Korištenje suvremenih izlaznih i ulaznih jedinica u nastavi programiranja – metodički pristup

Goran Igaly[1], Goranka Nogo[2]

1. Uvod

Posljednjih godinu dana u gotovo svim medijima mogli smo pročitati naslove poput *STEM revolucija: 920 ustanova dobiva 15.000 setova za robotiku*[1], *STEM revolucija u hrvatskim školama*[2], *Micro:bitovi stižu školama, STEM revolucija traje*[3], ... Tekstovi ispod navedenih naslova uvijek sadrže općenite pojmove kao što su *roboti*, *STEM revolucija* i *micro:bitovi*. No, u pravilu, iz njih ne možemo razabrati zašto bi npr. korištenje robota u nastavi informatike bilo korak (ili više njih) naprijed u odnosu na postojeće stanje.

Računarstvo i digitalna tehnologija prisutni su svuda oko nas, ali se s vremenom u formalnom obrazovanju naglasak stavio na korištenje gotovih sadržaja, dok se stvaranju vlastitih sadržaja, s izuzetkom škola orijentiranih prema obrazovanju, postupno pridaje sve manje pozornosti. To je vidljivo i u nastavi informatike gdje se nastava programiranja izvodi većinom u školama koje obrazuju kadrove koji će kasnije biti profesionalno vezani uz programiranje, dok se u većini ostalih škola programiranje niti ne spominje zbog shvaćanja da je znanje programiranja danas potrebno samo onima koji će se kasnije aktivno njime na neki način baviti. Međutim, razumijevanje koncepta programiranja bi, uz znanje korištenja računalom, trebalo biti dio opće pismenosti i za osobe koje se kasnije ne namjeravaju profesionalno time baviti budući da gotovo svi predmeti s kojima se susrećemo u svakodnevnom životu u sebi imaju ugrađen neki elektronički sklop koji izvodi određene programe.

U ovom ćemo članku učitelje informatike upoznati s mogućnostima osuvremenjivanja nastave programiranja korištenjem suvremenih i dostupnih izlaznih i ulaznih jedinica, čime se omogućuje temeljno razumijevanje načina na koji rade svi uređaji koji imaju neki oblik programiranog djelovanja.

[1]Goran Igaly, PMF-Matematički odsjek, Zagreb
[2]Goranka Nogo, PMF-Matematički odsjek, Zagreb

Navest ćemo primjere aktivnosti korištenja suvremenih izlaznih i ulaznih jedinica koje se mogu provoditi u nastavi informatike, prilikom učenja programiranja. Također, osvrnut ćemo se i na potencijalne probleme vezane uz programiranje fizičkih objekata. Aktivnosti iz članka provedene su sa studentima diplomskog sveučilišnog studija Matematika i informatika; smjer: nastavnički, u okviru kolegija Metodika nastave informatike 2, nastavne godine 2016./17. Kao izlazne i ulazne jedinice korišteni su BBC micro:bitovi te edukacijski robot mBot uz dodatne senzore i display.

2. Kratki opis pojmova

Radi boljeg razumijevanja, u ovom poglavlju dat ćemo kratki opis korištenih pojmova.

Mikrokontroler je malo računalo na jednom integriranom krugu (čipu) koje se uobičajeno sastoji od procesora, memorije i programabilnog vanjskog ulazno/izlaznog uređaja. Na čipu je obično i programabilna (flash) memorija (elektronički nepromjenjivi medij za pohranu podataka koji se može električki izbrisati i reprogramirati), te manja količina RAM-a. Dizajnirani su za ugrađene aplikacije, za razliku od mikroprocesora koji se koriste u osobnim računalima ili u drugim aplikacijama opće namjene.

BBC Micro, punim nazivom: *BBC Microcomputer System,* serija je mikroračunala koju je dizajnirala i proizvodila tvrtka **Acorn.** Razvijena je za BBC-jev projekt *Computer Literacy* (računalna pismenost). Postojala je i TV serija na BBC 2 *The Computer Programme* (1982.). Računala su na tržištu bila dostupna od 1981. do 1994. godine i u tom razdoblju prodano je 1.5 milijuna primjeraka.

Arhitektura ARM (1983. ARM = **Acorn** RISC Machine; 1990. ARM = **Advanced** RISC Machines) bazira se na arhitekturi RISC (*Reduced Instruction Set Computer*) čije su značajke jeftiniji procesor, manja potrošnja energije i manje zagrijavanje. Procesori temeljeni na ARM nalaze se svuda – mikrokontroleri u *ugrađenim sustavima*, pametni telefoni, tableti, pametni televizori itd. **Acorn** je britanska multinacionalna tvrtka za poluvodiče i softver, sa sjedištem u Cambridgeu.

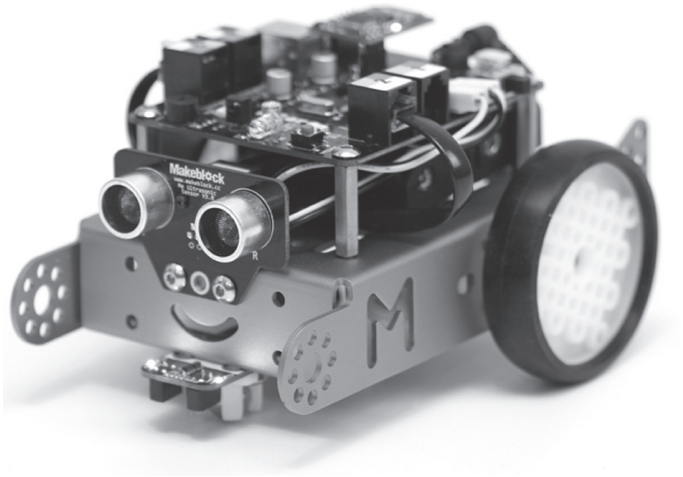
micro:bit je uklopljeni sustav utemeljen na ARM. Dizajnirao ga BBC za korištenje u računalnom obrazovanju u UK. Javnosti je predstavljen u veljači 2016. U listopadu 2016., BBC je predao micro:bit neprofitnoj organizaciji *Microbit Education Foundation*.

Više detalja može se pronaći na adresi [4].

Arduino je elektronička platforma otvorenog koda temeljena na jednostavnom hardveru i softveru. Arduino pločice mogu čitati različite ulaze – npr. svjetlo na senzoru, prst na gumbu ili Twitter poruku – i pretvoriti ga u izlaz – aktiviranje motora, uključivanje LED dioda, objavljivanje podataka na internetu. Ponašanjem pločice upravlja se slanjem niza naredbi (programa) *mikrokontroleru* koji se nalazi na pločici**.** Koristi se programski jezik Arduino (temeljen na jeziku Wiring) i Arduino softver (IDE), temeljen na sustavu Processing.

Programski jezik mBlock izuzetno je sličan programskom jeziku *Scratch***.** Ima intuitivno sučelje bazirano na tehnici *drag and drop* za povlačenje naredbi. Posjeduje i posebni izbornik sa svim naredbama za upravljanje robotom mBot. U pozadini je programski jezik Arduino C. Izvođenje programa zahtijeva prethodnu instalaciju (razlika u usporedbi s micro:bitom).

Robot mBot edukacijski je **Arduino** robot. Služi za učenje programiranja i elektronike. Programiranje robota jest upisivanje korisničkog programa u njegovu *flash memoriju***.** Robot ima senzore (ulaz) motore, lampice, display, zvučnik (izlaz), što sve skupa omogućava komunikaciju (ulaz/izlaz).



Više detalja o robotu može se pronaći na adresi [5].

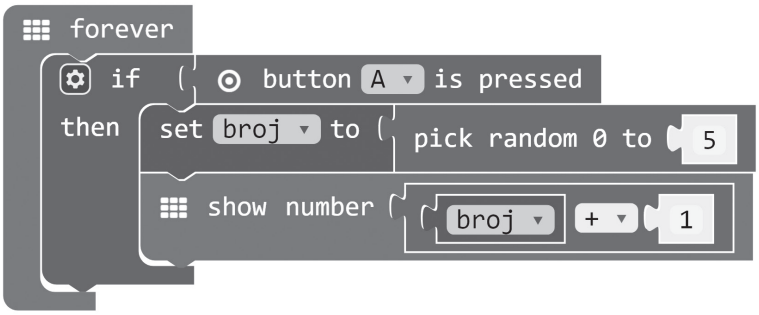
3. Primjeri aktivnosti – micro:bit

U ovom poglavlju bit će opisane dvije aktivnosti vezane uz korištenje micro:bita. Ukazat ćemo na razlike u odnosu na klasično programiranje, te na probleme na koje smo naišli. Programi će biti pisani u programskom jeziku Microsoft Blocks [6]. Programi se mogu pisati i u programskom jeziku Python, koristeći npr. editor Mu [7].

AKTIVNOST 1. Napisati program koji simulira bacanje simetrične igraće kocke (brojevi od *1* do *6*) kada na micro:bitu pritisnemo programsku tipku označenu s *A*.

Koji su nam blokovi (naredbe) potrebni? Idemo redom:

Primijetimo da prvo uvodimo petlju, a tek nakon toga naredbu *if*. Konačno rješenje je:



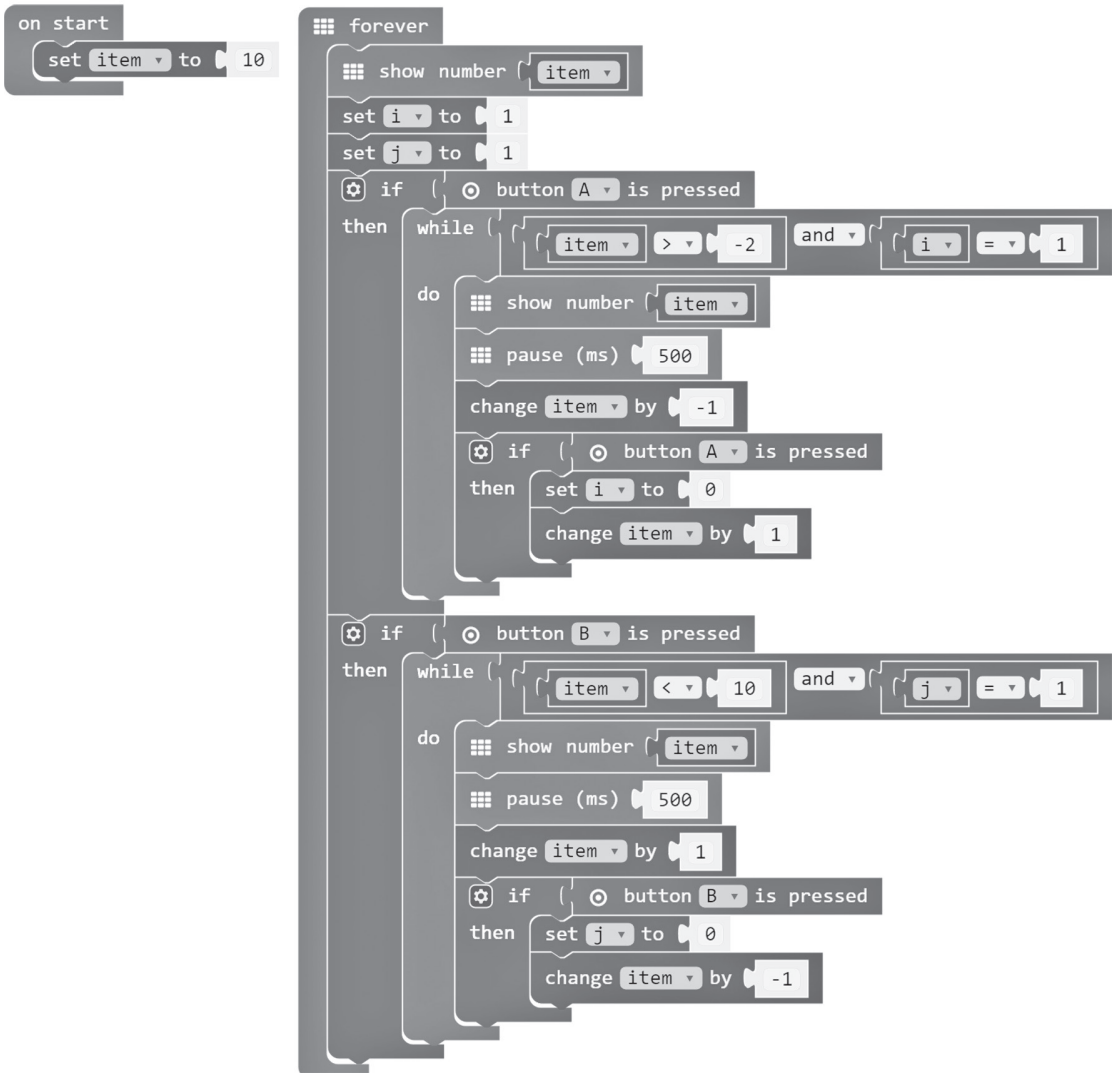
Napomenimo da se program sprema na računalo kao datoteka *.hex* klikom na *Download*. Analogno, za spremanje programa na uređaj, prethodno spojen USB kabelom na računalo, potrebno je kliknuti na *Download*.

Riječ je o poznatom programu koji se često navodi kao nešto složeniji motivacijski zadatak. Metodički gledano, već i u tako jednostavnom primjeru dolazi do izražaja posebnost programiranja micro:bita. U klasičnom programiranju tzv. beskonačnu petlju spominjemo (nakon što uvedemo pojam petlje) više kao izuzetak, odnosno kao primjer pogreške. Prilikom programiranja micro:bita, i kasnije mBota, beskonačna se petlja gotovo uvijek koristi budući da se mikrokontroleri obično koriste za odgovore na vanjske podražaje koje treba pratiti tijekom duljeg vremenskog razdoblja.

Sljedeći primjer je nešto kompliciraniji, a navodimo ga zato što su u njemu jako izraženi specifični problemi vezani uz programiranje fizičkih objekata.

AKTIVNOST 2. Napisati program koji simulira vožnju lifta u zgradi koja ima katove označene brojevima, od -2. do 10. Pritiskom na programsku tipku A lift se počinje gibati prema dolje, a ponovnim pritiskom na tu tipku lift se zaustavlja. Programska tipka B djeluje obratno.

Rješenje:



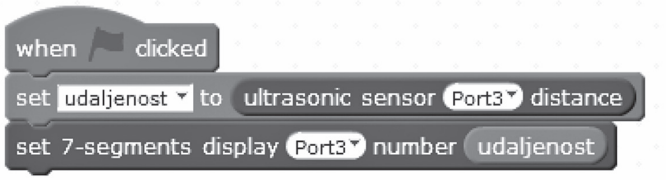
Od samoga koda u ovom su primjeru puno interesantniji problemi na koje smo naišli prilikom izvođenja programa. Prilikom izvođenja programa lift će se ponašati onako kako smo zamisliti samo ako odgovarajućom brzinom (ne presporo) pritiščemo programske tipke. Odgovarajuću programsku tipku potrebno je pritisnuti u trenutku izvršavanja naredbe *if*. Na primjer, ako zaustavnu tipku B pritisnemo nakon što se odgovarajuća naredba *if* izvrši, lift će se nastaviti kretati. Problem će biti manje izražen ako je razmak između pritiskanja pojedine tipke kratak.

4. Primjeri aktivnosti – mBot

U ovom poglavlju navest ćemo dvije povezane aktivnosti vezano uz korištenje edukacijskog robota mBot. Primjeri su zanimljivi kao motivacija za uvođenje petlje. I ovdje, za razliku od klasičnog programiranja, prvo uvodimo petlju, a tek nakon nje naredbu za kontrolu toka programa (grananje). Na kraju ćemo samo navesti nekoliko problema na koje smo naišli.

AKTIVNOST 3. Izmjeriti udaljenost pomoću ultrazvučnog senzora i prikazati je na 7-segmentnom displayu.

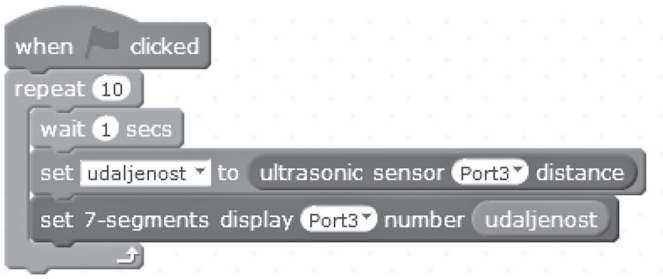
Rješenje:



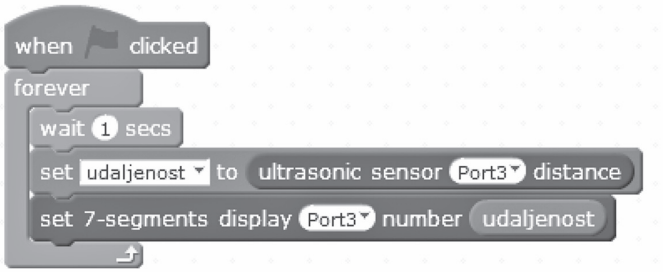
Primijetimo da smo ovdje uveli pojam varijable (*udaljenost*) na vrlo intuitivan način. Važno je učenicima skrenuti pozornost na činjenicu da se naredbe u petlji neprestano izvršavaju čak i onda kada se udaljenost robota od prepreke ne mijenja pa se na displayu uvijek prikazuje ista vrijednost.

AKTIVNOST 4. Udaljenost izmjerenu ultrazvučnim senzorom ispisati 10 puta na 7-segmentni display.

Rješenje:



Navedimo i rješenje prirodne generalizacije prethodnog problema:



Ako vrijeme dopušta, može se prvo napisati ovaj program bez naredbe „wait 1 secs" pri čemu bi učenici trebali uočiti da je program vrlo brzo završio s radom. Nakon dodavanja naredbe za čekanje, učenicima se može postaviti pitanje kako bi na displayu naizmjence prikazivali dva različita podatka, svaki u trajanju jedne sekunde.

Sljedeća varijanta aktivnosti može biti ispisivanje udaljenosti sve dok je ona manja od nekog zadanog broja. To je dobar motivacijski primjer za uvođenje neke od logičkih petlji.

Sada ćemo samo pobrojati neke od problema i nedostataka koje smo uočili (a o kojima ništa nismo mogli pročitati).

- Svaku izmjenu programa trebamo instalirati na robota kao novi program, što oduzima vrijeme za vrijeme rada u razredu, posebice ako se radi s početnicima kod kojih se očekuje pojava problema.
- Prilikom zakretanja robota, zbog sekvencijalnog izvršavanja programa kotači se zakreću sekvencijalno (prvo jedan, pa drugi), što rezultira isprekidanim kretanjem koje se ne podudara s očekivanim kretanjem koje bi nastupilo u idealnom slučaju pri kojem bi se oba kotača pokrenula istoga trenutka.
- Robota ne možemo nekom jednostavnom naredbom zakrenuti za točno određeni kut, poput kornjače u Logu. Možemo aproksimirati kut zakreta mjereći vrijeme potrebno robotu da se zaokrene za puni kut. No, za isto vrijeme kut zakreta može varirati u ovisnosti o stanju baterija.
- Komunikacija između dvaju robota je spora. Duže poruke (stringovi) stvaraju dodatne probleme.
- Ponekad jedan robot zadovoljava uvjet zaustavljanja (recimo kada se približe na 10 cm), a drugi ne. Ovaj problem trebalo bi detaljnije istražiti u obliku učeničkog projekta u kojem bi se mijenjao međusobni početni položaj robota, bilježio rezultat rada programa, pokušao otkriti uzrok različitog ponašanja robota koji izvode isti program i napravio neki konačni zaključak.
- ...

5. Zaključak

Iskustva koja smo stekli u radu sa studentima, budućim studentima na otvorenom danu Fakulteta te učenicima i njihovim učiteljima pokazuju da je programiranje fizičkih objekata, pogotovo edukacijskih robota, zabavno. Unatoč uočenim problemima svakako preporučamo ovakav način poučavanja programiranja. Smatramo da je ovakav način poučavanja svakako korak naprijed u odnosu na postojeće stanje. No, naglasimo da trenutno ne postoji dovoljno nastavnih sredstava. Robot (ili micro:bit)

nije sam sebi svrha, tj. nećemo ga cijele školske godine vozati po učionici već ćemo ga koristiti kada pomoću njega možemo objasniti neke koncepte puno zornije i uvjerljivije od prikaza na ekranu. Ovakav oblik poučavanja vidimo kao jedan bitan dio nastave informatike i u osnovnoj i u srednjim školama. Pomoću robota možemo uz dodatne senzore izmjeriti npr. temperaturu, dobivene podatke prenijeti na računalo, prikazati ih grafički te uočiti kojom ih funkcijom možemo dobro aproksimirati. Na taj smo način robota doveli u vezu s grafičkim prikazom podataka te uspostavili korelaciju s nastavom matematike. Također, učenici mogu, korištenjem mikrokontrolera i senzora, provesti razne pokuse koji u sebi uključuju mjerenja (npr temperature, tlaka zraka, intenziteta svjetlosti i slično) te na taj način kvalitetnije usvojiti znanja iz drugih nastavnih predmeta, pri čemu su motivirani za usvajanje određenih programskih koncepata budući da su programi za mikrokontroler potrebni za uspješnu provedbu određenih projekata.

Literatura:

1. https://www.vecernji.hr/techsci/stem-revolucija-920-ustanova-dobiva-15000-setova-za-robotiku-1149686
2. http://vijesti.hrt.hr/370908/poduprite-stem-revoluciju-u-hrvatskim-skolama
3. http://www.poslovni.hr/tehnologija/microbitovi-stizu-skolama-stem-revolucija-traje-324227
4. http://microbit.org/
5. http://izradi.croatianmakers.hr/project/uvodno-o-mbotu/
6. https://pxt.microbit.org/
7. https://codewith.mu/