Maths Quiz

1. Think through the quiz offline
2. Model a quiz algorithm
3. Welcome to the quiz
4. User input to ask the question
5. Selection
6. Multiple questions
   6a Different sounds if correct or wrong
7. Score Variable
   7a Take away score if answer wrong
   7b Questions that use < or >
   7c Repeat until correct
   7d Congratulations
8. Quiz procedure
9. Quiz procedure algorithm
10. Assessment Booklet

1. Think through quiz offline

Ask a simple maths quiz question to the class. Ask them in pairs to work out and record in bullet points the steps that humans have to say and do to ask and answer that question. Make sure they realise that this is nothing to do with code and it is what a humans would do if asking this question as part of a quiz. Once they have completed this draw out that the quiz master would need to think of the question and then think of the answer before they can ask the question. Once the user has answered the question ask them how they will know if the question is right or wrong. Indicate that the quiz master has got the original thought through answer and the answer the quiz user gave them. Hold one in each hand and look alternately at both like you are examining them. You could write RIGHT ANSWER on one piece of paper and USER ANSWER on another. You are looking for a pupil to indicate that they need to be the same. Often pupils will come up with the idea of comparing the answers which is a step in the right direction. Push pupils to be more specific and ask for a type of comparison. You can then ask what the maths symbol for the same is (=).

2. Model a quiz algorithm

What a human would do is a great start but converting our human algorithm to an algorithm that will work on a computer is a bit more complex. Use Quiz Algorithm slides 2 & 3 to take pupils through the human algorithm and a detailed quiz algorithm. This is the computing equivalent of a literacy modelled write. Can they spot what is similar and what is different between both algorithms (slide 3).

Program Aim Create a Maths quiz

Computer Science Concepts
- if else selection
- Broadcasting to trigger other blocks
- Variable for score
- Algorithm to code
- User typed input into program
- Abstract the code into a procedure

National Curriculum Programs of Study

Pupils should be taught to:
- design, write and debug programs that accomplish specific goals, including controlling or simulating physical systems; solve problems by decomposing them into smaller parts
- use sequence, selection, and repetition in programs; work with variables and various forms of input and output
- use logical reasoning to explain how some simple algorithms work and to detect and correct errors in algorithms and programs

See note on page 7
3, **Welcome to the quiz**

Explain that this program will mostly be one sequence of instructions from start to finish. Can pupils choose a good starting block and a way to welcome users to their quiz? Have they tested their code?

4, **User input to ask the question**

Drag the ask and the answer blocks out. Explain that this is a user input block. It puts in information from the person taking the quiz.

Now tick the answer block so that it is visible on the screen.

Explain that whatever is typed into the ask input block goes inside the answer block. Type in a Maths question that everyone will know the answer to and watch as it appears in the answer block after you tick to accept it.

You can also click on the answer box and a small speech bubble will appear showing what has been typed into the ask block.

5, **Selection**

If this is the first time that pupils have met the idea of selection it is worth using some real world examples which you can find at [http://code-it.co.uk/scratch/selectioninrealworld.pdf](http://code-it.co.uk/scratch/selectioninrealworld.pdf)

The last examples is important as it paves the way for the code. Make sure in the last example you draw out that equals means the same as.

Drag out these blocks and arrange them like this. Make sure you draw attention to the shape especially when inserting the equals block. Notice that the equals block goes inside the if else block and then the answer block goes inside the left hand side of the equals block.
Talk through the program line by line before running it to help pupils understand what is happening.

Pretend to be a user who gets an answer right. Pretend to type the answer and ask the pupils where the answer has gone into the code? (the blue answer block)

The program then checks to see if it is the same (point to equals sign) as the right answer this means that the condition has been met and only the top correct block is run. Do the same for a wrong answer pointing out that the answer is not the same as the right answer so only the else block is run.

6. **Multiple Questions**

Don’t show pupils how to duplicate the blocks until they have done a few questions block by block.

Now demonstrate how to duplicate blocks by dragging out a single question and right clicking on the top most blue ask block and choosing duplicate. You must right click on the top block or not all the code will be duplicated.

**Common Bug**

Once pupils have started to duplicate it is very easy to snap the question inside the bottom else part of the selection. Can they tell you when the second question would be asked?

**ANSWER** Only if you got question 1 wrong.

Do ask the question though as I have had pupils tell me that they are giving the user another opportunity to answer the question. This indicates that they understand the principles behind selection.

**Common Bugs**

Another common error is to use say blocks without timings. This runs the say block so fast that the user can’t see them at all. Comparing their code with yours is normally enough for most pupils to identify this bug.
Different sounds if correct or wrong

Ask pupils to make the quiz play a sound if the quiz question is correct. This is a further test to see if they understand what is happening inside the selection process. Many pupils will choose to add this without a teacher's prompt, especially if they have completed any musical programming in the past.

Score Variable

Ask pupils what it means if we say that the weather is variable? Draw out that it means changeable. Explain that we can get the program to keep score for us by creating a variable which is like a pencil pot which we can put pencils/numbers into to help us keep score.

Ask pupils what possible scores the user could get from a three question quiz if each question is worth one mark each? (0,1,2 or 3) Ask them if we know what score a user will get before they take the quiz? This is important as many pupils won't have considered the range of possibilities and may have already awarded a 'mark' via a say command at the end of each question.

Create a variable called score

Click on orange Date section

Click on make a variable

Type in a name like score

Select for all sprites

Click ok

Ask pupils what score the user should start with? Drag out the set score to 0 block and ask pupils where it should go? (At the top so the quiz always starts with zero points)

Now drag out one change score by 1 blocks and ask pupils to decide where they should go to put a point into the variable pot if the user gets the question correct?

Give pupils time to experiment with their first question to find the right location. Remind them that they must run the code for a right and a wrong answer to test it.

When pupils have come up with the right location talk through the code to explain it before running it. A great way to do this is with a box for the variable containing 4 pencils labelled with score that a pupil holds. Model setting the score to 0 by removing all the pencils. Model adding a pencil into the box if you get the answer right.

You may also wish to show pupils how you can get the sprite to say the score through the use of a say, join and score variable. Pupils love the cleverness of this.

Common Bugs

Inserting set score to 1 blocks instead of change score by 1. This sets the score back to 1 every time it is used.

Missing out a set score to 0 block
This means that when the quiz is run the second time the old score is still inside the variable and will be added to instead of having an empty score when you start.
7a, Takeaway a point if user gets answer wrong
Adding a change score by \(-1\) into the else section. Challenge pupils to takeaway a point if the user gets the answer wrong. Give pupils time to puzzle this out.

Formative Assessment Corrective Hint
It can help to walk an invisible number line, adding one to every number to count up and taking away one to count down.

7b, Code that uses less than or greater than
Can pupils create questions such as type a number less than 56. Use a > or < block instead of an = one.
If you right click on the = sign < and > are options you can choose.

7c, Repeat until answer correct
Can pupils get a question to repeat until the user answers is correctly? The top solution would affect a negative variable if you were using this.
The bottom right solution gives no feedback when the question is wrong.
As in all programming, more complex solutions are available.
8, Quiz Procedure
Ask pupils to look at the amount of code they have written. It seems like a lot of work for 5 to 10 questions. Would they be interested in writing one question that can be used over and over again so they don’t have to write so much code?

8a, Quiz Procedure Algorithm
Use slide 4 and 5 of the Quiz Algorithm to illustrate the procedure quiz algorithm.

Mention that the code on the left is the algorithm for a quiz question.

Ask pupils what changes from quiz question to quiz question?

Answer
The question
The right answer

So if we feed both of these into a procedure we can use one quiz question many times.

To create a procedure
Click on make a block.

Give the block (procedure) a name.
This one has been called quiz question.
Click on options.
Add two string inputs (strings include numbers and letters)
Rename them Question and Right Answer
The procedure will look like the one at the bottom.
Add the question code as shown top left.
Drag the Question and Right Answer blocks into the ask block and the other side of the =.
Show pupils slide 6 on the Quiz Algorithm as this has the diagram of the code on the top left.

Demonstrate this with your pupils before they go to make their own version.
Procedure Challenge

A great challenge is to mention that you would like every question to have a different score amount. Lower score for easier questions and higher score for harder questions.

Can pupils create a new procedure that included score as well as question and right answer? What type of input will this be?

Answer Number

NOTE

In effect the procedure simplifies the process of making a quiz so that the programmer can concentrate on thinking of questions and answers.

It is classic abstraction in action where the complexity has been removed so the user can concentrate on the most important elements, the questions and the right answers.

This method feeds much more directly into text based programming methodology.

I am in two minds if we should use the word abstraction at this point. I shall try with and without and see if it helps or not.

I know the use of procedures can be taught successfully with Year 5+ pupils once they have the basics of sequence, repetition, selection and simple variable use. However I am going to try this with Year 4 pupils and see if they understand what they are using.

10, Assessment Booklet

The assessment booklet and answer booklets are on the last six pages of this document.
I recognise there is more than one way to solve/describe a problem

I can evaluate my solutions against a set criteria

I can design criteria to evaluate my creations

I can contribute useful ideas to a partner or group

I can encourage others to share their ideas

I lead using all the people talent in my group

I learn from setbacks and don’t let them put me off

I can persevere even if the solution is not obvious

I don’t just accept the first solution

Handles Ambiguity

Open Ended Problem Solver

Copes with Complexity

Insert picture of your students here

I look for a range of solution to the same problem

I look for how a project can be extended

I can break complex problems into parts

I can identify patterns in problems & solutions

I can adapt existing ideas to solve new problems

I can develop, test and debug until a product is refined

I make predictions about what will happen

I repeatedly experiment through predicting, making, testing & debugging

Inspired by Behaviour Rubric created with @MarkDoliner and linked at http://code-it.co.uk/attitudes/
**Quiz Algorithm**

- Create a variable called **score**
- Use a variable called **answer**
- Ask the user a question
- Store their answer in **answer** variable
- If their answer stored in the **answer** variable is = to the right
- Else their answer stored in the **answer** variable is not = to the right

**Quiz Code**

- if **answer** = 52 then
- else

Use the algorithm and code above to answer these questions

<table>
<thead>
<tr>
<th>Question</th>
<th>Answer</th>
</tr>
</thead>
<tbody>
<tr>
<td>If the user typed in 52, what word would appear on the screen?</td>
<td></td>
</tr>
<tr>
<td>Can this quiz question ever say correct and wrong for the same answer?</td>
<td></td>
</tr>
<tr>
<td>If you wanted to give the user 5 points, which block would you change?</td>
<td></td>
</tr>
<tr>
<td>What is the name for the block of code that uses if else? (variable,</td>
<td></td>
</tr>
<tr>
<td>conditional selection, repeat loop, sequence, list)</td>
<td></td>
</tr>
</tbody>
</table>
Maisy created this program.

She duplicated the second question and placed it inside the else section as shown by the bracket.

What possible reasons could she have for putting it there?

[Blank]

Duplicated second section

---

Draw lines to show how the two quiz question boxes connect with parts of the quiz question procedure.

[Blank]  [Blank]
How many times will this program say **Score so far is** .... ?

If the quiz user gets every question correct, how many points will they have?

How many times will the quiz say **Your final score is** .... ?

Fiona made this program to help her younger brother learn how to takeaway.

Can you complete the output table below?

<table>
<thead>
<tr>
<th>Question</th>
<th>Right Answer</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>9</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Total out of 17
Match the red circled algorithm to the correct code. The first one has been done for you.

**Quiz Algorithm**

- Create a variable called **score**
- Use a variable called **answer**
- Ask the user a question
- Store their answer in **answer** variable
- If their answer stored in the **answer** variable is = to the right answer, say correct and increase **score** variable by 1
- Else their answer stored in the **answer** variable is not = to the right answer, say wrong and decrease **score** variable by 1

**Quiz Code**

If the user typed in 52, what word would appear on the screen?

- Correct

Can this quiz question ever say correct and wrong for the same answer?

- No

If you wanted to give the user 5 points, which block would you change?

- Change score by 5

What is the name for the block of code that uses if else? *(variable, conditional selection, repeat loop, sequence, list)*

- Conditional selection
Maisy created this program.

She duplicated the second question and placed it inside the else section as shown by the bracket.

What possible reasons could she have for putting it there?

- It is a bug that was dragged into the wrong place.
- She is giving the user a second chance to do the question if they got it wrong the first time.

Duplicates second section

Quiz question boxes

Either path is correct

Quiz question procedure

Draw lines to show how the two quiz question boxes connect with parts of the quiz question procedure.
How many times will this program say **Score so far is .... ?**?

- **5**

Fiona made this program to help her younger brother learn how to takeaway. Can you complete the output table below?

<table>
<thead>
<tr>
<th>Question</th>
<th>Right Answer</th>
</tr>
</thead>
<tbody>
<tr>
<td>10-1=</td>
<td>9</td>
</tr>
<tr>
<td>10-2=</td>
<td>8</td>
</tr>
<tr>
<td>10-3=</td>
<td>7</td>
</tr>
<tr>
<td>10-4=</td>
<td>6</td>
</tr>
</tbody>
</table>

Total out of 17