Assignment 5

Student Name			
Year/Group	2009/2010		
Assignment title:	**Introduction to functions**		
Unit title:	Computer Skills II (C Language)	**Subject Tutor:**	mhd. Mazen al-Mustafa
Start:		**Coordinator:** mhd. Mazen al-Mustafa	
Submission:			
Actual Submission date:		**Grading.**	
Learning Outcomes covered:			
	• **Function**		
Resources:			
	• C How To Program fifth Edition, by H.M. Deitel & P.J. Deitel, • Teacher's handouts • internet		

True/False

1.	A local variable of a function is not visible in any other functions.	[True]
2.	A global variable can be visible in any function.	[True]
3.	Every function prototype must include at least one formal parameter.	[False]
4.	A function call is a valid statement.	[True]
5.	A user-defined function can call library functions or user-defined functions.	[True]
6.	An expression cannot be used as an actual argument in a call to a function.	[False]
7.	Every function must return value	[False]
8.	Every function must declare at least one argument between its parentheses	[False]
9.	The *main()* function can call another function without its parameters	[False]
10.	A user-defined function can call any other function except *main()* function	[True]

Correct the Code

1. In a function that receives a value from the main function via a parameter and then displays the parameter value on the screen, that parameter is considered ___________.
   a. an input parameter
   b. an output parameter
   c. an input/output parameter
   d. a local variable
   e. all of the above

2. Which of the following statements is wrong:
   A function **prototype** tells the compiler ______________
   a. The name of the function.
   b. The types of the parameters that the function should receive.
   c. The value returned by the function.
   d. The order in which these parameters will be received
   e. The type of value that returned by the function

3. Which of the types listed below can be the type of the result value returned by a user-defined function?
   a. int
   b. double
   c. char
   d. long
   e. all of the above

4. Which of the following is a correctly written function that increment Number by one, which is returned to the main function?

a.
```
int  Increment(int Number)
{      Number +=1;
}
```
b.
```
void Increment(int Number)
{      Number +=1;
    return Number;
}
```
c.
```
int  Increment(int Number)
{    return Number +=1;
}
```
[✓ d.]
```
int Increment(int Number)
{      Number +=1;
    return Number;
}
```
e.
```
void Increment(int Number)
{      Number +=1;
```

5. What will be the output of the following program when the program is executed?

```
#include "stdio.h"
void ComputeArea(double , double );
int main()
{
      double  x,y,z;
        x=3;
        y=3;
       If (x> y)
         z= x + y;
       else
         z = y-x;
      return 0;
      printf("Z = %d\n", Z );
      }
```

a. 6
b. 3
c. 0
d. Runtime error
e. [✓ No output]

6. What will be displayed when the following program is executed?

```
#include "stdio.h"
void ComputeArea(double , double );
int main()
{
        double lng, wid;
        lng = 6;
        wid = 10;
        ComputeArea(lng, wid);
        return 0;
}

void ComputeArea(double length, double width)
        {double area;
        area = length * width;
        printf( "%.1f\n",area);
    }
```

a. 0.0

b. 10.0

c. 16.0

d. 60.0

e. No output

Short Answer

1. The region of a program where an identifier can be referenced is called the [scope] of the identifier.

2. What is displayed by the program that follows?

```
#include <stdio.h>
Void bizarre(int n)
{
        printf("%4d ", n);
}
int main(void)
{
        double x;

        x = 35.8;
        bizarre(x);
        printf("%.2f\n", x);

        return (0); }
```

[Answer: 35 35.80]

3. What is displayed by the program defined below?

```
#include <stdio.h>

Double ad1(double x)
{
        return (x + 1);
}

Double trpl(double x)
{
        return (3 * x);
}

Double hlf(double x);
{
        return 0.5 * x;
}

int
main()
{
        printf("%.3f\n", hlf(trpl(ad1(8.2))));
        return 0;
}
```

[Answer: 13.800]

4. Define a function named always_five that has no parameters and returns the integer 5 as its result.

[Answer:

```
int always_five()
{
        return 5;
}
```
]

5. Correct the following program to display a character input by the user.

```
#include  <stdio.h>
void f4(int);
int   main()
{
    char  mychar;
    printf ("enter a character\n");
    scanf("%c", mychar);
    f4 (mychar);
    return 0;
}
void f4 (char  c)
{
    printf("you just entered the character: %c \n", mychar);
    return   mychar;}
```

Write programs

1. Use switch case and Function to write a program that calculates shapes areas. (Circle, Rectangle, Triangle):
   1) Prompt the user to: (enter the appropriate Number) as in step3
   2) Read the Number
   3) Use ***switch*** statement to choose the shape of your choice:
      a) 1 : mean a circle
      b) 2 : mean a rectangle
      c) 3 : mean a triangle
      d) The default case is: prompt the user :(Error: invalid character of choice).

<u>Switch statement:</u>

- i. Case your choice a circle area (1) :
  1. Prompt the user to :(enter the radius)
  2. Read the radius
  3. Call the appropriate function
  4. Print the area
- ii. Case your choice a rectangle area (2) :
  1. Prompt the user to :(enter the width)
  2. Read the width
  3. Prompt the user to :(enter the height)
  4. Read the height
  5. Call the appropriate function
  6. Print the area
- iii. Case your choice a triangle area (3) :
  1. Prompt the user to :(enter the base)
  2. Read the base
  3. Prompt the user to :(enter the height)
  4. Read the height
  5. Call the appropriate function
  6. Print the area

<u>Note:</u>

Write three functions:

float circle (float);	to calculate area of circle (pi*r*r) **where** pi=3.14
int rectangle(int,int);	to calculate area of rectangle (width*height)
float triangle(int,int);	to calculate area of triangle ((base*height)/2)

2. Write a function named "sum_from_to" that takes two integer arguments, call them "first" and "last", and returns as its value the sum of all the integers between first and last inclusive. Thus, for example,
printf("%d\n",sum_from_to(4,7)); // will print 22 because 4+5+6+7 = 22
printf("%d\n",sum_from_to(-3,1));// will print -5 'cause (-3)+(-2)+(-1)+0+1 = -5
printf("%d\n",sum_from_to(7,6)); // will print 22 because 7+6+5+4 = 22
printf("%d\n",sum_from_to(9,9)); // will print 9

3. Write a function named "enough" that takes one integer argument, call it "goal" and returns as its value the smallest positive integer n for which 1+2+3+. . . +n is at least equal to goal . Thus, for example,
```
printf("%d\n", enough(9));     // will print 4 because 1+2+3+4 _ 9 but 1+2+3<9
printf("%d\n", enough(21));   // will print 6 'cause 1+2+ . . .+6 _ 21 but 1+2+ . . . 5<21
printf("%d\n", enough(-7));   // will print 1 because 1 _ -7 and 1 is the smallest
                                       // positive integer
printf("%d\n", enough(1));     // will print 1 because 1 _ 1 and 1 is the smallest
                                       // positive integer
```

DEFINITION: A positive integer d is called a ***divisor*** of an integer n if and only if the remainder after n is divided by d is zero. In this case we also say that " d divides n ", or that " n is divisible by d ". Here are some examples:

-> 7 is a divisor of 35 ; that is, 35 is divisible by 7 .

-> 7 is a not a divisor of 27 ; that is, 27 is not divisible by 7 .

-> 1 is a divisor of 19 ; that is, 19 is divisible by 1 (in fact, 1 is a divisor of every integer n).

-> 12 is a divisor of 0 ; that is, 0 is divisible by 12 .

In C can test the expression n % d to determine whether d is a divisor of n .
The ***greatest common divisor*** of a pair of integers m and n (not both zero) is the largest positive integer that is a divisor of both m and n. We sometimes use the abbreviation "gcd." for "greatest common divisor."
Here are some examples: 10 is the gcd. of 40 and 50; 12 is the gcd. of 84 and -132 ; 1 is the gcd. of 256 and 625 ; 6 is the gcd. of 6 and 42 ; 32 is the gcd. of 0 and 32 .

4. Write a function named "g_c_d" that takes two positive integer arguments and returns as its value the greatest common divisor of those two integers. If the function is passed an argument that is not positive (i.e.,greater than zero), then the function should return the value 0 as a sentinel value to indicate that an error occurred. Thus, for example,
printf("%d\n",g_c_d(40,50)); // will print 10

```
printf("%d\n",g_c_d(256,625)); // will print 1
printf("%d\n",g_c_d(42,6)); // will print 6
printf("%d\n",g_c_d(0,32)); // will print 0 (even though 32 is the g.c.d.)
printf("%d\n",g_c_d(10,-6)); // will print 0 (even though 2 is the g.c.d.)
```

5. A positive integer n is said to be prime (or, "a prime") if and only if n is greater than 1 and is divisible only by 1 and n. For example, the integers 17 and 29 are prime, but 1 and 38 are not prime. Write a function named "is_prime" that takes a positive integer argument and returns as its value the integer 1 if the argument is prime and returns the integer 0 otherwise. Thus, for example,

```
printf("%d\n",is_prime(19)); // will print 1
printf("%d\n",is_prime(1)); // will print 0
printf("%d\n",is_prime(51)); // will print 0
printf("%d\n",is_prime(-13)); // will print 0
```

6. Write a function named "digit_name" that takes an integer argument in the range from 1 to 9, inclusive, and prints the English name for that integer on the computer screen. No newline character should be sent to the screen following the digit name.
   The function should not return a value. The cursor should remain on the same line as the name that has been printed. If the argument is not in the required range, then the function should print "digit error" without the quotation marks but followed by the newline character. Thus, for example, the statement digit_name(7); should print seven on the screen; the statement digit_name(0); should print digit error on the screen and place the cursor at the beginning of the next line.

7. Write a function named "reduce" that takes two positive integer arguments, call them "num" and "denom", treats them as the numerator and denominator of a fraction, and reduces the fraction. That is to say, each of the two arguments will be modified by dividing it by the greatest common divisor of the two integers.

Student Declaration: I certify that the work contained in this assignment was researched and prepared by me:

Signature: ____ Date: / /

Remark: separate feedback sheet will be returned to you after your work has been marked

Feedback Sheet

Student's Notes:
Assistant's Feed Back:

Assistant: ________________________________

Signature: ________________________________ Date: ___________________________