

Unit 4: Programming

Assessment 2

Contents

What do requirements and specifications mean?	2
What are the requirements and specifications?	3
What is project management?	3
Software Development Lifecycle	3
What is project planning and why is it important?	4
What is design and why is it important?	4
What is a pseudocode?	5
How will I do pseudocode and why?	5
What is a flowchart?	5
How will I make a flowchart and why?	5
Original Pseudocode	5
Modified Pseudocode	6
Original Flowchart	8
Modified Flowchart	8
Original Source Code Data	9
Wireframe	11
Screenshots	12
Modified Source Code Data	13
Design Reviews	15
Program Reviews	15
How does the program meet the user requirements?	16
Why did I code C# and why did I use Visual Studio?	16
Why did I code the program this way?	16
Advantages and Disadvantages of the Original Code	16
Advantages	16
Disadvantages	16
References	16

What do requirements and specifications mean?

Requirements refer to sets of characteristics that are specific, measurable and testable that a software program needs to have so it meets the needs of their users. Essentially, it is just features the software must have so people can use it properly; so for example, if you are making a quiz, you are going to need to put in questions and answers down as users will require this sort of stuff in a quiz, otherwise if you make a quiz and put no questions down, then this will breach the user requirements and nobody will use it. So it is optimal that you research what sort of requirements your software must have so you know what requirements you need to meet otherwise you will end up making errors on your program that users don't want in terms of meeting requirements.

Specifications refer to how the software will do the quiz, so how is it going to generate questions? How is it going to mark your questions?

What are the requirements and specifications?

I am required to produce a program that will generate a set of 10 random multiplication questions that will use the numbers from 1 to 10 whilst the answers must be integers between from 1 to 100. The child that will be using this program will input their answer and the program would output if they got the question right or wrong. After the child has answered all the 10 questions it will then mark their child's answers out of 10, (which would output for example "Well done you've got 7 out of 10 correct); but I would also have to implement an error checking system on the input to ensure that only integers that are in the range 1 to 100 are entered. I will also have to develop a supportive message that will help the child improve their performance. How that would all work is it would open up a command prompt, then it will give you a question that would be randomly generated (that would only use numbers from 1 to 10, so for example: what is 8×4 ?), then when you answer it, it will give you another generated question 9 more times before you'll finish it and you'll get given how many questions you've got right by the marking system that will be in place (so 8/10) with a message that will encourage the child to improve on their performance if they get under 10.

What is project management?

It is the discipline of applying processes and principles to initiate, plan, execute new initiatives whilst also managing the way they are implemented within an organisation because it involves planning, controlling, monitoring and delegating all aspects of the project and motivating those who are involved so they can achieve the project's objectives.

The key aspects that should be managed is:

- Time - so how long it should take
- Benefits - how will the software development company benefit from this

By managing the project, they define a new project at a higher level so they have authorisation to start the multiplication program project, they will define the scope, schedule (so when should I work on the program), and resources which are needed to complete the project as well having an approach to managing this project.

Software Development Lifecycle

The software development lifecycle (which in short is SDLC) is the process that effectively costs less time and money; development teams use this in order to build and design software with high quality. The goal of this process is to minimise project risks by planning forwardly so that the software meets what customer is expecting during their production and beyond. However, they outline several tasks that are required to build a software application as it works by going through several stages as developers will be adding new features and fix bugs in their software.

- Plan – this involves tasks such as analysing, scheduling, estimating and allocating resources which will all be done in a way that will benefit costs. The development team collects requirements from customers, internal and external experts, and managers to create a software requirement specification document.
- Design – software engineers analyse requirements and identify best solutions to creating software. For example, they may consider integrating modules that already exist, making technological choices, and identifying developmental tools. They'll research the best way on how to integrate new software into existing IT infrastructure the organisation may have.
- Implement – the development team codes the program. They analyse requirements that's there to identify smaller coding tasks they can do daily to achieve the final result.
- Test – they combine automation and manual testing to check the software for bugs. Quality analysis includes testing software for errors and checking if it meets customer requirements. Because many development teams test the code they write immediately, the testing phase often runs parallel to the development phase.
- Deploy – when teams develop software, they code and test on a different copy of the software than the one that users have access to. The software customers use is called production, while other copies are in the build or testing environment. Having separate build and production environments ensures that customers can continue to use the software even while it's being changed or upgraded. The deployment phase includes several tasks to move the latest build copy to the production environment, such as packaging, environmental configuration and installation.
- Maintain – the development team fixes bugs, resolves customer issues and manages software changes. Additionally, they monitor overall system performance, security and user experience to identify new ways to improve existing software.

What is project planning and why is it important?

Project planning is when you make a plan to do a project, so say that you're making a project about a kids game (like I'm doing) you will need to plan how you will do that so what programming language? What are the requirements? What program do you need to use? And it is important because a plan will be treated as a manual that will guide you through the steps on how you will code a multiplication game, and you wouldn't have to keep guessing wildly on what to do now and next. It will also give you keep the deadlines on track as since you will gain confidence from the plan you will be rest assured that you can easily work on the project.

What is design and why is it important?

Design is a concept of a project; so it shows the programmer how the project should look like. Without a design, the programmer will have no idea how it should be made or structured; so it is important to have one because of that.

What is a pseudocode?

Pseudocode is fake representation of code which is used as part of planning or design as a way to make it easier for anyone with basic programming knowledge.

How will I do pseudocode and why?

I will do pseudocode in Microsoft Word (on this document) because it is just fictitious code that doesn't need to be perfect as long as everybody gets what it does, then they will take inspiration from that pseudocode and turn it into usable code on their project.

What is a flowchart?

Flowcharts are visual diagrams which represent steps in sequential order which is widely used when it comes to presenting the **flow** of algorithms, **workflow** and/or processes. Flowcharts typically contain boxes which will be where steps go (although the shapes are different depending what they do, so for example a choice would be a different shape, but a step would be a rectangle), and it gets ordered by connecting each other with arrows.

How will I make a flowchart and why?

I will make a flowchart by using draw.io because I have had experience with this website and it's got the shapes and connectors I need therefore I can easily make the flowchart on there, then I will screenshot it and put it on this document.

Original Pseudocode

```
Print "Welcome to the multiplications game! This will give you a set of 10 questions which can go up to the 10 times tables. Good luck!" // intro
```

```
Set correctAnswers = 0 // sets the correct answer to 0
```

```
Set wrongAnswers = 0 // sets the wrong answers to 0
```

```
Create a Random object called random
```

```
For i from 1 to 10 do
```

```
    Set number1 = random number between 1 and 10 // generates a random number between from 1 to 10
```

```
    Set number2 = random number between 1 and 10 // same as above
```

```
    Print "Question i | number1 x number2 = ?" // outputs the question
```

Declare playerGuess without initialization

Set letters to false

Repeat until letters is true

 Get input from user and store it in input

 Try to parse input into an integer and store the result in playerGuess, set letters to true if successful

 If letters is false then

 Print "Please enter a number." // it will output this whenever someone types a non-number (could be a letter or a spacebar)

 End If

End Repeat

Set correctAnswer = number1 * number2

If playerGuess is equal to correctAnswer then

 Print "Correct!"

 Increment correctAnswers by 1

Else

 Print "Wrong. The answer is: correctAnswer"

 Increment wrongAnswers by 1

End If

End For

Print ""

Print "You've got correctAnswers out of 10 correct"

If correctAnswers is equal to 10 then

 Print "Well done! You are now The Maths Superstar!"

Else

 Print "Good try, but be sure to keep practicing to become The Maths Superstar!"

End If

Modified Pseudocode

Print "Welcome to the multiplications game! This will give you a set of 10 questions which can go up to the 10 times tables. Good luck!" // intro

Set correctAnswers = 0 // sets the correct answer to 0

Set wrongAnswers = 0 // sets the wrong answers to 0

Create a Random object called random

For i from 1 to 10 do

 Set number1 = random number between 1 and 10 // generates a random number between from 1 to 10

 Set number2 = random number between 1 and 10 // same as above

 Print "Question i | number1 x number2 = ?" // outputs the question

 Declare playerGuess without initialization

 Set letters to false

 Repeat until letters is true

 Get input from user and store it in input

 Try to parse input into an integer and store the result in playerGuess, set letters to true if successful

 If letters is false then

 Print "Please enter a number." // it will output this whenever someone types a non-number (could be a letter or a spacebar)

 End If

 End Repeat

 Set correctAnswer = number1 * number2

 If playerGuess is equal to correctAnswer then

 Print "Correct!"

 Increment correctAnswers by 1

 Else

 Print "Wrong. The answer is: correctAnswer"

```

    Increment wrongAnswers by 1

End If

End For

Print ""

Print "You've got correctAnswers out of 10 correct"

If correctAnswers is equal to 10 then

    Print "Well done! You are now The Maths Superstar!"

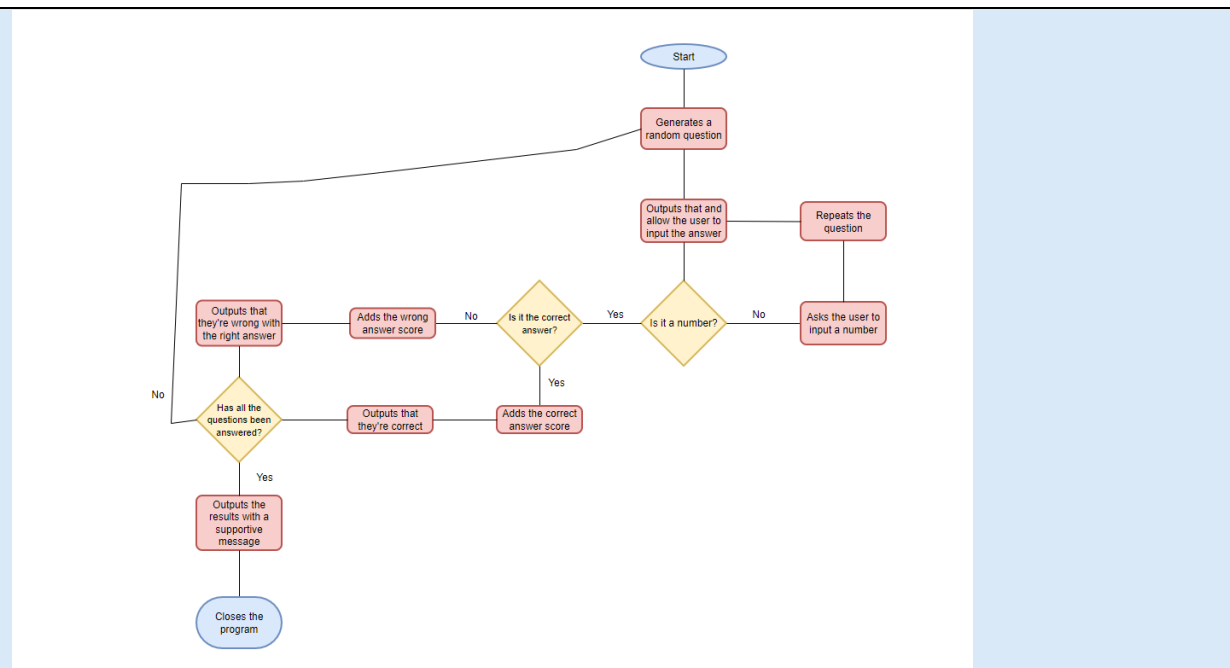
Else

    Print "Good try, but be sure to keep practicing to become The Maths Superstar!"

End If

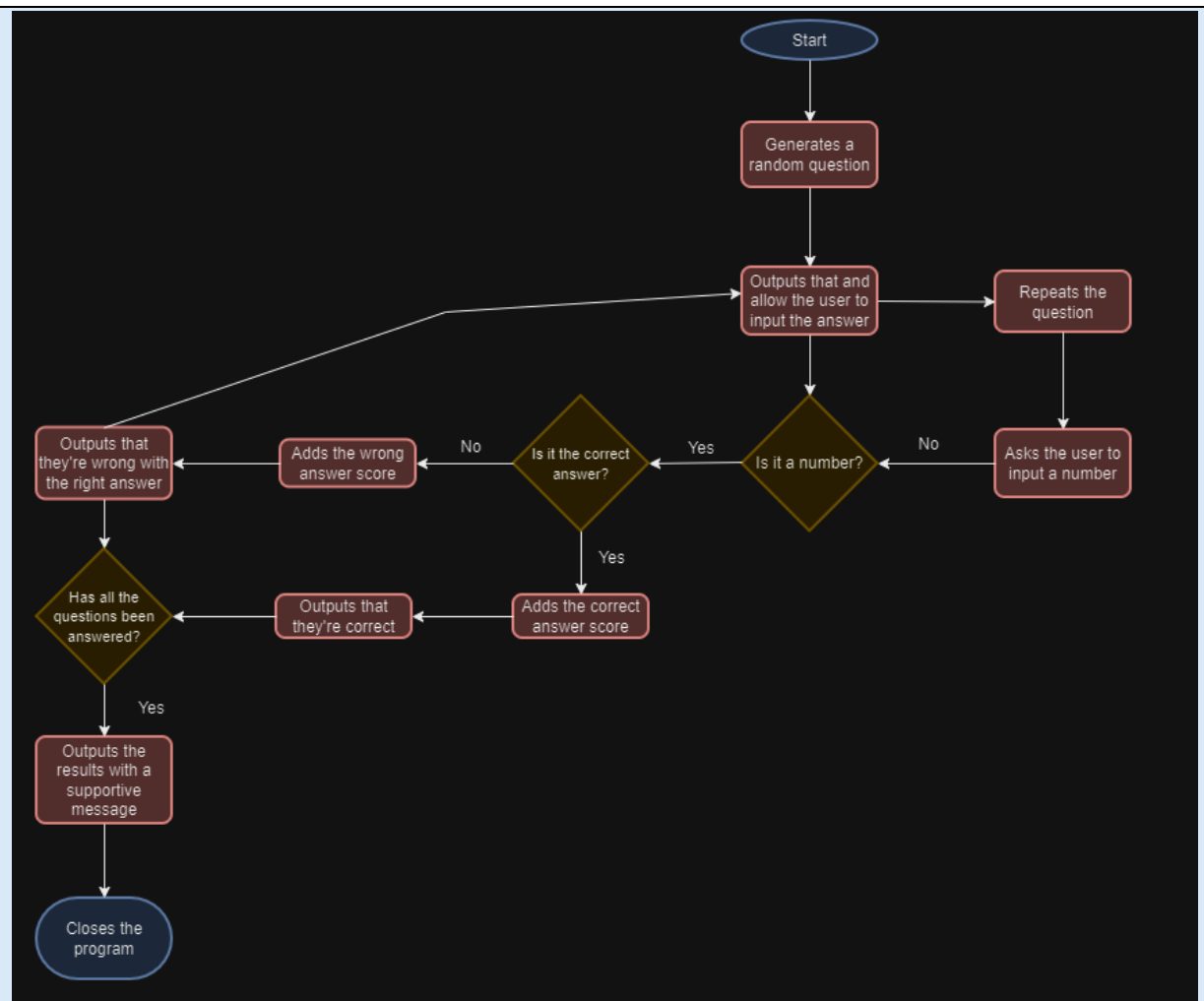
```

Original Flowchart



This shows how the program will work. So when the program starts up, it will generate a random question then it will output that question and allows the user to answer that question; if it's a number, it will be treated as an answer regardless if it's wrong or right, and if it's not, it will warn the user to type a number and it'll just repeat the question that's being asked already. If the answer is correct then the correct answer score will be added 1, and the same goes for the wrong answers just only that the wrong answer score will be added 1. After that, it will repeat itself 9 times; then if it gets to all 10 questions, it will output the results with a supportive message.

Modified Flowchart



This shows how the program will work. So when the program starts up, it will generate a random question then it will output that question and allows the user to answer that question; if it's a number, it will be treated as an answer regardless if it's wrong or right, and if it's not, it will warn the user to type a number and it'll just repeat the question that's being asked already. If the answer is correct then the correct answer score will be added 1, and the same goes for the wrong answers just only that the wrong answer score will be added 1. After that, it will repeat itself 9 times; then if it gets to all 10 questions, it will output the results with a supportive message. I have added arrows so it makes it easier for people to read the flowchart instead of plain lines.

Original Source Code Data

```
using System;
```

```
class Program
```

```
{
```

```

static void Main(string[] args)
{
    Console.WriteLine("Welcome to the multiplications game! This will give you a set of 10
questions which can go up to the 10 times tables. Good luck!"); // introduction

    int correctAnswers = 0; // sets the correct answer to 0

    int wrongAnswers = 0; // sets the wrong answers to 0

    Random random = new Random();

    for (int i = 1; i <= 10; i++)
    {
        int number1 = random.Next(1, 11); // generates a random number between from 1 to 10

        int number2 = random.Next(1, 11); // same as above

        Console.WriteLine("Question {0} | {1} x {2} = ?", i, number1, number2); // outputs the
question

        int playerGuess; // Declare playerGuess without initialization

        bool letters = false;

        do
        {
            string input = Console.ReadLine();

            letters = int.TryParse(input, out playerGuess);

            if (!letters)
            {
                Console.WriteLine("Please enter a number."); // it will output this whenever someone
types a non-number (could be a letter or a spacebar)
            }
        } while (!letters);

        int correctAnswer = number1 * number2;

        if (playerGuess == correctAnswer)
        {

```

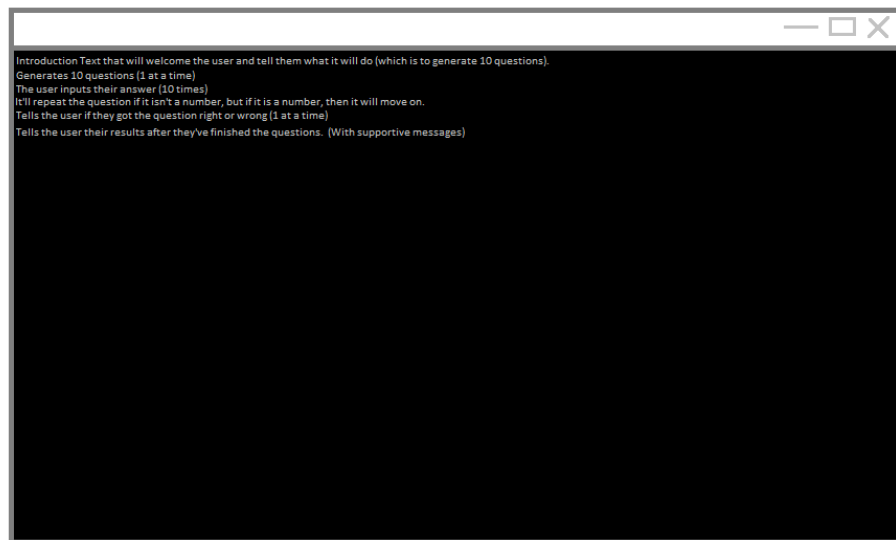
```
        Console.WriteLine("Correct!");
        correctAnswers++;
    }
    else
    {
        Console.WriteLine("Wrong. The answer is: {0}", correctAnswer);
        wrongAnswers++;
    }
}

Console.WriteLine();

Console.WriteLine("You've got {0} out of 10 correct", correctAnswers);

if (correctAnswers == 10)
{
    Console.WriteLine("Well done! You are now The Maths Superstar!");
}
else
{
    Console.WriteLine("Good try, but be sure to keep practicing to become The Maths Superstar!");
}
}
```

Wireframe



Screenshots

```
Welcome to the Multiplications game! This will give you a set of 10 questions which can go up to the 10 times tables. Good luck!
Question 1 | 4 x 1 = ?

Please enter a number.
65
Wrong. The answer is: 4
Question 2 | 4 x 10 = ?
40
Correct!
Question 3 | 10 x 3 = ?
30
Correct!
Question 4 | 6 x 3 = ?
12
Wrong. The answer is: 18
Question 5 | 5 x 5 = ?
25
Correct!
Question 6 | 1 x 10 = ?
10
Correct!
Question 7 | 5 x 5 = ?
25
Correct!
Question 8 | 8 x 6 = ?
34
Wrong. The answer is: 48
Question 9 | 4 x 9 = ?
17
Wrong. The answer is: 36
Question 10 | 5 x 9 = ?
50
Wrong. The answer is: 45

You've got 5 out of 10 correct
Good try, but be sure to keep practicing to become The Maths Superstar!

C:\Users\comp5248971\source\repos\MultiplicationGames\MultiplicationGames\bin\Debug\net6.0\MultiplicationGames.exe (process 22476) exited with code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops.
Press any key to close this window . . .
```

```
using System;

class Program
{
    static void Main(string[] args)
    {
        Console.WriteLine("Welcome to the multiplications game! This will give you a set of 10 questions which can go up to the 10 times tables. Good luck!"); // intro
        int correctAnswers = 0; // sets the correct answer to 0
        int wrongAnswers = 0; // sets the wrong answers to 0
        Random random = new Random();
        for (int i = 1; i <= 10; i++)
        {
            int number1 = random.Next(1, 11); // generates a random number between from 1 to 10
            int number2 = random.Next(1, 11); // same as above
            Console.WriteLine("Question {0} | {1} x {2} = ?", i, number1, number2); // outputs the question

            int playerGuess;

            bool letters = false;
            do
            {
                string input = Console.ReadLine();
                letters = int.TryParse(input, out playerGuess);

                if (!letters)
                {
                    Console.WriteLine("Please enter a number."); // it will output this whenever someone types a non-number (could be a letter or a spacebar)
                } while (!letters);
            } while (!letters);

            int correctAnswer = number1 * number2;
            if (playerGuess == correctAnswer)
            {
                Console.WriteLine("Correct!");
                correctAnswers++;
            }
            else
            {
                Console.WriteLine("Wrong. The answer is: {0}", correctAnswer);
                wrongAnswers++;
            }
        }

        Console.WriteLine();
        Console.WriteLine("You've got {0} out of 10 correct", correctAnswers);
        if (correctAnswers == 10)
        {
            Console.WriteLine("Well done! You are now The Maths Superstar!");
        }
        else
        {
            Console.WriteLine("Good try, but be sure to keep practicing to become The Maths Superstar!");
        }
    }
}
```

Modified Source Code Data

```
using System;
```

```
class Program
```

```
{
```

```
    static void Main(string[] args)
```

```
    {
```

Console.WriteLine("Welcome to the multiplications game! This will give you a set of 10 questions which can go up to the 10 times tables. Good luck!"); // outputs this line, meaning this will display when the user launches up the program

int correctAnswers = 0; // sets the correct answer score to 0

int wrongAnswers = 0; // sets the wrong answers score to 0

Random random = new Random(); // this will let it generate a random number

for (int i = 1; i <= 10; i++) // sets up minimum and maximum of randomised numbers

```
        {
```

int number1 = random.Next(1, 11); // generates a random number between from 1 to 10

int number2 = random.Next(1, 11); // same as above

Console.WriteLine("Question {0} | {1} x {2} = ?", i, number1, number2); // outputs the question, meaning the question will display

int playerGuess; // this is where the users type their answer

```

bool letters = false; // disallows letters

do
{
    string input = Console.ReadLine();

    letters = int.TryParse(input, out playerGuess);

    if (!letters)
    {
        Console.WriteLine("Please enter a number."); // it will display this if someone types a
non-number (could be a letter or a spacebar)

    }

} while (!letters);

int correctAnswer = number1 * number2; // calculates the generated question so it can
find the answer to see if the user's answer is correct or not

if (playerGuess == correctAnswer) // so if the answer is correct.
{
    Console.WriteLine("Correct!"); // it will display this message that's in quotes

    correctAnswers++; // adds 1 to the correct answers score
}

else
{
    Console.WriteLine("Wrong. The answer is: {0}", correctAnswer); // it will display the
correct answer

    wrongAnswers++; // adds 1 to the wrong answers score
}

}

Console.WriteLine();

Console.WriteLine("You've got {0} out of 10 correct", correctAnswers); // displays how many
you've got right, like a marking system.

if (correctAnswers == 10) // if the user got all the questions right
{

```

```
        Console.WriteLine("Well done! You are now The Maths Superstar!"); // displays this if the
user got it all right

    }

    else

    {

        Console.WriteLine("Good try, but be sure to keep practicing to become The Maths
Superstar!"); // displays this if the user didn't get it all right

    }

}

}
```

Design Reviews

William Sanders:

“For the pseudocode, it's good, but it is a bit hard to follow. As some of the code that is connected to each other isn't on the same line, perhaps separate the non-connected parts by a few lines to make it easier to read. For example, after "sets the wrong answers to 0" you could add a couple of lines before "create a random object called random". Additionally, you could link connected code by colour coding it. For example, you could colour the sequence from "if letter is false then" to "end repeat" in red, and then the next sequence in blue. For the flowchart, I would suggest adding in arrows instead of just lines, both single directional and bi-directional, to make it clearer which parts of the flowchart connect to other parts and in what way. For example, the "has all the questions been answered" part only links to "generates a random question", and not the other way round.”

Callum Holman-Hathaway:

“I have reviewed your document and I have seen a very good amount of information to describe to the user about different features and the overall design. Your overall program shows supportive messages to the children”

I will make changes to my flowchart and pseudocode based on reviews that's been given, and I will add a dark blue background on them to show that it's modified.

Program Reviews

Callum Holman-Hathaway:

“Hi Josh, I am reviewing the following code you have sent me is suitable towards the clients needs as it includes the following features of supportive messages, scores and a generator for the numbers that it needs to go up to. Your code is very simplistic and has been condensed so it doesnt use too much code which is beneficial for both the end users and the client.”

Lewis James:

“You could add some more comments on your code so that the people who are reading your program understand which code does what. For example, if you were to code in an error system

with 3-4 lines of code, you could explain which each line does so the person that's reading your source data will fully understand them."

How does the program meet the user requirements?

The program meets the user requirements because it has got everything from the user requirements: for example, the client wants the program to generate 10 questions that users can answer, that only has numbers in the question ranging from 1 to 10, and the answers of each question must be integers from 1 to 100. Not only that but I've also added other things such as an error checking system (where it would check if the answer is a number or not, and if it isn't, then it will tell the user to put in a number instead of a non-number like a letter or a symbol), a supportive message that tells the user (as they're a child according to the target audience) that they're a maths superstar whilst phrasing them for their good work.

Why did I code C# and why did I use Visual Studio?

I coded the program in C# as I understand the programming language more than other programming languages such as Python and JavaScript. I used Visual Studio because I am more familiar with the program than other coding editors such as Notepad++.

Why did I code the program this way?

Because if I don't code the program the way that it should be, then it probably either wouldn't work properly or I wouldn't understand the code, therefore I'll be unable to fix errors that the program might make since I would not be able to find solutions since I won't have understood the code.

Advantages and Disadvantages of the Original Code

Advantages

- The code works as intended.
- The error system works as intended.
- The marking system works as intended.

Disadvantages

- Not enough comments on the code. (Solved on the modified code)
-

References

<https://www.axelos.com/certifications/prince2/what-is-project-management>

