COE 301/ICS 233, Term 161
Computer Architecture & Assembly Language
HW# 3

Q.1. Write a MIPS assembly program to perform **signed division** of 32-bit numbers using the algorithm studied in class. The program should ask the user to inter two integers and then display the result of division displaying both the quotient and remainder. Test your program using the following numbers:

1. +17 ÷ +3
2. +17 ÷ -3
3. -17 ÷ +3
4. -17 ÷ -3

A sample execution of the program is shown below:

Enter the dividend: 17
Enter the divisor: -3
Result of division: Quotient = -5 Remainder = 2

Q.2. Write a procedure, **GCD**, that receives two positive numbers in \$a0 and \$a1 and returns their greatest common divisor in register \$v0. It is required that the procedure **preserves the content of all used registers** according to the MIPS programming convention by saving them and restoring them on the stack. The pseudo code of the GCD procedure is given below:

```
int gcd(int m, int n) {
   if ((m % n) == 0)
      return n;
   else
      return gcd(n, m % n);
}
```

Q.3.

(i) Given that **Multiplicand=1010** and **Multiplier=1011,** using **signed multiplication**, show the **signed** multiplication of **Multiplicand** by **Multiplier**. The result of the multiplication should be an 8 bit **signed** number in HI and LO registers. Show the steps of your work.

Iteration		**Multiplicand**	**Sign**	**Product = HI,LO**
0	Initialize			
1				
2				
3				
4				

(ii) Given that **Dividend=1011** and **Divisor=0010**, Using **unsigned division**, show the **unsigned** division of **Dividend** by **Divisor**. The result of division should be stored in the Remainder and Quotient registers. Show the steps of your work.

Iteration		**Remainder (HI)**	**Quotient (LO)**	**Divisor**	**Difference**
0	Initialize				
1					
2					
3					
4					

Q.4. What is the decimal value of the following single-precision floating-point numbers?

(i) `0010 0000 0001 1100 0000 0000 0000 0000`

(ii) `1100 1111 1110 1000 0000 0000 0000 0000`

Q.5. Show the IEEE 754 binary representation for: `−24.0625` in

(i) Single Precision

(ii) Double precision

Q.6. Perform the following floating-point operations rounding the result to the nearest even. Perform the operation assuming both infinite precision and using only guard, round and sticky bits. Compare your solution in both cases.

(i) `0100 0011 1000 0000 0000 0000 0000 0000`

`- 0100 0001 1000 0000 0000 0000 0000 1100`

(ii) `0011 1111 1000 0000 0000 0000 0000 0000`

`- 0011 1111 1000 0100 0000 0000 0000 0000`

(iii) `0011 1111 1111 1111 1111 1111 1111 1110`

`+0011 0100 0100 0000 0000 0000 0000 0000`

Q.7. Given that x = `0101 1111 1011 1110 0100 0000 0000 0000`
y = `0011 1111 1111 1000 0000 0000 0000 0000`
z = `1101 1111 1011 1110 0100 0000 0000 0000`
represent single precision IEEE 754 floating-point numbers. Perform the following operations showing all work:

(i) $x + y$

(ii) Result of (**i**) + z

(iii) Why is the result of (**ii**) counterintuitive?