# **SUBMITTED BY**:

# MUHAMMAD HAMZA NASIR

### 2016-CS-115

# **SUBMITTED TO**:

# Sir SHEHZAD Ali

# **TABLE OF CONTENTS**

Contents

[**TABLE OF CONTENTS** 1](#_Toc509997183)

[Lab No. 1 4](#_Toc509997184)

[Basic Instructions in Assembly Language: 4](#_Toc509997185)

[**Lab1 Task Description:** 4](#_Toc509997186)

[**Instructions to Perform above Tasks:** 5](#_Toc509997187)

[**Lab No. 2** 6](#_Toc509997188)

[Add Two Numbers using Single Digit: 6](#_Toc509997189)

[**Lab2 Task Description:** 6](#_Toc509997190)

[**Emulator Code to Perform the Task:** 7](#_Toc509997191)

[**Output for Task:** 7](#_Toc509997192)

[**Lab No. 3** 8](#_Toc509997193)

[Sum of Two Numbers using Two Digit: 8](#_Toc509997194)

[**Lab3 Task Description:** 8](#_Toc509997195)

[**Emulator Code to Perform the Task:** 9](#_Toc509997196)

[**Output for Task:** 9](#_Toc509997197)

[**Lab No. 4(a)** 10](#_Toc509997198)

[Sum of Two Numbers Using Functions: 10](#_Toc509997199)

[**Lab4 (a) Task Description:** 10](#_Toc509997200)

[**Code for the Task:** 11](#_Toc509997201)

[**Output:** 12](#_Toc509997202)

[**Lab No. 4(b)** 13](#_Toc509997203)

[Check that Number is Even or Odd: 13](#_Toc509997204)

[**Lab4 (b) Task Description:** 14](#_Toc509997205)

[**Code for the Task:** 15](#_Toc509997206)

[**Output:** 16](#_Toc509997207)

[**Home Task for Lab 4:** 17](#_Toc509997208)

[Convert Uppercase Letter to Lowercase Letter and Vice Versa: 17](#_Toc509997209)

[**Home Task Description:** 17](#_Toc509997210)

[**Code for task:** 18](#_Toc509997211)

[**Output of Task:** 20](#_Toc509997212)

[**Lab No. 5:** 21](#_Toc509997213)

[Sorting of three Numbers: 21](#_Toc509997214)

[**Code for Task:** 22](#_Toc509997215)

[**Output:** 27](#_Toc509997216)

[**Lab No. 6** 28](#_Toc509997217)

[Numbers of Vowels in the String and Calculate it: 28](#_Toc509997218)

[**Lab Task to Be Performed:** 28](#_Toc509997219)

[**Code for Lab:** 29](#_Toc509997220)

[**Output:** 32](#_Toc509997221)

[**Lab 6 (b)** 33](#_Toc509997222)

[Print Numbers with and without Loop: 33](#_Toc509997223)

[**Lab 6 (b) Task Description:** 33](#_Toc509997224)

[**Emulator code to perform the task:** 34](#_Toc509997225)

[**Output:** 34](#_Toc509997226)

# Lab No. 1

## Basic Instructions in Assembly Language:

In the first lab, we talked about how an Assembler works. We downloaded Emulator for the code purpose. It works like an Assembler. We talked about Registers in the first lab. There are many types of Registers.

**Registers** are basically used for the Movement of Data. Some Registers are **General Purpose Registers** .There are four basic General Purpose registers that we have studied in our lab which are divided into 16-bit Registers and these are further Divided into 8-bit Registers. The Description of these Registers are given as:

|  |  |  |  |
| --- | --- | --- | --- |
| **General purpose 32-Bit Registers** | **General Purpose 16-Bit Registers** | **General Purpose 8-Bit Registers** | **General Purpose 8-Bit Registers** |
| EAX | AX | AH | AL |
| EBX | BX | BH | BL |
| ECX | CX | CH | CL |
| EDX | DX | DH | DL |

**AX** is known as Accumulator Register.

**CX** is used in Loop counter.

**BX** is called Base Register.

**DX** is called Data Register.

These registers are simply used for the Movement of Data. We have studied a lot about the registers we used these registers to store some data.

## **Lab1 Task Description:**

1. To get some **Input from the User.**
2. To produce **interrupts.**
3. To **display input** that user has given
4. To Move the cursor on the Next line we use **Line Feed**.
5. To Place the cursor at the Start of line we use **Carriage** **Return.**

## **Instructions to Perform above Tasks:**

* MOV AH, 1

INT 21h

* MOV AH, 2

MOV DL, 0AH

INT 21H

* MOV DL, 0DH

INT 21H

# **Lab No. 2**

## Add Two Numbers using Single Digit:

In the Second lab, We Performed a task i.e. how to get input of two Numbers from the user And Sum these numbers and Display the Sum of the given numbers in the given format. We used Interrupts for the following purpose. We used **Line Feed** and **Carriage Return**. The Sum of the Numbers will be in **Single Digit**. If the Sum is Greater than 9 then the Screen shows the respective **ASCII** symbol for that Number That user have Typed.

**ADD:** It is used to find sum of data in the registers and as well as Variables.

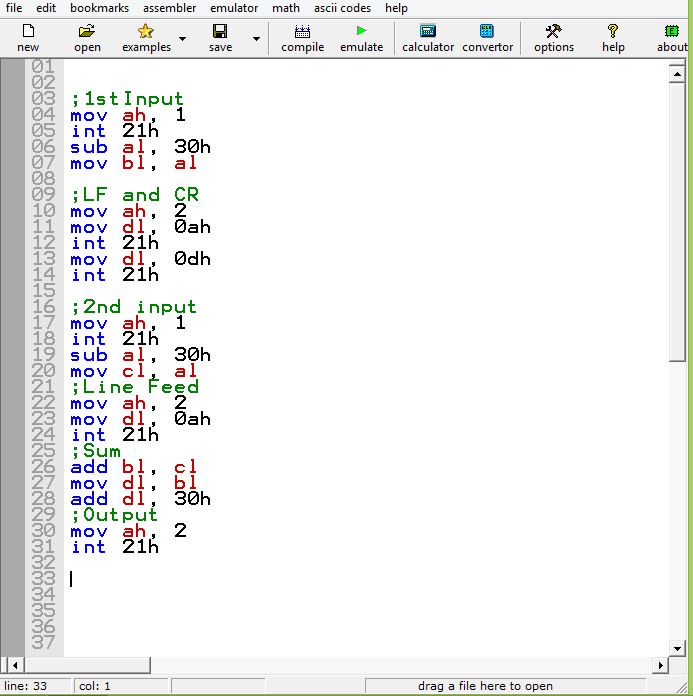
**Format:** ADD Register, Register

**Format:** SUB Register, Register

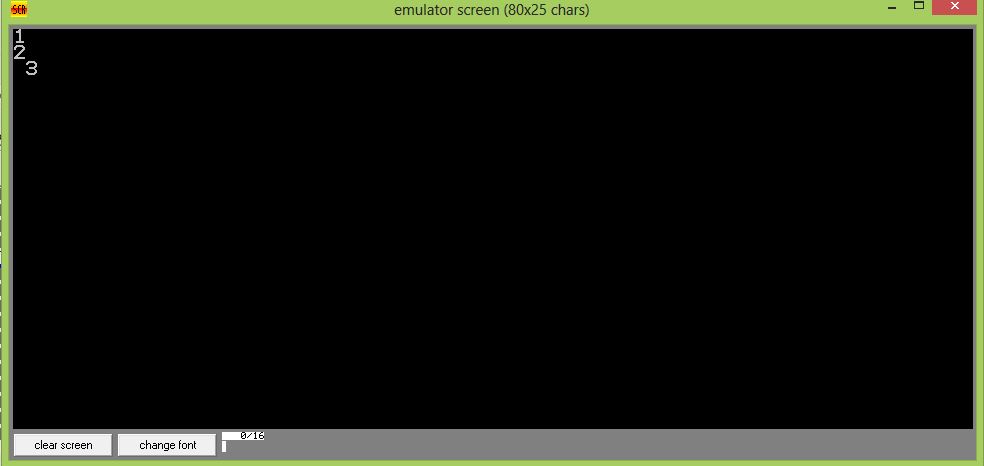
## **Lab2 Task Description:**

1. Take Two Numbers as **Input** from the User.
2. Find **Sum** of these Numbers.
3. **Display the Sum** in the given Format.
4. Sum will be a **Single Digit.**
5. We Displayed Sum from **1 to 9.**

## **Emulator Code to Perform the Task:**



## **Output for Task:**



# **Lab No. 3**

## Sum of Two Numbers using Two Digit:

In the Third lab, we performed a task i.e. How to get input of two Numbers from the user And **Sum** these numbers and Display the sum of the given numbers in the given format. We **used interrupts** for the following purpose. We used **Line Feed** and **Carriage Return**. The Sum of the Numbers will be in two Digit. This is done by the operation of division. The sum is divided by 10, the **quotient** of the division goes into AL register and the **remainder** goes into AH register. These registers are then moved to other registers and displayed.

**ADD:** It is used to find sum of value in the registers and as well as Variables.

**SUB:** It is used to find difference of value in the registers and as well as Variables.

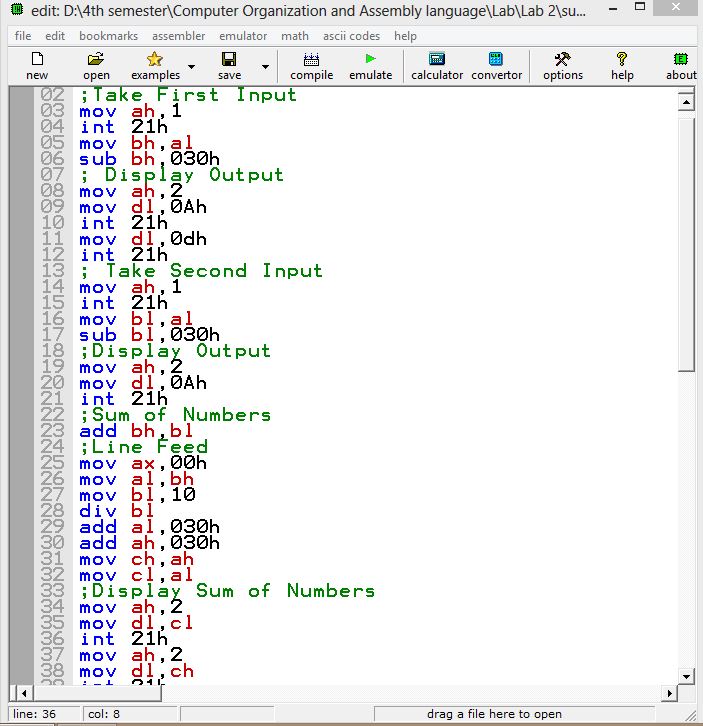
**Format:** ADD Register, Register

**Format:** SUB Register, Register

## **Lab3 Task Description:**

1. Take Two Numbers as **Input** from the User.
2. Find **Sum** of these Numbers.
3. Display the Sum in the **given Format.**
4. Sum will be in **Two Digits**.
5. We **Displayed Sum**.

## **Emulator Code to Perform the Task:**



## **Output for Task:**



# **Lab No. 4(a)**

## Sum of Two Numbers Using Functions:

In the Fourth lab, we performed a task i.e. how to get input of two Numbers from the user And Sum these numbers and **Display the sum** of the given numbers in the given format. We used interrupts for the following purpose. We used **Line Feed** and **Carriage Return**. The Sum of the Numbers will be in two digit. This is done by the operation of division. The sum is divided by 10, the quotient of the division goes into AL register and the remainder goes into AH register. These registers are then moved to other registers and displayed. We performed the following task with the help of Functions We use Main Function to perform the Task We Use **New Line Function**. The Number will display with **operators.**

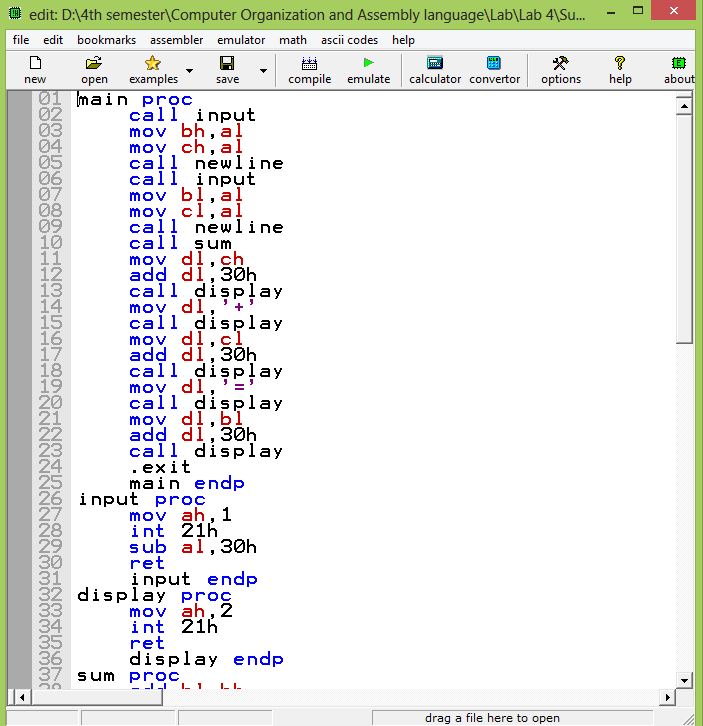
**Main**: We write our specific code there to increase the Readability. Proc is used for default Function.

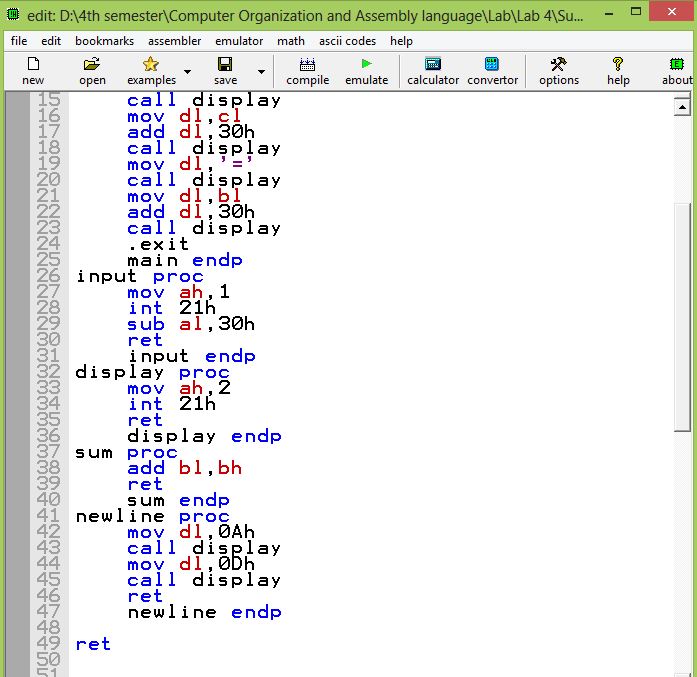
**Main Endp:** It is used to end the main function.

## **Lab4 (a) Task Description:**

1. Take two numbers as **Input from the User**.
2. Find **Sum** these Numbers.
3. Display the sum in **The given Format**.
4. Sum will be a **Two Digits**.
5. We **Displayed Sum.**
6. We use **Function** such as Main, Newline.

## **Code for the Task:**





## **Output:**



# **Lab No. 4(b)**

## Check that Number is Even or Odd:

In the Following Lab, We learnt How to **Declare a String.** How to **Display the String** and How **to End a String**. We make **Labels** in this Lab. Label are of Different types Such as Code Label and Data Label. We learnt How to make Jumps from one Label to Another Label and Within Label. Jumps are Conditional and Unconditional. We learnt How to make a **LOOP.** We Take Input and Checked Whether It is even or odd and asked the user whether he wants to continue the Program or Exit.

**Label:** is a place maker for Instructions.

**Format:** Labelname:

.**data:** It is used to define variable and data in the data segment.The description of this is given in the code.

.**code:** It is used to write code in the segment we are using data from data segment.

**CMP:** It is used to compare data in the registers etc. **Format:** CMP Reg,val

**MOV:** It is used to copies data from source operand to destination operand. We cannot move memory to memory. **Format:** MOV Destination,Source

**LEA**: It is used to input a string. **Format:** LEA Register, 09h

**JMP:** It is used for jumping from one label to another label. Three Jumps are used such as **JE, JNE, JG.**

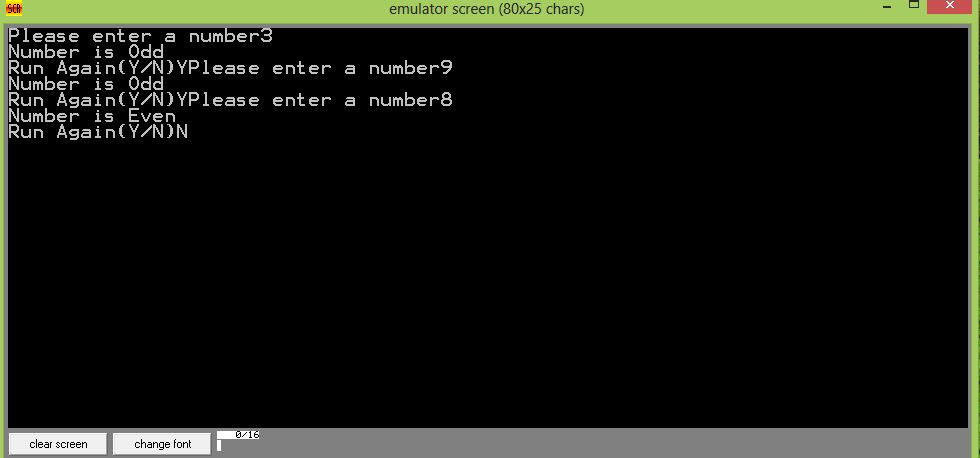


## **Lab4 (b) Task Description:**

1. To **Declare Strings**.
2. To **Display Strings**.
3. To Take an **Input of a number**.
4. Check whether the Number is **Even or Odd**.
5. To Ask the User to **Continue or Quit the Program**.
6. To make **loop with the help of Labels**.

## **Code for the Task:**Capture4.JPGCapture5.JPG

## **Output:**



# **Home Task for Lab 4:**

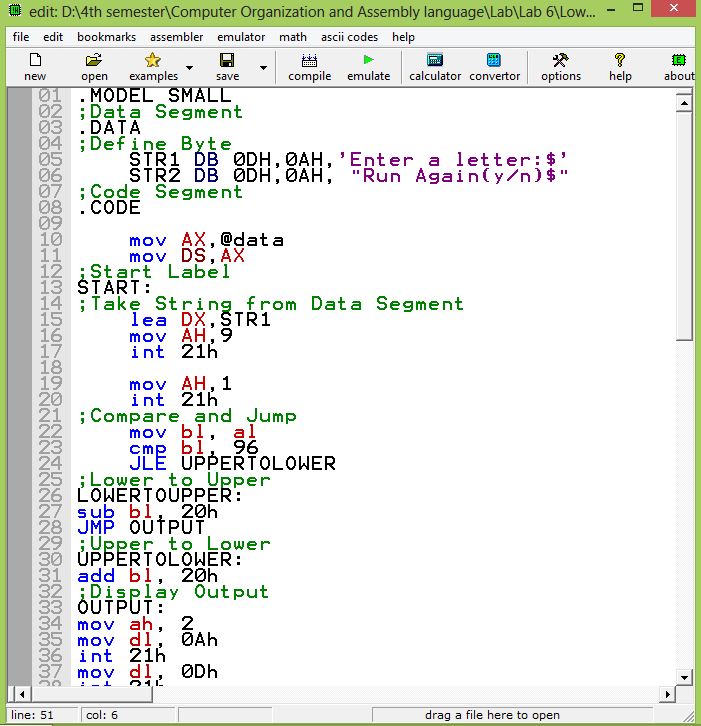
## Convert Uppercase Letter to Lowercase Letter and Vice Versa:

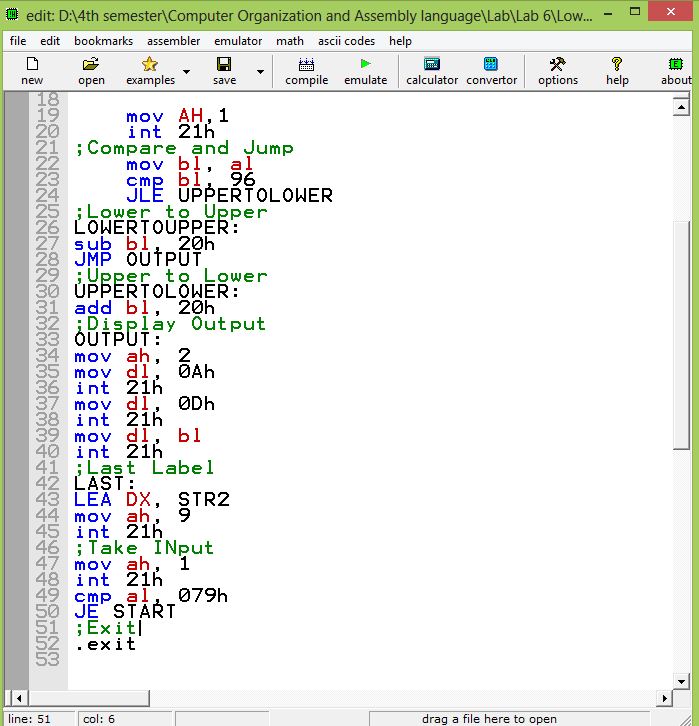
The Home Task is to convert a Lowercase **letter into an uppercase letter** and **uppercase letter into lowercase letter.** User will give Input a letter Either Uppercase or Lowercase by his own choice and Program Converts it to Lowercase or Uppercase respectively. And creates a loop to whether continue or not. We can perform the Task with the Help of Strings and tells the user Input.

## **Home Task Description:**

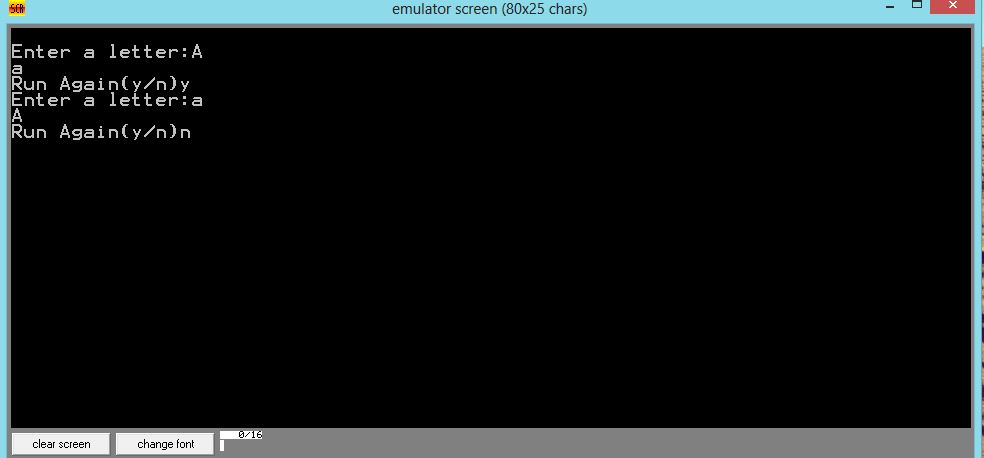
1. **Input is a Letter**.
2. Define String for **Display.**
3. Tells User either it is **Lowercase or Uppercase Letter**.
4. To make a **case sensitive program**.
5. To make loop for **Continue or Discontinue.**

## **Code for task:**





## **Output of Task:**



# **Lab No. 5:**

## Sorting of three Numbers:

In lab we performed to **sort three numbers** in the **Ascending Order**. We have to **sort number from Smallest to Largest** we performed lab with the help of our previous Knowledge. We are doing with strings **loops and jumps** to perform it.

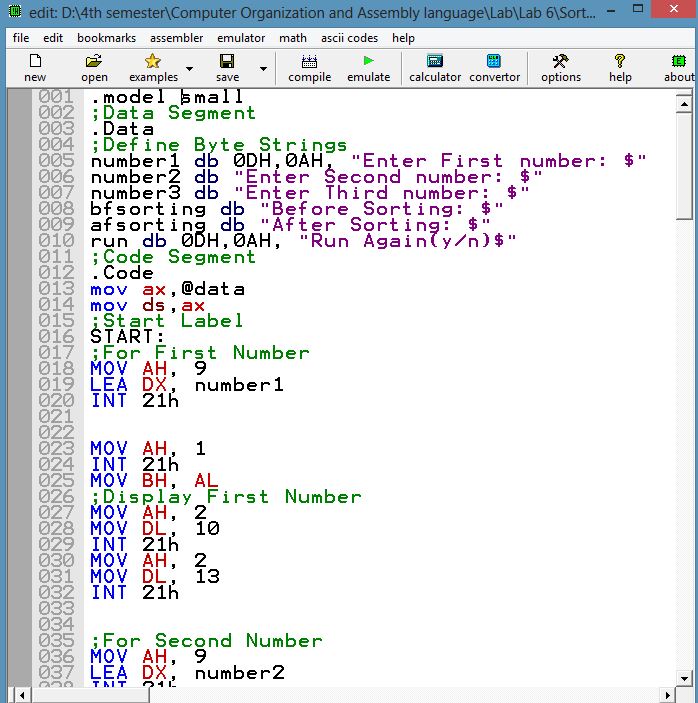
**DB:** It is used define byte. Description is given in the code.

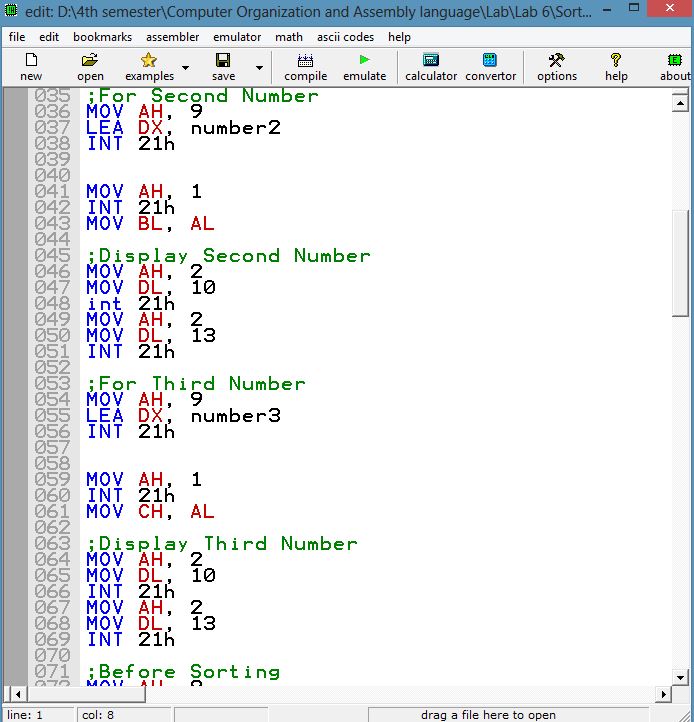
**DW:** It is used to define word. Description is given in the code.

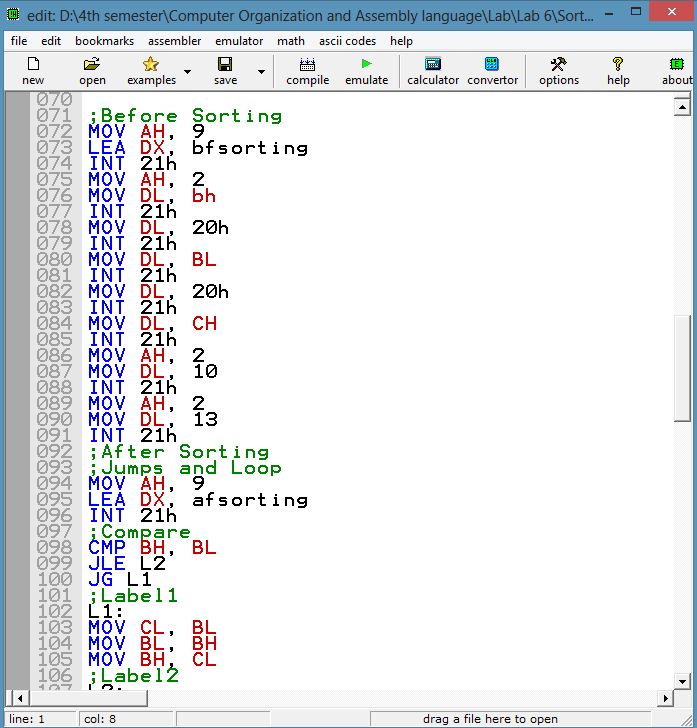
**Lab5 Task Description:**

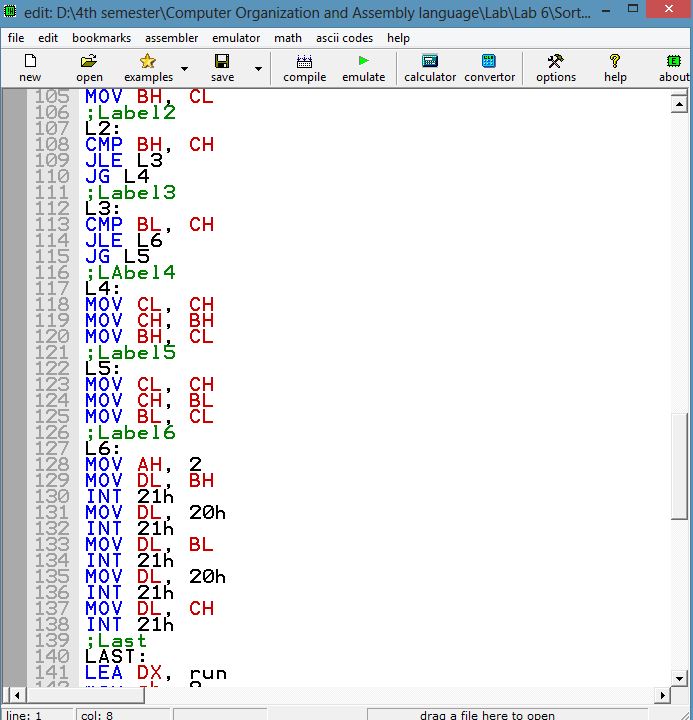
1. Take **Input** of Three Numbers.
2. **Sort** The Three Numbers
3. **Sort in Ascending Order**.
4. **Display** the Sorted Number.
5. Make it **case Sensitive**.
6. Check it on **All Cases.**
7. There are Six Different **Scenarios** for arrangement.

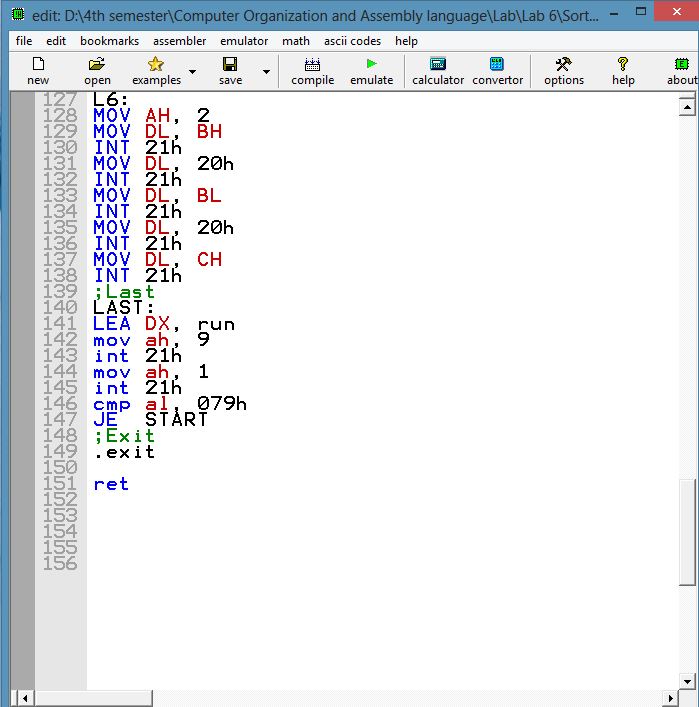
## **Code for Task:**



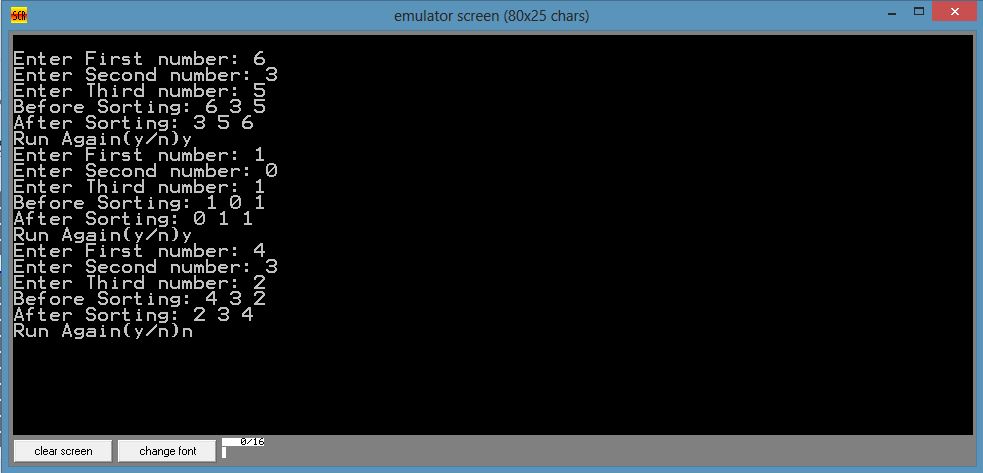








## **Output:**



# **Lab No. 6**

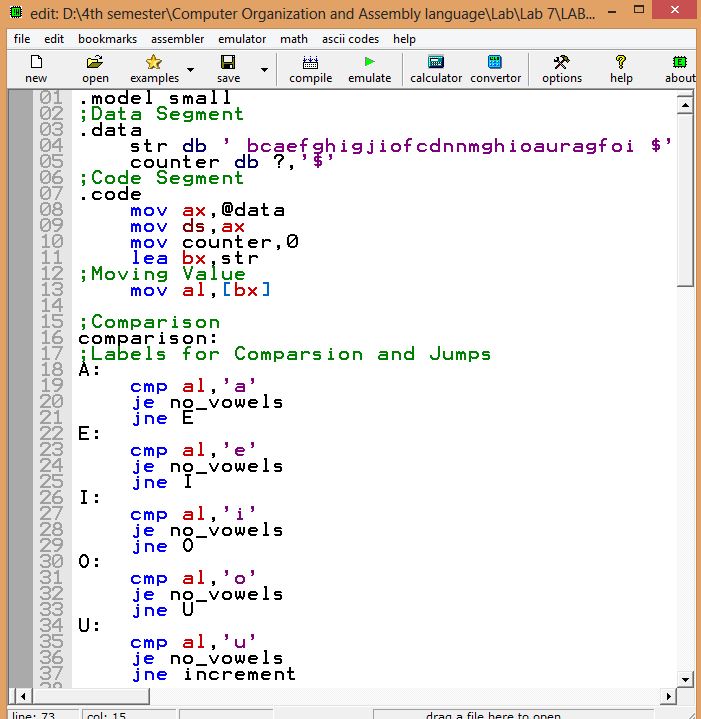
## Numbers of Vowels in the String and Calculate it:

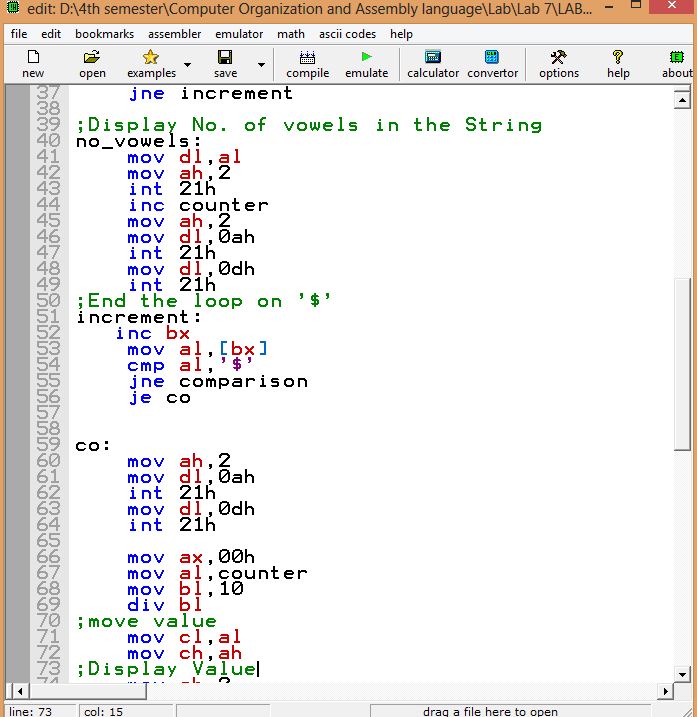
In Lab, We performed a task to calculate the Number of **vowels in the String**. We also **displayed the Vowels** which are Present in the String in the Arrangement. We also calculate vowels on both cases which are greater or less Than 9. We calculate with the help of Strings and Define Variables in the Data Segment.

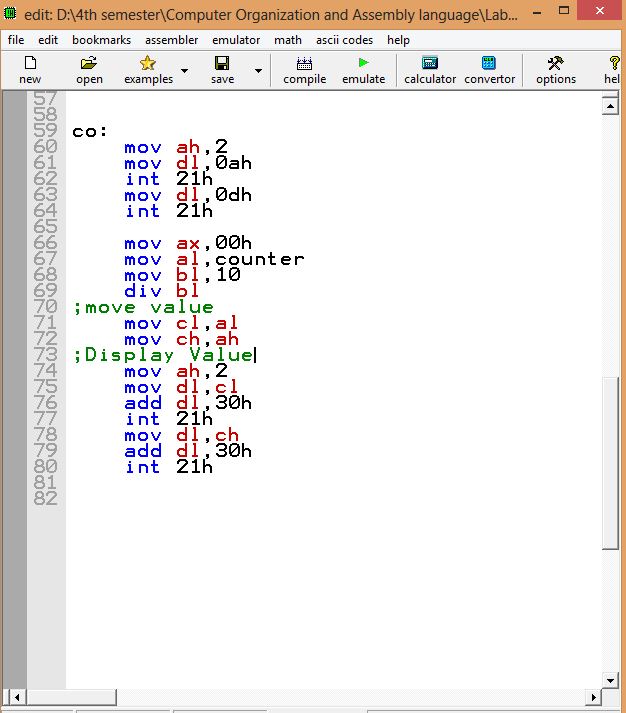
## **Lab Task to Be Performed:**

1. **Calculate Number of Vowels** in String.
2. Make the Program **Case Sensitive**.
3. **Display The Vowels** in the String.
4. Check it **for Less Than 9**.
5. Check it also **for Greater than 9**.

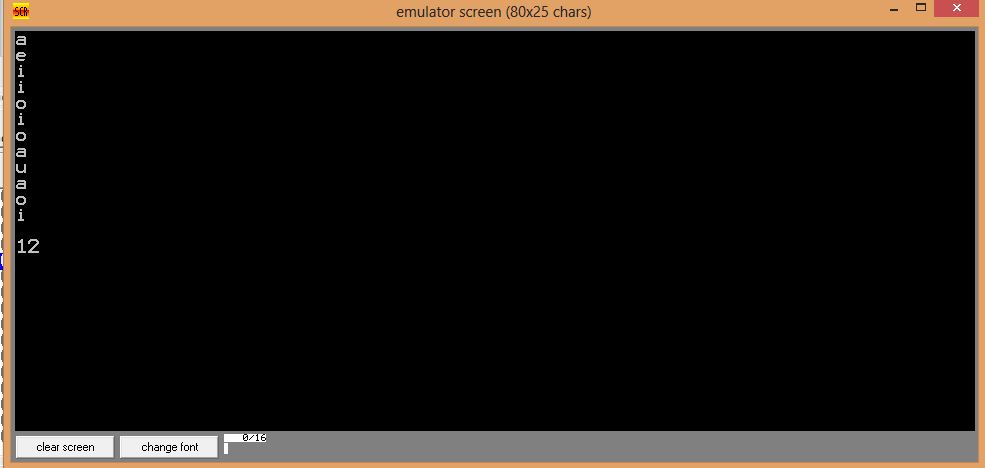
## **Code for Lab:**

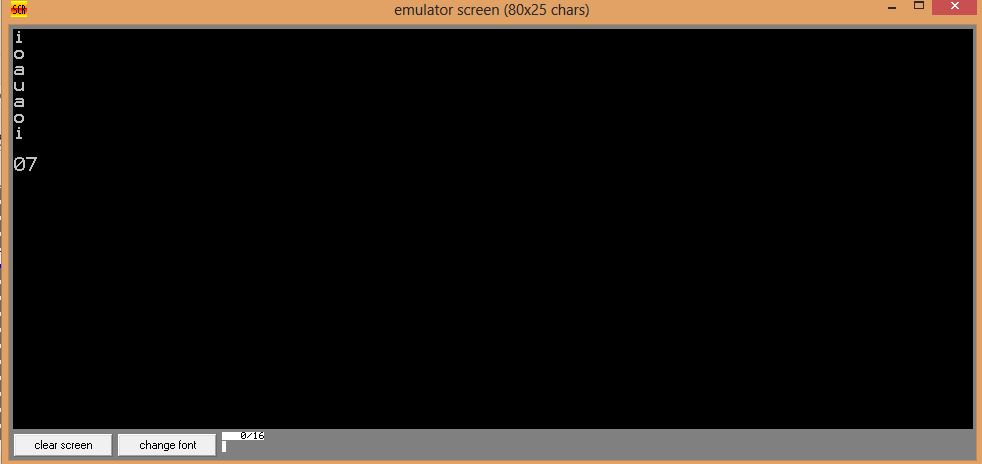






## **Output:**





# **Lab 6 (b)**

## Print Numbers with and without Loop:

We have to Print Digit from 0 to 9 in this case These Digits will be Print out with the help of Loop and Without Loop.

**LOOP:** It provides a simple way to repeat specific block of code.

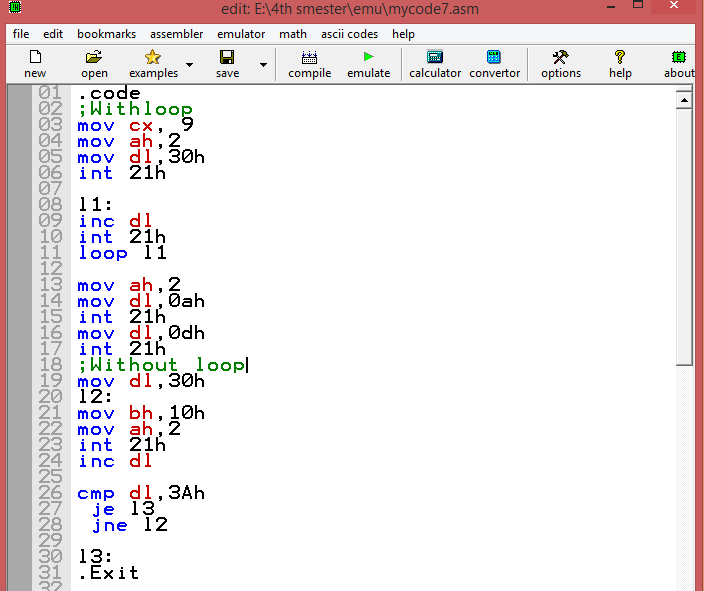
**INC:** It is used to add up 1 in the given memory. **Format:** INC Register.

**DEC:** It is used to decrease 1 in the memory. **Format:** DEC Register.

## **Lab 6 (b) Task Description:**

1. To **increment a Number** from 0-9 without Loop
2. **Print** out Numbers.
3. Make it happen **without loop**.
4. Print out These Numbers **Using Loop.**

## **Emulator code to perform the task:**



## **Output:**

