**Standard Input system**

No

No

Yes

Yes

Start

Input how many questions the student would like and store as ‘*questions\_left*’

Input what year in school the student is in

Pick question set based on year in school

Set a variable ‘*correct*’ to 0

Set a variable ‘*total\_questions*’ to ‘*questions\_left*’

Check if all data inputted is valid

Output the question

Input what the student believes the answer is

If inputted answer is correct

Add one to ‘*correct*’

Choose two random numbers, and an operation

Calculate what the answer is

If ‘*questions\_left*’ is equal to 0

Subtract one from ‘*questions\_left*’

Output ‘*correct*’ / ‘*total\_questions*’

End

START

INPUT ‘*questions\_left’* as Integer greater than 1

SET ‘*total\_questions’* to ‘*questions\_left*’

INPUT ‘*year*’ as Integer between 1 and 6

SET ‘*correct*’ to 0

WHILE ‘*questions\_left*’ > 0

SET ‘*Num1*’ to Random Number

SET ‘*Num2*’ to Random Number

SET ‘*Operation*’ to a random character in ‘+ - / \*’

SET ‘*Answer*’ to Correct Answer to the equation ‘*Num1*’ ‘*Operation*’ ‘*Num2*’

OUTPUT ‘*Num1*’ ‘*Operation*’ ‘*Num2*’

INPUT ‘*user\_answer*’

IF ‘*user\_answer*’ = ‘*Answer*’ THEN

OUTPUT “Correct”

CHANGE ‘*correct*’ by 1

ENDIF

CHANGE ‘*questions\_left*’ by -1

ENDWHILE

OUTPUT ‘*correct*’ / ‘*total\_questions*’

END

**Types of inputting data**

Start

Calculate correct answer

Generate 3 other random wrong answers

Randomise order, so correct button is not in the same place every time

Output four buttons on the student’s screen with the answers

Student inputs/ presses on the one that is correct

End

Multiple choices

START

SET ‘*Answers\_list*’ to a list of the correct answer ‘*Answer*’ and three randomly generated numbers between 0 and the power of ten closest to ‘*answer*’

RANDOMISE order of ‘*Answers\_list*’

OUTPUT four buttons with text inside for each respective button

INPUT ‘*button\_pressed*’

IF ‘*button\_pressed*’ value = ‘*Answer*’ THEN

OUTPUT “Correct”

ENDIF

END

An issue with multiple choice questions is that students can be lead to thinking one answer is right instinctively. For example, longer addition problems, such as 652+347, can be checked against the other answers, such as A) 997 B) 998 C) 999 D) 1000, just by adding the last digits together, 2+7. This does not necessarily prove the student is able to do maths, but can get around doing the entire question.

Start

Calculate correct answer

Input answer that student thinks is correct

End

Check if it is valid and correct

Open ended

START

INPUT ‘*user\_answer*’

IF ‘*user\_answer*’ = ‘*Answer*’ THEN

OUTPUT “Correct”

ENDIF

END

An issue with open-ended questions is that there are so many possible combinations of what could be entered. For example, the question ‘If I had 3 apples and ate one, how many would be left?’ could be answered in many ways, including ‘2’, ‘two’, ‘Two’, ‘TWO’, ‘Two Apples’, ‘two apples’, ‘02’. This means each question will need to be specific, and avoid confusion of what is needed to be inputted.

True/ False

Start

Calculate correct answer

Output ‘Is NUMBER the correct answer?’

Input True or False (buttons)

End

Check if it is correct

START

SET ‘*Answers\_list*’ to a list of the correct ‘*Answer*'

and one randomly generated number

PICK 1 from ‘*Answers\_list*’ and save the number as ‘*number*’

IF ‘*number*’ = ‘*Answer’* THEN

SET ‘*correct*’ to True

ELSE

SET ‘*correct*’ to False

ENDIF

OUTPUT ‘Is *number* the correct answer?’

INPUT ‘*user\_answer*’ as Boolean

IF ‘*user\_answer*’ = ‘*correct*’ THEN

OUTPUT ‘Correct’

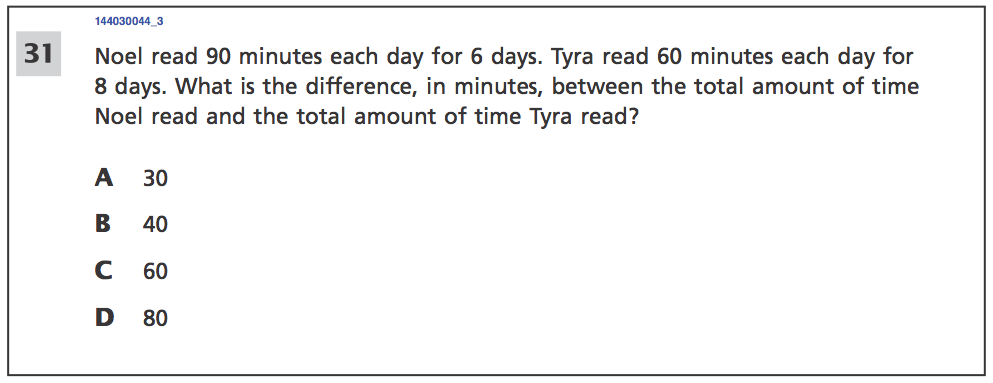
ENDIF

END

One of the issues with True/ False questions is that the question itself is giving the student an answer, which if seems roughly correct probably is. On the other hand, True/ False questions are normally set to a ‘trick’ answer that many students would fall for, to try and ‘catch them out’. For example, in a multiple-stage problem, an answer may be given that is correct but doesn’t answer the question, however, appears earlier on in the question to trick students. This means attention to detail is necessary, yet can be quite tedious for extremely long questions.

**Types of games/ outputting data**

When a student gets the correct answer, what should be the result be? There needs to be a differentiation between correct and incorrect answers, and if a student gets a wrong answer it should be explained what the steps are to make it a better, more accurate answer. If a student gets it wrong, it is either a silly mistake or the student doesn’t understand certain aspects of the course, meaning the correct answer and working out should be shown to the student.

Just Questions

Questions can appear on the screen, and the user can input an answer as suggested above. This is good because it makes each question clear, concise, ensures the user is not distracted and allows for people to work at their own pace, whether limited by mathematics skills or computer skills. However, there is no real incentive to make the user continue doing questions and no reward for doing one correctly.



Interactive Game

Questions can appear in the form of a game, where there are questions that advance the game onwards if answered correctly. These add incentive to the questions and make maths fun and enjoyable. However, some can be a little distracting as there are more games than actual mathematics, which students may favour as being more fun and less work or learning.

Combination of both

Some games require you to complete a certain amount of questions, in this example 10, and afterwards play a short mini game depending on how many questions were correct. The final score you got was dependent on how many questions you got right and how many games you won after completion of the questions. This allows for a reward system to be implemented for a positive reinforcement of answers, yet also ensures the student is having a fun time answering questions.

**Conclusion**

The main input device I will be using will be open-ended, as I think it is a better system because it makes sure the student understands the mathematics, however I must be careful about only allowing the user to input numbers, otherwise spelling may affect the student’s score if it is spelt wrong. In the past, I have had some programs, especially with Modern Foreign Languages, that have marked me down for a single space being wrong and forcing me to restart the entire module. This is not the appropriate outcome; therefore I will design my game to make sure silly inputting errors do not occur and it is the mathematics that is being assessed rather than the typing. I will also be outputting the student’s progress using a simple game with the set amount of questions, as it encourages the user to finish the questions, and as a result, beat the ‘computer’ opponent overall for their progress. The game I have been thinking of is a race in which two animals are selected, and for every question correct one advances, and every question incorrect the other advances. This means students will have to get above 50% to pass and ‘win’.