**Programming Fundamentals**

**Lab 8**

**Submitted To:**

Mr. Dilshad Sabir

**Submitted By:**

Manaal Waseem

FA18-BCE-074

**In Lab:**

**Task 1:**

**For the given program find (and print) the addresses assigned to all the variables against their names.**

**Code Listing 1**

**#include<stdio.h>**

**void test(void);**

**int m = 22, n = 44;**

**float a = 50.5, b = 8.79;**

**int main()**

**{**

**int x, y=5, z = 5;**

**test();**

**}**

**void test()**

**{**

**char ch1 = ‘F’, ch2 = 69, ch3 = 100;**

**int x = 5, y = 55, z = 8;**

**}**

**Program:** In this program, I have made additions to the program that was already given to us. The addresses of the variables have been assigned to pointers. Then the value as well as address of the variable is printed through **“printf”** statements by using **%d** and **%u** format specifiers respectively.

1 #include<stdio.h>

2

3 **void** test(**void**);

4

5 **int** m = 22, n = 44;

6 **float** a = 50.5, b = 8.79;

7

8 **int** \* ptr\_global1=&m;

9 **int** \* ptr\_global2=&n;

10 **float** \* ptr\_global3=&a;

11 **float** \* ptr\_global4=&b;

12

13

14 **int** main()

15 {

16 17 **int** x, y=5, z = 5;

18

19 **int** \* ptr\_main1=&x;

20 **int** \* ptr\_main2=&y;

21 **int** \* ptr\_main3=&z;

22

23 printf("Address of x is:%u and its value is:%d\n\n",ptr\_main1,\*ptr\_main1);

24 printf("Address of y is:%u and its value is:%d\n\n",ptr\_main2,\*ptr\_main2);

25 printf("Address of z is:%u and its value is:%d\n\n",ptr\_main3,\*ptr\_main3);

26

27

28 test();

29

30 printf("Address of m is:%u and its value is:%d\n\n",ptr\_global1,\*ptr\_global1);

31 printf("Address of n is:%u and its value is:%d\n\n",ptr\_global2,\*ptr\_global2);

32 printf("Address of a is:%u and its value is:%f\n\n",ptr\_global3,\*ptr\_global3);

33 printf("Address of b is:%u and its value is:%f\n\n",ptr\_global4,\*ptr\_global4);

34

35 }

36

37 **void** test()

38 {

39 **char** ch1 ='F', ch2 = 69, ch3 = 100;

40

41 **char** \* ptr\_test1=&ch1;

42 **char** \* ptr\_test2=&ch2;

43 **char** \* ptr\_test3=&ch3;

44

45 printf("Address of ch1 is:%u and its value is:%d\n\n",ptr\_test1,\*ptr\_test1);

46 printf("Address of ch2 is:%u and its value is:%d\n\n",ptr\_test2,\*ptr\_test2);

47 printf("Address of ch3 is:%u and its value is:%d\n\n",ptr\_test3,\*ptr\_test3);

48

49 **int** x = 5, y = 55, z = 8;

50

51 **int** \* ptr\_test4=&x;

52 **int** \* ptr\_test5=&y;

53 **int** \* ptr\_test6=&z;

54

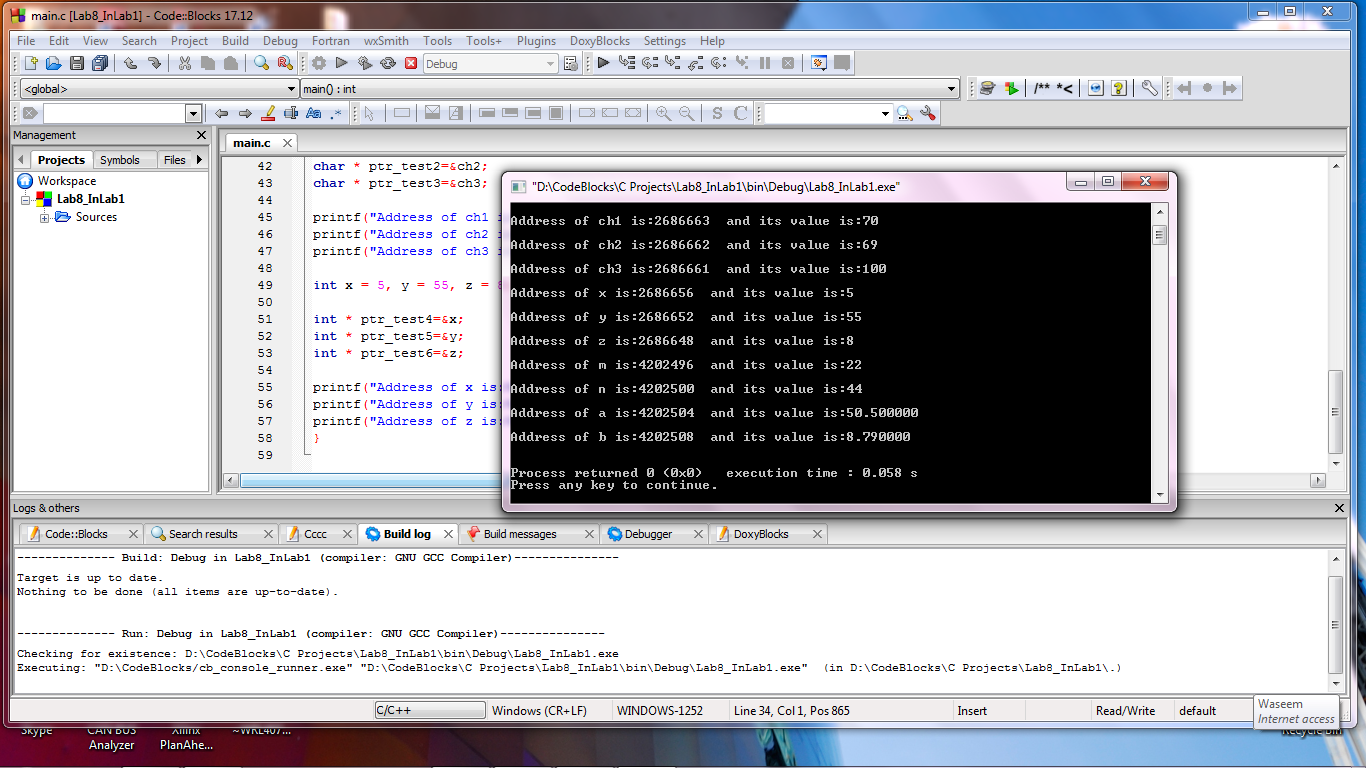
55 printf("Address of x is:%u and its value is:%d\n\n",ptr\_test4,\*ptr\_test4);

56 printf("Address of y is:%u and its value is:%d\n\n",ptr\_test5,\*ptr\_test5);

57 printf("Address of z is:%u and its value is:%d\n\n",ptr\_test6,\*ptr\_test6);

58 }

**Output:**



**Task 2:**

**Write a C function which takes three integer inputs, corresponding to RGB components of a colored pixel, and coverts them to the YUV color space.**

**Program:** In this program, first the function **“rgb2ycbcr”** is completed as per instructions in the reading material. Then the program is compiled via. **Command Prompt** by using the following command given in the reading material:

gcc main.c my\_lib.o -o output.exe

Hence, the **output.exe** file is created. When executed the output is displayed on console with **“Conversion Successful! Well Done!!”** message.

1 #include<stdio.h>

2 #include"my\_lib.h"

3

4 **int** main (**void**)

5 {

6 printf("Program to test color space conversion\n\n");

7

8 printf("Testing color space conversion code: \n\n");

9 **if**(test\_code() == 0)

10 printf("\n\n Conversion Successful! Well Done!!\n");

11 **else**

12 printf("\n\nOops! Conversion failed!!!\n\n");

13 }

14

15 **void** rgb2ycbcr(**int** r, **int** g, **int** b, **int** \* y, **int** \* cb, **int** \* cr)

16 {

17 \*y = (**int**)(16 + (0.257\*r+0.504\*g+0.098\*b));

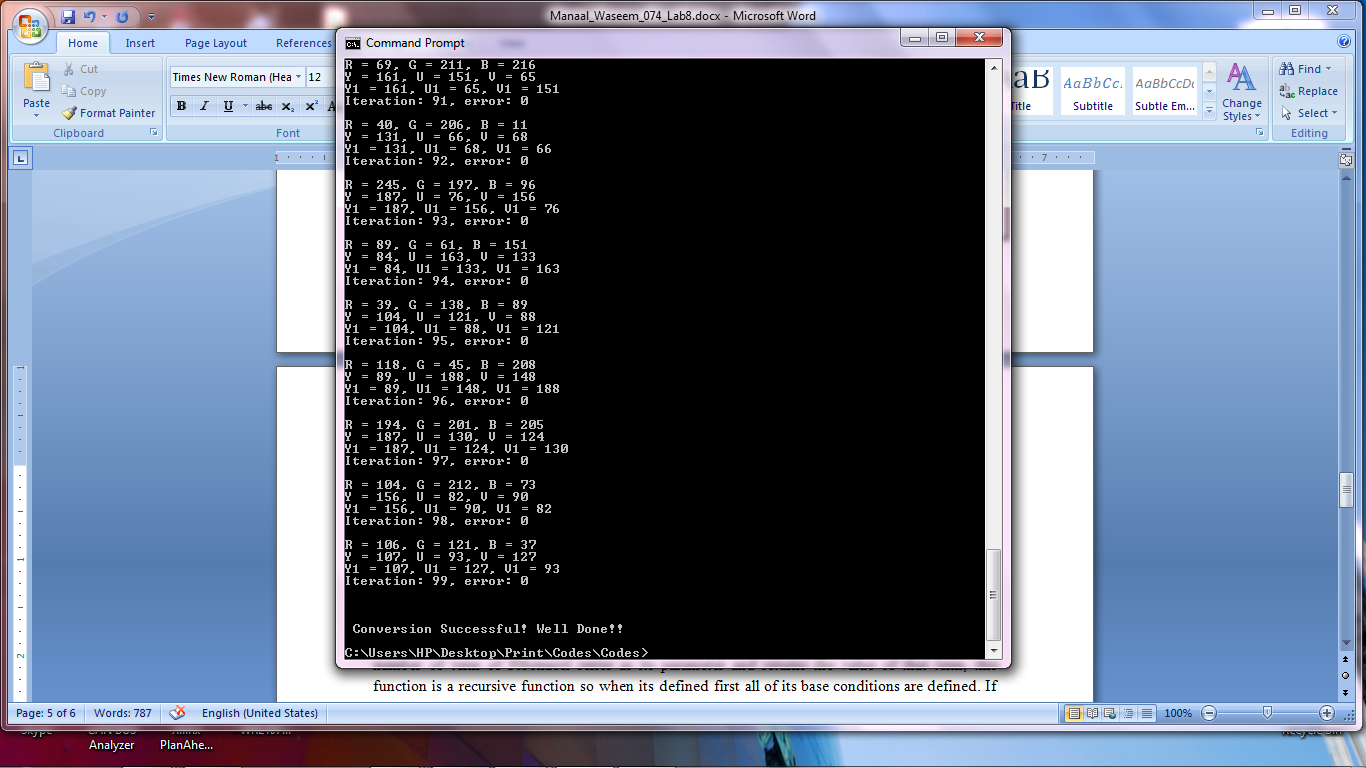
18 \*cb =(**int**)(128 + (-0.148\*r-0.291\*g+0.439\*b));

19 \*cr =(**int**)(128 + (0.439\*r-0.368\*g-0.071\*b));

20 }

21

**Output:**



**Post Lab:**

**Write a recursive version of the Fibonacci function developed in class.**

**Program:** In this program, first of all a function **“fibonacci\_recursive”** is declared that take the number of term of Fibonacci series as its parameter and returns the value of that term, this function is a recursive function so when its defined first all of its base conditions are defined. If no base condition is matched then function call itself recursively as shown in the code. In main function first the variables are declared. A statement is printed to input value on command prompt. Next instruction gets the number of term of Fibonacci series from the user as input on the Command Prompt. Then the declared function is called and hence the required term is printed.

1 #include <stdio.h>

2 #include <stdlib.h>

3

4 **int** fibonacci\_recursive(**int** n);

5 **int** main()

6 {

7 **int** num;

8

9 printf("Enter a number: ");

10 scanf("%d",&num);

11

12

13 printf("\nThe nth term of fibonacci series is %d.",fibonacci\_recursive(num));

14

15 **return** 0;

16 }

17 **int** fibonacci\_recursive(**int** n)

18 {

19 **int** fibonacci;

20

21 **if** (n==0)

22 **return** 0;

23 **else**

24 **if** (n==1)

25 **return** 1;

26 **else**

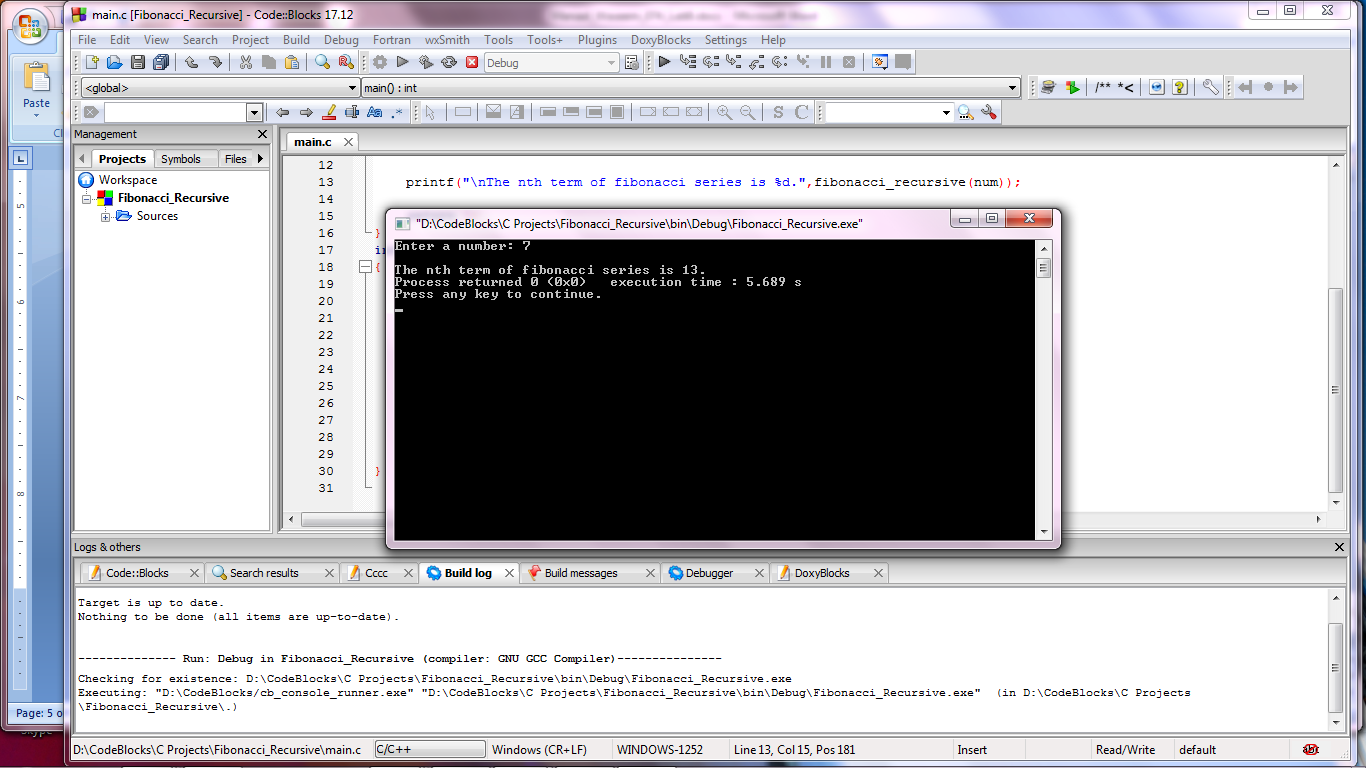
27 fibonacci=fibonacci\_recursive(n-1) + fibonacci\_recursive(n-2);

28

29 **return** fibonacci;

30 }

**Output:**

****

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**THE END**