Recitation - 1

Solutions

Problem 1:

a. $(7)_{10} = (111)_2$

$(12)_{10} = (1100)_2$

$(19)_{10} = (10011)_2$

b. $(1101)_2 = (13)_{10}$

$(011001)_2 = (25)_{10}$

$(1111)_2 = (15)_{10}$

c. $(7)_{10} = (000111)_2$

$(12)_{10} = (001100)_2$

$(19)_{10} = (010011)_2$

For d, we are not considering a signed representation. We are just converting the given 2's compliment binary string into decimal.

d. $(000111)_2 \rightarrow (111001) \rightarrow (57)_{10}$

$(001100)_2 \rightarrow (110100) \rightarrow (52)_{10}$

$(010011)_2 \rightarrow (101101) \rightarrow (45)_{10}$

e. $(.75)_{10} \rightarrow (.11000)_2$

$(.3)_{10} \rightarrow (.01001)_2$

f $(.11000)_2 \rightarrow (.75)_{10}$

$(.01001)_2 \rightarrow (0.28125)_{10}$ -- Note the relative error in the representation.

g $(7)_{10} = (00111)_2$

$(3)_{10} = (00011)_2$

$(00111)_2 + (00011)_2 = (01010)_2$

$(01010)_2 = (10)_{10}$

Part-2

$(8)_{10} = (01000)_2$

$(00111)_2 + (01000)_2 = (01111)_2 = (15)_{10} = (7 + 8)_{10}$

It is good to note here that the result (01111) is the highest positive value. If the value was greater by at least 1, the output will overflow into the sign bit and make the number negative. for eg. if we add binary string of 8 and 8 in a 5 bit signed representation we get an overflow.

Problem-2

The programmer interprets the problem and puts down the steps (the algorithm) to solve it. The programmer will usually use a programming language to 'encode' this algorithm.

The program tells the computer how to execute this step by step solution and based on inputs executes a specific instance of the algorithm and outputs a specific value or response.

Problem-3

```
import java.util.Scanner;

public class SumDivideSample {

	int x, y;

	public void getInput() {
		Scanner sin = new Scanner(System.in);
		x = sin.nextInt();
		y = sin.nextInt();
	}

	public void computeAndPrint() {
		System.out.println("x + y = " + (x + y));
		System.out.println("x/y quotient = " + x / y + " remainder = "
				+ (x % y));
	}

	public static void main(String args[]) {
		SumDivideSample obj = new SumDivideSample();
		obj.getInput();
		obj.computeAndPrint();
	}
}
```