

Assignment computer programming

by Jericho Uzzell

FILE	12014_JERICO_UZZELL_ASSIGNMENT_COMPUTER_PROGRAMMING_173935_1156020913.DOCX (704.39K)		
TIME SUBMITTED	05-MAY-2019 11:22PM (UTC+0100)	WORD COUNT	941
SUBMISSION ID	106368864	CHARACTER COUNT	4956

University of Wales Trinity Saint David

School of Engineering

Programming

Introduction to Computer Programming

M4X01508

Jericho Uzzell

1702201

22/03/19



Prifysgol Cymru

Y Drindod Dewi Sant

University of Wales

Trinity Saint David

Part A

Q1 – What is a variable and why are they important in writing programs?

A variable is dynamic, it can change at any point throughout the program. Programmers should name their variables correctly for what task they may do, for example, if I am going to be counting two numbers together, I would use an integer and call the variable countA.

The different types of variables within C are: int, char, bool, float; An integer (int) is used to store whole numbers, a character (char) is used to store singular characters however can be used to store strings and as an array but this must be defined in square brackets after the char. Bool also known as a boolean is a true and false variable that can be used for logic statements – true is known as 1 and false is known as 0. Floats are used for numbers that have a decimal point; this can be used for currencies.

For example:

```
Int i=0;
char i="a";
char l[8]="Jericho";
Bool i=false;
Float i=0.2;
```

Q2 – Explain why the for-loop can also be referred to as a counting-loop and give an example of when you would use it.

A for-loop is referred to as a “counting-loop” because it will always have a number to define the start of the loop and a number to end the loop. The number to end the loop can either be equal to, less than or greater than, less than or equal to and greater than or equal to.

For example:

```
For(i=0;i<10;i++)
For(i=0;i>10;i++)
For(i=0;i==10;i++)
For(i=0;i>=10;i++)
For(i=0;i<=10;i++)
```

Q3 - Explain what an array is, what their features are, and give an example of when you would use one.

An array is a collection of data that can be accessed with an index, if the array is not an associated array then a number would be its index. Within this array (which is not an associated array), you would use a number that would range from zero to the last element plus one. With that said, some languages allow something known as an associated array that will hold a key as a string, this is useful to access information about the array without knowing the element number, but C does not offer this feature.

An example of when an array would be used for could be for a user logs in. You might want to store their username, tags, user type etc. You are not limited to what was listed but you must consider the length of your arrays for if your application can support it in memory.

With all that said, arrays can range from two dimensional all the way through to eight (it does depend on what your application requirements are, and the memory allocated – it can also get confusing the more dimensions you go into). This will allow you to track data, for example, if you were coding a game of battleships you might make it so index zero would be player one. Index zero (player one) could have items within itself such as coordinates and guesses.

For example:

```
char ships[3][3][2]={ {"1","0"}, {"2","2"} };
for(int i=0;i<2;i++){
    printf("Player: %s\n",ships[i][0]);
    printf("\tGuesses: %s\n",ships[i][1]);
}
```

Exercise 1 – Show how variables can contain different types of values by declaring three variables that contain a name, a measure of height (e.g. 1.81cm) and a person's age (e.g. 32). Input these values and display them on the screen.

```
#include <stdio.h>
void main(){
    int age=30;
    char name[8]="Jackson";
    float height=5.10;
    printf("My name is %s, I am %.2fcm tall and I am aged %d.",name,height,age);
}
```

Output:

```
My name is Jackson, I am 5.10cm tall and I am aged 30
```

Exercise 2 – Allow the user to enter the dimensions of shapes, then calculate the area of the blue shaded part. Note: The triangular area in the centre is empty.

```
#include <stdio.h>

void main(){
    float height=0,base=0,area=0;
    printf("Height of triangle: ");
    if(scanf("%f",&height)==0) {
        printf("\nYou need to enter a valid number for this input.");return;
    }
    printf("Base of triangle: ");
    if(scanf("%f",&base)==0){
        printf("\nYou need to enter a valid number for this input.");return;
    }
    //half height times the base
    area=height / 2 * base;
    printf("Area of triangle: %.2f",area);
}
```

Output:

Normal output:

```
Height of triangle: 15
```

```
Base of triangle: 10
```

```
Area of triangle: 75.00
```

If user enters anything other than a number:

```
You need to enter a valid number for this input.
```

Part B

Exercise 3 – Write a program that allows the user to enter the price of 5 products into an array. Once all the prices are entered calculate the total price, and then calculate the total price plus 20% VAT.

```
#include <stdio.h>

void main(){
    //Declare variables.
    // @prices = five available elements;
    // @i      = counter (loops);
    float prices[5]={},i=0, calc, vat=20.00,vatout=0;
    printf("Enter the five different products: ");
    for(int i = 0; i < 5;i++) {
        printf("\n%d: ", i+1);
        scanf("%f", &prices[i]);
    }
    for(int i = 0; i < 5; i++) {
        calc=calc+prices[i];
        printf("%.2f ",prices[i]);
    }
    // 20 divided by 100 times the price
    vatout=calc-(calc * 100.0 / (100.0 + vat));
    printf("\n\nPrice = %.2f | Inc VAT = %.2f",calc,calc+vatout);
}
```

Output:

3: 0

4: 0

5: 0

122.00 0.00 0.00 0.00 0.00

Price = 122.00 | Inc VAT = 142.33

Exercise 4 – Build a fruit machine game which represents the 3 reels of symbols using numbers ranging from 0 to 5. Allow the user to have as many goes as they like (the numbers are chosen randomly each time).

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
int generateRamdom(int lower,int upper){
    int num=(rand())%(upper-lower+1)+lower;
    return num;
}
void main(){
    //Declare variables.
    // @prices = five available elements;
    // @i      = counter (loops);
    char con='y';
    int i = 0,number=0,lastno=0,looptime=3;
    srand(time(0));
    while(con!='n'){
        int columns=0, mainNumber=-1,prices[3]={};
        for(i = 0;i<looptime;i++){
            number=generateRamdom(0,5);
            //printf("\n%d > %d",i,number);
            prices[i]=number;
        }
        for(i=0;i<looptime;i++){
            if(i==0){mainNumber=prices[i];}
            else if(mainNumber==prices[i]){columns++;}
            printf("%d ",prices[i]);
        }
        if((looptime - 1)==columns){printf("\nJackpot!");}
        printf("\nDo you want to continue, type \"n\" for no: ");

        scanf(" %c",&con);
    }
}
```

Output:

0 0 1

Do you want to continue, type "n" for no: h

1 1 1

Jackpot!

References

- [1 G. Thomas, UWTSD 2019;
]

Assignment computer programming

GRADEMARK REPORT

FINAL GRADE

58/100

GENERAL COMMENTS

Instructor

Part A

Variables - You have covered most of the points, you could also have talked about the naming rules 9/10

Counting Loops - the point with a for loop is that the number of iterations is known before the loop starts. Example is OK, I would have liked to see the contents of the statement rather than just the controller and some of them have a capital I rather than small i. 6/10

Arrays - You need to state that the collection of data stored by an array is the same types. Example is OK, although a 3D array is a bit complex! 8/10

Part B - you haven't included a logical breakdown of the program in the documentation (eg pseudocode) - please read the assignment brief carefully

Exercise 1

Code-Annotation A bit limited - it would be better to have the user enter the values, then you can also show validation 2/5

Documentation - you need to break down stages. Testing is limited because the code is limited - nothing to test without data entry 2 /5

Exercise 2

Code-Annotation A reasonable attempt, you need to validate the inputs ie not to allow negative numbers and the size of the triangle should not be bigger than the circle. You also have not shown the area of the shaded part? 5/10

Documentation No breakdown, testing is limited to the happy path and impossible, you need the full range eg negative numbers etc 2/5

Exercise 3

Code-Annotation - Good overall, you just need to validate for negative prices 8/10

Documentation No breakdown, testing is limited, a wider range of values need to be tested eg negatives etc 2.5/5

Exercise 4

Code-Annotation Good use of functions. You have not allocated the scoring as required in the assignment brief eg points for 2 5s and so on.10 /20

Documentation No breakdown, testing is limited. 3.5/10

Total: 58%

PAGE 5

PAGE 6

PAGE 7
