BIRLA INSTTUTE OF TECHNOLOGY & SCIENCE, PILANI (RAJ.)
CS F111 Computer Programming

LAB SESSION #4

(More C Programming and usage of <math.h> library)

Create a directory "**lab4**" inside "**myprogs**" directory that you have created in the last week's lab session. This week, we will write a bit more advanced C programs. We will also learn how to use <math.h> library for evaluating various mathematical expressions. You shall need to use the basics that were covered in the previous lab session. I hope that you are now familiar with the following arithmetic operations:

Addition:
```
num3 = num1 + num2
```

Subtraction:
```
num3 = num1 - num2
```

Multiplication:
```
num3 = num1 + num2
```

Integer Division:
```
num3 = num1 / num2
```

Modulus (or remainder after division)
```
num3 = num1 % num2
```

Let us write a few more C programs using the fundamentals covered in the previous lab session.

1. Write a C program that reads in an integer denoting number of days. It prints the number of years, number of months and the number of days that constitute the input number of days. For example, if the input number is 403, it should print 1(year), 1(month), 13(days). For simplicity: there is no need to consider leap years and assume all months have 30 days. [**Hint:** Use modulus (**%**) and division (**/**) operators. **End of Hint**]

2. In this question, you will learn more about data types and qualifiers. Copy the following program in "**lab4_sizes.c**", and then compile and execute it.

```
#include<stdio.h>
int main()
{
float f;
printf("Sizeof (char) = %lu bytes\n", sizeof(char)); // datatype
printf("Sizeof (short)= %lubytes\n", sizeof(short));
printf("Sizeof (int)= %lu bytes\n", sizeof(int));
printf("Sizeof (long)= %lu bytes\n", sizeof(long));
printf("Sizeof (float)= %lu bytes\n", sizeof(f)); // variable
printf("Sizeof (double)= %lu bytes\n", sizeof(double));
printf("Sizeof (1.55)= %lu bytes\n", sizeof(1.55)); // constant
printf("Sizeof (1.55L)= %lu bytes\n", sizeof(1.55L));
printf("Sizeof (str)= %lu bytes\n", sizeof("Hello")); // string
return 0;
}
```

a. Find out the sizes of data types when prefixed with the keywords signed and unsigned.
b. Now, try various combinations of qualifiers (**short and long**) with the keywords **unsigned** and **signed** keywords. Try which of these the compiler accepts, and which are not. For example, **long unsigned int** is valid, whereas **long unsigned double** is not.

3. Write a C program, **"swap.c"**, to swap the values of two integer numbers `a` and `b` entered by the user, and display the new numbers to the user. Do this with and without using a third variable.

4. A computer manufacturing company has the following monthly compensation policy to their salespersons:

Minimum base salary : 1500.00
Bonus for every computer sold : 200.00
Commission on the total monthly sales : 2 per cent

Since the prices of computers are changing, the sale price of each computer is fixed at the beginning of every month. Write a C program, "**computer_sales.c**", to compute a sales-person's bonus, commission and gross salary. Your program should take the number of computers sold (in a month) and the sale price of a computer as user input.

5. Write a C program, "**ascii_test.c**", which takes two character as input and returns the sum of their ASCII as output. For instance, input is A and B, the output should be 131 (sum of the ASCII of A and B). [**Hint:** Use explicit typecast to covert character to integer values. **End of Hint**]

6. Explain why the following code prints the largest integral value on your system:

```
unsigned long long val = -1;
printf("The biggest integer value: %llu\n", val);
```

Note the conversion specifier `u` (for unsigned integers) preceded by the length modifier `ll` (ell-ell) used to print the value stored in val. Explore other length modifiers and conversion specifiers by reading the online manual for `printf()`, by typing the following command at the Linux shell prompt: `man 3 printf`.

(Contd. in the next page…)

Using <math.h> library

C supports extensive mathematics operations using various functions implemented in a library called <math.h>. We will learn how to use this library for evaluating various mathematical expressions. Your program should start with importing this library using #include <math.h>. And then we can call the functions that are available in it. A sample program is shown below:

myFirstMathProg.c

```
#include <stdio.h>
#include <math.h>

int main()
{
      int a = 4;
      double b = sqrt((double)a);
      /* Computes square root of "a" after converting it into a
      double variable */
      printf("Printing the value of square root of a: %lf \n", b);
      double angle = 2.3; // angle in radians
      double c = sin(angle);
      printf("Sin %lf is: %lf \n", angle, c);
      return 0;
}
```

This program computes square root of a number using sqrt() function and sin value of an angle in radians using sin() function. Both these functions are available in math.h library. Both of them take double values as input and return double values as output which is to be captured in a separate variable. We will study more about *functions* in subsequent classes and lab sessions.

One thing to be noted here is that while compiling any C program that uses math.h library, you must add -lm flag as shown below:

```
gcc myFirstMathProg.c -o math_exe -lm
./math_exe // runs the program
```

Without this flag, the compiler will throw an error. Now it is an exercise for you to figure out what is -lm and why do you need it. Please take help of GOOGLE!

The complete list of functions available in math.h library is as follows:

Function	Description
floor( )	This function returns the nearest integer which is less than or equal to the argument passed to this function.
round( )	This function returns the nearest integer value of the float/double/long double argument passed to this function. If decimal value is from ".1 to .5", it returns integer value less than the argument. If decimal value is from ".6 to .9", it returns the integer value greater than the argument.
ceil( )	This function returns nearest integer value which is greater than or equal to the argument passed to this function.

sin( )	This function is used to calculate sine value.
cos( )	This function is used to calculate cosine.
cosh( )	This function is used to calculate hyperbolic cosine.
exp( )	This function is used to calculate the exponential "e" to the x^{th} power.
tan( )	This function is used to calculate tangent.
tanh( )	This function is used to calculate hyperbolic tangent.
sinh( )	This function is used to calculate hyperbolic sine.
log( )	This function is used to calculates natural logarithm.
log10( )	This function is used to calculates base 10 logarithm.
sqrt( )	This function is used to find square root of the argument passed to this function.
pow( )	This is used to find the power of the given number.
trunc( )	This function truncates the decimal value from floating point value and returns integer value.

Now, let us try to write some programs using `math.h` library.

7. Write a C program in "**math_ops.c"** that evaluates and prints the values of the following arithmetic expressions. Here *x* and *y* are floating point numbers to be taken as input from the user. The value of *pi* is 3.142. Use *exp*, *sin*, *cos* and *tan* functions from `math.h` library. It is now your task to figure out how to use the above functions, what is their syntax. Take help of GOOGLE!

$$\text{expr1} = \frac{e^x \sin 60^\circ + 5.6\times10^{-5}}{3\cos 30^\circ}$$

$$\text{expr2} = \sin\left(\frac{\tan^{-1} 0.33 + \pi}{2y}\right)$$

8. As you know, the roots x1 and x2 of a quadratic equation $ax^2 + bx + c = 0$ are calculated by:

$$x1 = (-b + \sqrt{(b^2 - 4ac)})/2a,$$

$$x2 = (-b - \sqrt{(b^2 - 4ac)})/2a$$

Write a C program named "**quadroots.c"**, which should take a, b and c as inputs, and output the values of x1 and x2.