Computing

GCSE

Computing
Exemplar Candidate Work

J275

Unit A453 Sample Material C1

Version 1

Disclaimer on use of Sample Material

Confidentiality

These tasks are taken from legacy Controlled Assessment tasks, undertaken and submitted by candidates. Where possible, we have removed all identifying information from these assessments. Should any data remain, you are requested to treat this confidentially and inform OCR as soon as possible highlighting the data concerned.

Use of URS Sheets and Sample Material

These tasks have all been moderated as part of the relevant exam series in which they were submitted and the marks submitted have all been allowed to stand. However, schools should bear in mind that this only indicates that the **overall assessment** of the Controlled Assessment is within tolerance and not necessarily each individual mark band. There may be instances where the mark scheme has been applied too generously, or similarly too harshly. This would have been identified in the reports to the centre – but will not be evident from URS alone. The spirit of the release of these samples is to give teachers better understanding of what High, Medium and Low graded coursework would feel like as an entity, rather than exact definitions of requirements for mark bands independently.

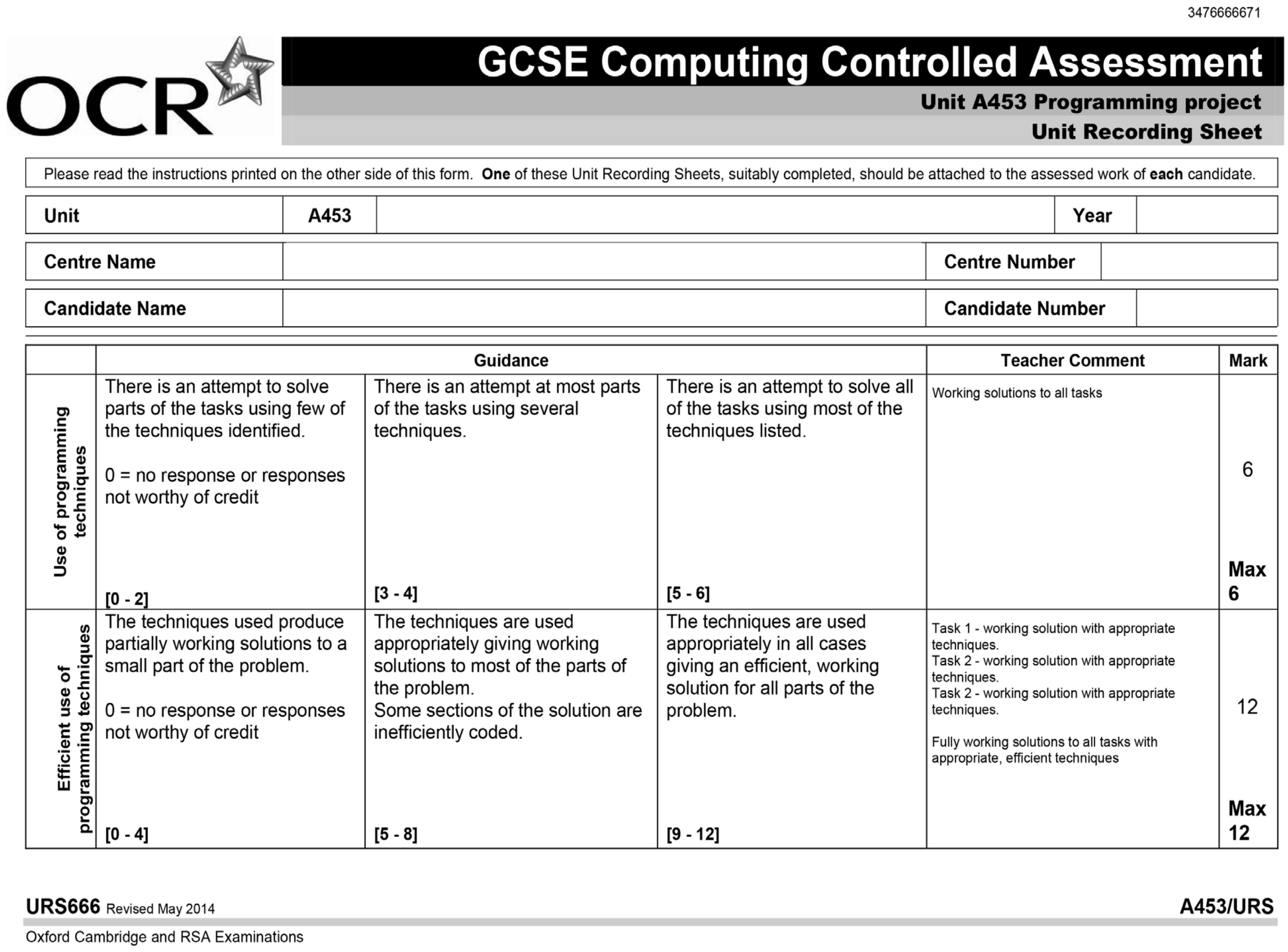
The provision of high graded work should **not infer** that this is the only, or best way of writing up a Controlled Assessment Task. Candidates are encouraged to map their personal journey through the tasks. Writing frames, or 'guides' for documentation are against the spirit of the coursework and constitute malpractice.

Each set of materials released contains a High, Middle and Low grade band. This should allow teachers to gain good understanding of the general standard of work quality required for each mark band, and as a whole – especially when comparing each set side by side.

Teachers are encouraged to seek further support when they feel clarification is needed in applying the mark scheme. We would also recommend regular CPD in respect of Controlled Assessment delivery and marking.

Accuracy

All work has, where possible, remained unaltered from the original submission. There may well be grammatical errors and poor layout in diagrams. This is to allow better matching of mark band criteria, where specific bullet points refer to quality of Spelling, Punctuation and Grammar, and also ease of navigation etc. Any significant changes are clearly marked. Some data that is perceived sensitive may be blocked out in black.



3476666671

GCSE Computing Controlled Assessment

Unit A453 Programming project

Unit Recording Sheet

Please read the instructions printed on the other side of this form. **One** of these Unit Recording Sheets, suitably completed, should be attached to the assessed work of **each** candidate.

Unit	A453	**Year**	
Centre Name		**Centre Number**	
Candidate Name		**Candidate Number**	

	Guidance			**Teacher Comment**	**Mark**
Use of programming techniques	There is an attempt to solve parts of the tasks using few of the techniques identified. 0 = no response or responses not worthy of credit **[0 - 2]**	There is an attempt at most parts of the tasks using several techniques. **[3 - 4]**	There is an attempt to solve all of the tasks using most of the techniques listed. **[5 - 6]**	Working solutions to all tasks	6 **Max 6**
Efficient use of programming techniques	The techniques used produce partially working solutions to a small part of the problem. 0 = no response or responses not worthy of credit **[0 - 4]**	The techniques are used appropriately giving working solutions to most of the parts of the problem. Some sections of the solution are inefficiently coded. **[5 - 8]**	The techniques are used appropriately in all cases giving an efficient, working solution for all parts of the problem. **[9 - 12]**	Task 1 - working solution with appropriate techniques. Task 2 - working solution with appropriate techniques. Task 2 - working solution with appropriate techniques. Fully working solutions to all tasks with appropriate, efficient techniques	12 **Max 12**

URS666 Revised May 2014 **A453/URS**

Oxford Cambridge and RSA Examinations

Design	There are comments on what the task involves and a limited outline describing the intended approach to some parts of the problem. There are brief comments on how this might be tested but with no mention of success criteria. 0 = no response or responses not worthy of credit **[0 - 3]**	There is a brief analysis of the tasks indicating what is required for each of the tasks. There is a set of basic algorithms outlining a solution to most parts of the problem. There is some discussion of how this is tested and how this compares to the identified outcomes in the tasks. There is discussion of the variables to be used and some general discussion of validation. **[4 - 6]**	There is a detailed analysis of what is required for these tasks justifying their approach to the solution. There will be a full set of detailed algorithms representing a solution to each part of the problem. There is detailed discussion of testing and success criteria. The variables and structures are identified together with any validation required **[7 - 9]**	Task 1 - success criteria, flow chart, pseudo code, test plan, variables and validation. Task 2 - success criteria, flow chart, pseudo code, variables and validation, test plan. Task 3 - success criteria, flow chart, pseudo code, variables and validation stated, test plan. Excellent design throughout, covering all elements with good algorithm designs	9 **Max 9**
Development	There is some evidence to show a solution to part of the problem with some evidence to show that it works. Code is presented with little or no annotation, the variable names not reflecting their purpose and with little organisation or structure. 0 = no response or responses not worthy of credit **[0 - 3]**	There is evidence to show how the solutions were developed. There is some evidence of testing during development showing that many parts of the solution work. The code is organised with sensible variable names and with some annotation indicating what sections of the code does. **[4 – 6]**	There is detailed evidence showing development of the solution with evidence of systematic testing during development to show that all parts work as required. The code is well organised with meaningful variable names and detailed annotation indicating the function of each section. **[7- 9]**	Task 1 - shows order of creation and testing during development. Code has appropriate variable and object names, with annotation. Task 2 - evidence of how they were developed, with testing. Code is appropriate with annotation. Task 3 - shows how the solution was developed and describes testing. Code is appropriate with annotation. Evidence of how the solutiosn were developed with evidence of systematic testing. Code is well organised with annotation	8 **Max 9**

URS666 Revised May 2014 **A453/URS**

Oxford Cambridge and RSA Examinations

3476666671

Testing and evaluation	There is evidence to show that the system has been tested for function but the test plan is limited in scope with little structure. There is limited evidence to show how the result matches the original criteria. The evidence of written communication is limited with little or no use of specialist terms. Errors in spelling, punctuation and grammar may be intrusive. Information may be ambiguous or disorganised. 0 = no response or responses not worthy of credit **[0 - 3]**	There is a test plan covering many parts of the problem with some suggested test data. There is evidence that the system has been tested using this data. There is some evidence to show how the results of testing have been compared to the original criteria. There is a brief discussion of how successful or otherwise the solutions are. The quality of written communication is good using some specialist terms. There are few errors in spelling, grammar and punctuation. Information for the most part is presented in a structured format. **[4 - 6]**	The test plan covers all major success criteria for the original problem with evidence to show how each of these criteria have been met, or if they have not been met, how the issue might be resolved. There is a full evaluation of the final solution against the success criteria. A high level of written communication is obvious throughout the task and specialist terms/ technology with accurate use of spelling will have been used. Grammar and punctuation are used correctly and information is presented in a coherent and structured format. **[7 - 9]**	Task 1 - Completed test table with evidence. Comparison to success criteria. Task 2 - completed test table, with evidence. Comparison to success criteria. Task 3 - completed test table with evidence, comparison to success criteria. Through evidence of testing for all tasks, with comparison to success criteria	9 **Max 9**
				Total/45	**44**

Guidance on Completion of this Form

1 **One** sheet should be used for each candidate.
2 Please ensure that the appropriate boxes at the top of the form are completed.
3 Using the guidance identify the most appropriate mark range for the work and enter the mark awarded for each element in the mark column.
4 Add appropriate comments to assist the moderator in the 'Teacher Comment' column.
5 Add the marks for the strands together to give a total out of 45. Enter this total in the relevant box.

A453

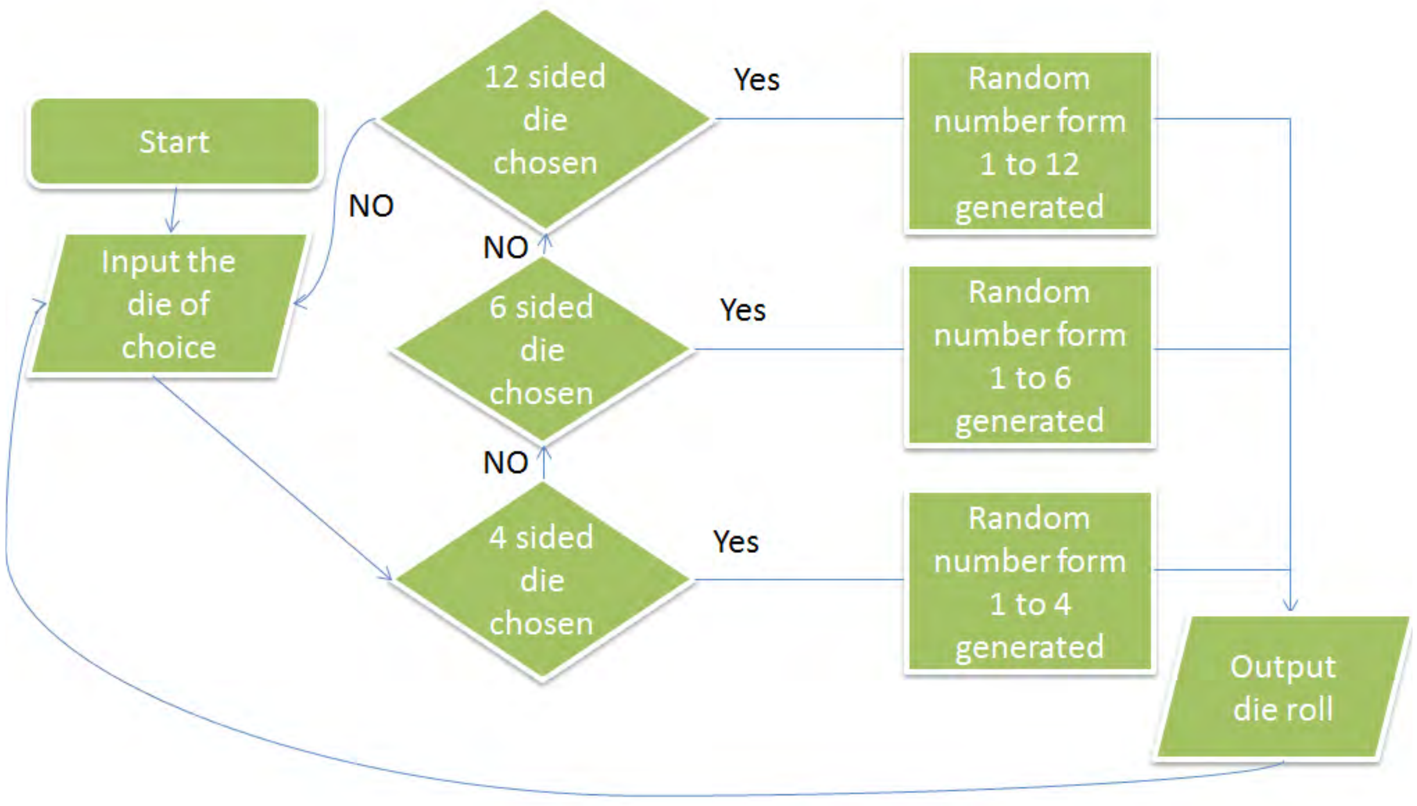
Task 1

What I'm going to do.
Create program that simulates a roll of a 4, 6 or 12 sided dice. The user will pick what dice is to be thrown the program will then output the result of the throw. This will be able to be done as many time as required.

Success Criteria

- The system should display a choice of a 4, 6 or 12 die.
- The user will then be able to choose which die that will be rolled
- The system will then 'roll' the chosen die
- The output of die chosen and the die roll will then be displayed
- The process will then be able to be repeated

Flow chart



Pseudo Code

```
4 sided die clicked
Dice = 4
No = random number between 1 and 4
Output "Dice" & (Dice) & "Has been chosen. The result was " & (No)
6 sided die clicked
Dice = 6
No = random number between 1 and 6
Output "Dice" & (Dice) & "Has been chosen. The result was " & (No)
12 sided die clicked
Dice = 12
No = random number between 1 and 12
Output "Dice" & (Dice) & "Has been chosen. The result was " & (No)
```

Test plan

What I'm testing	How I will test it	Expected Output
That the 4 sided die works	Click 4 sided die	"The 4 sided dice has been rolled." & A number between 1 and 4
That the 6 sided die works	Click 6 sided die	"The 6 sided dice has been rolled." & A number between 1 and 6
That the 12 sided die works	Click 12 sided die	"The 12 sided dice has been rolled." & A number between 1 and 12
That the user can use the program as many time as desired	Click the dice randomly for 60 seconds	No crashes

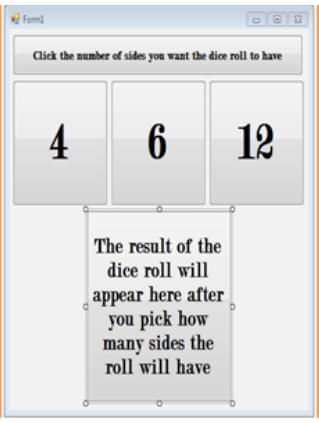
Variables

- No = this will store the number generated locally ready for output. (Integer)
- Die = This will store the dice chosen. (Integer)
- Ran = This is where the random number will be generated (As New random)

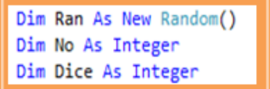
Validation

The will be no way to enter an invalid input as the only input methods will be the selection buttons.

Development

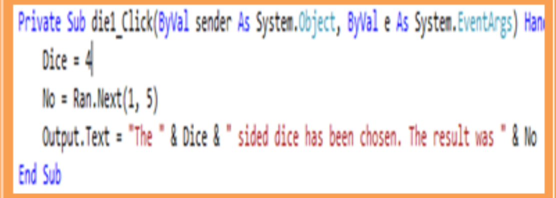

Form1
Click the number of sides you want the dice roll to have
4
6
12
The result of the dice roll will appear here after you pick how many sides the roll will have

First I set up the form design with the dice option input as 3 buttons.



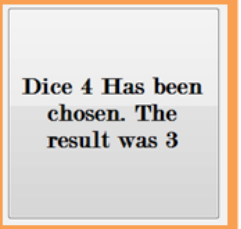
```
Dim Ran As New Random()
Dim No As Integer
Dim Dice As Integer
```

Second I declared my variables. (Refer to variables and validation)



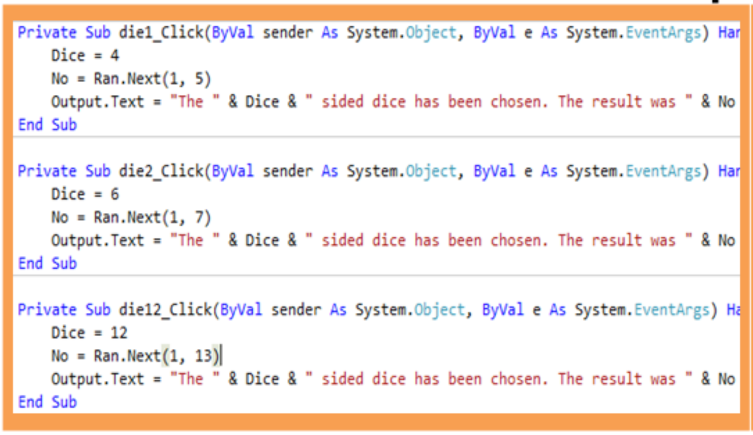
```
Private Sub die1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Han
    Dice = 4
    No = Ran.Next(1, 5)
    Output.Text = "The " & Dice & " sided dice has been chosen. The result was " & No
End Sub
```

Then I made the first die roll code this "rolls the dice" then outputs the dice rolled and the result.



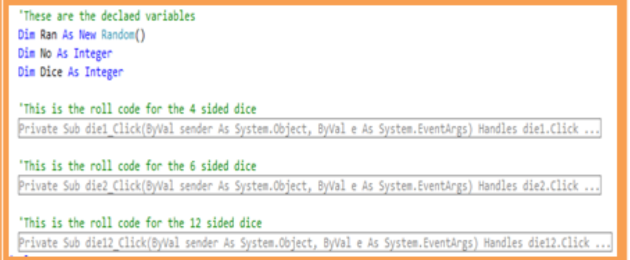
Dice 4 Has been chosen. The result was 3

I then tested the code I'd written by clicking dice 4 The expected output was displayed.



```
Private Sub die1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Har
    Dice = 4
    No = Ran.Next(1, 5)
    Output.Text = "The " & Dice & " sided dice has been chosen. The result was " & No
End Sub

Private Sub die2_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Har
    Dice = 6
    No = Ran.Next(1, 7)
    Output.Text = "The " & Dice & " sided dice has been chosen. The result was " & No
End Sub

Private Sub die12_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Ha
    Dice = 12
    No = Ran.Next(1, 13)
    Output.Text = "The " & Dice & " sided dice has been chosen. The result was " & No
End Sub
```

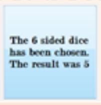
I repeated this for the 6 and 12 sided die they also worked as expected after their individual tests.



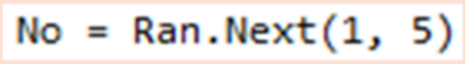
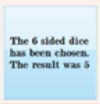
```
'These are the declaed variables
Dim Ran As New Random()
Dim No As Integer
Dim Dice As Integer

'This is the roll code for the 4 sided dice
Private Sub die1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles die1.Click ...

'This is the roll code for the 6 sided dice
Private Sub die2_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles die2.Click ...

'This is the roll code for the 12 sided dice
Private Sub die12_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles die12.Click ...
```

After I appropriately commented the code the programme was now finished so I proceeded to the testing.

Testing

What I'm testing	How I will test it	Expected Output	Actual output
That the 4 sided die works	Click 4 sided die	"The 4 sided dice has been rolled." & A number between 1 and 4	"The 4 sided dice has been rolled." & A number between 1 and 4
That the 6 sided die works	Click 6 sided die	"The 6 sided dice has been rolled." & A number between 1 and 6	"The 6 sided dice has been rolled." & A number between 1 and 6  The 6 sided dice has been chosen. The result was 5
That the 12 sided die works	Click 12 sided die	"The 12 sided dice has been rolled." & A number between 1 and 12	"The 12 sided dice has been rolled." & A number between 1 and 12
That the user can use the program as many time as desired	Click the dice randomly for 60 seconds	No crashes	No crashes

Criteria Check

Criteria point	How has it been met.
The system should display a choice of a 4,6 or 12 die.	There are 3 selection buttons with the number of sides that dice has displayed one for 4,6 and 12 sided dice
The user will then be able to choose which die that will be rolled	There are 3 selection buttons with the number of sides that dice has displayed one for 4,6 and 12 sided dice
The system will then 'roll' the chosen die	The system has a random number picker for each dice  `No = Ran.Next(1, 5)`
The output of die chosen and the die roll will then be displayed	After a dice roll the output of die chosen and the die roll will then be displayed in the button  The 6 sided dice has been chosen. The result was 5
The process will then be able to be repeated	In testing the dice could be rolled repeatedly without crashes

Overall my system met my success criteria, passed all of my system tests and was simple to use and interact with so, overall was a successes although I could improve it by making the interface more engaging by, for example making an visual dice to output the result.

Full Code

```
Public Class Form1
   'These are the declaed variables
   Dim Ran As New Random()
   Dim No As Integer
   Dim Dice As Integer
   'This is the roll code for the 4 sided dice
   Private Sub die1_Click(ByVal sender As System.Object, ByVal e As System.
EventArgs) Handles die1.Click
       Dice = 4
       No = Ran.Next(1, 5) 'This will generate a randome number between 0 and 4
       Output.Text = "The " & Dice & " sided dice has been chosen. The result was
" & No
   End Sub
'This is the roll code for the 6 sided dice
   Private Sub die2_Click(ByVal sender As System.Object, ByVal e As System.
EventArgs) Handles die2.Click
       Dice = 6
       No = Ran.Next(1, 7) 'This will generate a randome number between 0 and 6
       Output.Text = "The " & Dice & " sided dice has been chosen. The result was
```

```
" & No
  End Sub
  'This is the roll code for the 12 sided dice
  Private Sub die12_Click(ByVal sender As System.Object, ByVal e As System.
EventArgs) Handles die12.Click
     Dice = 12
     No = Ran.Next(1, 13) 'This will generate a randome number between 0 and 12
     Output.Text = "The " & Dice & " sided dice has been chosen. The result was
" & No
  End Sub
```

Task 2

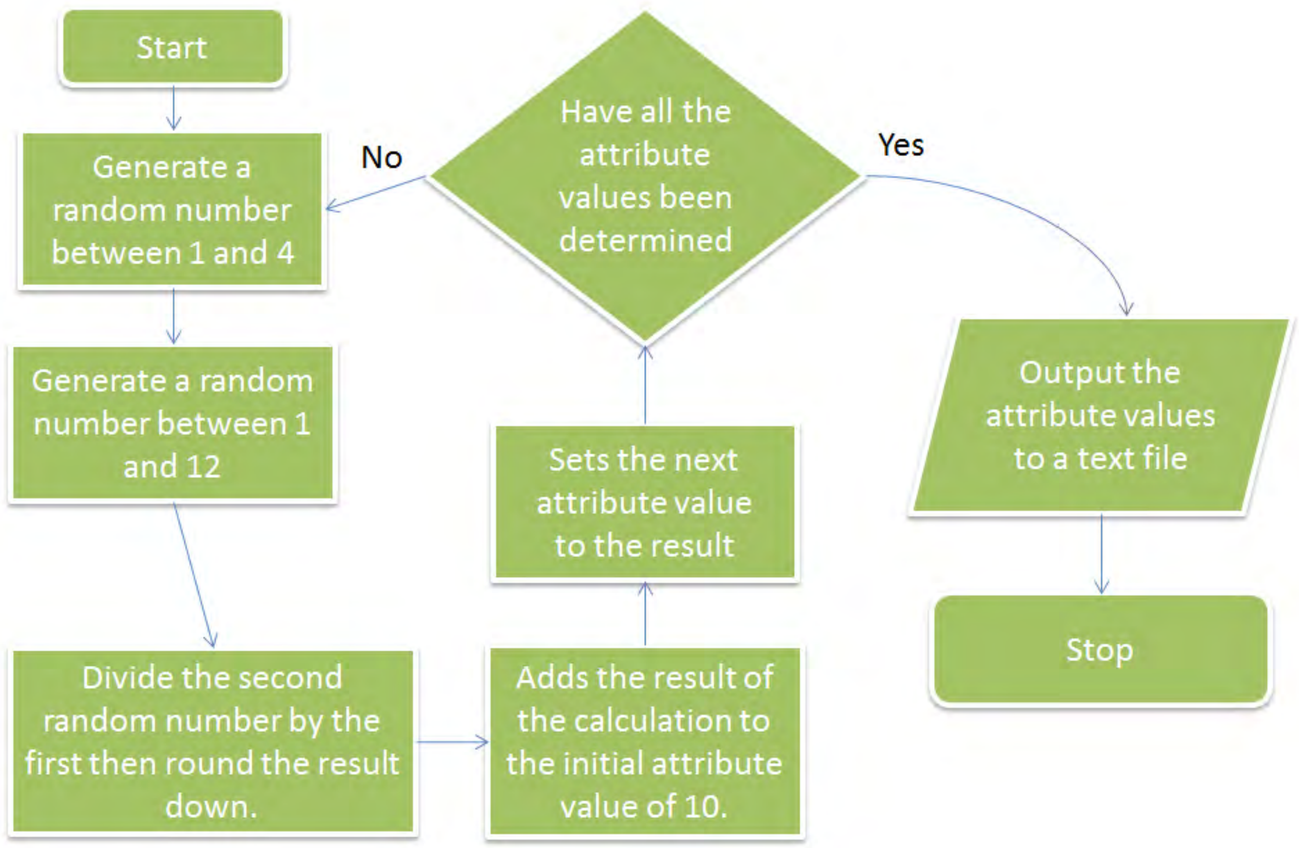
What I'm going to do.

I'm going to make the start of a game with two characters who each have two attributes ,strength and skill. These will be determined by a 4 and 12 sided dice being rolled then the value of the 12 sided dice will be divided by the value of the 4 sided dice. The result will then be rounded down then added to the initial attribute value of 10. This will be repeated for each attribute for each character. The game will then store a sample of the two attributes for both characters to a text file.

Success Criteria

- At the start of the game a 4 and 12 sided dice are rolled
- The value of the 12 sided dice is divided by the value of the 4 sided dice.
- The result will be rounded down then added to the initial attribute value of 10.
- The calculation is repeated while setting the results to the attributes corresponding variable
- The results are outputted to the text file

Flow chart


Start
Generate a random number between 1 and 4
Generate a random number between 1 and 12
Divide the second random number by the first then round the result down.
Adds the result of the calculation to the initial attribute value of 10.
Sets the next attribute value to the result
Have all the attribute values been determined
No
Yes
Output the attribute values to a text file
Stop

Pseudo Code

```
Form load
X=0
Do Until x = 4
No1 = Random number between 1 and 4 generated
No2 = Random number between 1 and 12 generated
Attribute(x) = Math.Floor(No2/No1)
X = X + 1
Loop
Output all the attribute values to a text file
```

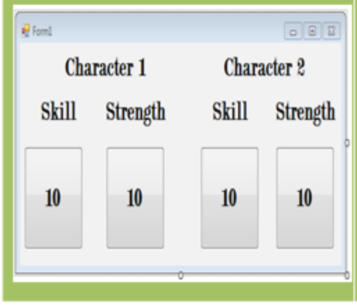
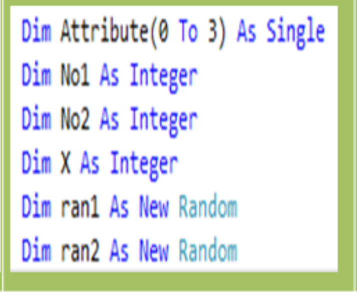
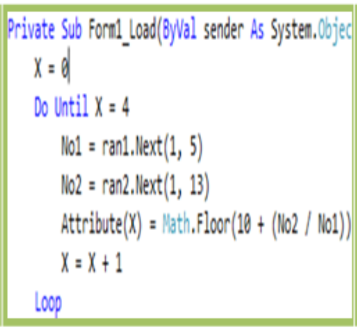
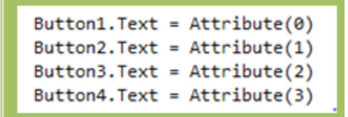
Variables

Filename String
Attribute(0 To 3) Single = The array to store the attribute values
No1 Integer = This to store the 1st random numbers generated
No2 Integer This to store the 2nd random numbers generated
X Integer = This to store the number the loop uses
ran1 Random= The random number generator

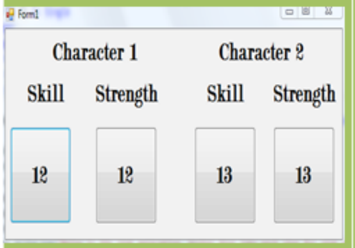
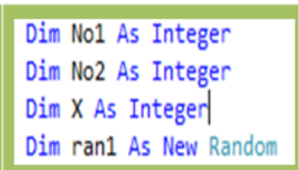
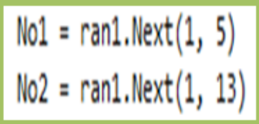
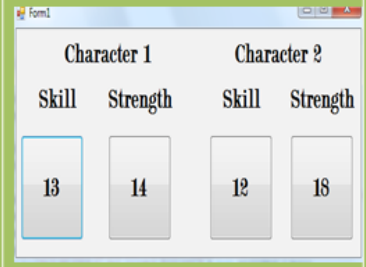
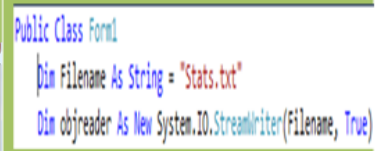
Test plan

What I'm testing	How will I test it	Expected output
That the random numbers generated are random	By running the program multiple times then consulting the text file	A random number between 1 and 4 each time and A random number between 1 and 12
That the program divides the numbers correctly	By running the program multiple times then consulting the text file Testing numbers close, far away and the same as each other	The correct answer to the division
That the result from the division is rounded down correctly	By running the program multiple times then consulting the text file	The answer to the division correctly rounded down
That the result is added to 10 correctly	By running the program multiple times then consulting the text file	The result correctly added to 10 correctly
The results of the operations are outputted to the text file correctly	By running the program multiple times then consulting the text file	The results of the operations are outputted to the text file correctly

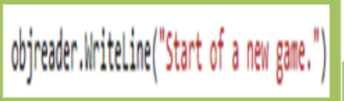
Development

 Form1
Character 1
Character 2
Skill
Strength
Skill
Strength
10
10
10
10
	`Dim Attribute(0 To 3) As Single`
	`Dim No1 As Integer`
	`Dim No2 As Integer`
	`Dim X As Integer`
	`Dim ran1 As New Random`
	`Dim ran2 As New Random`
		`Private Sub Form1_Load(ByVal sender As System.Objec`
		`    X = 0`
		`    Do Until X = 4`
		`        No1 = ran1.Next(1, 5)`
		`        No2 = ran2.Next(1, 13)`
		`        Attribute(X) = Math.Floor(10 + (No2 / No1))`
		`        X = X + 1`
		`    Loop`
			`Button1.Text = Attribute(0)`
			`Button2.Text = Attribute(1)`
			`Button3.Text = Attribute(2)`
			`Button4.Text = Attribute(3)`

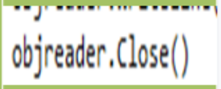
First I made my form layout the 4 buttons to display the output of the stats labelled with the correct titles.	Then I made declared my variables for; all of the stat numbers in an array, the random numbers and the numbers used in the operation.	Next I made the loop that would generate the 4 stats by a 4 and 12 sided dice being rolled then the value of the 12 sided dice will be divided by the value of the 4 sided dice. The result will then be rounded down then added to the initial attribute value of 10. The result of that will then be assigned to the first slot in the array, (the value of X) it would then be repeated for the other 3 slots then stop.	Then I added the code above after the loop this allowed me to test if it was working by outputting the stat values to the 4 buttons on the form display.

 Form1
Character 1
Character 2
Skill
Strength
Skill
Strength
12
12
13
13
	`Dim No1 As Integer`
	`Dim No2 As Integer`
	`Dim X As Integer`
	`Dim ran1 As New Random`
	`No1 = ran1.Next(1, 5)`
	`No2 = ran1.Next(1, 13)`
		 Form1
		Character 1
		Character 2
		Skill
		Strength
		Skill
		Strength
		13
		14
		12
		18
			`Public Class Form1`
			`    Dim Filename As String = "Stats.txt"`
			`    Dim objreader As New System.IO.StreamWriter(Filename, True)`

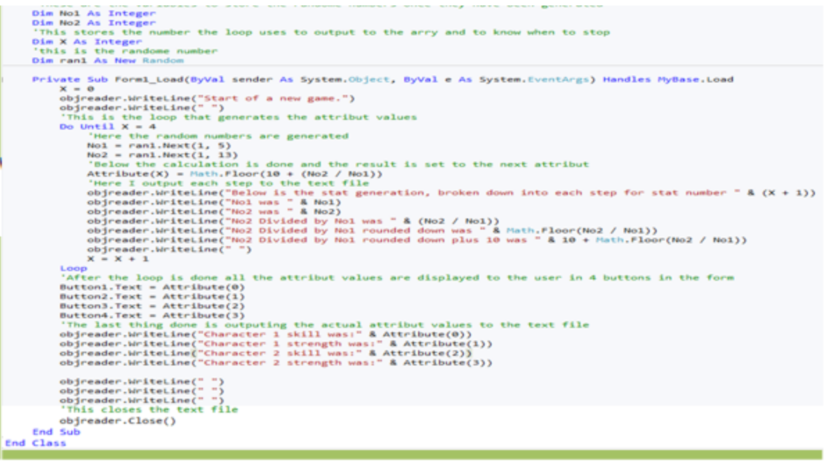
After testing it a few times I noticed the first two number and the last two numbers Were always the same. This was because I had declared two random numbers instead of just one.	To fix this I deleted the ran2 variable and changed the loop to only use ran1 to generate numbers.	This fix the problem and the numbers randomized properly.	Now I needed to output the results to a text file so I added the above to the start of the program to make and open the text file.



```
objreader.WriteLine("Start of a new game.")
```



```
objreader.Close()
```

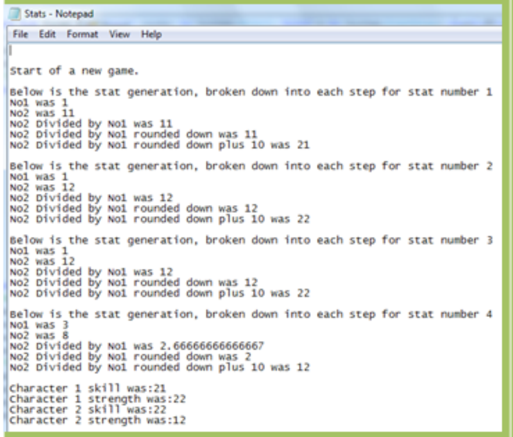


```
        Dim No1 As Integer
        Dim No2 As Integer
        'This stores the number the loop uses to output to the arry and to know when to stop
        Dim X As Integer
        'this is the randome number
        Dim ran1 As New Random

        Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
            X = 0
            objreader.WriteLine("Start of a new game.")
            objreader.WriteLine(" ")
            'This is the loop that generates the attribut values
            Do Until X = 4
                'Here the random numbers are generated
                No1 = ran1.Next(1, 5)
                No2 = ran1.Next(1, 13)
                'Below the calculation is done and the result is set to the next attribut
                Attribute(X) = Math.Floor(10 + (No2 / No1))
                'Here I output each step to the text file
                objreader.WriteLine("Below is the stat generation, broken down into each step for stat number " & (X + 1))
                objreader.WriteLine("No1 was " & No1)
                objreader.WriteLine("No2 was " & No2)
                objreader.WriteLine("No2 Divided by No1 was " & (No2 / No1))
                objreader.WriteLine("No2 Divided by No1 rounded down was " & Math.Floor(No2 / No1))
                objreader.WriteLine("No2 Divided by No1 rounded down plus 10 was " & 10 + Math.Floor(No2 / No1))
                objreader.WriteLine(" ")
                X = X + 1
            Loop
            'After the loop is done all the attribut values are displayed to the user in 4 buttons in the form
            Button1.Text = Attribute(0)
            Button2.Text = Attribute(1)
            Button3.Text = Attribute(2)
            Button4.Text = Attribute(3)
            'The last thing done is outputing the actual attribut values to the text file
            objreader.WriteLine("Character 1 skill was:" & Attribute(0))
            objreader.WriteLine("Character 1 strength was:" & Attribute(1))
            objreader.WriteLine("Character 2 skill was:" & Attribute(2))
            objreader.WriteLine("Character 2 strength was:" & Attribute(3))

            objreader.WriteLine(" ")
            objreader.WriteLine(" ")
            objreader.WriteLine(" ")
            'This closes the text file
            objreader.Close()
        End Sub
End Class
```

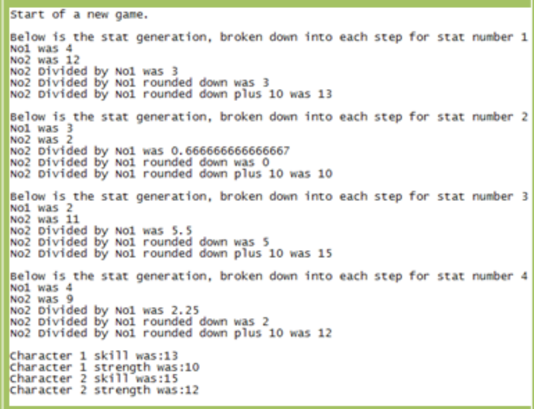
Throughout the code I added the line 'objreder.Writeline()' At the required times to write whatever I put in the brackets to the text file. I outputed after each step of the calculation so I could test every aspect of the code.

I added the above at the end of the programme to close the text file after all the calculation s have been done.

This is the code after I had added the output to the text file at each step of the calculations and commented the code.



```
Start of a new game.

Below is the stat generation, broken down into each step for stat number 1
No1 was 1
No2 was 11
No2 Divided by No1 was 11
No2 Divided by No1 rounded down was 11
No2 Divided by No1 rounded down plus 10 was 21

Below is the stat generation, broken down into each step for stat number 2
No1 was 1
No2 was 12
No2 Divided by No1 was 12
No2 Divided by No1 rounded down was 12
No2 Divided by No1 rounded down plus 10 was 22

Below is the stat generation, broken down into each step for stat number 3
No1 was 1
No2 was 12
No2 Divided by No1 was 12
No2 Divided by No1 rounded down was 12
No2 Divided by No1 rounded down plus 10 was 22

Below is the stat generation, broken down into each step for stat number 4
No1 was 3
No2 was 8
No2 Divided by No1 was 2.66666666666667
No2 Divided by No1 rounded down was 2
No2 Divided by No1 rounded down plus 10 was 12

Character 1 skill was:21
Character 1 strength was:22
Character 2 skill was:22
Character 2 strength was:12
```

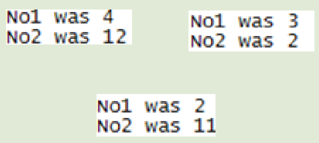
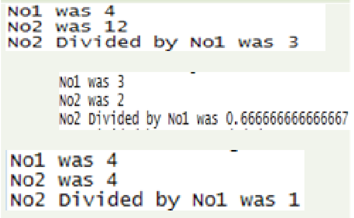
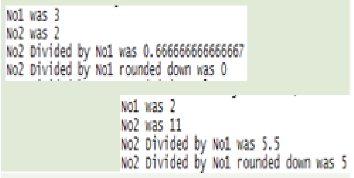
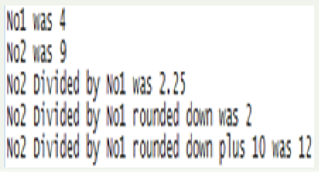
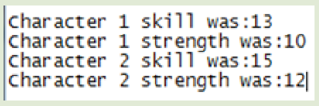


```
Start of a new game.

Below is the stat generation, broken down into each step for stat number 1
No1 was 4
No2 was 12
No2 Divided by No1 was 3
No2 Divided by No1 rounded down was 3
No2 Divided by No1 rounded down plus 10 was 13

Below is the stat generation, broken down into each step for stat number 2
No1 was 3
No2 was 2
No2 Divided by No1 was 0.666666666666667
No2 Divided by No1 rounded down was 0
No2 Divided by No1 rounded down plus 10 was 10

Below is the stat generation, broken down into each step for stat number 3
No1 was 2
No2 was 11
No2 Divided by No1 was 5.5
No2 Divided by No1 rounded down was 5
No2 Divided by No1 rounded down plus 10 was 15

Below is the stat generation, broken down into each step for stat number 4
No1 was 4
No2 was 9
No2 Divided by No1 was 2.25
No2 Divided by No1 rounded down was 2
No2 Divided by No1 rounded down plus 10 was 12

Character 1 skill was:13
Character 1 strength was:10
Character 2 skill was:15
Character 2 strength was:12
```

I ran the program to test the text file output. It outputted the correct information which showed everything was working correctly.

I then move on to using the text file to test every aspect of my code.

Testing

What I'm testing	How will I test it	Expected output	Actual output
That the two numbers generated were random	By running the program multiple times then consulting the text file	A random number between 1 and 4 each time and A random number between 1 and 12	 No1 was 4 No2 was 12 No1 was 3 No2 was 2 No1 was 2 No2 was 11
That the program divides the numbers correctly	By running the program multiple times then consulting the text file Testing numbers close, far away and the same as each other	The correct answer to the division	 No1 was 4 No2 was 12 No2 Divided by No1 was 3 No1 was 3 No2 was 2 No2 Divided by No1 was 0.666666666666667 No1 was 4 No2 was 4 No2 Divided by No1 was 1
That the result from the division is rounded down correctly	By running the program multiple times then consulting the text file	The answer to the division correctly rounded down	 No1 was 3 No2 was 2 No2 Divided by No1 was 0.666666666666667 No2 Divided by No1 rounded down was 0 No1 was 2 No2 was 11 No2 Divided by No1 was 5.5 No2 Divided by No1 rounded down was 5
That the result is added to 10 correctly	By running the program multiple times then consulting the text file	The result added to 10 correctly	 No1 was 4 No2 was 9 No2 Divided by No1 was 2.25 No2 Divided by No1 rounded down was 2 No2 Divided by No1 rounded down plus 10 was 12
The results of the operations are outputted to the text file correctly	By running the program multiple times then consulting the text file	The results of the operations are outputted to the text file correctly	 Character 1 skill was:13 Character 1 strength was:10 Character 2 skill was:15 Character 2 strength was:12

Criteria point	How they are met

At the start of the game a 4 and 12 sided dice are rolled	When the form loads this code is run
	`Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load`
	`No1 = ran1.Next(1, 5)`
	`No2 = ran1.Next(1, 13)` this rolls a 4 and 12 sided dice

The value of the 12 sided dice is divided by the value of the 4 sided dice.	This is done by this code `(No2 / No1)`
The result will be rounded down then added to the initial attribute value of 10.	This is done by this code `Math.Floor(10 + (No2 / No1))`

The calculation is repeated while setting the results to the attributes corresponding variable	This code is run in a loop
	`X = 0`
	`Do Until X = 4`
	`    No1 = ran1.Next(1, 5)`
	`    No2 = ran1.Next(1, 13)`
	`    Attribute(X) = Math.Floor(10 + (No2 / No1))`
	`    X = X + 1`
	`Loop`

The results are outputted to the text file	This is done by the following code
	`objreader.WriteLine("Character 1 skill was:" & Attribute(0))`
	`objreader.WriteLine("Character 1 strength was:" & Attribute(1))`
	`objreader.WriteLine("Character 2 skill was:" & Attribute(2))`
	`objreader.WriteLine("Character 2 strength was:" & Attribute(3))`

The program was very successful as all the success criteria was met and the program does exactly what the specification states although the program could be improved if you could choose the difficulty at the start of the game that would make the range of random numbers generated larger therefore making the attribute values vary more.

```
Public Class Form1
  'This is declartion of the text file and the opening of it
  Dim Filename As String = "Stats.txt"
  Dim objreader As New System.IO.StreamWriter(Filename, True)
  'This is the array to store the attribute values after they have been
determined
  Dim Attribute(0 To 3) As Single
  'These are the variables to store the random numbers once they have been
generated
  Dim No1 As Integer
  Dim No2 As Integer
  'This stores the number the loop uses to output to the array and to know when
to stop
  Dim X As Integer
  'this is the randome number
  Dim ran1 As New Random
  Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.
EventArgs) Handles MyBase.Load
```

```
     X = 0
     objreader.WriteLine("Start of a new game.")
     objreader.WriteLine(" ")
     'This is the loop that generates the attribute values
Do Until X = 4
         'Here the random numbers are generated
         No1 = ran1.Next(1, 5)
         No2 = ran1.Next(1, 13)
         'Below the calculation is done and the result is set to the next
attribute
         Attribute(X) = Math.Floor(10 + (No2 / No1))
         'Here I output each step to the text file
         objreader.WriteLine("Below is the stat generation, broken down into
each step for stat number " & (X + 1))
         objreader.WriteLine("No1 was " & No1)
         objreader.WriteLine("No2 was " & No2)
         objreader.WriteLine("No2 Divided by No1 was " & (No2 / No1))
         objreader.WriteLine("No2 Divided by No1 rounded down was " & Math.
Floor(No2 / No1))
         objreader.WriteLine("No2 Divided by No1 rounded down plus 10 was " & 10
+ Math.Floor(No2 / No1))
         objreader.WriteLine(" ")
         X = X + 1
     Loop
     'After the loop is done all the attribute values are displayed to the user
in 4 buttons in the form
     Button1.Text = Attribute(0)
     Button2.Text = Attribute(1)
     Button3.Text = Attribute(2)
     Button4.Text = Attribute(3)
     'The last thing done is outputting the actual attribute values to the text
file
     objreader.WriteLine("Character 1 skill was:" & Attribute(0))
     objreader.WriteLine("Character 1 strength was:" & Attribute(1))
     objreader.WriteLine("Character 2 skill was:" & Attribute(2))
     objreader.WriteLine("Character 2 strength was:" & Attribute(3))
     objreader.WriteLine(" ")
     objreader.WriteLine(" ")
     objreader.WriteLine(" ")
     objreader.WriteLine(" ")
     'This closes the text file
     objreader.Close()
  End Sub
End Class
```

Task 3

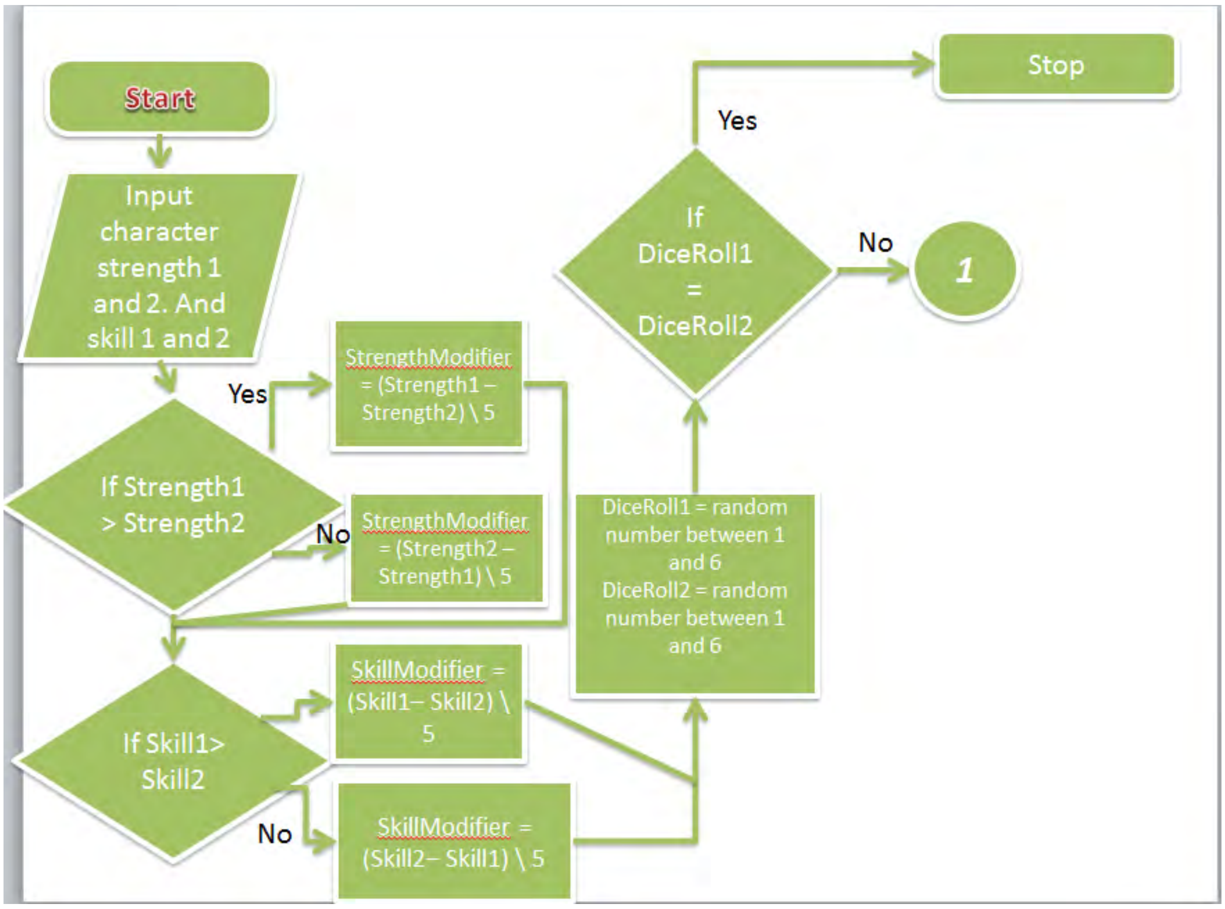
What I'm going to do

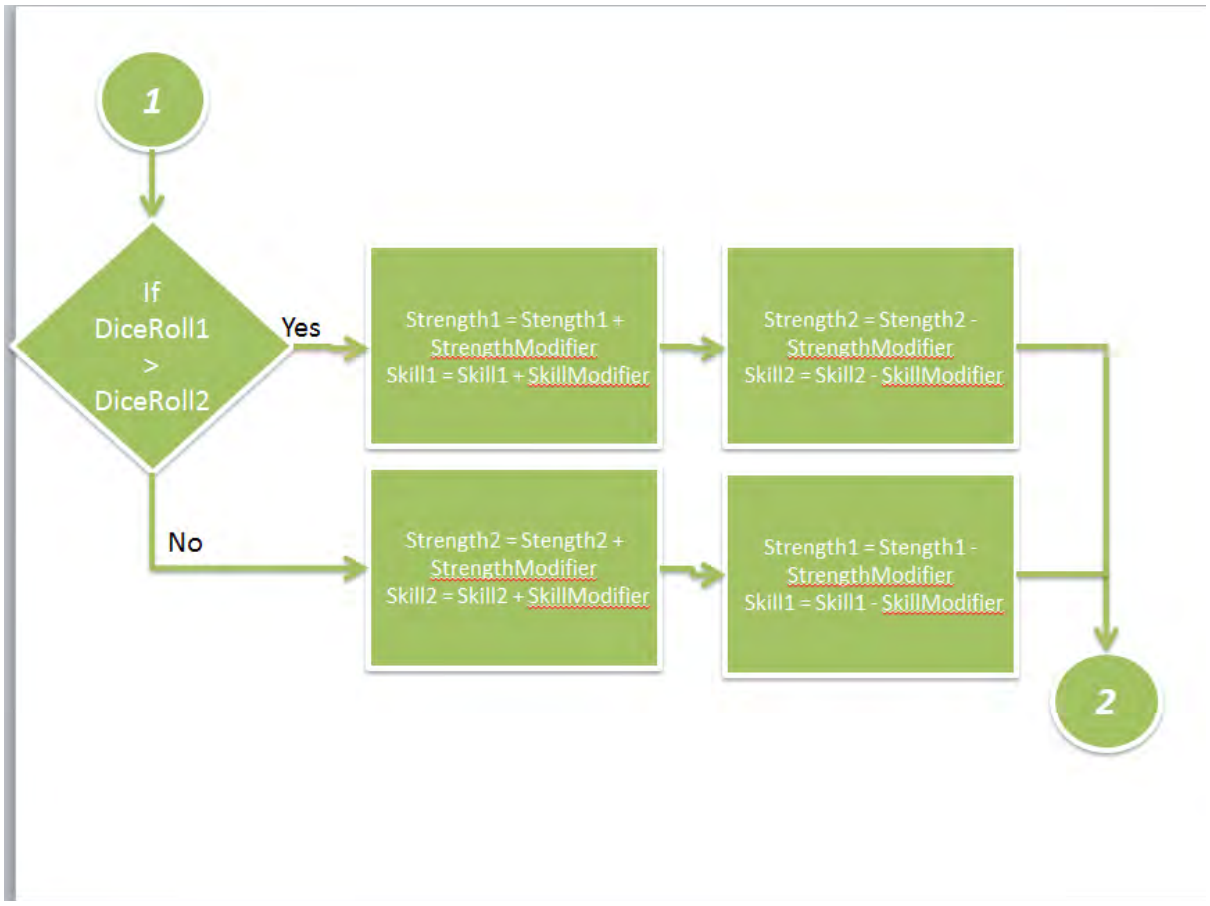
I'm going to make a program that determines the outcome of an encounter it will allow the user to input the strength and skill for the two characters. The program will then take those values put then though an algorithm to make a strength and skill modifier. A dice will then be rolled for each player the one with the highest dice roll will get the modifiers added to their respected attributes, the one with the lowest will get the modifiers subtracted from their respected attributes. If any characters skill becomes zero or negative it will be stored as zero. If any characters strength becomes zero or negative that character dies this will be the end of the game.

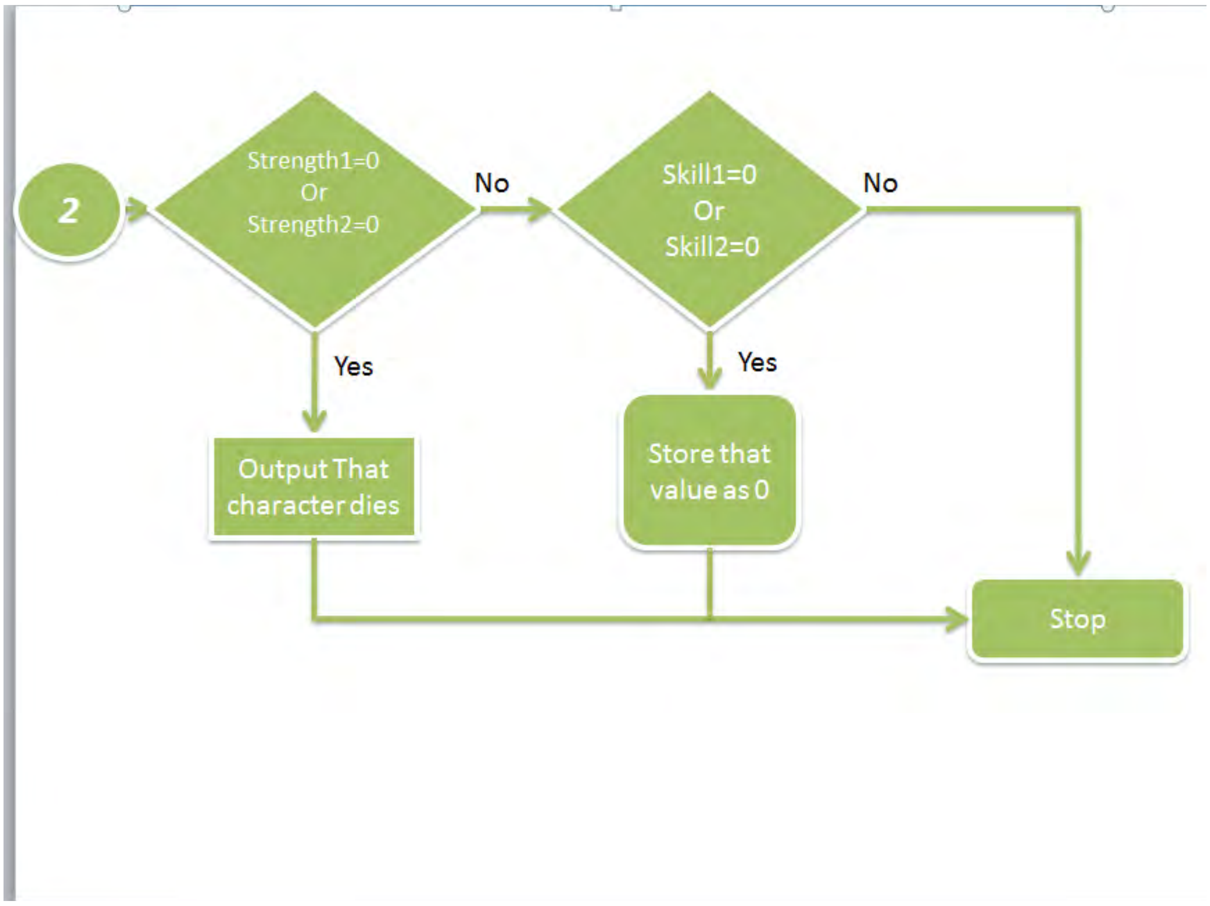
Success criteria

- The program will allow the user to input the strength and skill for both character
- The difference between the strength attributes is calculated, divided by 5 and rounded down then stored as the strength modifier.
- The difference between the skill attributes is calculated, divided by 5 and rounded down then stored as the skill modifier.
- Both players can roll a 6 sided dice once
- The results of the dice rolls will be will be compared, if the results are the same nothing will change if the results are different the player with the highest dice roll will get the modifiers added to their respected attributes, the one with the lowest will get the modifiers subtracted from their respected attributes.
- If any characters skill becomes zero or negative it will be stored as zero.
- If any characters strength becomes zero or negative that character dies this will be the end of the game.

Flow chart


Start
Input character strength 1 and 2. And skill 1 and 2
If Strength1 > Strength2
Yes
StrengthModifier = (Strength1 – Strength2) \ 5
No
StrengthModifier = (Strength2 – Strength1) \ 5
If Skill1> Skill2
SkillModifier = (Skill1– Skill2) \ 5
No
SkillModifier = (Skill2– Skill1) \ 5
DiceRoll1 = random number between 1 and 6
DiceRoll2 = random number between 1 and 6
If DiceRoll1 = DiceRoll2
Yes
Stop
No
1


1
If DiceRoll1 > DiceRoll2
Yes
No
Strength1 = Stength1 + StrengthModifier
Skill1 = Skill1 + SkillModifier
Strength2 = Stength2 - StrengthModifier
Skill2 = Skill2 - SkillModifier
Strength2 = Stength2 + StrengthModifier
Skill2 = Skill2 + SkillModifier
Strength1 = Stength1 - StrengthModifier
Skill1 = Skill1 - SkillModifier
2


2
Strength1=0 Or Strength2=0
No
Yes
Skill1=0 Or Skill2=0
No
Yes
Output That character dies
Store that value as 0
Stop

Pseudo Code

```
Input Stength1, Strength2, Skill1, Skill2
If Strength1 > Strength2 then
    StrengthMod = (Strength1 - Strength2 \ 5)
Else
    StrengthMod = (Strength2 – Strength1 \ 5)
End if
If Skill1 > Skill2 then
    SkillMod = (Skill1 - Skill2 \ 5)
Else
    SkillMod = (Skill2 – Skill1 \ 5)
End if
Dice1 = random number between 1 and 6
Dice2 = random number between 1 and 6
If Dice1 > Dice2 then
    Strength1 = Strength1 + StrengthMod
    Strength2 = Strength2 – StrengthMod
    Skill1 = Skill1 + SkillMod
    Skill2 = Skill2 - SkillMod
Else
    Strength1 = Strength1 - StrengthMod
    Strength2 = Strength2 + StrengthMod
    Skill1 = Skill1 - SkillMod
    Skill2 = Skill2 + SkillMod
End if
If Skill1 < 0 then
    Skill1 = 0
End if
If Skill2 < 0 then
    Skill2 = 0
End if
If Strength1 <= 0 then
    Output (Character1 Has Died)
End if
If Strength2 <= 0 then
    Output (Character2 Has Died)
End if
```

Variables

Variable name	What is does
ran Random	This will generate the random numbers
Dice1 Integer Dice2 Integer	These will store random numbers that are generated
Strength1 Integer Strength2 Integer Skill1 Integer Skill2 Integer	These will store the entered attributes ready for the calculation and store the new attributes after the calculation
StrengthMod Integer SkillMod Integer	These will store the calculated modifiers

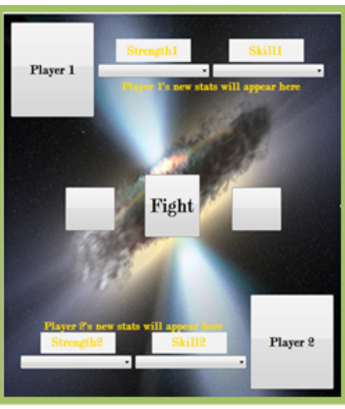
Validation

- No validation is needed as the user can only enter things already in the combo box's
- All the numbers there are valid and require no validation.

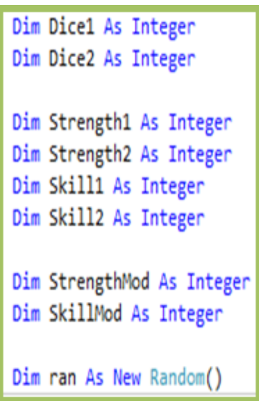
Test Plan

What I'm testing
If the user can enter the strength and skill attributes
The difference between the strength attributes is calculated the divided by 5 and then rounded down then stored as the strength modifier
The difference between the Skill attributes is calculated the divided by 5 and then rounded down then stored as the skill modifier
Two dice are rolled for both character and compared. Whoever has the highest roll wins the encounter
The player that wins the dice roll gets the strength modifier added to their strength attribute and The player that wins the dice roll gets the skill modifier added to their skill attribute
The player that loses the dice roll gets the strength modifier subtracted from their strength attribute and The player that loses the dice roll gets the skill modifier subtracted from to their skill attribute
If any skill modifier becomes below zero it is changed to zero
if any strength modifier becomes 0 or below the character that it belongs to Dies
If the dice roll is the same it outputs as a draw

Development

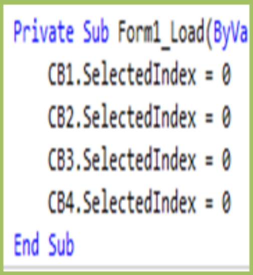

Player 1
Strength1
Skill1
Player 1's new stats will appear here
Fight
Player 2's new stats will appear here
Strength2
Skill2
Player 2

First I made the form layout the combo boxes to input the player attributes and the fight button to run the calculation and the labels to output the result of the encounter



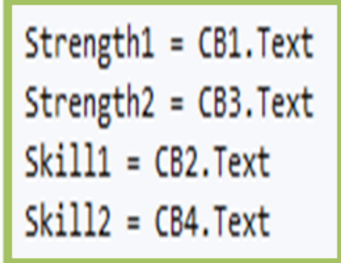
```
Dim Dice1 As Integer
Dim Dice2 As Integer

Dim Strength1 As Integer
Dim Strength2 As Integer
Dim Skill1 As Integer
Dim Skill2 As Integer

Dim StrengthMod As Integer
Dim SkillMod As Integer

Dim ran As New Random()
```

Then I declared my variables; the two dice rolls, the character attributes, the modifiers and the random number



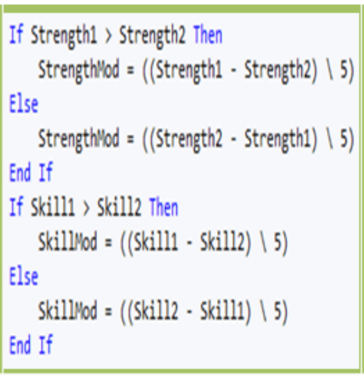
```
Private Sub Form1_Load(ByVa
    CB1.SelectedIndex = 0
    CB2.SelectedIndex = 0
    CB3.SelectedIndex = 0
    CB4.SelectedIndex = 0
End Sub
```

Then I made the combo boxes equal 1 when the form loads



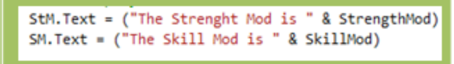
```
Strength1 = CB1.Text
Strength2 = CB3.Text
Skill1 = CB2.Text
Skill2 = CB4.Text
```

This code would make the attribute variables equal to the users input in the combo box's



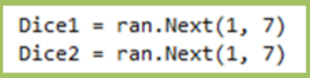
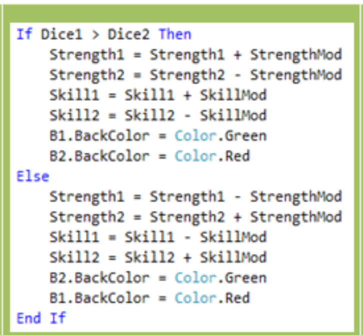
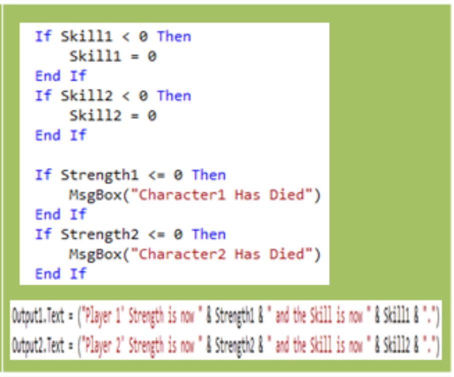
```
If Strength1 > Strength2 Then
    StrengthMod = ((Strength1 - Strength2) \ 5)
Else
    StrengthMod = ((Strength2 - Strength1) \ 5)
End If
If Skill1 > Skill2 Then
    SkillMod = ((Skill1 - Skill2) \ 5)
Else
    SkillMod = ((Skill2 - Skill1) \ 5)
End If
```

Then I made the modifier calculation to make both the modifiers. At first I used a forward slash to do the calculation then I realized after testing it retuned decimals which needed to be rounded down so I changed it to a back slash which would divide then round down



```
StM.Text = ("The Strenght Mod is " & StrengthMod)
SM.Text = ("The Skill Mod is " & SkillMod)
```

Then I made the code that would display both modifiers in the form I also used it to test if the modifier calculation was working.
After running it a few times everything seemed to be working properly

		
I then made the dice roll code to generate 2 numbers between or including 1 and 6	Then I made the dice comparison that then would apply the modifiers depending on who won the dice roll it also displayed who won the dice roll by changing the colour of the player buttons	Then I made the code that would check if any of the attributes where 0 or less then store it as 0 if it was the Skill and outputting the player dies if it was Strength It would

```
Dice1 = ran.Next(1, 7)
Dice2 = ran.Next(1, 7)
```

```
If Dice1 > Dice2 Then
    Strength1 = Strength1 + StrengthMod
    Strength2 = Strength2 - StrengthMod
    Skill1 = Skill1 + SkillMod
    Skill2 = Skill2 - SkillMod
    B1.BackColor = Color.Green
    B2.BackColor = Color.Red
Else
    Strength1 = Strength1 - StrengthMod
    Strength2 = Strength2 + StrengthMod
    Skill1 = Skill1 - SkillMod
    Skill2 = Skill2 + SkillMod
    B2.BackColor = Color.Green
    B1.BackColor = Color.Red
End If
```

```
If Skill1 < 0 Then
    Skill1 = 0
End If
If Skill2 < 0 Then
    Skill2 = 0
End If

If Strength1 <= 0 Then
    MsgBox("Character1 Has Died")
End If
If Strength2 <= 0 Then
    MsgBox("Character2 Has Died")
End If
```

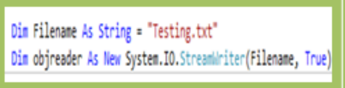
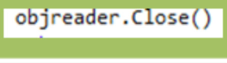
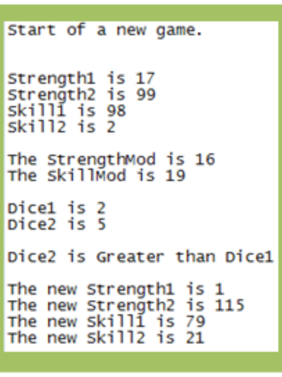
```
Output1.Text = ("Player 1' Strength is now " & Strength1 & " and the Skill is now " & Skill1 & ".")
Output2.Text = ("Player 2' Strength is now " & Strength2 & " and the Skill is now " & Skill2 & ".")
```

```
Public Class Form1
  'Here are the variables for the; the dice rolls
  'The player attributes
  'the Strength and Skill modifier
  'and the random number
  Dim Dice1 As Integer
  Dim Dice2 As Integer

  Dim Strength1 As Integer
  Dim Strength2 As Integer
  Dim Skill1 As Integer
  Dim Skill2 As Integer

  Dim StrengthMod As Integer
  Dim SkillMod As Integer

  Dim ran As New Random()

  Dim Filename As String = "Testing.txt"


  Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.
EventArgs) Handles MyBase.Load
     Dim objreader As New System.IO.StreamWriter(Filename, True)

     objreader.WriteLine("Start of a new game.")
     objreader.WriteLine(" ")
     objreader.WriteLine(" ")
     'This sets the combo boxes value to 1
     CB1.SelectedIndex = 0
     CB2.SelectedIndex = 0
     CB3.SelectedIndex = 0
     CB4.SelectedIndex = 0
```

```
        objreader.Close()
    End Sub

    'This runs the algorithm to determine the outcome of the encounter
    Private Sub Run_Click(ByVal sender As System.Object, ByVal e As System.
EventArgs) Handles Run.Click
        Dim objreader As New System.IO.StreamWriter(Filename, True)

        'This sets the attribute variables to the players input
        Strength1 = CB1.Text
        Strength2 = CB3.Text
        Skill1 = CB4.Text
        Skill2 = CB2.Text



        objreader.WriteLine("Strength1 is " & Strength1)
        objreader.WriteLine("Strength2 is " & Strength2)
        objreader.WriteLine("Skill1 is " & Skill1)
        objreader.WriteLine("Skill2 is " & Skill2)
        objreader.WriteLine(" ")

        'This calculates the modifiers for the encounter
        If Strength1 > Strength2 Then
            StrengthMod = ((Strength1 - Strength2) \ 5)
        Else
            StrengthMod = ((Strength2 - Strength1) \ 5)
    End If
    objreader.WriteLine("The StrengthMod is " & StrengthMod)

    If Skill1 > Skill2 Then
        SkillMod = ((Skill1 - Skill2) \ 5)
    Else
        SkillMod = ((Skill2 - Skill1) \ 5)
    End If
    objreader.WriteLine("The SkillMod is " & SkillMod)
    objreader.WriteLine(" ")

    'This displays the modifiers
    StM.Text = ("The Strength Mod is " & StrengthMod)
    SM.Text = ("The Skill Mod is " & SkillMod)

    'Here the dice are rolled
    Dice1 = ran.Next(1, 7)
    Dice2 = ran.Next(1, 7)
    objreader.WriteLine("Dice1 is " & Dice1)
    objreader.WriteLine("Dice2 is " & Dice2)
    objreader.WriteLine(" ")
```

```
'This is the dice comparison that calculates the new attributes
If Dice1 > Dice2 Then
   objreader.WriteLine("Dice1 is Greater than Dice2")
   objreader.WriteLine(" ")
   Strength1 = Strength1 + StrengthMod
   Strength2 = Strength2 - StrengthMod
   Skill1 = Skill1 + SkillMod
   Skill2 = Skill2 - SkillMod
   B1.BackColor = Color.Green
   B2.BackColor = Color.Red
ElseIf Dice2 > Dice1 Then
   objreader.WriteLine("Dice2 is Greater than Dice1")
   Strength1 = Strength1 - StrengthMod
   Strength2 = Strength2 + StrengthMod
   Skill1 = Skill1 - SkillMod
   Skill2 = Skill2 + SkillMod
   B2.BackColor = Color.Green
   B1.BackColor = Color.Red
Else
   MsgBox("Draw")
End If


'This calculates the out come and if either character dies
If Skill1 < 0 Then
   Skill1 = 0
   objreader.WriteLine("Skill1 was equal or lower than zero")
End If
If Skill2 < 0 Then
   Skill2 = 0
   objreader.WriteLine("Skill2 was equal or lower than zero")
End If

If Strength1 <= 0 Then
   Strength1 = 0
   objreader.WriteLine("Strength1 was equal or lower than zero so Charecter1 
has died")
   MsgBox("Character1 Has Died")
End If
If Strength2 <= 0 Then
   Strength2 = 0
   objreader.WriteLine("Strength2 was equal or lower than zero so Charecter2 
has died")
   MsgBox("Character2 Has Died")
End If
'This outputs the new attribute values
Output1.Text = ("Player 1' Strength is now " & Strength1 & " and the Skill is 
now " & Skill1 & ".")
Output2.Text = ("Player 2' Strength is now " & Strength2 & " and the Skill is 
now " & Skill2 & ".")
objreader.WriteLine(" ")
objreader.WriteLine("The new Strength1 is " & Strength1)
```

```
    objreader.WriteLine("The new Strength2 is " & Strength2)
    objreader.WriteLine("The new Skill1 is " & Skill1)
    objreader.WriteLine("The new Skill2 is " & Skill2)
    objreader.WriteLine(" ")
    objreader.WriteLine(" ")
    objreader.Close()
    End Sub

End Class
```

 `Dim Filename As String = "Testing.txt"`
`Dim objreader As New System.IO.StreamWriter(Filename, True)`
 `objreader.Close()`
	 `Start of a new game.`
	
	`Strength1 is 17`
	`Strength2 is 99`
	`Skill1 is 98`
	`Skill2 is 2`
	
	`The StrengthMod is 16`
	`The SkillMod is 19`
	
	`Dice1 is 2`
	`Dice2 is 5`
	
	`Dice2 is Greater than Dice1`
	
	`The new Strength1 is 1`
	`The new Strength2 is 115`
	`The new Skill1 is 79`
	`The new Skill2 is 21`	

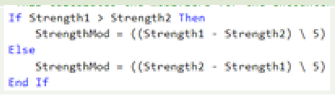
To thoroughly test my code I made it output to a text file after each step so I could test that each step worked properly. I did this by adding the code (objreader.WriteLine(" ")) after each step	This is what is out putted to the text file after a test play through	

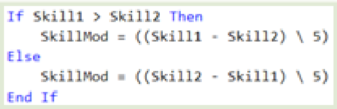
Testing

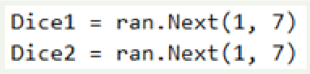
What I'm testing	How I'm testing it	Input	Video that proves on the next slide
If the user can enter the strength and skill attributes	By trying to enter the attributes	Attribute values	The Combo boxes that are used in Video 1
The difference between the strength attributes is calculated the divided by 5 and then rounded down then stored as the strength modifier	Running the program and consulting the text file and video	Strength 1 = 3 Strength 2 = 86	The expected outputted in the two buttons under strength modifier Video 1
The difference between the Skill attributes is calculated the divided by 5 and then rounded down then stored as the skill modifier	Running the program and consulting the text file and video	Skill 1 = 91 Skill 2 = 7	The expected outputted in the two buttons under Skill modifier Video 1
Two dice are rolled for both character and compared. Whoever has the highest roll wins the encounter	Running the program and consulting the text file and video	Fight button (Click)	Player that wins tile turn green Video 1
The player that wins the dice roll gets the strength modifier added to their strength attribute and The player that wins the dice roll gets the skill modifier added to their skill attribute	Running the program and consulting the text file and video	Strength 1 = 3 Strength 2 = 86 Skill 1 = 91 Skill 2 = 7 Fight button (Click)	The expected outputs outputted to the new attributes label Video 1
The player that loses the dice roll gets the strength modifier subtracted from their strength attribute and The player that loses the dice roll gets the skill modifier subtracted from to their skill attribute	Running the program and consulting the text file and video	Strength 1 = 3 Strength 2 = 86 Skill 1 = 91 Skill 2 = 7 Fight button (Click)	The expected outputs outputted to the new attributes label Video 1
If any skill modifier becomes below zero it is changed to zero	Running the program and consulting the text file and video	Strength 1 = 3 Strength 2 = 86 Skill 1 = 91 Skill 2 = 7 Fight button (Click)	The expected outputs outputted to the new attributes label Video 1
if any strength modifier becomes 0 or below the character that it belongs to Dies	Running the program and consulting the text file and video	Strength 1 = 3 Strength 2 = 86 Skill 1 = 91 Skill 2 = 7 Fight button (Click)	The expected outputs outputted to the new attributes label Video 1
If the dice roll is the same it outputs as a draw	Running the program and consulting the text file and video	Clicking on the fight button lots until a draw result is outputted	Video 2

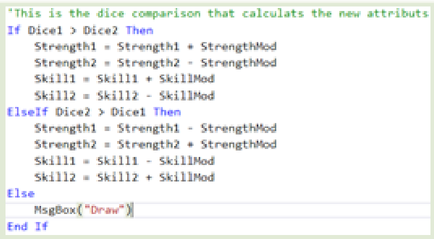
Criteria check

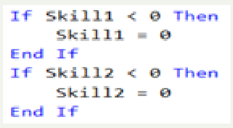
The program will allow the user to input the strength and skill for both character	With 4 drop down combo boxes labelled with the attribute they correspond to (Video 1)

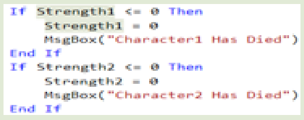
The difference between the strength attributes is calculated, divided by 5 and rounded down then stored as the strength modifier	The code here does this  `If Strength1 > Strength2 Then`
	`    StrengthMod = ((Strength1 - Strength2) \ 5)`
	`Else`
	`    StrengthMod = ((Strength2 - Strength1) \ 5)`
	`End If`

The difference between the skill attributes is calculated, divided by 5 and rounded down then stored as the skill modifier	The code here does this  `If Skill1 > Skill2 Then`
	`    SkillMod = ((Skill1 - Skill2) \ 5)`
	`Else`
	`    SkillMod = ((Skill2 - Skill1) \ 5)`
	`End If`

Both players roll a 6 sided dice once	The code here does this  `Dice1 = ran.Next(1, 7)` `Dice2 = ran.Next(1, 7)`

The results of the dice rolls will be will be compared, if the results are the same nothing will change if the results are different the player with the highest dice roll will get the modifiers added to their respected attributes, the one with the lowest will get the modifiers subtracted from their respected attributes.	The code here does this  `'This is the dice comparison that calculats the new attributs`
	`If Dice1 > Dice2 Then`
	`    Strength1 = Strength1 + StrengthMod`
	`    Strength2 = Strength2 - StrengthMod`
	`    Skill1 = Skill1 + SkillMod`
	`    Skill2 = Skill2 - SkillMod`
	`ElseIf Dice2 > Dice1 Then`
	`    Strength1 = Strength1 - StrengthMod`
	`    Strength2 = Strength2 + StrengthMod`
	`    Skill1 = Skill1 - SkillMod`
	`    Skill2 = Skill2 + SkillMod`
	`Else`
	`    MsgBox("Draw")`
	`End If`

If any characters skill becomes zero or negative it will be stored as zero	The code here does this  `If Skill1 < 0 Then`
	`    Skill1 = 0`
	`End If`
	`If Skill2 < 0 Then`
	`    Skill2 = 0`
	`End If`

If any characters strength becomes zero or negative that character dies this will be the end of the game.	The code here does this  `If Strength1 <= 0 Then`
	`    Strength1 = 0`
	`    MsgBox("Character1 Has Died")`
	`End If`
	`If Strength2 <= 0 Then`
	`    Strength2 = 0`
	`    MsgBox("Character2 Has Died")`
	`End If`