Computer Organization
And
Assembly Language

```
00000000: 01001101 01011010 10010000 00000000 00000011 00000000  MZ····
00000006: 00000000 00000000 00000100 00000000 00000000 00000000  ······
0000000c: 11111111 11111111 00000000 00000000 10111000 00000000  ······
00000012: 00000000 00000000 00000000 00000000 00000000 00000000  ······
00000018: 01000000 00000000 00000000 00000000 00000000 00000000  @·····
0000001e: 00000000 00000000 00000000 00000000 00000000 00000000  ······
00000024: 00000000 00000000 00000000 00000000 00000000 00000000  ······
0000002a: 00000000 00000000 00000000 00000000 00000000 00000000  ······
00000030: 00000000 00000000 00000000 00000000 00000000 00000000  ······
00000036: 00000000 00000000 00000000 00000000 00000000 00000000  ······
0000003c: 10000000 00000000 00000000 00000000 00001110 00011111  ······
00000042: 10111010 00001110 00000000 10110100 00001001 11001101  ······
00000048: 00100001 10111000 00000001 01001100 11001101 00100001  !··L·!
0000004e: 01010100 01101000 01101001 01110011 00100000 01110000  This p
00000054: 01110010 01101111 01100111 01110010 01100001 01101101  rogram
0000005a: 00100000 01100011 01100001 01101110 01101110 01101111   canno
00000060: 01110100 00100000 01100010 01100101 00100000 01110010  t be r
00000066: 01110101 01101110 00100000 01101001 01101110 00100000  un in 
0000006c: 01000100 01001111 01010011 00100000 01101101 01101111  DOS mo
00000072: 01100100 01100101 00101110 00001101 00001101 00001010  de.···
00000078: 00100100 00000000 00000000 00000000 00000000 00000000  $·····
0000007e: 00000000 00000000 01010000 01000101 00000000 00000000  ··PE··
```

Lab Manual 06

Objectives:

1. To learn how can we pass arguments to the procedures(advanced)
2. To learn using logic instructions
3. To learn shift and rotate instructions
4. To learn how to use logic and shift/rotate instructions to print data in hex and binary format

Lab Instructor:

Mr. Tariq Mehmood Butt tariq.butt@pucit.edu.pk

Teaching Assistants:

Hafiz Muhammad Ahmad	bcsf21m502@pucit.edu.pk
Syed Muhammad Zain Raza	bcsf21m510@pucit.edu.pk
Zahra Malik	bcsf21m551@pucit.edu.pk
Bilal	bsdsf21m022@pucit.edu.pk

LAB TASKS

You are required to prepare the ASM file of each task and run the executable file in the debugger. Use maximum number of procedures to do the tasks.

Task 1:

Write a program that will

a) Prompt the user to enter a character
b) On the next line prints its ASCII code in binary
c) The number of 1 bit in its ASCII code.

Sample execution:

Enter a character: **A**
The ASCII code of **A** in binary is 01000001
The number of 1 bit is 2

Task 2:

Write a program that will

a) Prompt the user to enter a character
b) On the next line prints its ASCII code in HEX
c) Repeat this process until the user type a carriage return (Enter button)

Sample execution:

Enter a character: **Z**
The ASCII code of Z in HEX is 5A
Enter a character: **A**
The ASCII code of **A** in HEX is 41
Enter a character:

Task 3:

Write a program that will

a) Prompt the user to enter a hex number of four digits or less, if the user enters an illegal character, he or she should be prompted to begin again. Accept only uppercase letters.

b) On the next line prints it in binary

Your program ignores any input beyond four characters.

Sample execution:

Enter a hex number (0 to FFFF): **1a**
Illegal hex digit, try again: **1ABC** In binary
it is: 0001101010111100

Task 4:

Write a program that will

a) Prompt the user to enter a binary number of 16 digits or less, if the user enters an illegal character, he or she should be prompted to begin again.

b) On the next line prints it in Hex

Your program ignores any input beyond 16 characters.

Sample execution:

Enter a binary number up to 16 digits: **11100001**
In Hex it is E1

Task 5:

Write a program that will

a. Prompt the user to enter 2 binary numbers of 8 digits each, if the user enters an illegal character, he or she should be prompted to begin again.

b. On the next line prints it sum in binary and hex formats

Sample execution:

Sample execution:

```
Enter a binary number up to 8 digits: 11001010

Enter a binary number up to 8 digits: 10011100

The binary sum is: 101100110
The hex sum is: 0166
```

Sample execution: