

Assignment 1 Donna Wright

by Donna Wright

FILE	11968_DONNA_WRIGHT_ASSIGNMENT_1_DONNA_WRIGHT_157877_70 3406467.DOCX (654.27K)		
TIME SUBMITTED	25-FEB-2019 12:03PM (UTC+0000)	WORD COUNT	3063
SUBMISSION ID	101166118	CHARACTER COUNT	18570

Programs in C and documentation

Parts A and B
By Donna Wright

Contents

Part A (Understanding).....	4
A1. Variables	4
A2. Loops	4
A3. Arrays	4
Part B (Implementation)	5
Exercise 1: Different types of variables.....	5
1.1 Problem breakdown.....	5
1.2 Annotated code.....	5
1.3 Testing	5
1.3.1 Normal values	5
1.3.2 Impossible values.....	6
Exercise 2: Area	7
2.1 Problem breakdown.....	7
2.1.1 Determine the area of the circle	7
2.1.2 Determine the area of the triangle.....	7
2.1.3 Determine the area of the shaded part.....	7
2.2 Annotated code.....	8
2.3 Testing	9
2.3.1 Correct values	9
2.3.2 Incorrect values.....	9
2.3.3 Invalid characters	9
Exercise 3: Products, price plus VAT.....	10
3.1 Problem breakdown.....	10
3.1.1 Determine the products values in an array and calculate the total	10
3.1.2 Calculate the total price plus VAT for the product list.....	10
3.2 Annotated code.....	11
3.3 Testing	12
3.3.1 Normal values	12
3.3.2 Boundary values	12
3.3.3 Invalid values	12
Exercise 4: The fruit machine	13
4.1 Problem breakdown.....	13
4.1.1 Generate random numbers	13
4.1.2 Generate loop so user can continue playing	13

4.1.3 Assign points for spin	13
4.1.4 Accumulate points for users spins.....	13
4.2 Annotated code.....	14
4.2.1 Code to generate random numbers as per 4.1.1:.....	14
}.....	14
4.2.2 Code to allow user to spin as many times as they want as per 4.1.2:	15
}.....	15
4.2.3 Code to add scores to spins:.....	16
4.2.4 Accumulate scores	17
4.3 Testing.....	18
4.3.1 Random numbers.....	18
4.3.2 Loop game	19
4.3.3 Add scores together	20
4.3.4 Accumulate scores	20
References	21

Part A (Understanding)

A1. Variables

Within C program data values are stored within containers within the computer's memory. These containers are called variables.

The programmer can select any name for their variable and there are few restriction on what a variable can be called, however it is best practice to use something meaningful in order to ensure the program code is understandable.

Variables are important when writing code as they are a means for the programmer to store data in a meaningful way. The use of variables enables the programmer to label and store data in the computer's memory reducing the need to write data in its entirety in their code.

A2. Loops

A piece of code within a program which repeats is known as a loop. There are three loop types, while, do while and for, with for loops being the most commonly used.

The while loop is used when there is no known total of iterations.

In contrast, the for loop (also referred to as the counting loop) will repeat until a set number of iterations have been met. The programmer can set an integer outlining an initial value and an upper limit and the code will repeat until this upper limit has been reached.

An example of when a for loop may be used is when checking a fixed amount of entries, i.e. grades of students. The program will run from 1 until the number of students and then stop, providing a list of grades for the students.

A3. Arrays

In A1 variables were discussed as containers for data. However, these containers can only hold one item of data at a time. Arrays are capable of holding more than one item of related data.

Structured data types are an example of an array, where a number of the same type of data can be stored in one place. For example a list of numbers or words.

To put this in context, individual items from a shopping list can be stored separately in variables, whereas the entire shopping list can be stored in an array.

However, arrays are limited in that they can only hold the same types of data, for example a list of words or a list of numbers. An array cannot be made up of a mix of numbers and words.

Part B (Implementation)

Exercise 1: Different types of variables.

1.1 Problem breakdown

As explained in A1, variables are containers for data. Each data type has a different variable, for example numbers are an int variable, letters and words are char variables and percentages are float variables. Therefore variables can hold many different types of data.

1.2 Annotated code

```
#include <stdio.h>

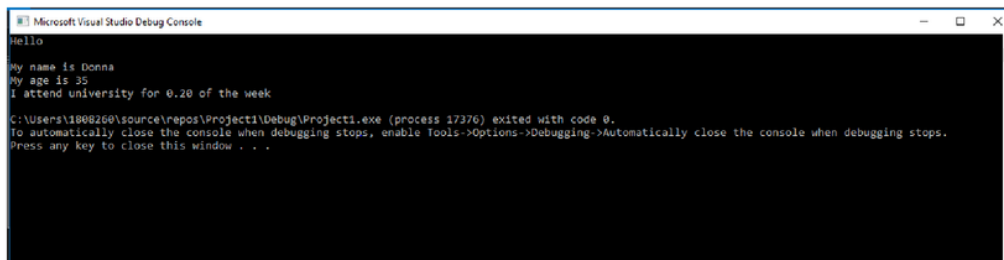
void main()
{
    char myname[6] = { 'D', 'o', 'n', 'n', 'a' };
    int myage = 35;
    float mydecimal = 0.20;

    printf("Hello\n\n");
    printf("My name is %s\n", myname); //myname is a string so %s is needed to
return the value
    printf("My age is %d\n", myage); //myname is an int so %d is needed to return
the value
    printf("I attend university for %.2f of the week\n", mydecimal);
    //mydecimal is a float so %f is needed to return the value. .2 ensures only 2
decimal spaces returned.
}
```

1.3 Testing

1.3.1 Normal values

Valid data entered as the variables

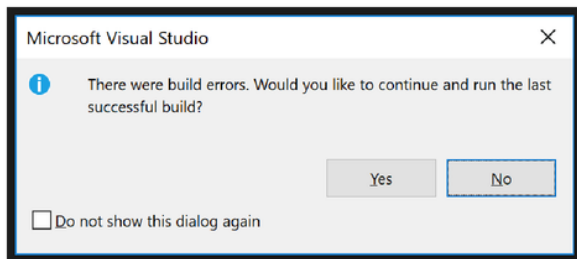
A screenshot of the Microsoft Visual Studio Debug Console window. The window title is "Microsoft Visual Studio Debug Console". The output text is as follows:

```
Hello
My name is Donna
My age is 35
I attend university for 0.20 of the week
C:\Users\1808260\source\repos\Project1\Debug\Project1.exe (process 17376) 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 . . .
```

Result: Program ran as expected.

1.3.2 Impossible values

Invalid values entered against variables, letters in place of numbers and vice versa

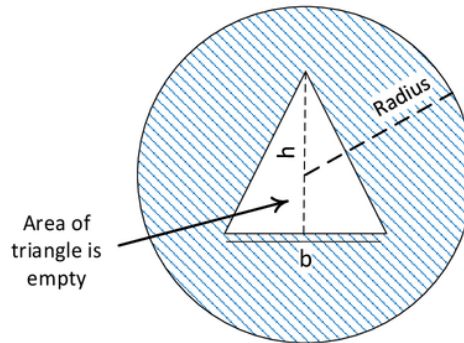


Result: Error message.

Exercise 2: Area

2.1 Problem breakdown

For this exercise we are asked to calculate area of the shaded part of the following diagram, dependant on dimensions provided by a user:



As π plays a large part in calculating area for both circles and triangles, its value should be defined as a constant.

2.1.1 Determine the area of the circle

The area of a circle is calculated using the formula $A = \pi r^2$. Therefore in order to determine the area we need the user to specify the radius.

1. User enters radius
2. Calculation completed ($A = \pi r^2$)
3. Number stored in area1 (float)

2.1.2 Determine the area of the triangle

The area of a triangle is calculated using the formula $A = \frac{(b \times h)}{2}$. Therefore in order to determine the area we need the user to specify the base and height dimensions

1. User enters base
2. User enters height
3. Calculation completed ($A = \frac{(b \times h)}{2}$)
4. Number stored in area2 (float)

2.1.3 Determine the area of the shaded part

Then determine the area of the shaded part the area calculated in 2.1.2 needs to be subtracted from the area calculated in 2.1.1.

If the value for area 2 is greater than area 1, print "The dimensions for your triangle are greater than your circle". Otherwise:

1. Area1 – area2
2. Number stored in area3
3. Print area3 on screen

2.2 Annotated code

```
#include <stdio.h>
#define PI 3.142 //defined the value of PI as a constant

void main()
{
    float radius, area1, height, base, area2, area3; //floats chosen as area is
    likely to contain decimals

    //First calculate the area of the circle
    printf("Let's calculate the remaining area of a triangle within a circle\n\n");
    printf("First we need to determine the area of our circle.\n");
    printf("Please enter the radius of your circle in cm:\n");
    scanf("%f", &radius);

    area1 = (PI * radius * radius);

    printf("the area of your circle is %.2f\n", area1);

    //second calculate the area of the triangle
    printf("Second we need to determine the area of our triangle (this should be
    smaller than your circle!).\n");
    printf("Please enter the height of your triangle in cm:\n");
    scanf("%f", &height);
    printf("Please enter the base of your triangle in cm:\n");
    scanf("%f", &base);
    area2 = (height * base) / 2;

    printf("the area of your triangle is %.2f\n", area2);

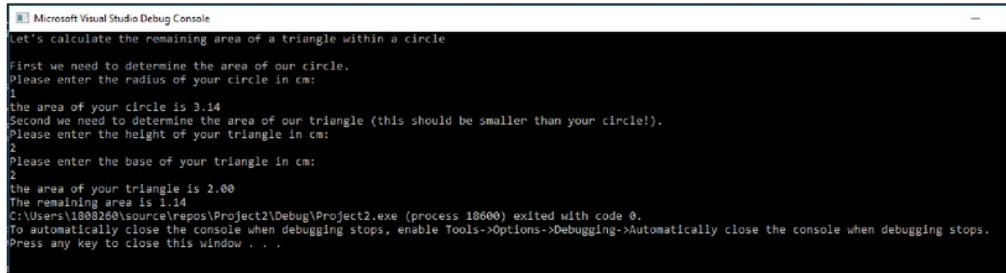
    //If the triangle is bigger than the circle
    if (area2 > area1)
    {
        printf("The dimensions for your triangle are greater than your
    circle.");
    }

    //if the triangle is smaller than the circle
    else
    {
        area3 = (area1 - area2);
        printf("The remaining area is %.2f", area3);
    }
}
```

2.3 Testing

2.3.1 Correct values

User enters valid and correct values for shapes

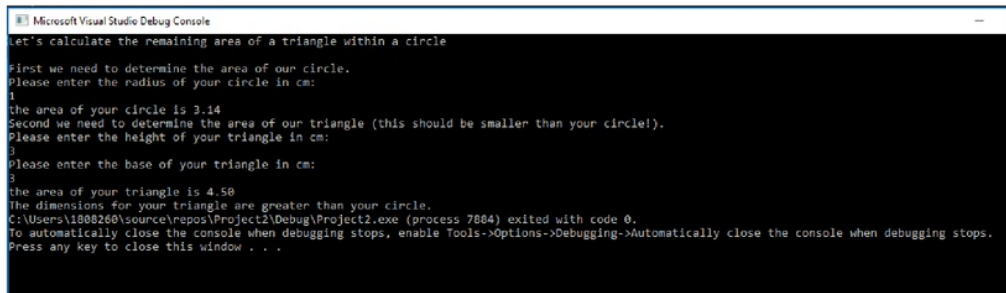


```
Microsoft Visual Studio Debug Console
Let's calculate the remaining area of a triangle within a circle
First we need to determine the area of our circle.
Please enter the radius of your circle in cm:
1
the area of your circle is 3.14
Second we need to determine the area of our triangle (this should be smaller than your circle!).
Please enter the height of your triangle in cm:
2
Please enter the base of your triangle in cm:
2
the area of your triangle is 2.00
The remaining area is 1.14
C:\Users\1808260\source\repos\Project2\Debug\Project2.exe (process 18600) 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 . . .
```

Result: Program ran as expected.

2.3.2 Incorrect values

User enters values where triangle is bigger than square

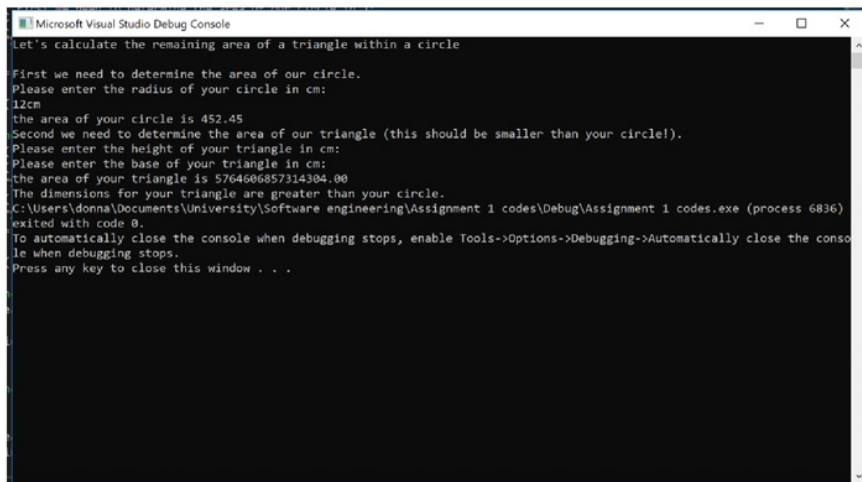


```
Microsoft Visual Studio Debug Console
Let's calculate the remaining area of a triangle within a circle
First we need to determine the area of our circle.
Please enter the radius of your circle in cm:
1
the area of your circle is 3.14
Second we need to determine the area of our triangle (this should be smaller than your circle!).
Please enter the height of your triangle in cm:
3
Please enter the base of your triangle in cm:
3
the area of your triangle is 4.50
The dimensions for your triangle are greater than your circle.
C:\Users\1808260\source\repos\Project2\Debug\Project2.exe (process 7884) 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 . . .
```

Result: Program ran as expected.

2.3.3 Invalid characters

User enters invalid characters, letters included



```
Microsoft Visual Studio Debug Console
Let's calculate the remaining area of a triangle within a circle
First we need to determine the area of our circle.
Please enter the radius of your circle in cm:
12cm
the area of your circle is 452.45
Second we need to determine the area of our triangle (this should be smaller than your circle!).
Please enter the height of your triangle in cm:
Please enter the base of your triangle in cm:
the area of your triangle is 5764606857314304.00
The dimensions for your triangle are greater than your circle.
C:\Users\donna\Documents\University\Software engineering\Assignment 1 codes\Debug\Assignment 1 codes.exe (process 6836)
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 . . .
```

Result: Program failed.

Exercise 3: Products, price plus VAT

3.1 Problem breakdown

This exercise requires the program to allow a user to determine the price of 5 specified products, calculate the total price for all the items for them and then the total plus VAT.

3.1.1 Determine the products values in an array and calculate the total

The exercise requires that the program should allow the user to enter the price of 5 products and for the total to be calculated.

1. Create an array of 5 products
2. User enters prices for each product
3. Values added stored in product variables as floats
4. Add prices of all 5 products together
5. Print totalprice on screen (float)

3.1.2 Calculate the total price plus VAT for the product list

As we now have the total price for the products we can now calculate the VAT and provide a total plus VAT.

1. Create variables VAT and sumtotal
2. Multiply totalprice by .20 to calculate 20%
3. Store as VAT
4. Add totalprice and VAT
5. Store in sumtotal
6. Print sumtotal on screen

3.2 Annotated code

```
#include <stdio.h>

void main()
{
    float products[5]; //declare the array
    float totalprice, VAT, sumtotal; //declare variables

    //First the user needs to populate the array totals
    printf("What's the cost?\n");
    printf("Please determine a price (i.e 1.50) for the following 5 products and I  
will tell you their total cost before and after VAT.\n");
    printf("Shoes:\n\x9c"); //The code \x9c is used to print a f
    scanf("%f", &products[0]);
    printf("Dress:\n\x9c");
    scanf("%f", &products[1]);
    printf("Coat:\n\x9c");
    scanf("%f", &products[2]);
    printf("Bag:\n\x9c");
    scanf("%f", &products[3]);
    printf("Purse:\n\x9c");
    scanf("%f", &products[4]);

    //The total for the prices entered can now be calculated

    totalprice = (products[0] + products[1] + products[2] + products[3] +  
products[4]);

    printf("\nThe total price for the 5 products is \x9c%.2f\n", totalprice);

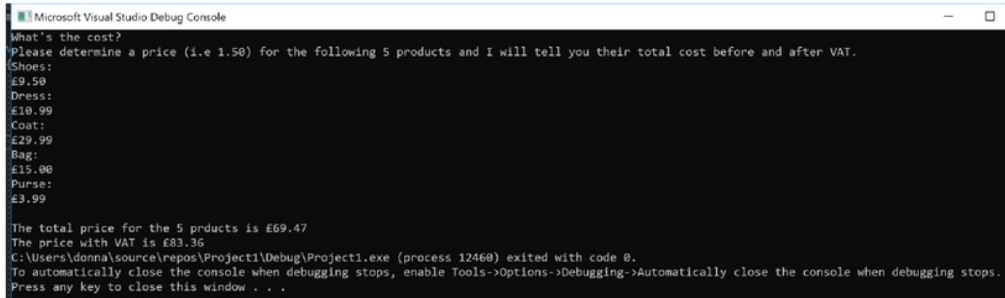
    VAT = (totalprice * 0.20); //this will calculate 20% of the totalprice
    sumtotal = (totalprice + VAT);

    printf("The price with VAT is \x9c%.2f", sumtotal);
}
```

3.3 Testing

3.3.1 Normal values

User enters valid numbers (in decimal)



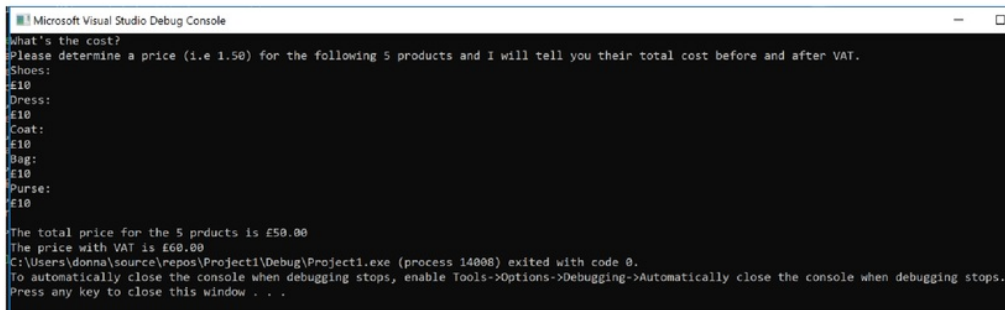
```
Microsoft Visual Studio Debug Console
What's the cost?
Please determine a price (i.e 1.50) for the following 5 products and I will tell you their total cost before and after VAT.
Shoes:
£9.50
Dress:
£10.99
Coat:
£29.99
Bag:
£15.00
Purse:
£3.99

The total price for the 5 products is £69.47
The price with VAT is £83.36
C:\Users\donna\source\repos\Project1\Debug\Project1.exe (process 12460) 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 . . .
```

Result: program ran as expected. Correct totals calculated

3.3.2 Boundary values

User enters all values the same



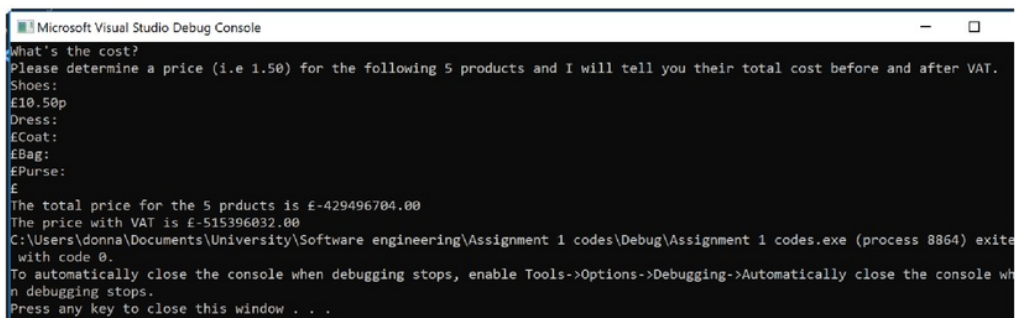
```
Microsoft Visual Studio Debug Console
What's the cost?
Please determine a price (i.e 1.50) for the following 5 products and I will tell you their total cost before and after VAT.
Shoes:
£10
Dress:
£10
Coat:
£10
Bag:
£10
Purse:
£10

The total price for the 5 products is £50.00
The price with VAT is £60.00
C:\Users\donna\source\repos\Project1\Debug\Project1.exe (process 14008) 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 . . .
```

Result: Program ran as expected. Correct totals calculated

3.3.3 Invalid values

User enters invalid characters, letters included



```
Microsoft Visual Studio Debug Console
What's the cost?
Please determine a price (i.e 1.50) for the following 5 products and I will tell you their total cost before and after VAT.
Shoes:
£10.50p
Dress:
£Coat:
£BAG:
£Purse:
£

The total price for the 5 products is £-429496704.00
The price with VAT is £-515396032.00
C:\Users\donna\Documents\University\Software engineering\Assignment 1 codes\Debug\Assignment 1 codes.exe (process 8864) 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 . . .
```

Result: Program failed.

Exercise 4: The fruit machine

4.1 Problem breakdown

This purpose of this exercise is to build a fruit machine game which allows a user to “spin” 3 reels in order to attain points. Each reel should range from 1 to 5, with points granted for “landing” 2 or 3 5's or three of a kind (with different points awarded depending on the combination).

4.1.1 Generate random numbers

Additional libraries are needed in order to generate random numbers in C, these are `<stdlib.h>` and `<time.h>`.

1. Add additional libraries to code
2. Assign variable and set an array for numbers generated
3. Dictate range for random number
4. State seed randomness
5. Create for statement to generate random numbers
6. Print generated numbers on the screen

4.1.2 Generate loop so user can continue playing

This stage requires input from the user.

1. Invite user to play game
2. Allow user to enter choice
3. Store entry in choice variable
4. Create while loop to repeat process
5. Print random values for that spin
6. Offer user choice to play again
7. If user says yes print new random numbers on screen
8. If user says no print “you're no fun” on screen

4.1.3 Assign points for spin

The exercise details the points that should be awarded for certain spins. These should be printed on screen for each spin.

1. Create variable score
2. Within the while loop created in 4.1.2, create if statements to determine scores for each spin.
3. Store score generated by random spin in score variable
4. Print score on screen in line with random numbers generated

4.1.4 Accumulate points for users spins

The final step is to accumulate the scores gained for the user.

1. Create variable accscore
2. For each spin, create new score total by adding score and accscore totals
3. Store new total in accscore
4. Print accscore on screen

4.2 Annotated code

4.2.1 Code to generate random numbers as per 4.1.1:

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
void main()
{
    int spin[3];
    int low = 1;
    int high = 5;
    srand(time(NULL));
    int i;
    for (i = 0; i < 3; i++)
    {
        spin[i] = rand() % (high - low + 1) + low;
    }

    printf("You have pulled: \n");
    for (i = 0; i < 3; i++)
    {
        printf("%d ", spin[i]);
    }
    printf("\n\n");
}
```

4.2.2 Code to allow user to spin as many times as they want as per 4.1.2:

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
void main()
{
    int spin[3];
    int low = 1;                      //lowest of range
    int high = 5;                     //highest of range
    srand(time(NULL));                //seed randomness
    int i;
    char choice;

    printf("Greetings player\n");
    printf("Should we play a game? (y/n)\n");
    scanf(" %c", &choice);

    while (choice == 'y' || choice == 'Y') //both uppercase and lowercase covered
    {
        for (i = 0; i < 3; i++)
        {
            spin[i] = rand() % (high - low + 1) + low;
        }

        printf("You have pulled: \n");
        for (i = 0; i < 3; i++)
        {
            printf("%d ", spin[i]);
        }
        printf("\n\n");

        choice = 'n'; //error check to ensure player wants to go again

        printf("Would you like to spin again?\n");
        scanf(" %c", &choice);
    }
    if (choice == 'n' || choice == 'N')
    {
        printf("You're no fun!");
    }
    else //this will trigger if the user enters any other character
        printf("invalid entry");
}
```

4.2.3 Code to add scores to spins:

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
void main()
{
    int spin[3];
    int low = 1;
    int high = 5;
    srand(time(NULL));
    int i, score;
    char choice;

    printf("Greetings player\n");
    printf("Should we play a game? (y/n)\n");
    scanf(" %c", &choice);

    while (choice == 'y' || choice == 'Y') //both uppercase and lowercase covered
    {
        for (i = 0; i < 3; i++)
        {
            spin[i] = rand() % (high - low + 1) + low;
        }

        printf("You have pulled: \n");
        for (i = 0; i < 3; i++)
        {
            printf("%d ", spin[i]);
        }
        printf("\n");

        if (spin[0] == 5 && spin[1] == 5 && spin[2] == 5) // score for 3 5's
        {
            score = 2000;

            printf("You scored %d points\n\n", score);
        }
        else if (spin[0] == 5 && spin[1] == 5 || spin[0] == 5 && spin[2] == 5 ||
spin[1] == 5 && spin[2] == 5) // score for 2 5's
        {
            score = 100;

            printf("You scored %d points\n\n", score);
        }
        //for variations of 3 of a kind
        else if (spin[0] == 1 && spin[1] == 1 && spin[2] == 1 || spin[0] == 2 &&
spin[1] == 2 && spin[2] == 2 || spin[0] == 3 && spin[1] == 3 && spin[2] == 3 ||
spin[0] == 4 && spin[1] == 4 && spin[2] == 4) // score for 2 5's
        {
            score = 100;

            printf("You scored %d points\n\n", score);
        }
        //for losing combinations
        else if (spin[0] != 5 && spin[1] != 5 && spin[2] || spin[0] != 1 &&
spin[1] != 1 && spin[2] != 1 || spin[0] != 2 && spin[1] != 2 && spin[2] != 2 ||
spin[0] != 3 && spin[1] != 3 && spin[2] != 3 || spin[0] != 4 && spin[1] != 4 &&
spin[2] != 4)
        {
            score = 0;

            printf("You scored %d points\n\n", score);
        }
    }
}
```

```

    }

    choice = 'n'; //error check to ensure player wants to go again

    printf("Would you like to spin again?\n");
    scanf(" %c", &choice);

}
if (choice == 'n' || choice == 'N')
{
    printf("You're no fun!");
}
else //this will trigger if the user enters any other character
    printf("invalid entry");
}

```

4.2.4 Accumulate scores

```

#include <stdio.h>
#include <stdlib.h>
#include <time.h>
void main()
{
    int spin[3];
    int low = 1; //lowest of range
    int high = 5; //highest of range
    srand(time(NULL)); //seed randomness
    int i, score, accscore;
    char choice;

    accscore = 0;

    printf("Greetings player\n");
    printf("Should we play a game? (y/n)\n");
    scanf(" %c", &choice);

    while (choice == 'y' || choice == 'Y') //both uppercase and lowercase covered
    {
        for (i = 0; i < 3; i++)
        {
            spin[i] = rand() % (high - low + 1) + low;
        }

        printf("You have pulled: \n");
        for (i = 0; i < 3; i++)
        {
            printf("%d ", spin[i]);
        }
        printf("\n");

        if (spin[0] == 5 && spin[1] == 5 && spin[2] == 5) // score for 3 5's
        {
            score = 2000;
            accscore = score + accscore;

            printf("You scored %d points\n\n", score);
        }
        else if (spin[0] == 5 && spin[1] == 5 || spin[0] == 5 && spin[2] == 5 ||
spin[1] == 5 && spin[2] == 5) // score for 2 5's
        {

```

```

        score = 100;
        accscore = score + accscore;

        printf("You scored %d points\nThat's %d in total\n\n", score,
accscore);
    }
    //for variations of 3 of a kind
    else if (spin[0] == 1 && spin[1] == 1 && spin[2] == 1 || spin[0] == 2 &&
spin[1] == 2 && spin[2] == 2 || spin[0] == 3 && spin[1] == 3 && spin[2] == 3 ||
spin[0] == 4 && spin[1] == 4 && spin[2] == 4)// score for 2 5's
    {
        score = 100;
        accscore = score + accscore;

        printf("You scored %d points\nThat's %d in total\n\n", score,
accscore);
    }
    else if (spin[0] != 5 && spin[1] != 5 && spin[2] || spin[0] != 1 &&
spin[1] != 1 && spin[2] != 1 || spin[0] != 2 && spin[1] != 2 && spin[2] != 2 ||
spin[0] != 3 && spin[1] != 3 && spin[2] != 3 || spin[0] != 4 && spin[1] != 4 &&
spin[2] != 4)
    {
        score = 0;
        accscore = score + accscore;

        printf("You scored %d points\nThat's %d in total\n\n", score,
accscore);
    }

    choice = 'n';//error check to ensure player wants to go again

    printf("Would you like to spin again?\n");
    scanf(" %c", &choice);

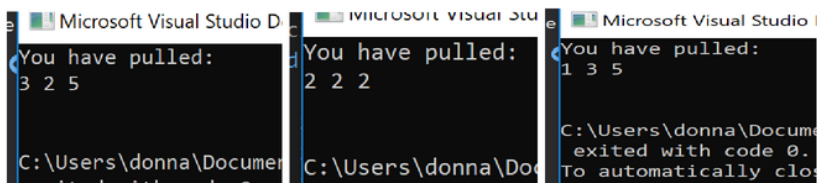
}
if (choice == 'n' || choice == 'N')
{
    printf("You're no fun!");
}
else //this will trigger if the user enters any other character
    printf("invalid entry");
}

```

4.3 Testing

4.3.1 Random numbers

Program creates 3 random numbers between 1 and 5.



Result: Program ran as expected.

4.3.2 Loop game

Allow user to play repeatedly when entering y or Y

```
C:\Users\donna\Documents\University\St
Greetings player
Should we play a game? (y/n)
y
You have pulled:
2 3 2

Would you like to spin again?
y
You have pulled:
2 4 3

Would you like to spin again?
y
You have pulled:
5 5 5
```

Result: Program ran as expected.

Allow user to stop playing

```
Microsoft Visual Studio Debug Console
Greetings player
Should we play a game? (y/n)
y
You have pulled:
2 4 2

Would you like to spin again?
y
You have pulled:
2 5 5

Would you like to spin again?
n
You're no fun!
```

Result: Program ran as expected.

User enters invalid value

<pre>Microsoft Visual Studio Debug Console Greetings player Should we play a game? (y/n) y You have pulled: 5 4 2 Would you like to spin again? n invalid entry C:\Users\donna\Documents\University\St</pre>	<pre>Microsoft Visual Studio Debug Console Greetings player Should we play a game? (y/n) y You have pulled: 5 3 2 Would you like to spin again? 0 invalid entry</pre>
---	--

Result: Program ran as expected.

4.3.3 Add scores

Program assigns scores to spin and prints them on screen

```
Would you like to spin again?  
y  
You have pulled:  
3 3 3  
You scored 100 points  
  
Would you like to spin again?  
y  
You have pulled:  
5 4 4  
You scored 0 points  
  
Would you like to spin again?  
y  
You have pulled:  
4 3 5  
You scored 0 points
```

Result: Program ran as expected.

4.3.4 Accumulate scores

Program assigns a scores for spins and provides the accumulated total.

```
Would you like to spin again?  
y  
You have pulled:  
5 1 4  
You scored 0 points  
That's 0 in total  
  
Would you like to spin again?  
y  
You have pulled:  
3 3 3  
You scored 100 points  
That's 100 in total  
  
Would you like to spin again?  
y  
You have pulled:  
1 5 5  
You scored 100 points  
That's 200 in total
```

Result: Program ran as expected.

References

- [1] M. McGrath, C Programming, vol. 4th edition, Warwickshire: In easy steps, 2014.

Assignment 1 Donna Wright

GRADEMARK REPORT

FINAL GRADE

/100

GENERAL COMMENTS

Instructor

Part A

Variables - You need to talk about the different types of variables here and the types of values that they hold eg int, float, char etc (Actually I found a bit further down in Part B, I'll give you an extra mark here) 3.5 /5

Counting Loops A good explanation, but you need to include a coded example 4/5

Arrays You have covered the main points, again, show a coded example 4 /5

Part B

Exercise 1

Code-Annotation You need to have the user input the numbers rather than hard code them, this in turn would allow you to look at validation which is implicit here for age and height eg age should be between 0-120 in this context 2/5

Documentation As the code is limited, the breakdown and the testing is limited here too 2/5

Exercise 2

Code-Annotation - Working solution, validation on the relative sizes of the triangle and the circle is

good. You need to think about the user entering negative values 4 /5

Documentation - Good breakdown, good range of tests other than negative values 4/5

Exercise 3

Code-Annotation - A working solution, but it would be more efficient to use a for loop for entering the products and adding the prices. Also, the price here needs to be validated so that it is not negative. 2.5/5

Documentation Good breakdown, testing is OK, boundary values would be more if the user entered all zeroes. Also, negative values not tested. 3/5

Exercise 4

Code-Annotation You only need to put in the last version of the code :) A good implementation, with the accumulating total implemented. Some of your if statements are very long, you could have used a for loop to count how many 5s were in the array. Also, you don't need to compare for 3 1s, 3 2s and so on - once you've established that it's not 3 5s, all that's needed is to know if all 3 are equal, it doesn't matter about the value. This would shorten your if statement considerably. However, this is a minor point. 8.5/10

Documentation - Good breakdown, testing is good overall, just make sure you show all the combinations of 5s in particular as these have their own if statement that needs to be tested. 8/10

Total: 45.5/65 (70%)

PAGE 4

PAGE 5

PAGE 6

PAGE 7

PAGE 8

PAGE 9

PAGE 10

PAGE 11

PAGE 12

PAGE 13

PAGE 14

PAGE 15

PAGE 16

PAGE 17

PAGE 18

PAGE 19

PAGE 20

PAGE 21
