
```

Επιλογή Γ

```
while pixels!=target:
    for i in range(3):
        pacman.forward(30)
        pixels+=30
    for i in range(5):
        strawberries-=1
```

Επιλογή Δ

```
while pixels!=target:
    pacman.forward(30)
    pixels+=30
for i in range(3):
    strawberries-=1
```

Επιλέξτε τη σωστή απάντηση:

Να επισημαίνεται μόνο μία έλλειψη.

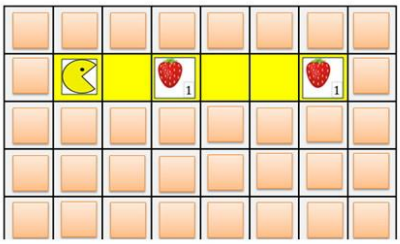
Επιλογή Α Επιλογή Β

Επιλογή Γ Επιλογή Δ

Ερώτηση 23

Τι λείπει από τις παρακάτω οδηγίες για να οδηγηθεί ο Pac-Man από το σκιαγραφημένο μονοπάτι στις φράουλες ώστε να καταναλώσει όλες τις φράουλες που υπάρχουν στα εικονίδια; Κάθε τετράγωνο έχει μέγεθος πλευράς 30 pixels.

```
while pixels!=destination:
    for i in range(????????):
        pacman.forward(30)
        pixels+=30
    if pixels==target1 or pixels==target2:
        strawberries-=1
```



```
from turtle import *
canvas = Screen()
pacman = Turtle()
pixels = 0
destination = 150
target1 = 60
target2 = 150
strawberries = 2
```

Επιλογή Α

1 φορά

Επιλογή Β

2 φορές

Επιλογή Γ

3 φορές

Επιλογή Δ

5 φορές

Επιλέξτε τη σωστή απάντηση:

Να επισημαίνεται μόνο μία έλλειψη.

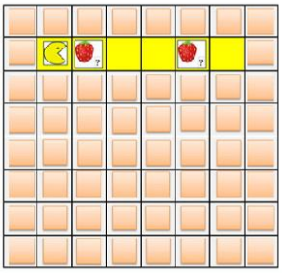
Επιλογή Α Επιλογή Β

Επιλογή Γ Επιλογή Δ

Ερώτηση 24

Ποιο βήμα λείπει από τις παρακάτω οδηγίες για να οδηγηθεί ο Pac-Man από το σκιαγραφημένο μονοπάτι στις φράουλες ώστε να καταναλώσει όλες τις φράουλες που υπάρχουν στα εικονίδια (άγνωστος αριθμός); Κάθε τετράγωνο έχει μέγεθος πλευράς 30 pixels.

```
while pixels!=destination:
    pacman.forward(30)
    pixels+=30
    if pixels==anyStrawberries1:
        ????????
            strawberries1-=1
    elif pixels==anyStrawberries2:
        ????????
            strawberries2-=1
```



```
from turtle import *
import random
canvas = Screen()
pacman = Turtle()
pixels = 0
destination = 150
strawberries1=random.randint(1,10)
strawberries2=random.randint(1,10)
anyStrawberries1,anyStrawberries2=30,120
```

Επιλογή Α

```
while pixels!=destination/ while pixels!=destination
```

Επιλογή Β

```
while pixels==destination/ while pixels==destination
```

Επιλογή Γ

```
while strawberries1!=0/ while strawberries2!=0
```

Επιλογή Δ

```
while strawberries1==0/ while strawberries2==0
```

Επιλέξτε τη σωστή απάντηση:

Να επισημαίνεται μόνο μία έλλειψη.

Επιλογή Α Επιλογή Β

Επιλογή Γ Επιλογή Δ

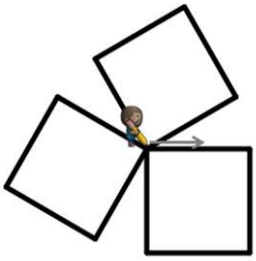
[9]Ερωτήσεις 25-28

Ερώτηση 25

Το παρακάτω σύνολο εντολών ονομάζεται 'my function' και σχεδιάζει ένα τετράγωνο με μέγεθος πλευράς 100 pixels:

```
def myfunction():
    for i in range(4):
        artist.forward(100)
        artist.right(90)
```

Ποιες οδηγίες χρειάζεται να ακολουθήσει ο καλλιτέχνης για να σχεδιάσει το ακόλουθο σχήμα; Κάθε τετράγωνο έχει μέγεθος πλευράς 100 pixels.



```
from turtle import *
canvas = Screen()
artist = Turtle()
pixels =0
```

Επιλογή Α

```
for i in range(3):
    myfunction()
artist.right(120)
```

Επιλογή Β

```
for i in range(3):
    myfunction()
    artist.right(120)
```

Επιλογή Γ

```
for i in range(4):
    myfunction()
artist.right(90)
```

Επιλογή Δ

```
for i in range(4):
    myfunction()
    artist.right(90)
```

Επιλέξτε τη σωστή απάντηση:

Να επισημαίνεται μόνο μία έλλειψη.

◯ Επιλογή Α ◯ Επιλογή Β

◯ Επιλογή Γ ◯ Επιλογή Δ

Ερώτηση 26

Το παρακάτω σύνολο εντολών ονομάζεται 'my function' και σχεδιάζει ένα τρίγωνο με μέγεθος πλευράς 50 pixels:

```
def myfunction():
    for i in range(3):
        artist.forward(50)
        artist.left(120)
```

Οι παρακάτω οδηγίες θα βοηθήσουν τον καλλιτέχνη να σχεδιάσει το ακόλουθο σχήμα. Κάθε τρίγωνο έχει μέγεθος πλευράς 50 pixels. Τι λείπει από τις οδηγίες;

```
for i in range(???):
    myfunction()
    penup()
    artist.forward(50)
    pendown()
```

```
from turtle import *
canvas = Screen()
artist = Turtle()
pixels = 0
```

Επιλογή Α

15

Επιλογή Β

5

Επιλογή Γ

4

Επιλογή Δ

3

Επιλέξτε τη σωστή απάντηση:

Να επισημαίνεται μόνο μία έλλειψη.

◯ Επιλογή Α ◯ Επιλογή Β

◯ Επιλογή Γ ◯ Επιλογή Δ

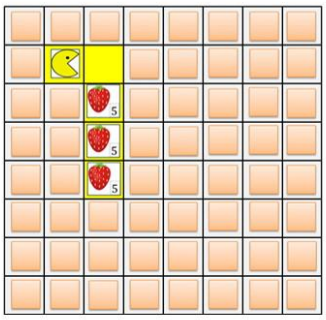
[9] Το τεστ στην αρχική του μορφή βρίσκεται στη διεύθυνση: http://goo.gl/IYEKMB (Spanish version)

Ερώτηση 27

Το παρακάτω σύνολο εντολών ονομάζεται 'get 5':

```
def get5():
    for i in range(5):
        global strawberries
        strawberries-=1
```

Ποιες οδηγίες χρειάζεται ο Pac-Man για να οδηγηθεί από το σκιαγραφημένο μονοπάτι στις φράουλες ώστε να καταναλώσει όλες τις φράουλες που υπάρχουν στα εικονίδια; Κάθε τετράγωνο έχει μέγεθος πλευράς 30 pixels.



```
from turtle import *
canvas = Screen()
pacman = Turtle()
pixels =0
strawberries=15
```

Επιλογή Α

```
pacman.forward(30)
pixels+=30
pacman.right(90)
for i in range(3):
    pacman.forward(30)
    pixels+=30
    get5()
```

Επιλογή Β

```
pacman.forward(30)
pixels+=30
pacman.right(90)
for i in range(3):
    get5()
pacman.forward(30)
pixels+=30
```

Επιλογή Γ

```
pacman.forward(30)
pixels+=30
pacman.right(90)
for i in range(5):
    pacman.forward(30)
    pixels+=30
    get5()
```

Επιλογή Δ

```
pacman.forward(30)
pixels+=30
pacman.right(90)
for i in range(5):
    get5()
pacman.forward(30)
pixels+=30
```

Επιλέξτε τη σωστή απάντηση:

Να επισημαίνεται μόνο μία έλλειψη.

◯ Επιλογή Α ◯ Επιλογή Β

◯ Επιλογή Γ ◯ Επιλογή Δ

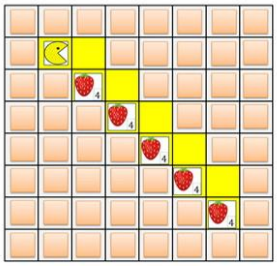
Ερώτηση 28

Το παρακάτω σύνολο εντολών ονομάζεται 'move and get 4':

```
def moveandget4():
    global pixels
    pacman.forward(30)
    pixels+=30
    pacman.right(90)
    pacman.forward(30)
    pixels+=30
    for i in range(4):
        global strawberries
        strawberries-=1
    pacman.left(90)
```

Τι λείπει από τις παρακάτω οδηγίες για να οδηγηθεί ο Pac-Man από το σκιαγραφημένο μονοπάτι στις φράουλες ώστε να καταναλώσει όλες τις φράουλες που υπάρχουν στα εικονίδια; Κάθε τετράγωνο έχει μέγεθος πλευράς 30 pixels.

```
for i in range(???):
    moveandget4()
```



```
from turtle import *
canvas = Screen()
pacman = Turtle()
pixels = 0
strawberries = 20
```

Επιλογή Α

3

Επιλογή Β

4

Επιλογή Γ

5

Επιλογή Δ

6

Επιλέξτε τη σωστή απάντηση:

Να επισημαίνεται μόνο μία έλλειψη.

◯ Επιλογή Α ◯ Επιλογή Β

◯ Επιλογή Γ ◯ Επιλογή Δ