**Total Marks: \_7.5\_\_\_\_\_\_\_\_\_**

**Obtained Marks: \_\_\_\_\_\_\_\_\_\_\_\_\_**

**Computer Organization & Assembly Language (Lab)**

**Final Project**

**Loops and If-else Project**

**Submitted To: Engr. Muhammad Tauqeer Ali**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Group Members: Farooq Ahmed Khan (2012302)  
 Atika Khan (2012298)  
 Turab Ali Khan (2012339)  
 Fahad Wakeel (2012292)**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Contribution of Members**

|  |  |
| --- | --- |
| **Farooq Ahmed Khan** | **CMP, JMP instructions, Structure** |
| **Daniyal Shakeel** | **Program Structure, Interrupts** |
| **Atika Khan** | **Addressing Modes, Keywords (other than**  **Basic ones)** |
| **Turab Ali Khan** | **Data Segment** |
| **Fahad Wakeel** | **Loops, Registers** |

**Conditional Statements and Loops’ Program in**

**Assembly Language**

**Introduction:**

## This is a simple program in assembly language which is based on a few functionalities. In this game, we have set 6 choices, all for a different functionality following the functions, keywords, and concepts we learnt in the Lab. It contains CMP, JMP instructions for conditional statements. Loops, counter register, labels and different flags were also used in it.

**Procedure:**

## Data Segment (Initialization and Definition of Variables): After the size allocation for program, and other necessary keywords, and in the .data section, we initialized variables. We have about 41 variables which were used throughout the program, some for a single purpose, and some with different purposes. Also, some of them were used more than once.

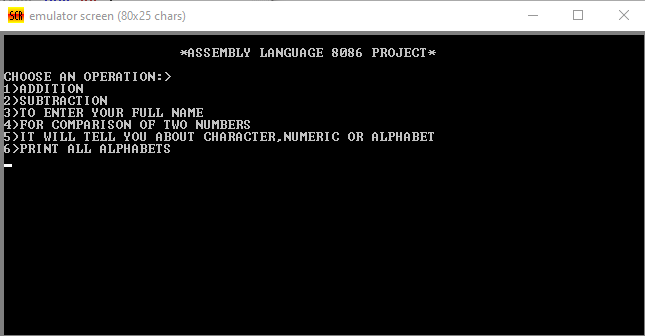
## Code Segment (Choices, Labels): The program starts from the “begin” label, and welcomes the user, and then displays a list of 6 options. The user chooses one operation from the list, and the program will then go to the label of that specific operation and perform the operation.

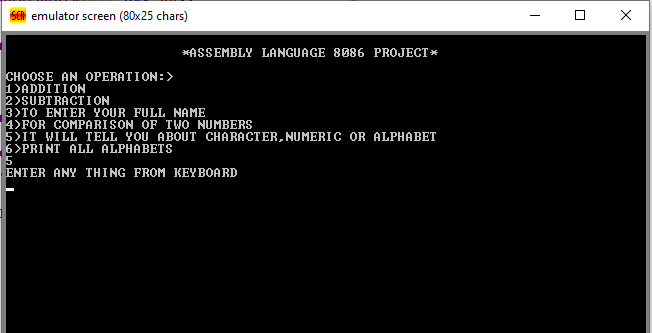
* **Code Segment (**Arithmetic and Logical Operations**):**In this section, the program performs arithmetic and logical operations according to the user’s choice which he makes in starting screen of the program. Program goes through the loops, and if-else statements in this section, and then after performing the operation the user asked for. It asks the use whether he wants to go the main menu or wants to terminate the program. The user replies with y for yes and n for no. Accordingly, the program runs after that.

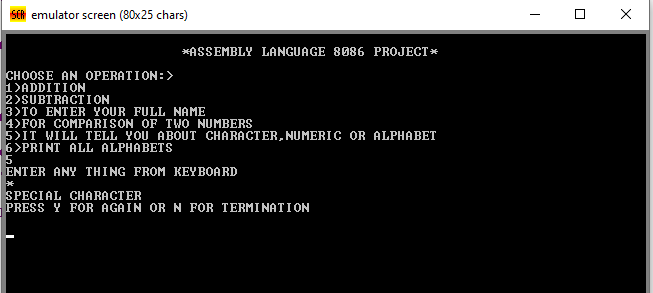
**Code:**

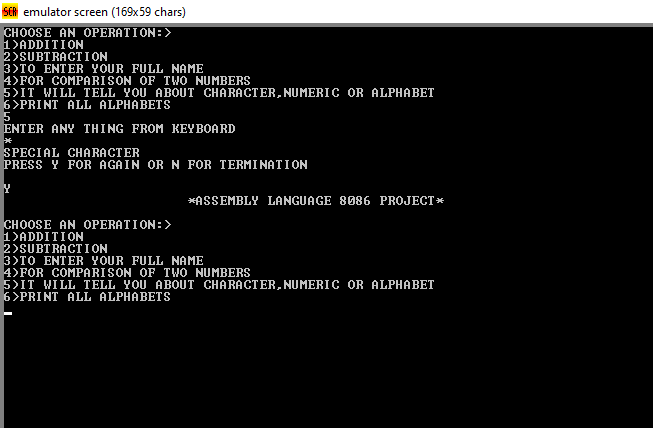
|  |
| --- |
| .MODEL SMALL  .STACK 100H  .DATA  MSG1 DB 10," \*ASSEMBLY LANGUAGE 8086 PROJECT\*$"  MSG2 DB 10,13,10,13,"CHOOSE AN OPERATION:>$"  MSG3 DB 10,13,"1)ADDITION $"  MSG4 DB 10,13,"2)SUBTRACTION$"  MSG5 DB 10,13,"3)TO ENTER YOUR NAME$"  MSG6 DB 10,13,"4)FOR COMPARISON OF TWO NUMBERS$"  MSGC DB 10,13,"5)IT WILL TELL YOU ABOUT CHARACTER,NUMERIC OR ALPHABET$"  MSGA DB 10,13,"6>PRINT ALL ALPHABETS$"  MSG7 DB 10,13,"INVALID INPUT$"  MSG8 DB 10,13,"ENTER 2 DIGITS (0-9) FOR SUBTRACTON $"  MSG9 DB 10,13,"THE SUBTRACTION OF $"  MSG10 DB " AND $"  MSG11 DB " IS $"  MSG12 DB 10,13,"ENTER 2 DIGITS (0-9) FOR ADDITION $"  MSG13 DB 10,13,"THE SUM OF $"  MSG14 DB " AND $"  MSG15 DB " IS $"  MSG17 DB 10,13,"ENTER YOUR NAME :$"  MSG18 DB 10,13,"YOU Entered $"  MSG19 DB 10,13,"ENTER TWO NUMBER FOR COMPARISON$"  MSG20 DB 10,13,"A IS GREATER $"  MSG21 DB 10,13,"B IS GREATER $"  MSG22 DB 10,13,"ENTER ANY THING FROM KEYBOARD$"  MSG23 DB 10,13,"NUMERIC$"  MSG24 DB 10,13,"IT IS A CAPITAL ALPHABET$"  MSG25 DB 10,13,"SPECIAL CHARACTER$"  MSG26 DB 10,13,"IT IS A SMALL ALPHABET$"  MSG27 DB 10,13,"PRESS Y FOR AGAIN OR N FOR TERMINATION$"    S1 DB ?  S2 DB ?  SV1 DB ?  SV2 DB ?  Q1 DB ?  A2 DB ?  Z3 DB ?  W4 DB ?  S5 DB ?  X6 DB ?  E7 DB ?  D8 DB ?  C9 DB ?  R10 DB ?    .CODE  MAIN PROC  MOV AX,@DATA  MOV DS,AX    BEGIN:  MOV AH,9  LEA DX,MSG1  INT 21H    LEA DX,MSG2  INT 21H    LEA DX,MSG3  INT 21H    LEA DX,MSG4  INT 21H    LEA DX,MSG5  INT 21H    LEA DX,MSG6  INT 21H    LEA DX,MSGC  INT 21H    LEA DX,MSGA  INT 21H      MOV AH,2  MOV DL,10  INT 21H    MOV DL,13  INT 21H    MOV AH,1  INT 21H    CMP AL,"1"  JE ADDITION    CMP AL,"2"  JE SUBTRACTION    CMP AL,"3"  JE NAME2    CMP AL,"4"  JE COMPARISON    CMP AL,"5"  JE NAS\_C    CMP AL,"6"  JE LOOP1    MOV AH,9  LEA DX,MSG7  INT 21H    JMP EXIT    ADDITION:  MOV AH, 9  LEA DX, MSG12  INT 21H    MOV AH,2  MOV DL,10  INT 21H    MOV DL,13  INT 21H    MOV AH, 1  INT 21H  MOV S1, AL    INT 21H  MOV S2, AL    MOV AH, 9  LEA DX, MSG13  INT 21H    MOV AH, 2  MOV DL, S1  INT 21H    MOV AH, 9  LEA DX, MSG14  INT 21H    MOV AH, 2  MOV DL, S2  INT 21H    MOV AH, 9  LEA DX, MSG15  INT 21H    MOV BL, S1  ADD BL, S2    SUB BL, 48    MOV AH, 2  MOV DL, BL  INT 21H  JMP EXIT      SUBTRACTION:  MOV AH, 9  LEA DX, MSG8  INT 21H    MOV AH,2  MOV DL,10  INT 21H    MOV DL,13  INT 21H    MOV AH, 1  INT 21H  MOV SV1, AL    INT 21H  MOV SV2, AL    MOV AH, 9  LEA DX, MSG9  INT 21H    MOV AH, 2  MOV DL, SV1  INT 21H    MOV AH, 9  LEA DX, MSG10  INT 21H    MOV AH, 2  MOV DL, SV2  INT 21H    MOV AH, 9  LEA DX, MSG11  INT 21H    MOV BL, SV1  SUB BL, SV2    ADD BL, 48    MOV AH, 2  MOV DL, BL  INT 21H  JMP EXIT      NAME2:  MOV AH,9  LEA DX,MSG17  INT 21H    MOV AH,2  MOV DL,10  INT 21H    MOV DL,13  INT 21H      MOV AH, 1  INT 21H  MOV Q1, AL    INT 21H  MOV A2,AL    INT 21H  MOV Z3,AL    INT 21H  MOV W4,AL    INT 21H  MOV S5,AL    INT 21H  MOV X6,AL    INT 21H  MOV E7,AL    INT 21H  MOV D8,AL    INT 21H  MOV C9,AL    INT 21H  MOV R10,AL    MOV AH,9  LEA DX,MSG18  INT 21H    MOV AH,2  MOV DL,Q1  INT 21H    MOV DL,A2  INT 21H    MOV DL,Z3  INT 21H    MOV DL,W4  INT 21H    MOV DL,S5  INT 21H    MOV DL,X6  INT 21H    MOV DL,E7  INT 21H    MOV DL,D8  INT 21H    MOV DL,C9  INT 21H    MOV DL,R10  INT 21H  JMP EXIT      COMPARISON:  MOV AH,9  LEA DX,MSG19  INT 21H    MOV AH,2  MOV DL,10  INT 21H    MOV DL,13  INT 21H      MOV AH,1  INT 21H  MOV BL,AL    INT 21H  MOV BH,AL    CMP BL,BH  JL B\_GREAT    MOV AH,9  LEA DX,MSG20  INT 21H    JMP EXIT    B\_GREAT:  MOV AH,9  LEA DX,MSG21  INT 21H  JMP EXIT    NAS\_C:  MOV AH,9  LEA DX,MSG22  INT 21H    MOV AH,2  MOV DL,10  INT 21H    MOV AH,2  MOV DL,13  INT 21H    MOV AH,1  INT 21H  MOV BL,AL    MOV AH,9    CMP BL,48  JL S\_CH    CMP BL,57  JG CAPITAL    MOV AH,9  LEA DX,MSG23  INT 21H    JMP EXIT    CAPITAL:  CMP BL,65  JL S\_CH    CMP BL,90  JG SMALL    MOV AH,9  LEA DX,MSG24  INT 21H    JMP EXIT    SMALL:  CMP BL,97  JL S\_CH    CMP BL,122  JG S\_CH    MOV AH,9  LEA DX,MSG26  INT 21H    JMP EXIT    S\_CH:    MOV AH,9  LEA DX,MSG25  INT 21H    JMP EXIT  LOOP1:  MOV CX,26  MOV DX,65  L1:  MOV AH,2    INT 21H    INC DX    LOOP L1    EXIT:    MOV AH,9  LEA DX,MSG27  INT 21H    MOV AH,2  MOV DL,10  INT 21H    MOV DL,13  INT 21H    MOV AH,2  MOV DL,10  INT 21H    MOV DL,13  INT 21H    MOV AH,1  INT 21H    CMP AL,"Y"  JE BEGIN    CMP AL,"y"  JE BEGIN    MOV AH,4CH  INT 21H    MAIN ENDP  END MAIN |

**Screen Shots:**









**Problems:**

Biggest problem was getting multiple digit input, and also the string output of any length. Which was not achieved. Upon research, it came to our knowledge that Arrays can be used to do that. We are not good in Arrays at the moment, and time was not left, so we made different variables instead and stored the string and numbers in them to make it work somehow to some extent.

**Conclusions:**

The program did run smoothly after an exhausting struggle. We have found keywords very confusing and hard to remember as they’re not based on straightforward English like high level languages. Also, the concepts around registers, addressing modes, ASCII codes are confusing too, and it’s hard to grasp one’s mind around it without enough practice.

We conclude that people who made high level languages are the very people because of whom the world is better as we experience today, and we also share our sympathy with those who had to work with assembly language regularly.  
Humor apart, we have learned much more about Assembly language and its keywords, components etc. by practicing it in this project, and we are confident that we can use assembly language to make more better projects than this one.