SETTING OBJECTIVE TESTS IN MATHEMATICS USING QM PERCEPTION

Martin Greenhow and Mundeep Gill

Setting objective tests in mathematics using QM Perception

Martin Greenhow and Mundeep Gill,
Department of Mathematical Sciences,
Brunel University,
Uxbridge, UB8 3PH

Abstract

We here describe technical issues in setting objective tests in various areas of mathematics using Question Mark Perception's QML language and format files, coupled with MathML mathematics mark-up and the Scalable Vector Graphics (SVG) syntax for producing diagrams. The plain text MathML and SVG coding can replace graphics files commonly used to display equations and diagrams in CAA packages and web pages, and have the overwhelming advantage that random parameters can be dropped into the interpreted plain text at runtime, thereby producing many millions of realisations of the underlying question style.

Introduction

This paper updates previous reports of Nichols & Greenhow (2002) and Nichols, Gill & Greenhow (2003) on the development of online objective questions for mathematics formative and diagnostic testing using Question Mark Perception. The most significant enhancements for questions involving mathematics and diagrams is likely to be the replacement of graphics by plain text coding of MathML for equations and SVG for diagrams. As well as being easily editable, the syntax incorporates random parameters thereby making all elements of the question and feedback fully dynamic. It is worth remarking that much of the work we have done is independent of Perception, or indeed of any CAA package, being standard html/Javascript that can be incorporated into any web page, for example, a page of teaching material with ever changing examples.

For CAA, one needs, from the outset, a clear specification of what skills are assumed and which skill is being tested. This can be different for algebraically equivalent questions, for example, the tested skill *completing the square* is in some ways easier if all the coefficients of the quadratic are positive. This would result in a question style, to be realised at runtime with only positive coefficients, and other question styles where this condition is relaxed in different ways (with the assumed skill that students can handle negative numbers). Having specified a question style, the author then needs to decide on question type: multi-choice, multi-response, numerical input etc. We have developed these to include random parameters, which require that distracters are generated at runtime using algebraic expressions or algorithms that would arise if the student applies sensible, but incorrect, rules of their own. These mal-rules can be identified by analysis of past exam papers, CAA answer files and students' and teachers' experiences. A good deal of work will therefore be needed to identify and assign metadata to such mal-rules.

New question types such as *hot line* (typically the student is asked to identify where an error lies in a fully-worked solution) or *responsive numerical input* also rely on mal-rules. A responsive numerical input question looks the same as a numerical input question, except that mal-rule-generated input is detected and an appropriate message given, e.g. "You have forgotten the minus sign when differentiating cos(*ax*) " (here *a* could be a number in the actual question realisation). Multi-stage question types that allow for partial credit have also been developed.

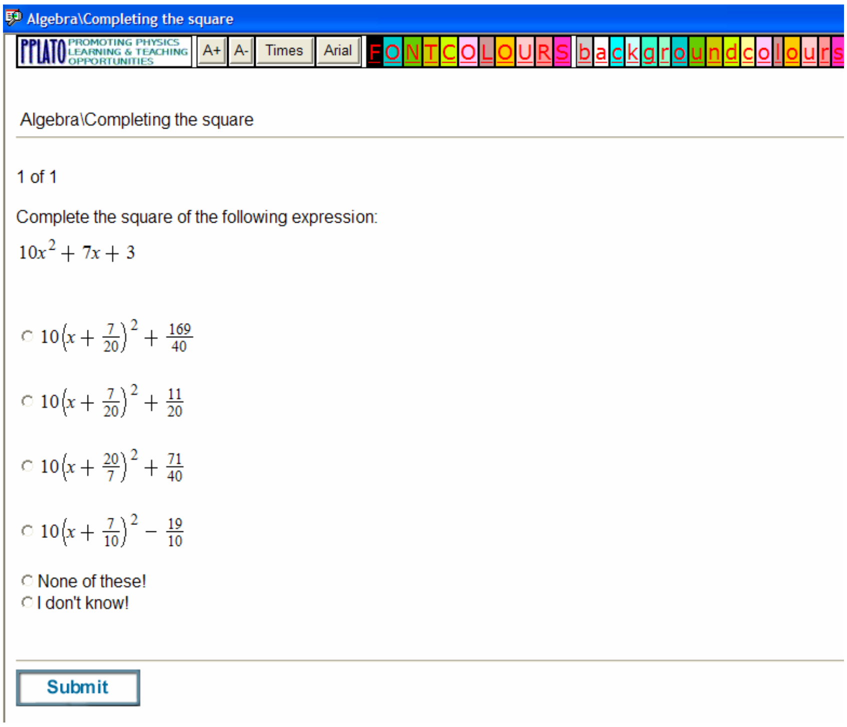
A strong feature of our question styles is the feedback to students (we imagine the questions being used mainly in diagnostic and formative tests). Typically, the feedback for an incorrect answer comprises: the correct answer; the general theory and the actual example asked of the student with fully-worked solutions done in parallel and possibly including an SVG diagram; any "target of opportunity" that will teach the (hopefully) engaged and interested student, such as alternative solution methods or tips ("You could have simplified the log terms first by …"); and finally the question setter's interpretation of why the student went wrong (based on the mal-rule that generated his/her choice).

Many of the question styles call external (Javascript) functions; these divide into two types; those that return the value(s) of some mathematical operation, e.g. multiplication of two polynomials, and those that return the MathML string needed to display the mathematics, e.g. a matrix. Interesting examples of the first include recursive programming, whilst the second incorporates a higher level of abstraction whereby the degree of the polynomial or size of the matrix is used to loop round, each pass concatenating the next bit of the string. Many of the display problems, such as uncancelled fractions, and expressions such as *a* + *bx* realising as 2 + -3*x* etc, are also resolved at this stage. As stated above, much of this functionality is exportable to other CAA systems.

We illustrate these ideas by presenting a series of examples from the existing database of questions developed as our contribution to the FDTL4 PPLATO consortium.

Example 1

The general appearance of the screen shows a typical multi-choice question. Note that the user has the option of changing the font sizes and colours in all questions; this might help certain types of dyslexic students and those with partial sight (further accessibility functionality using cookies is planned). The randomised coefficients of the quadratic are used, in conjunction with various mal-rules, to give the four distracters shown here; in this realisation, the correct answer is “None of these”. This will happen randomly, with an author-prescribed probability, and is considered an important option to lessen the tendency for students to attempt to eliminate all answers bar one, or to do the question backwards.


Algebra\Completing the square
PPLATO PROMOTING PHYSICS LEARNING & TEACHING OPPORTUNITIES
A+
A-
Times
Arial
FONTCOLOURS
backgroundcolours

Algebra\Completing the square

1 of 1

Complete the square of the following expression:

$10x^2 + 7x + 3$

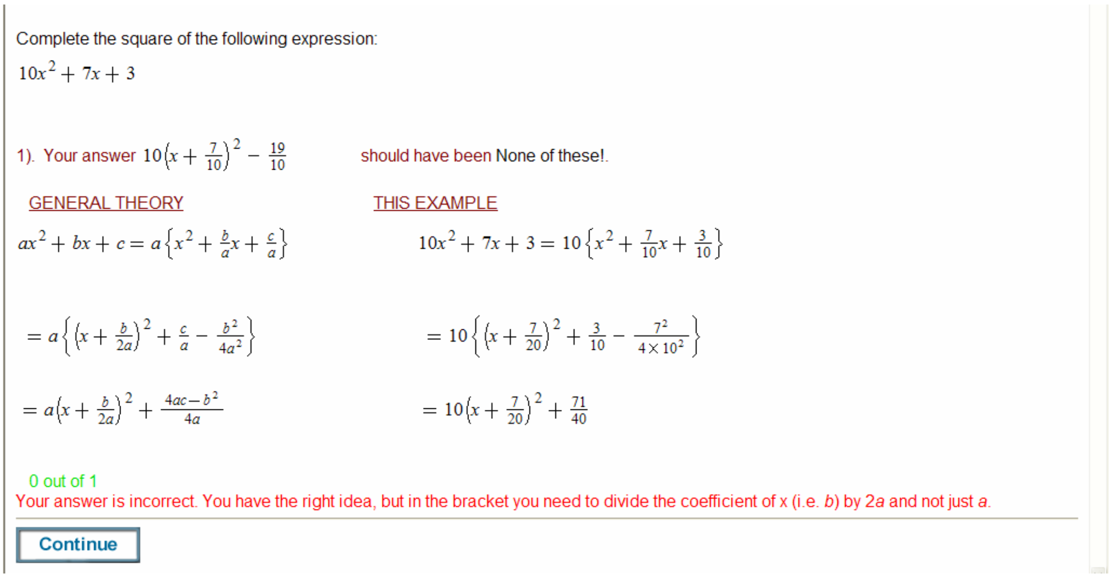
○ $10\left(x + \frac{7}{20}\right)^2 + \frac{169}{40}$

○ $10\left(x + \frac{7}{20}\right)^2 + \frac{11}{20}$

○ $10\left(x + \frac{20}{7}\right)^2 + \frac{71}{40}$

○ $10\left(x + \frac{7}{10}\right)^2 - \frac{19}{10}$

○ None of these!
○ I don't know!



Complete the square of the following expression:

$10x^2 + 7x + 3$

1). Your answer $10\left(x + \frac{7}{10}\right)^2 - \frac{19}{10}$ should have been None of these!.

GENERAL THEORY	THIS EXAMPLE
$ax^2 + bx + c = a\left\{x^2 + \frac{b}{a}x + \frac{c}{a}\right\}$	$10x^2 + 7x + 3 = 10\left\{x^2 + \frac{7}{10}x + \frac{3}{10}\right\}$
$= a\left\{\left(x + \frac{b}{2a}\right)^2 + \frac{c}{a} - \frac{b^2}{4a^2}\right\}$	$= 10\left\{\left(x + \frac{7}{20}\right)^2 + \frac{3}{10} - \frac{7^2}{4 \times 10^2}\right\}$
$= a\left(x + \frac{b}{2a}\right)^2 + \frac{4ac - b^2}{4a}$	$= 10\left(x + \frac{7}{20}\right)^2 + \frac{71}{40}$

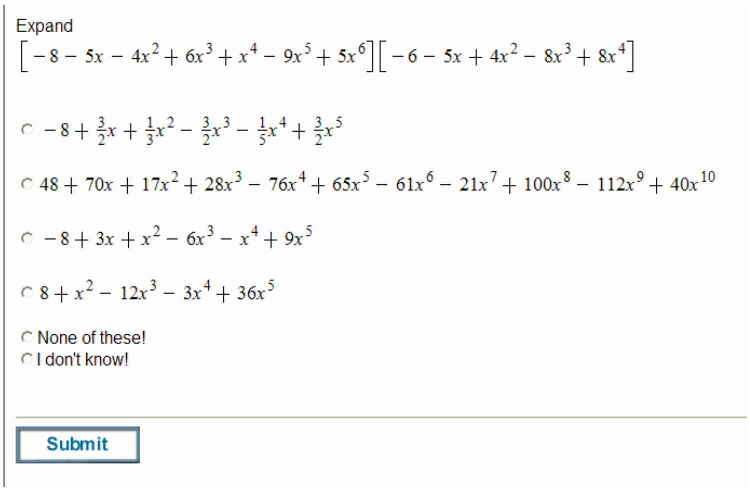
0 out of 1

Your answer is incorrect. You have the right idea, but in the bracket you need to divide the coefficient of x (i.e. *b*) by 2*a* and not just *a*.

Typical feedback is shown above, where the general theory is illuminated by inputting the random parameter in the parallel worked example. The actual response of the student is assumed to follow from the mal-rule referred to in the last sentence of the feedback.

Example 2

The following question is under development (the distracters are meaningless), and is not intended for direct use with students: it is included here to show a new methodology for question setting. The question is written in a very general way whereby random length polynomials (themselves with random, possibly fractional, coefficients) are displayed and multiplied together (the correct answer is shown) by calling the external Javascript functions to carry out the mathematics and display. In an actual question style, the lengths of the polynomials may not be random so that an author would be certain that the question tests e.g. multiplication of a quadratic by a cubic (simply by inserting the lines n=2; m=3; into the code). Writing general questions, and then restricted them, is a highly-efficient way of authoring a large number of related question styles.



Expand

$\left[-8-5x-4x^2+6x^3+x^4-9x^5+5x^6\right]\left[-6-5x+4x^2-8x^3+8x^4\right]$

○ $-8+\frac{3}{2}x+\frac{1}{3}x^2-\frac{3}{2}x^3-\frac{1}{5}x^4+\frac{3}{2}x^5$

○ $48+70x+17x^2+28x^3-76x^4+65x^5-61x^6-21x^7+100x^8-112x^9+40x^{10}$

○ $-8+3x+x^2-6x^3-x^4+9x^5$

○ $8+x^2-12x^3-3x^4+36x^5$

○ None of these!
○ I don't know!

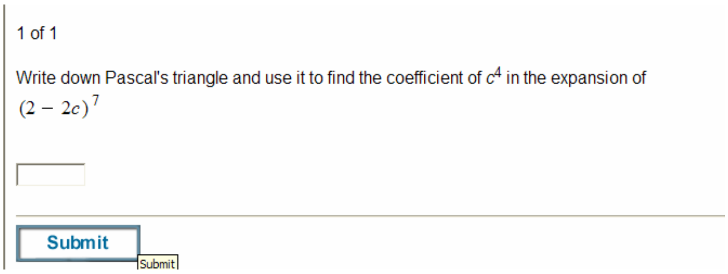
The following function (held in a Perception format file) is used to generate the MathML display string of a polynomial of degree n whose coefficients are held in the coeffs array:

```
function displaypolynomial(x,coeffs,n) {
var n

if(coeffs[0] > 0){terms = "<mn>" + coeffs[0] + "</mn>"}else{if(coeffs[0] < 0){terms = "<mo>-
</mo>" + "<mn>" + Math.abs(coeffs[0]) + "</mn>"}else{terms = ""}};

 for (i = 1; i <= n; i++)
     {if (coeffs [i] >= 0)
         {sign = "<mo>+</mo>";
}
else
       {sign = "<mo>-</mo>";
}

if(i == 1){di = "";}else{di = i};
coeff = Math.abs(coeffs[i]);
if(terms == "" & coeffs [i] >= 0){sign = ""};

   if(coeff != 1){term = sign +
"<mn>"+coeff+"</mn><msup><mi>"+x+"</mi><mn>"+di+"</mn></msup>"}else{term = sign +
"<msup><mi>"+x+"</mi><mn>"+di+"</mn></msup>"};

       if (coeffs [i] != 0 ) {terms = terms + term;}
}
             return ("<mrow>" + terms + "</mrow>");
}
```

The for loop successively concatenates the required MathML string to be returned to the question for display. Various if statements suppress terms with zero coefficient and avoid returning strings that would display e.g. 2+-4x^1.

Example 3

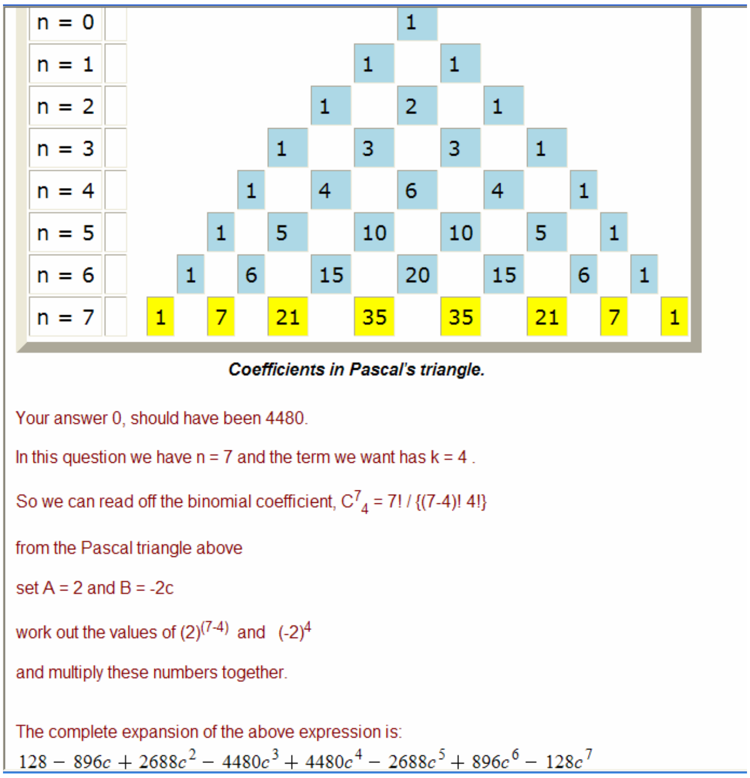
The following is a numerical input question (with zero tolerance in this case so an exact answer is required) for testing Pascal's triangle; the coefficients, the name of the variable (c in this case) and the power are randomised.



1 of 1

Write down Pascal's triangle and use it to find the coefficient of c^4 in the expansion of $(2 - 2c)^7$

An incorrect response produces extensive feedback, part of which is shown below. There is a balance to be struck here, of course. Students may not engage with, or even read, feedback if it is too long, but may not understand if too many of the steps are omitted. An alternative, sometimes used for steps or general theory/formulae, is to use pop-up boxes allowing the student to see the details if they wish.



n = 0								1							
n = 1							1		1						
n = 2						1		2		1					
n = 3					1		3		3		1				
n = 4				1		4		6		4		1			
n = 5			1		5		10		10		5		1		
n = 6		1		6		15		20		15		6		1	
n = 7	1		7		21		35		35		21		7		1

Coefficients in Pascal's triangle.

Your answer 0, should have been 4480.

In this question we have n = 7 and the term we want has k = 4 .

So we can read off the binomial coefficient, C^7_4 = 7! / {(7-4)! 4!}

from the Pascal triangle above

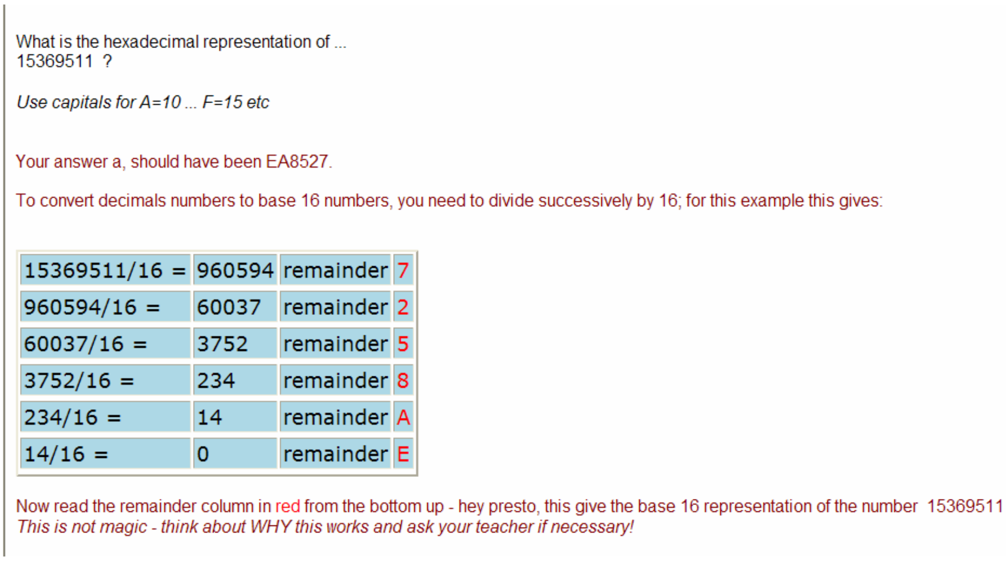
set A = 2 and B = -2c

work out the values of $(2)^{(7-4)}$ and $(-2)^4$

and multiply these numbers together.

The complete expansion of the above expression is:
$128 - 896c + 2688c^2 - 4480c^3 + 4480c^4 - 2688c^5 + 896c^6 - 128c^7$

The above feedback uses string concatenation to display the triangle to the required size. Building tables in this way is also useful for statistics where data tables of random length are required and for truth tables with an arbitrary number of propositions and logical connectives. Another example might be to illustrate an algorithm, as follows:



What is the hexadecimal representation of ...
15369511 ?

Use capitals for A=10 ... F=15 etc

Your answer a, should have been EA8527.

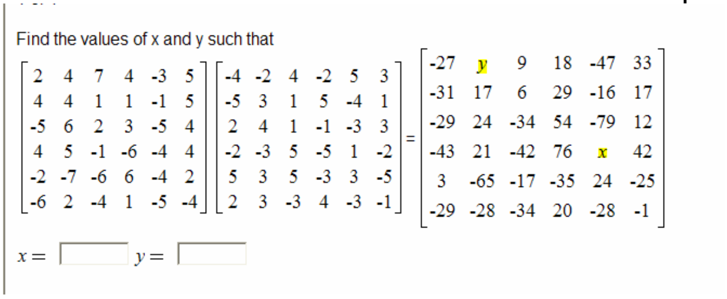
To convert decimals numbers to base 16 numbers, you need to divide successively by 16; for this example this gives:

15369511/16 =	960594	remainder	7
960594/16 =	60037	remainder	2
60037/16 =	3752	remainder	5
3752/16 =	234	remainder	8
234/16 =	14	remainder	A
14/16 =	0	remainder	E

Now read the remainder column in red from the bottom up - hey presto, this give the base 16 representation of the number 15369511
This is not magic - think about WHY this works and ask your teacher if necessary!

Example 4

The following example shows the considerable power of randomisation and MathML string concatenation. Two random sized (conformable) matrices with elements [-9,9] are displayed together with their (function calculated) product with two randomly chosen element positions overwritten with x and y. This question style would produce roughly 10^{100} question realisations, although much of this variation is irrelevant to the actual question as stated here.



Find the values of x and y such that

$$\begin{bmatrix} 2 & 4 & 7 & 4 & -3 & 5 \\ 4 & 4 & 1 & 1 & -1 & 5 \\ -5 & 6 & 2 & 3 & -5 & 4 \\ 4 & 5 & -1 & -6 & -4 & 4 \\ -2 & -7 & -6 & 6 & -4 & 2 \\ -6 & 2 & -4 & 1 & -5 & -4 \end{bmatrix} \begin{bmatrix} -4 & -2 & 4 & -2 & 5 & 3 \\ -5 & 3 & 1 & 5 & -4 & 1 \\ 2 & 4 & 1 & -1 & -3 & 3 \\ -2 & -3 & 5 & -5 & 1 & -2 \\ 5 & 3 & 5 & -3 & 3 & -5 \\ 2 & 3 & -3 & 4 & -3 & -1 \end{bmatrix} = \begin{bmatrix} -27 & y & 9 & 18 & -47 & 33 \\ -31 & 17 & 6 & 29 & -16 & 17 \\ -29 & 24 & -34 & 54 & -79 & 12 \\ -43 & 21 & -42 & 76 & x & 42 \\ 3 & -65 & -17 & -35 & 24 & -25 \\ -29 & -28 & -34 & 20 & -28 & -1 \end{bmatrix}$$

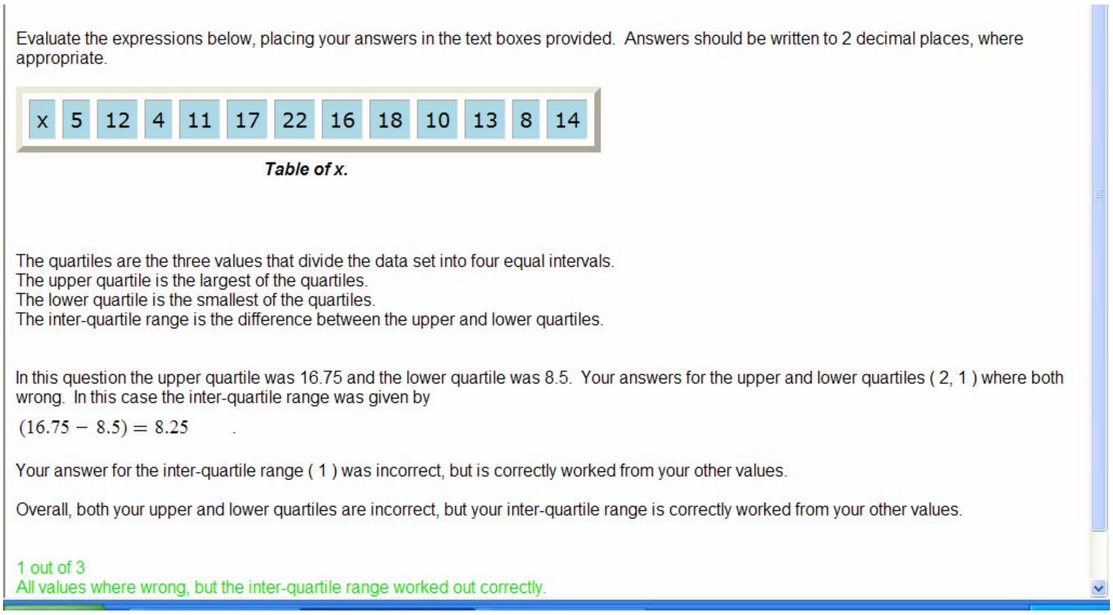
$x =$ □ $y =$ □

Typically matrix manipulation requires several functions that can call each other; the following determinant demonstrates this (matrix size and minor are called functions) and the idea of recursive coding where the function calls itself:

```
function determinant(Rmtrix) {
var number=0;
var i;
var Nrow;
var size = new Array();
size = MatrixSize(Rmtrix);
Nrow = size[0];

var number;

if(Nrow == 1){number=Rmtrix[1][1]}else{for(i=1 ; i<=Nrow ; i++)number +=
Rmtrix[1][i]*Math.pow(-1,(i+1))*determinant(dominor(Rmtrix,1,i))};

return number;
}
```

Example 5

Sequential questions allow for partial credit whereby incorrect inputs from the first part(s) of the question can be carried through in the marking to see if the methods for calculation of the latter parts are correct (even through the actual input is incorrect). For example:



Evaluate the expressions below, placing your answers in the text boxes provided. Answers should be written to 2 decimal places, where appropriate.

x	5	12	4	11	17	22	16	18	10	13	8	14

Table of x.

The quartiles are the three values that divide the data set into four equal intervals.
The upper quartile is the largest of the quartiles.
The lower quartile is the smallest of the quartiles.
The inter-quartile range is the difference between the upper and lower quartiles.

In this question the upper quartile was 16.75 and the lower quartile was 8.5. Your answers for the upper and lower quartiles (2, 1) where both wrong. In this case the inter-quartile range was given by

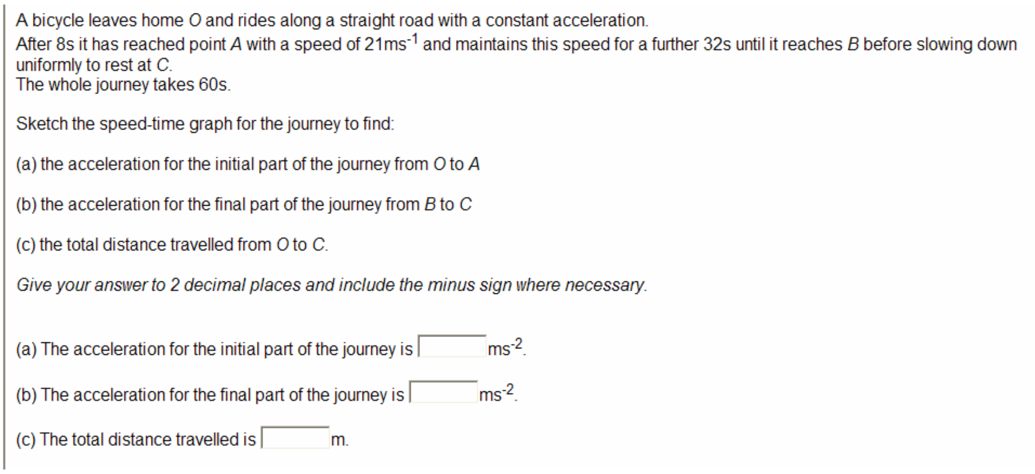
$$(16.75 - 8.5) = 8.25 \quad .$$

Your answer for the inter-quartile range (1) was incorrect, but is correctly worked from your other values.

Overall, both your upper and lower quartiles are incorrect, but your inter-quartile range is correctly worked from your other values.

1 out of 3
All values where wrong, but the inter-quartile range worked out correctly.

The following question also has randomised word pairs (bicycle/home; train/station etc) and realistic velocities and accelerations for the mode of transport chosen:



A bicycle leaves home *O* and rides along a straight road with a constant acceleration.
After 8s it has reached point *A* with a speed of 21ms^{-1} and maintains this speed for a further 32s until it reaches *B* before slowing down uniformly to rest at *C*.
The whole journey takes 60s.

Sketch the speed-time graph for the journey to find:

(a) the acceleration for the initial part of the journey from *O* to *A*

(b) the acceleration for the final part of the journey from *B* to *C*

(c) the total distance travelled from *O* to *C*.

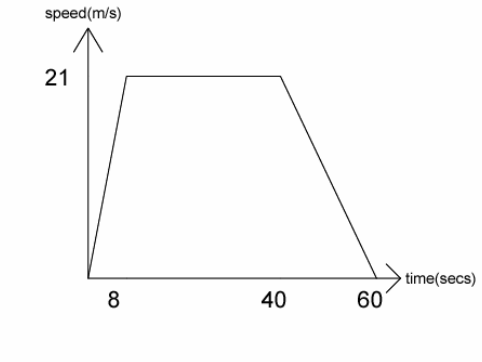
Give your answer to 2 decimal places and include the minus sign where necessary.

(a) The acceleration for the initial part of the journey is ☐ ms^{-2}.

(b) The acceleration for the final part of the journey is ☐ ms^{-2}.

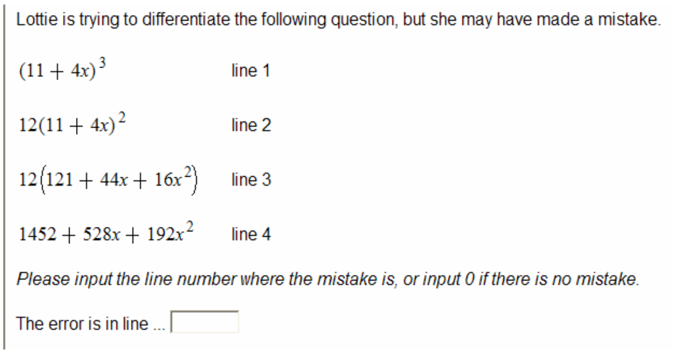
(c) The total distance travelled is ☐ m.

and demonstrates the use of Scalable Vector Graphics (SVG), in feedback in this case. The required speed time graph is realised with the correct aspect and labels.



Example 6

The following example illustrates a HotLine question. There is a lot of coding behind this question where the MathML for any number of scenarios is coded and one is chosen at random for presentation. Thus the mistake can occur in any (or no) line, and any line can have any one of several mistakes. Once a mistake has been made, the rest of the display is consistent with this mistake. Obviously each scenario has its own feedback. Finally the name (Lottie in this case) is selected from a list of either male or female names with the correct ethnic mix for 16-25 year olds in the UK taken from the last census data.



Lottie is trying to differentiate the following question, but she may have made a mistake.

$(11+4x)^3$ line 1

$12(11+4x)^2$ line 2

$12\left(121+44x+16x^2\right)$ line 3

$1452+528x+192x^2$ line 4

Please input the line number where the mistake is, or input 0 if there is no mistake.

The error is in line ... □

Conclusion

The open code nature of Question Mark Perception means that its functionality can be considerably extended both in the question QML coding and in the provision of Javascript functions that are available to all questions. Given that one generates typically thousands or even millions of realisations for each question style, the trade-off between ease of use of the standard Perception question wizard and functionality presented here seems, at least for objective questions with significant mathematics, to be heavily in favour of the present approach. It is hoped to export these ideas and much of the coding to other subject areas and other CAA systems.

References

1. Nichols D & Greenhow M (2002) “Using Question Mark Perception v3 for testing mathematics” MSOR Connections Volume 2, Number 3: Aug. Available on http://ltsn.mathstore.ac.uk/newsletter/vol2.shtml#no3
2. Nichols D, Gill M & Greenhow M (2003) “Pedagogic issues in setting online questions” MSOR Connections Volume 3, Number 4: Nov. Available on http://ltsn.mathstore.ac.uk/newsletter/vol4.shtml#no4