Name: Laiba Gul Semester: 7th

Sap ID: 55740 Dep: Computer Science

**Lab Task 02**

**Question no: 01**

**Describe function of following**

* **Accumulator Register:**

AC register is used for arithmetic and input or output instructions .It is firstly introduced as 8 bits and then due to higher data, other version is also introduced like

A 8bits

AX 16bits

EAX 32bits

RAX 64bits

* **Base Register:**

In this register address is saved like address of data of RAM

B 8bits

BX 16bits

EBX 32bits

RBX 64bits

* **Count Register:**

In this register two main work is done which is counting and looping.

* **Data Register:**

In this register data of output is stored.

**Question no:02**

**Describe the role of index registers in source and destination operations. Provide an example using SI and DI in an assembly instruction**

Index registers, such as the Source Index (SI) and Destination Index (DI), are crucial in assembly language for efficient data manipulation. They help point to the source and destination operands in memory, facilitating fast access and data transfer

**Question No:03**

1. **Briefly describe the functions of these bits:**

* **Overflow Flag (OF)**
* **Zero Flag (ZF)**

**Carry Flag (CF**

1. **Overflow Flag (OF)**:
   * **Function**: Set if the result is too large a positive number or too small a negative number to fit into the destination operand. It indicates that an arithmetic overflow has occurred.
2. **Zero Flag (ZF)**:
   * **Function**: Set if the result of an operation is zero. This flag indicates that the operation resulted in a zero value, which is often used in conditional statements.
3. **Carry Flag (CF)**:
   * **Function**: Set if there was a carry from or borrow to the most significant bit during the last result calculation. This flag is important for detecting overflow in unsigned arithmetic operations.

**Question No:04**

**Write a simple program using MOV, ADD, and SUB instructions:**

* **Move values to registers**
* **Perform addition and subtraction**
* **Store and display results**

MOV CL, 6 :Move value into CL, CL = 6

MOV BL, 3 : Move value into BL, BL = 3

SUB CL, BL :Subtract BL from CL, CL = 3

MOV BL, 4 : Move value into BL, BL = 4

ADD CL, BL ;: Add BL to CL, CL = 7