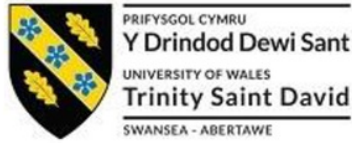


programs in C and documentation

by Hafiz Uddin

FILE	11625_HAFIZ_UDDIN_PROGRAMS_IN_C_AND_DOCUMENTATION_17393 5_328089285.DOCX (126.48K)		
TIME SUBMITTED	05-MAY-2019 10:29PM (UTC+0100)	WORD COUNT	2006
SUBMISSION ID	106366015	CHARACTER COUNT	10274



University of Wales Trinity Saint David

School of Applied Computing

Programs in C and Documentation

Introduction to Computer Programming

Hafiz Uddin

1807533

02/05/2019

Contents Page

Contents

Contents Page	2
Understanding basic computer programming	3
Introduction	3
Methodology	3
A1- What is a “variable”	3
A2 – for-loop	3
A3 – array	4
Exercise 1	5
Exercise 2	6
Exercise 3	8
Exercise 4	9
Conclusions	12
References	13

Understanding basic computer programming

Introduction

[1] To understand the basic functions of computer programming, to understand and explain what a variable is and what functions it has, to understand and explain for-loop and when to use one, to explain what an array is and its features, and to write basic computer programmes from well-defined scenarios or problems. To understand the basic functions, the syntax, the semantics and software construction of computer programming using the C language [1].

Methodology

A1- What is a “variable” = [2] A variable is storage of computer memory which contains information that can be referenced and manipulated later in a computer program (a variable must be declared and initialized before it can be executed in a program). Each variable uses different amount of computer memory, the smallest being the char variable it uses one byte, the float and int uses four bytes and the double data uses eight bytes of memory, as the string can hold multiple letters it is allocated one byte per letter. These variable values can vary depending on the processor in the CPU, eg- for a float data type 16-bit processor will allocate two bytes, for a 32-bit processor it will allocate four bytes, for a 64-bit processor it will allocate eight bytes [2].

[1] variables start with a lowercase letter, they cannot start with spaces, special characters or keywords.

Special characters are for example – (< >), (), *, &, @, #, ect.

Keywords for example – scanf, printf, null, true, false, if, are already built into the compiler for functions.

There are four basic variables types –

int = integers, are used for whole numbers, eg 5, 10, 21, 36 ect.

floats = are used for numbers with decimal points in them, eg 1.2, 2.5, 10.9 ect.

char = characters, are used for single letters, eg “a”, “d”, “w”, “z”, ect.

string = are used for multiple letters, eg “Hafiz”, “university”, “work”, ect.

A2 – for-loop = [3] used in a program to repeat a section of the code and is controlled by an integer. The amount of times the section is to be repeated is always known, either it is declared. A for loop is flexible and can be either increased or decreased the counter, the loop will have a initial value and it will increase to an upper limit, it is also known as a counting loop.

For example, if you want to know the ages of all your employees, the for-loop will repeat the section of the code until all the employees ages have been known, it will then move onto the next part of the program or end.

A3 – array = [4] A fixed size data belonging to the same data type, a structure of data that can store several variables of the same data type in a sequence, these include, "int", "char", "float",

As "int", "char" and "float" can hold only one instruction (data) at a time whereas an array can hold a collection of related data types.

[4] An array can be used to collect the same data from a group of people, for example if you wanted to find out what shoe sizes this group of people wore it would be put into an array. The array would repeat the question to each person until it has collected everyone's shoe size.

Exercise 1

This program shows three variables – A name, height and age of a person.

```
#include<stdio.h> // contains input and output functions.

void main()
{
    int age = 0; // lowest range of age being declared (whole number).
    float height = 0; //lowest range of height being declared (decimal number).
    char name[40]; //highest number of letters in first name.
    char sname[40]; //highest number of letters in surname.
    printf("\nplease answer the following three questions:\n");

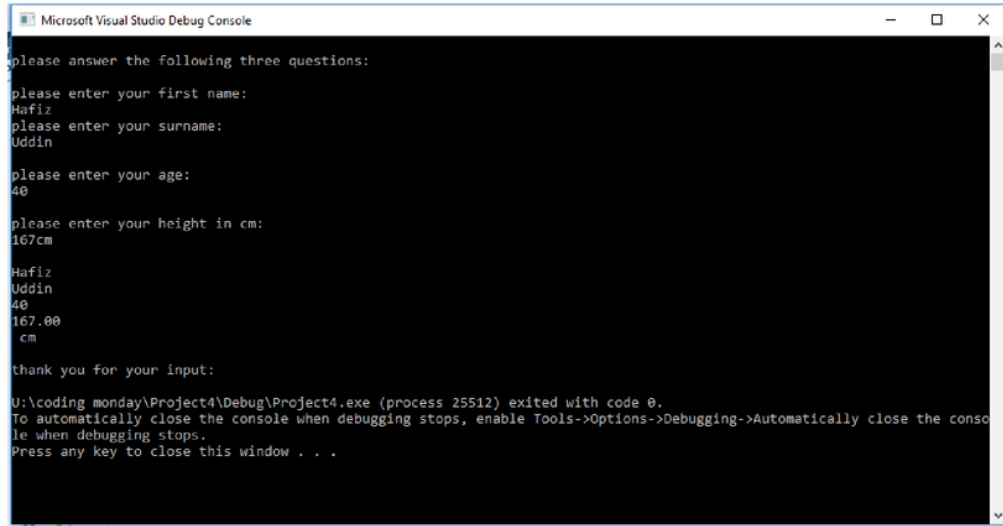
    printf("\nplease enter your first name:\n");
    scanf("%s", &name);
    printf("\nplease enter your surname:\n");
    scanf("%s", &sname);
    while (age <= 0 || age > 150) //lowest and highest range of age.

    {
        printf("\nplease enter your age:\n");
        scanf("%d", &age);
    }
    while (height <= 0 || height > 250) //lowest and highest range of height.

    {
        printf("\nplease enter your height in cm:\n");
        scanf(" %f", &height);
    }

    printf("\n%s\n", name);
    printf("%s\n", sname);
    printf("%d\n", age);
}
```

```
printf("%.2f\n cm\n", height);
printf("\nthank you for your input:\n");
}
```



```
Microsoft Visual Studio Debug Console
please answer the following three questions:
please enter your first name:
Hafiz
please enter your surname:
Uddin
please enter your age:
40
please enter your height in cm:
167cm
Hafiz
Uddin
40
167.00
cm
thank you for your input:
U:\coding monday\Project4\Debug\Project4.exe (process 25512) 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 . . .
```

Example of exercise 1.

Exercise 2

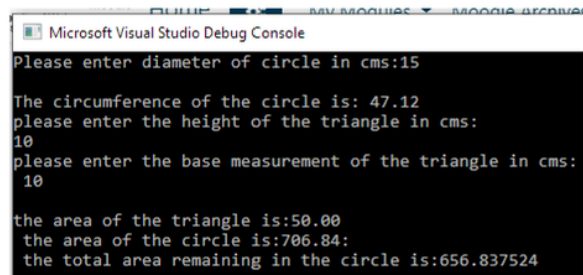
This program allows users to input dimensions of a circle and a triangle. The triangle is in the middle of the circle. The program will calculate the radius of the circle then take away the area of the triangle and give you the remaining area of the circle.

```
#include <stdio.h> //input and output functions.
#define PI 3.1415 // contains pi function
#include<math.h> //contains mathematical functions
void main()
{
    float base = 0, height = 0, total1, total2, radius, total3, total4; //declaring ranges.
    printf("Please enter diameter of circle in cms:");
    scanf("%f", &radius);
    total1 = radius * PI;
    printf("\nThe circumference of the circle is: %.2f", total1);
    printf("\nplease enter the height of the triangle in cms:\n");
    scanf("%f", &height);
    printf("please enter the base measurement of the triangle in cms:\n ");
    scanf("%f", &base);
    total2 = (height * base) / 2;
    while (height > radius) //loop to repeat this part.
    {
        printf("please enter the height of the triangle in:\n");
        scanf("%f", &height);
        if (height > radius);
        {
            printf("please enter a height less than the radius in:\n");
        }
    }
}
```

```

    }
    while (base > radius)//loop to repeat.
    {
        printf("please enter the base measurement of the triangle:\n ");
        scanf("%f", &base);
        if (base > radius);
        {
            printf("please enter base less than the radius:\n");
        }
    }
    printf("\nthe area of the triangle is:%.2f\n", total2);
    total3 = PI * (radius*radius);
    {
        printf(" the area of the circle is:%.2f:\n", total3);
    }
    total4 = total3-total2; //mathematical equation declared.
    {
        printf(" the total area remaining in the circle is:%2f\n", total4);
    }
}

```



The screenshot shows the Microsoft Visual Studio Debug Console with the following output:

```

Please enter diameter of circle in cms:15
The circumference of the circle is: 47.12
please enter the height of the triangle in cms:
10
please enter the base measurement of the triangle in cms:
10
the area of the triangle is:50.00
the area of the circle is:706.84:
the total area remaining in the circle is:656.837524

```

Example of exercise 2.

Exercise 3

This program allows users to enter prices of five products, it then calculates the total price and adds a 20% VAT and gives the user the answer.

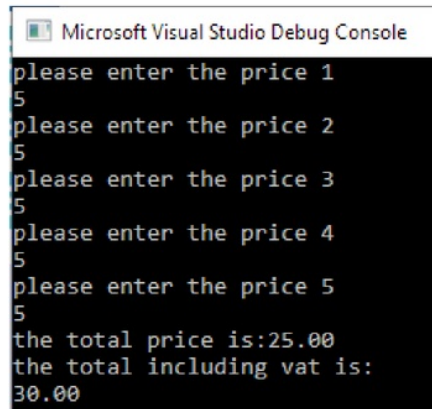
```
#include<stdio.h> //functions for input and output

void main()
{
    float vat = 20.0; //declaring variables
    float total = 0;
    int sum = 0;
    int x = 0;
    float total2;
    int num[5];

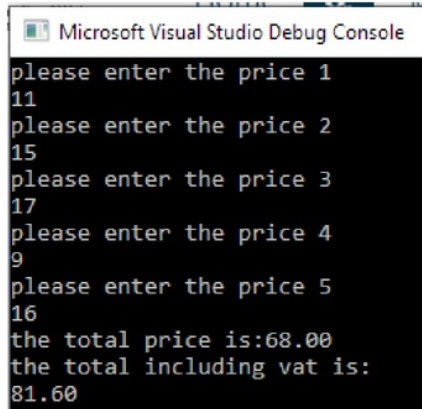
    for (x = 0; x < 5; x++)
    {
        printf("please enter the price %d\n", (x + 1));
        scanf("%d", &num[x]);
    }
    for (x = 0; x < 5; x++) //mathematical equation declared.
    {
        sum = sum + num[x]; //mathematical equation declared.
    }
    total = sum;
    printf("the total price is:%.2f\n", total); ////mathematical equation declared.

    total2 = total*.20+total; //VAT calculation.

    printf("the total including vat is:\n%.2f",total2);
}
```



```
Microsoft Visual Studio Debug Console
please enter the price 1
5
please enter the price 2
5
please enter the price 3
5
please enter the price 4
5
please enter the price 5
5
the total price is:25.00
the total including vat is:
30.00
```



```
Microsoft Visual Studio Debug Console
please enter the price 1
11
please enter the price 2
15
please enter the price 3
17
please enter the price 4
9
please enter the price 5
16
the total price is:68.00
the total including vat is:
81.60
```

Working program for VAT calculator.

Exercise 4

A fruit machine style game that represents three reels of numbers ranging from 0 to 5. The numbers are chosen randomly each time. It allows the user to have as many goes as they wish, it will allocate points for wins and calculate the total.

```
#include<stdio.h> // for input and output functions to be defined
#include<stdlib.h> //for srand, rand and number conversion to be defined
#include<math.h> // for mathematical functions to be defined
#include<time.h> //for time to be defined
#include<string.h> // functions for manipulating strings

int main()
{
    int spins;
    int wheel1; int wheel2; int wheel3; // declaring the number of wheels in the game.
    int low = 0; //lowest range of numbers in the game.
    int high = 6; // highest range in the game.
    rand() % 5 + 1;
    srand(time(NULL));
    float total;
    char choice;

    wheel1 = ((rand() % 5) + 1);
    wheel2 = ((rand() % 5) + 1);
    wheel3 = ((rand() % 5) + 1);
    //wheel1 = 2;
    //wheel2 = 5;
    //wheel3 = 5;

    printf("wheel1:%d\n", wheel1);
    printf("wheel2:%d\n", wheel2);
    printf("wheel3:%d\n", wheel3);
    printf("-----\n");
    if (wheel1 == wheel2 && wheel2 == wheel3 && wheel1 == 5)//jackpot (all 5s)
    {
        printf("fantastic you have hit the jackpot man:\n ");
        printf("5000 points");
    }
    else if (wheel1 == wheel2 && wheel2 == wheel3)//all 1s 2s 3s 4s
    {
        printf("Thats 20 points man");
    }
    else if (wheel1==5 && wheel2==5
||wheel2==5&&wheel1==5||wheel3==5&&wheel2==5||wheel3==5&&wheel1==5)
    {
        printf("\nThats 100 points man ");
    }
    else
    {
        printf(" you lose man ");
    }

    return 0;
}
```

```
Microsoft Visual Stu
wheel1:1
wheel2:1
wheel3:1
-----
20 points
```

All 1s working.

```
Microsoft Visual Studi
wheel1:2
wheel2:2
wheel3:2
-----
20 points
```

All 2s working

```
Microsoft Vi
wheel1:3
wheel2:3
wheel3:3
-----
20 points
```

all 3s working

```
Microsoft Vi
wheel1:4
wheel2:4
wheel3:4
-----
20 points
```

all 4s working

```
Microsoft Visual Studio Debug Console
wheel1:5
wheel2:5
wheel3:5
-----
fantastic you have hit the jackpot man:
```

Jackpot all 5s working.

```
Microsoft Visual Studio Debu
12 = 1 * 5 + 1
11 = 1 * 5 + 1
10 = 1 * 5 + 1
9 = 1 * 5 + 1
8 = 1 * 5 + 1
7 = 1 * 5 + 1
6 = 1 * 5 + 1
5 = 1 * 5 + 1
4 = 1 * 5 + 1
3 = 1 * 5 + 1
2 = 1 * 5 + 1
1 = 1 * 5 + 1
-----
Thats 100 points man
```

Two 5s and another number.

My attempt at all parts of exercise 4 adding a repeat play function and a counting score function.

```
#include<stdio.h> // for input and output functions to be defined
#include<stdlib.h> //for srand and rand to be defined
#include<math.h> // for mathematical functions to be defined
#include<time.h> //for time to be defined
#include<string.h> // functions for manipulating strings
```

```
int main()
{
    int score;
    int wheel1; int wheel2; int wheel3;
    int low = 0;
    int high = 6;
    int count_score(int wheel1, int wheel2, int wheel3);
    int wheel = 0;
    rand() % 5 + 1;
    srand(time(NULL));
    float total;
    char choice;
    char play;
    bool repeat = true;
    int random_numbers (int wheel);
    wheel1 = ((rand() % 5) + 1);
    wheel2 = ((rand() % 5) + 1);
    wheel3 = ((rand() % 5) + 1);
    //wheel1 = 2;
    //wheel2 = 5;
    //wheel3 = 5;
    while (repeat)
    {
        printf("would you like to have a go?");
        scanf("%s", &choice);
        if ((play == 'y') || (play == 'Y'))
        {
            wheel1 = random_numbers(wheel1);
            printf("%d", wheel1);
            wheel2 = random_numbers(wheel2);
            printf("%d", wheel2);
            wheel3 = random_numbers(wheel3);
            printf("%d", wheel3);
```

```

    }
    score = count_score(wheel1, wheel2, wheel3);
    printf("%d", count_score);
    if (score > 0)
    {
        printf("you win man your score is %d \n", score);
    }
    else
    {
        printf("\nsorry you lose man, your score is 0\n");
    }
    if ((play == 'n') || (play == 'N'))
    {
        printf("\n Thank you for playing man\n");
        repeat = false;
    }
    else
    {
        printf("\n sorry man this option is not allowed, please press y/Y or n/N \n");
    }

    printf("wheel1:%d\n", wheel1);
    printf("wheel2:%d\n", wheel2);
    printf("wheel3:%d\n", wheel3);
    printf("-----\n");
    if (wheel1 == wheel2 && wheel2 == wheel3 && wheel1 == 5)//jackpot (all 5s)
    {
        printf("fantastic you have hit the jackpot man:\n ");
        printf("5000 points");
    }
    else if (wheel1 == wheel2 && wheel2 == wheel3)//all 1s 2s 3s 4s
    {
        printf("Thats 20 points man");
    }
    else if (wheel1 == 5 && wheel2 == 5 || wheel2 == 5 && wheel1 == 5 || wheel3 == 5 &&
wheel2 == 5 || wheel3 == 5 && wheel1 == 5)
    {
        printf("\nThats 100 points man ");// win for two 5s and another number
    }
    else
    {
        printf(" you lose man ");
    }

    return 0;
}
}

```

Conclusions

Knowing the basic declarations of programming gives you the foundation of coding, learning the variables and understanding what they can do would enable you to program with success.

As programming is a language learning the basics is a good start and then building on that foundation, as with any language it all comes down to understanding and practice. In the exercises knowing and understanding the variables and which ones to declare helped to complete and run these programs, with all language mistakes can have effects on the running of the program these can be something as small semicolon(;) or speech marks(") which is why when learning to program it can be very frustrating, perseverance and patients is a key to succeed. As more practice takes place these programs can be tidied up and made to look a lot fancier. Also, a lot more can be added to each of these programs for it to do a lot more functions, again this will come down to learning and practice.

References

- [1] M.Mcgrath, C Programming, *grasp the nuts and bolts of programming*, 5th Edition, Leamington Spa: Warwickshire, 2018.
- [2] Tutorials point, "C – Data Types, tutorial point simple easy learning", Tutorialspoint.com 2019. [Online] Available: https://www.tutorialspoint.com/cprogramming/c_data_types.htm. [Accessed May.01, 2019].
- [3] zentut, "C for loop, *C tutorial*", Zen tut .com 2019. [Online] Available: <http://www.zentut.com/c-tutorial/c-for-loop/>. [Accessed May.02, 2019].
- [4] zentut, "C for loop, *C tutorial*", Zen tut .com 2019. [Online] Available: <http://www.zentut.com/c-tutorial/c-array/> [Accessed May.02, 2019].

programs in C and documentation

GRADEMARK REPORT

FINAL GRADE

68/100

GENERAL COMMENTS

Instructor

Part A

Variables A comprehensive explanation! 10/10

Counting Loops Be careful in your expression "A for loop is flexible" - it's not really. As you stated in your first sentence, the number of times the section is to be repeated is known ie inflexible. You need to include a coded example. 6/10

Arrays You've covered the main points, but you need to include a coded example. You could also talk about the index of the array and how it counts from 0 in C 6/10

Part B

Exercise 1

Code-Annotation A good implementation with validation included. 5/5

Documentation No breakdown? You need to produce a numbered list of logical steps. Testing is OK, but limited to the "happy path". You need to test for out of range values (eg negative values) and also boundary values ie 0 and 150 for age, 0 and 250 for height. 2.5 /5

Exercise 2

Code-Annotation Generally good, although some of your if statements are not needed because the same thing is being done by the while. in fact, the if statements are not being run because you've put a semicolon on then. No semicolons on if statements. You also need to check for negative numbers being entered A few sets of random brackets are present 6/10

Documentation - Same comment above on numbered steps. Same comment as above on testing 2.5 /5

Exercise 3

Code-Annotation - Good use of the array, you just need to add validation to stop the user entering a negative price 8/10

Documentation Same comment above on numbered steps. Same comment as above on testing 2.5/5

Exercise 4

Code-Annotation A good effort! You have not done the accumulating score, but nevertheless, you have tested efficiently for all the possibilities. 13.5/20

Documentation Same comment above on numbered steps. Testing is better, just show all combination of 2 5s so that you've tested all conditions on your if statements 6 /10

Total: 68%

PAGE 1

PAGE 2

PAGE 3

PAGE 4

PAGE 5

PAGE 6

PAGE 7

PAGE 8

PAGE 9

PAGE 10

PAGE 11

PAGE 12

PAGE 13
