BIRLA INSTTUTE OF TECHNOLOGY & SCIENCE, PILANI (RAJ.)

Second Semester 2019-20

CS F111 Computer Programming

LAB SESSION #5

(Number system conversion and IEEE floating point representation)

In this lab, you will practice your programming skills over number systems and their conversions. As a quick recall, a binary number system is used to represent the underlying numbers in base 2, an octal for base 8 and hexadecimal for base 16. The generic number system (we call as decimal number system) is by default represented in base 10. Below, find the example conversions from binary to decimal and vice-versa.

Example 1: *Binary to decimal conversion*. Let us assume the binary number as 10101. The conversion to its equivalent decimal number is evaluated as:

Step 1	1	0	1	0	1	*Bits (A)*
Step 2	2^4	2^3	2^2	2^1	2^0	*Weights (B)*
Step 3	16	0	4	0	1	*Weighted values (A) × (B)*
Step 4	Finally add all the weighted values as: 16+0+4+0+1= 21					

Thus, $(10101)_2 = (21)_{10}$

Example 2: *Decimal to binary conversion*. For reverse process, i.e., to get the binary equivalent of a decimal number, the following steps need to executed.

Step1: Divide the decimal number *N* by 2.

- the remainder becomes the Least Significant Bit (LSB)

Step 2: Divide the quotient by 2 (if it is not 0).

- the remainder becomes the next bit to the left

Step 3: Repeat step 2 until the quotient becomes 0.

Thus, for 21:

1. 21/2, remainder is **1**
2. 10/2, remainder is **0**
3. 5/2, remainder is **1**
4. 2/2, remainder is **0**
5. ½, remainder is **1**

Write the remainders in their increasing order as **10101**, which is the binary equivalent of **21**.

Note: (i) For decimal to octal/hexadecimal conversion, simply replace the base 2 with base 8/base 16 in example 1. (ii) For octal/hexadecimal to decimal conversion, follow the same procedure as mentioned in example 2 by replacing base 2 with base 8/base 16.

Questions:

1. The binary addition is performed to add two or more binary numbers. It is performed the similar way we used to add two decimal numbers. But, as the binary represents base 2, we cannot expect the resulting addition value to be greater than 1. In this case, simply take one carry, and follow the similar binary operation as of the decimal numbers.

   Example: Let the binary numbers 110011 and 10011100 needs to be added. The procedure consists of the following:

   - Right aligning the numbers first, and
   - Adding the two corresponding bit and a carry-in bit (if any)

```
      1 1 0 0 1 1
+   1 0 0 1 1 1 0 0
----------------------
    1 1 0 0 1 1 1 1
----------------------
```

   Write an algorithm to take two binary numbers b_1 and b_2 as the input, and output the decimal value of their sum. Write separate methodologies for the following conditions:

   a) When the input binary numbers are of the equal length
   b) When the input binary numbers are of unequal length

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

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

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

   Your program, named lab5_quadroots.c should take a, b and c as inputs, and output the values of x1 and x2. For calculating the square root, you can use the function called sqrt(), the declaration of which is in the header file <math.h> that you must #include, just as you do <stdio.h>.

   How to use this function in a C program? To find out the square root of x and store the result in y, you will write the following C code:

```
int x, y;

y = sqrt(x);
```

   Using this information, write a well-commented C program to output x1 and x2 (the roots). At this point of time, you can neglect any error checks in your code.

   Now compile the file in gcc lab5_quadroots.c as follows: **gcc lab5_quadroots.c –lm** (note the use of –lm for linking the math library).

   Test your program with the following cases, and record their answers in your notebook:

   a) a = 3, b = 5, c = 2
   b) a = 3, b = –6, c = 3
   c) a = 2, b = 1, c = 5

3. Given two decimal numbers as $x = 21$ and $y = -17$. Transform them into their equivalent 2's complement representation and evaluate their sum D_1.

    a. Now, for the same set of values i.e. $x = 21$ and $y = -17$, evaluate their equivalent 1's complement and find their sum D_2. Did you find any difference between D_1 and D_2?

Additional Exercises

4. Write a program to find the decimal equivalent value of addition of two signed binary numbers 10111 and 01100, represented in 2's complement form.

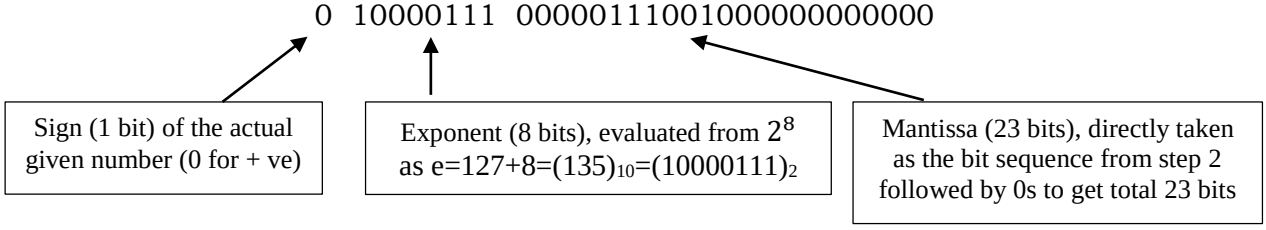
5. Given a number as 263.125, its conversion to IEEE 754 floating point representation is performed by the following steps:

    Step 1: Convert the given number 263.125 to its equivalent binary form, i.e., 100000111.001.

    Step 2: Shift the decimal (.) to the left by keeping only one bit before that. Also multiply the resulting binary representation with 2^k, where k is the number of bits shifted. The resulting conversion will become:

    $$1.00000111001 \times 2^8$$

    Step 3: Represent the given number in IEEE 754 floating point representation as follows:

    Write a program to find the IEEE 754 floating point representation of a given number x = 263.125. Your program should first print the binary conversion of x, and then the complete representation in IEEE 754 floating point format.

6. Given a decimal number as 5002. Write a C program to find its equivalent binary, octal and hexadecimal representation and print them on your screen. Also, evaluate -5002 in 1's and 2's complement representation and print them on your screen.

7. Given a hexadecimal number as 40400000. Write a program to find its equivalent IEEE 754 floating point representation in decimal form.