Lab Sheet -1

19CSE102, Computer Programming

Common for Second Semester B.Tech ECE, EAC, EEE, ELC, MEE and CSE, 2019 Batch

Write the programs of the questions 7, 8, 10, 12, 13, 14, 15 in the lab record.

---

1. Writing, compiling and executing your first C Program

   To write a C program, enter the following command in the terminal. This command creates a new file with name ***first.c*** and opens it in a text editor called ***gedit.***

   ```
   gedit first.c
   ```

   Write the following code in the file.

   ```
   #include<stdio.h>
   int main()
   {
       printf(“Hello World”);
       return 0;
   }
   ```

   Next you need to compile your C code. To do this, type in the following command in the terminal:

   ```
   gcc first.c
   ```

   The compiled program will be saved in a new file called **a.out**. Note that "a.out" is the default output file name for executables created by **gcc** compilers and linkers when no output name is specified.

   To run the executable, type the following command in the terminal.

   ```
   ./a.out
   ```

2. Escape sequence in a programming language is a sequence of two or more characters that has a special meaning. For example, n is an escape sequence in C language that prints a new line on the output screen.

   Write the following code in the file.

   ```
   #include<stdio.h>
   int main()
   {
       printf(“Hello \n World”);
       return 0;
   }
   ```

   The above code prints Hello on a line and World on a new line on the output screen.

3. The format string in a printf statement can contain several *escape sequences* besides \n. The complete list is:

Sequence	Meaning
\a	Alert (bell)
\b	Backspace
\f	Formfeed
\n	Newline
\r	Carriage return
\t	Horizontal tab
\v	Vertical tab
\\	Backslash
\?	Question mark
\’	Single quote
\”	Double quote

Write a program with printf () statements containing several of these escape sequences.

4. The printf () function returns the number of characters it has printed on the screen. This return value may be ignored. Execute the following program that records the return value of first printf () statement in the integer variable c, and then prints out the value of the variable.

```
int c;
c= printf(“123abc”);
printf(“The number of characters printed = %d”,c);
```

Add a newline to the string to be printed and run the program again.

5. Format specifiers are special sequence in C that are used to specify the type of data in an input or output operation. For example, to print an integer data, we write the printf statement as follows.
printf(“This is an integer %d”,10);
The sequence **%d** is the format specifier of integers.

Here is a list of format specifiers in C.

Format Specifier	**Type**
%c	Character
%d	Signed integer
%e or %E	Scientific notation of floats

Format Specifier	**Type**
%f	Float values
%g or %G	Similar as %e or %E
%hi	Signed integer (short)
%hu	Unsigned Integer (short)
%i	Unsigned integer
%l or %ld or %li	Long
%lf	Double
%Lf	Long double
%lu	Unsigned int or unsigned long
%lli or %lld	Long long
%llu	Unsigned long long
%o	Octal representation
%p	Pointer
%s	String
%u	Unsigned int
%x or %X	Hexadecimal representation
%n	Prints nothing
%%	Prints % character

6. ***scanf*** is a function used in C to read formatted input. For example, to read an integer from the keyboard we write the following code.

```
int x;
scanf (“%d”, &x);
```

   where %d is the format specifier for integer. The scanf function reads the integer entered in the keyboard and stores it in the variable x.

   Note that in C, all variables must be declared before use. (Unlike Python).

7. Write a c program to read an integer, a character and a float using ***scanf*** and display them using ***printf***.
8. Write a C program to read two integers and print their sum, difference, product, quotient, remainder using various arithmetic operators.
9. The following C program finds the memory allocated for variables of various type. The operator *sizeof()* is used to find the memory-size of variables.

```
#include <stdio.h>
int main()
{
    int a;
    float b;
    double c;
    char d;
    short e;
    long int f;
    long long int g;
    printf("Size of int: %ld bytes\n",sizeof(a));
    printf("Size of long int: %ld byte\n",sizeof(f));
    printf("Size of long long int: %ld byte\n",sizeof(g));
    printf("Size of float: %ld bytes\n",sizeof(b));
    printf("Size of double: %ld bytes\n",sizeof(c));
    printf("Size of char: %ld byte\n",sizeof(d));
    printf("Size of short: %ld byte\n",sizeof(e));

  return 0;
}
```

10. Type in, compile, and run the following program.

```
    #include <stdio.h>
    int main( )
    {
        float   rupees, dollars, exchangerate=50.0 ;
        printf(“Enter dollars:   “)  ;
        scanf (“%f ” , &dollars) ;
        rupees  =  dollars * exchangerate;
        printf ( “%f  dollars  = %f  rupees. \n”, dollars, rupees);
    }
```

    Modify the exchange rate program to read the current exchange rate from the user.

11. scanf() returns the number of variables to which it assigns a value. The return value can be used to check whether or not scanf() was successful. For example, if scanf() is trying to read an integer but the user types in letters, scanf() will return 0 to indicate that it failed. Write a program which reads in an integer and then prints out the integer it reads as well as the return value of scanf(). Run the program twice, once entering a valid integer and once entering alphabetic characters.

12. Write a program which reads in an amount of seconds from the user and outputs the equivalent amount of minutes and the equivalent number of hours. Use preprocessor constants (i.e #define) for the conversion factors. Use integer variables to store the seconds, minutes, and hours. When you run the program, enter 5000 seconds. Notice how the division of one integer by another results in an integer (i.e. the remainder of the division is lost).

13. Modify the time conversion program to use float variables for the number of seconds, minutes and hours. Be sure to modify the printf() and scanf() statements. Print out the results to two decimal places.

14. Write a program which reads the radius of a circle and computes its area and perimeter. use **#define** to make PI a constant.

15. Write a program to calculate the distance between two points. Make use of ***math.h*** header file to use the ***sqrt()*** and ***pow()*** functions.