

**Rajshahi Univarsity of Engineering and Technology**

**Course No: CSE-3110**

**Course Title: Sessional Based on CSE-3109**

**Lab No:** 01

**Problem Name:**  Write an assembly code that calculate sum=A+B-C

**Submitted To:**

**Name:** Sadia Zaman Mishu

**Designation:** Assistant Professor

**Department:** CSE, RUET

**Submitted By:**

**Name:** Md Al Amin Tokder

**Roll:** 1803078

**Section:** B

**Department:** CSE, RUET

**Lab-1**

**Problem Description:**

Write an assembly code that calculate sum=A+B-C

**Theory:**

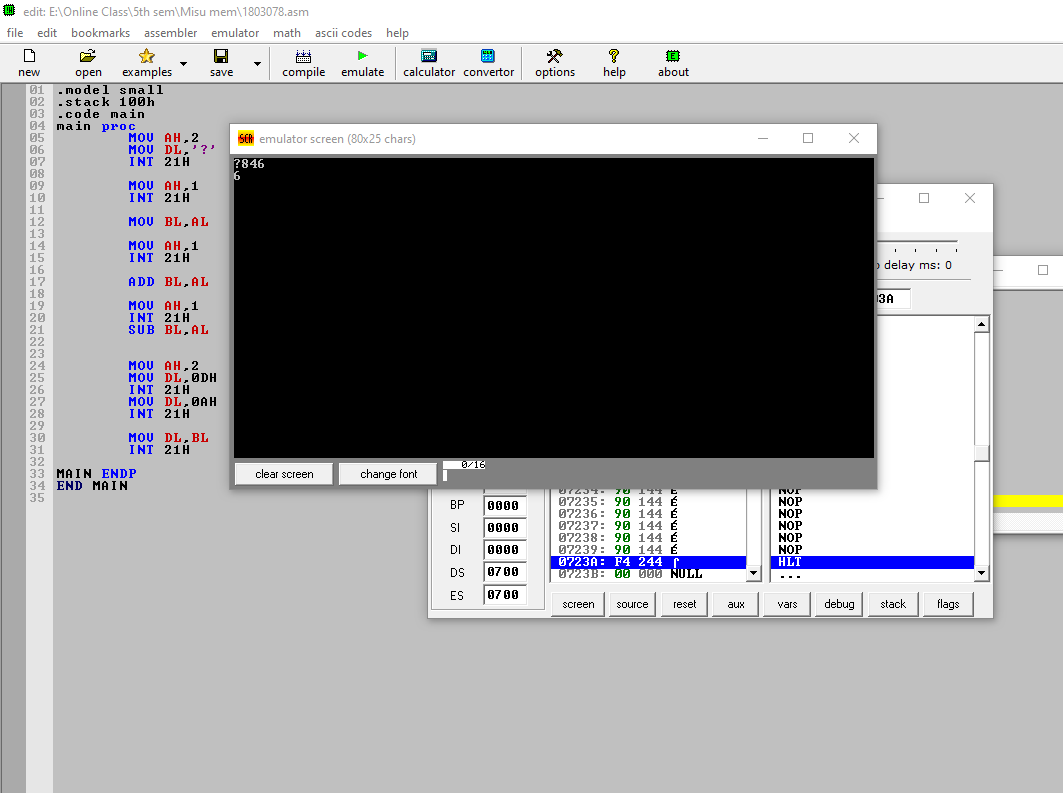
Assembly programming 8086 is hardware oriented programming language which provides architecture and registers functionality for 8086 processors.

For storing value in the predeclared variable in data segment we use Three character using MOV AH,1 function. For print the stored values using MOV AH,2 function. Mov operation is applicable between memory location and register but not both memory location. MOV,SUB,ADD,JMP etc operations are used to execute instruction .

**Source code:**

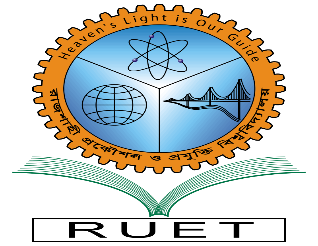
|  |
| --- |
| .model **small**  .**stack** 100h  .**code** main  main proc  **MOV** **AH**,2  **MOV** **DL**,'?'  **INT** 21H    **MOV** **AH**,1  **INT** 21H    **MOV** **BL**,**AL**    **MOV** **AH**,1  **INT** 21H    **ADD** **BL**,**AL**    **MOV** **AH**,1  **INT** 21H  **SUB** **BL**,**AL**      **MOV** **AH**,2  **MOV** **DL**,0DH  **INT** 21H  **MOV** **DL**,0AH  **INT** 21H    **MOV** **DL**,**BL**  **INT** 21H    MAIN ENDP  END MAIN |

**Output:**



**Conclusion:**

In this program when we input any value for A,B,C 8086- microprocessor use the ASCII value and calculated result.



**Rajshahi Univarsity of Engineering and Technology**

**Course No: CSE-3110**

**Course Title: Sessional Based on CSE-3109**

**Lab No:** 02

**Problem Name:** Write a program to display a “?”, read two capital letters and display them on the next line in alphabetical order.

**Submitted To:**

**Name:** Sadia Zaman Mishu

**Designation:** Assistant Professor

**Department:** CSE, RUET

**Submitted By:**

**Name:** Md Al Amin Tokder

**Roll:** 1803078

**Section:** B

**Department:** CSE, RUET

**Lab-2**

**Problem Description:**

Write a program to display a “?”, read two capital letters and display them on the next line in alphabetical order.

**Theory:**

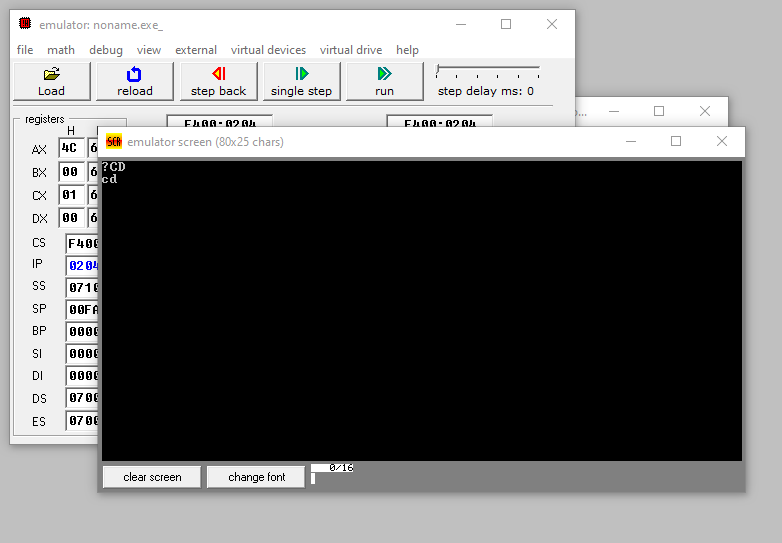
In assembly language, data are stored in register and use the data to solve arithmetic and logical problem. Memory-Memory, Memory-Variable, Variable-Memory data transfer are allowed. But, Variable-Variable data transfer is not allowed in assembly language. ADD, MOV, CMP, SUB, MUL, DIV these keywords are used for different purposes.

The CMP instruction compares two operands. It is generally used in conditional execution. This instruction basically subtracts one operand from the other for comparing whether the operands are equal or not. It does not disturb the destination or source operands.

**Source code:**

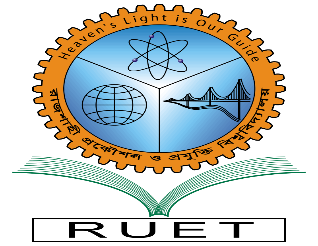
|  |
| --- |
| .MODEL **SMALL**  .**STACK** 100H  **.DATA**      .**CODE**  MAIN PROC    **MOV** **AH**,2  **MOV** **DL**, '?'  **INT** 21H    **MOV** **AH**, 1  **INT** 21H  **MOV** **BL**, **AL**  **ADD** **BL**, 32    **MOV** **AH**,1  **INT** 21H  **MOV** **CL**, **AL**  **ADD** **CL**, 32    **MOV** **AH**, 2  **MOV** **DL**, 0AH  **INT** 21H  **MOV** **DL**, 0DH  **INT** 21H      **CMP** **CL**, **BL**  **JG** NEXT    **MOV** **AH**, 2  **MOV** **DL**, **CL**  **INT** 21H  **MOV** **DL**, **BL**  **INT** 21H  **MOV** **AH**, 4CH  **INT** 21H    NEXT:  **MOV** **AH**, 2  **MOV** **DL**, **BL**  **INT** 21H  **MOV** **DL**, **CL**  **INT** 21H  **MOV** **AH**, 4CH  **INT** 21H  MAIN ENDP  END MAIN |

**Output:**



**Conclusion :**

Here in this program ,using CMP instruction we check and then by adding 32 capital letter is converted small letter.



**Rajshahi Univarsity of Engineering and Technology**

**Course No: CSE-3110**

**Course Title: Sessional Based on CSE-3109**

**Lab No:** 03

**Problem Name:** Write a program to count number of 1 bits in a binary number and check the number is odd or even.

**Submitted To:**

**Name:** Sadia Zaman Mishu

**Designation:** Assistant Professor

**Department:** CSE, RUET

**Submitted By:**

**Name:** Md Al Amin Tokder

**Roll:** 1803078

**Section:** B

**Department:** CSE, RUET

**Lab-3**

**Problem Description:**

Write a program to count number of 1 bits in a binary number and check the number is odd or even.

**Theory:**

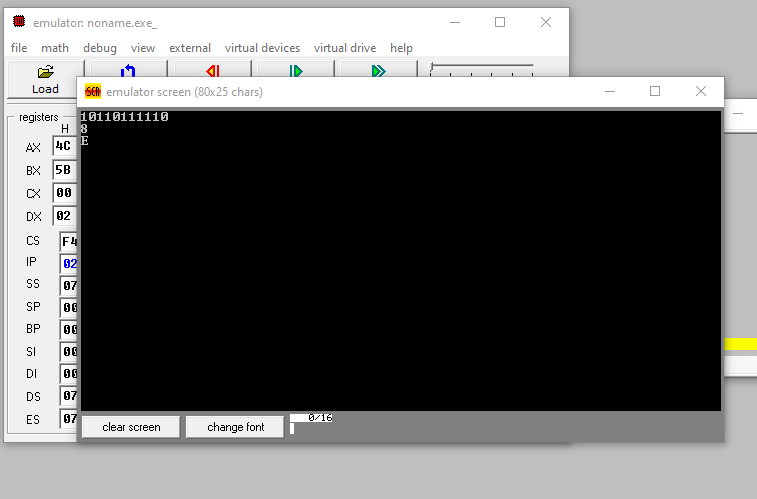
For counting number of 1 bits in a binary number Loop through all bits check if a bit is set and if it is, then increment the set bit count.

For Odd Even checking ,we can determine one number is odd or even by checking only the LSB. When LSB is 1, the number is odd, otherwise it is even. In this program we are taking a number from memory and then ANDing 01H with it. if the result is nonzero, then the number is odd, otherwise it is even.

**Source code:**

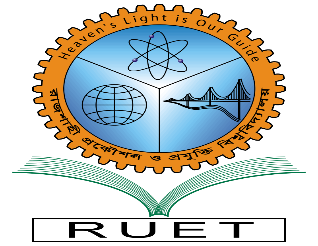
|  |
| --- |
| .MODEL **SMALL**  .**STACK** 100H  **.DATA**  .**CODE**  MAIN PROC    **XOR** **BX**, **BX**  **MOV** **CL**, 0  **MOV** **AH**, 1  **INT** 21H    WHILE:  **CMP** **AL**, 0DH  **JE** END\_WHILE  **AND** **AL**, 0FH  **SHL** **BX**, 1  **OR** **BL**, **AL**  **INT** 21H  **JMP** WHILE  END\_WHILE:  **MOV** **AH**, 2  **MOV** **DL**, 0AH  **INT** 21H  **MOV** **DL**, 0DH  **INT** 21H    **XOR** **AX**, **AX**  **MOV** **CX**, 16  TOP:  **ROL** **BX**, 1  **JNC** NEXT  **INC** **AX**  NEXT:  **LOOP** TOP  **ADD** **AX**, 48  **MOV** **AH**, 2  **MOV** **DX**, **AX**  **INT** 21H  **MOV** **AH**, 2  **MOV** **DL**, 0AH  **INT** 21H  **MOV** **DL**, 0DH  **INT** 21H  **AND** **BL**,1  **CMP** **BL**,0  **JE** NEXT1  **MOV** **AH**,2  **MOV** **DL**,'O'  **INT** 21H  **JMP** END  NEXT1:  **MOV** **AH**, 2  **MOV** **DL**, 'E'  **INT** 21H  END:  **MOV** **AH**, 4CH  **INT** 21H  MAIN ENDP  END MAIN |

**Output:**



**Conclusion:**

Here RCL or RCR instructions are used for counting 1 bits in the number.The right shift shifts the bits to the right and the LSB is shifted into Carry Flag. The left shift shifts the bits to the left and the MSB is shifted into Carry Flag.



**Rajshahi Univarsity of Engineering and Technology**

**Course No: CSE-3110**

**Course Title: Sessional Based on CSE-3109**

**Lab No:** 04

**Problem Name:** Write a program to input a string and reverse every word of the string.

**Submitted To:**

**Name:** Sadia Zaman Mishu

**Designation:** Assistant Professor

**Department:** CSE, RUET

**Submitted By:**

**Name:** Md Al Amin Tokder

**Roll:** 1803078

**Section:** B

**Department:** CSE, RUET

**Lab-4**

**Problem Description:**

Write a program to input a string and reverse every word of the string.

**Theory:**

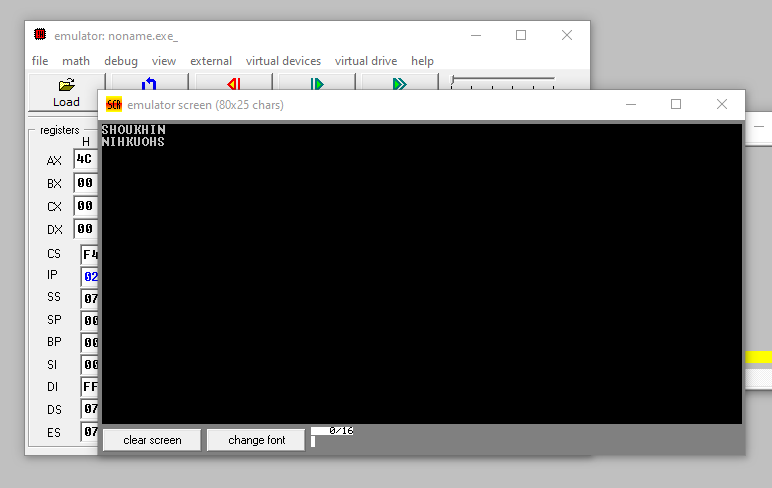
For this program first of all we need to take a array which can contain duplicate letters and push letter into stack until getting space or carriage return. SI point to the initial address of the array. Then Take CX=0 , to count the number of letter in a word and BX=0 ,to use as a flag. If BX=1 ,then program don’t take any input otherwise take input from user. When get space or carriage return then perform pop operation. When get a small vowel then turn it into capital letter using CAPITAL level, otherwise no change.

DUP instructions, an array is initialized by a common value.

**Source code:**

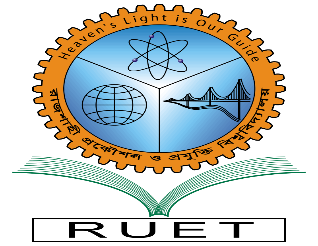
|  |
| --- |
| .MODEL **SMALL**  .**STACK** 100H  **.DATA**  MSG **DB** 50 DUP**(**'?'**)**  .**CODE**  MAIN PROC  **MOV** **AX**,@**DATA**  **MOV** **DS**,**AX**  **MOV** **BX**,0  **LEA** **SI**,MSG  **MOV** **AH**,1  **INT** 21H  INPUT:  **INC** **BX**  **CMP** **AL**,0DH  **JE** OUTPUT  **MOV** MSG**[SI]**,**AL**  **ADD** **SI**,1  **INT** 21H  **JMP** INPUT  OUTPUT:  **MOV** MSG**[SI]**,'$'  **MOV** **AH**,2  **MOV** **DL**,0AH  **INT** 21H  **MOV** **DL**,0DH  **INT** 21H  **LEA** **SI**,MSG  **MOV** **BX**,0  **JMP** OUTPUT1  OUTPUT1:  **CMP** **[SI]**,20H  **JE** PRINT  **CMP** **[SI]**,24H  **JE** PRINT  **INC** **BX**  **INC** **SI**  **JMP** OUTPUT1  PRINT:  **MOV** **CX**,**BX**  **LEA** **DI**,**SI**  **DEC** **DI**  PRINT1:  **MOV** **AH**,2  **MOV** **DL**,**[DI]**  **INT** 21H  **DEC** **DI**  **LOOP** PRINT1  **CMP** **[SI]**,24H  **JE** EXIT  **MOV** **BX**,0  **INC** **SI**  **MOV** **AH**,2  **MOV** **DL**,20H  **INT** 21H  **JMP** OUTPUT1  EXIT:  **MOV** **AH**,4CH  **INT** 21H  MAIN ENDP  END MAIN |

**Output:**



**Discussion :**

This program gives write output according to the given input. Knowledge of Loop,jmp and stack is needed to solve this problem. Without using $ sign, the program will display a wrong answer.Here (SI) is used to address the array source index.



**Rajshahi Univarsity of Engineering and Technology**

**Course No: CSE-3110**

**Course Title: Sessional Based on CSE-3109**

**Lab No:** 05

**Problem Name:** Write a program that lets the user enter time in seconds, up to 65535 and outputs the time as hours, minutes and seconds

**Submitted To:**

**Name:** Sadia Zaman Mishu

**Designation:** Assistant Professor

**Department:** CSE, RUET

**Submitted By:**

**Name:** Md Al Amin Tokder

**Roll:** 1803078

**Section:** B

**Department:** CSE, RUET

**Lab-5**

**Problem Description:**

Write a program that lets the user enter time in seconds, up to 65535 and outputs the time as hours, minutes and seconds

**Theory:**

For unsigned division, DIV instruction is used. IDIV is used for signed division.

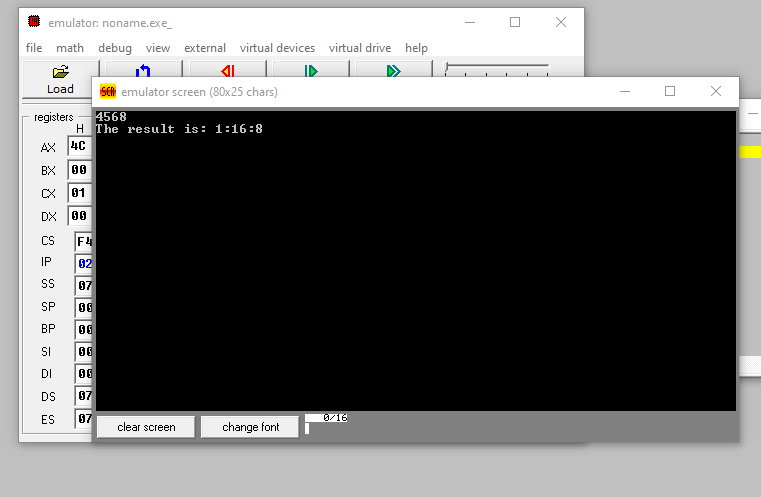
Unsigned divide AX by divisor BL with result stored in AL = Quotient, AH = Remainder.Unsigned divide DX:AX by divisor BX, with result stored in AX = Quotient, DX = Remainder.

If the divisor is much smaller than the dividend, the divide overflow will occur.

**Source code:**

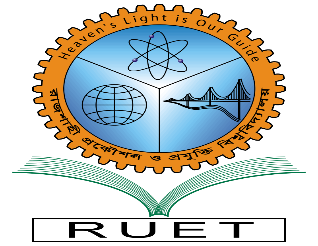
|  |
| --- |
| .MODEL **SMALL**  .**STACK** 100H  **.DATA**  MSG **DB** 0AH,0DH,'The result is : $'  .**CODE**  MAIN PROC  **MOV** **AX**,@**DATA**  **MOV** **DS**,**AX**  **CALL** INDEC  **MOV** **DX**,0  **MOV** **BX**,3600  **DIV** **BX**  **PUSH** **AX**  **PUSH** **DX**  **MOV** **AH**,9  **LEA** **DX**,MSG  **INT** 21H  **POP** **DX**  **POP** **AX**  **CALL** OUTDEC  **PUSH** **DX**  **MOV** **AH**,2  **MOV** **DL**,':'  **INT** 21H  **POP** **AX**  **MOV** **BX**,60  **MOV** **DX**,0  **DIV** **BX**  **CALL** OUTDEC  **PUSH** **DX**  **MOV** **AH**,2  **MOV** **DL**,':'  **INT** 21H  **POP** **AX**  **CALL** OUTDEC  **MOV** **AH**,4CH  **INT** 21H  MAIN ENDP  INDEC PROC  **PUSH** **BX**  **PUSH** **CX**  **PUSH** **DX**  BEGIN:  **XOR** **BX**,**BX**  **XOR** **CX**,**CX**  **MOV** **AH**,1  **INT** 21H  **CMP** **AL**,'-'  **JE** MINUS  **CMP** **AL**,'+'  **JE** PLUS  **JMP** REPEAT2  MINUS:  **MOV** **CX**,1  PLUS:  **INT** 21H  REPEAT2:  **CMP** **AL**,'0'  **JNGE** NOT\_DIGIT    **CMP** **AL**,'9'  **JNLE** NOT\_DIGIT  **AND** **AX**,000FH  **PUSH** **AX**  **MOV** **AX**,10  **MUL** **BX**  **POP** **BX**  **ADD** **BX**,**AX**  **MOV** **AH**,1  **INT** 21H  **CMP** **AL**,0DH  **JNE** REPEAT2  **MOV** **AX**,**BX**  **OR** **CX**,**CX**  **JE** EXIT  **NEG** **AX**  EXIT:  **POP** **DX**  **POP** **CX**  **POP** **BX**  **RET**  NOT\_DIGIT:  **MOV** **AH**,2  **MOV** **DL**,0DH  **INT** 21H  **MOV** **DL**,0AH  **INT** 21H  **JMP** BEGIN  INDEC ENDP  OUTDEC PROC  **PUSH** **AX**  **PUSH** **BX**  **PUSH** **CX**  **PUSH** **DX**  **OR** **AX**,**AX**  **JGE** END\_IF1  **PUSH** **AX**  **MOV** **DL**,'-'  **MOV** **AH**,2  **INT** 21H  **POP** **AX**  **NEG** **AX**  END\_IF1:  **XOR** **CX**,**CX**  **MOV** **BX**,10D  REPEAT1:  **XOR** **DX**,**DX**  **DIV** **BX**  **PUSH** **DX**  **INC** **CX**  **OR** **AX**,**AX**  **JNE** REPEAT1  **MOV** **AH**,2  PRINT:  **POP** **DX**  **OR** **DL**,30H  **INT** 21H  **LOOP** PRINT  **POP** **DX**  **POP** **CX**  **POP** **BX**  **POP** **AX**  **RET**  OUTDEC ENDP  END MAIN |

**Output:**



**Conclusion:**

In this program, if the input is a non-digit, the program will skip the line and take a new input. But for a negative number, it will display a wrong answer. We can solve this problem just like non-digit input. INDEC and OUTDEC procedure is used for input and output.



**Rajshahi Univarsity of Engineering and Technology**

**Course No: CSE-3110**

**Course Title: Sessional Based on CSE-3109**

**Lab No:** 06

**Problem Name:** Write a program to sort an array in descending order and display numbers, capital letters, small letters in descending order on different line.

**Submitted To:**

**Name:** Sadia Zaman Mishu

**Designation:** Assistant Professor

**Department:** CSE, RUET

**Submitted By:**

**Name:** Md Al Amin Tokder

**Roll:** 1803078

**Section:** B

**Department:** CSE, RUET

**Lab-6**

**Problem Description:**

Write a program to sort an array in descending order and display numbers, capital letters, small letters in descending order on different line.

**Theory:**

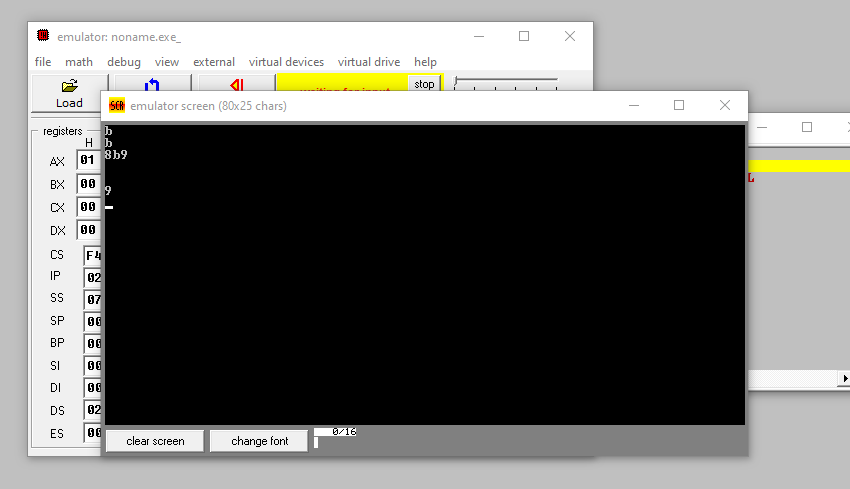
An array is implemented at the assembly language level by a block of memory in which the variables of the array are stored, contiguously, one right after the other. When the length value stored as well, it's generally stored just before the beginning of the block, so that it can be easily found. SI register is used for addressing.

Here selection sort is used for sorting the array. Select procedure is created for selection sort. Besides, to display the result, output procedure is used.

**Source code:**

|  |
| --- |
| .MODEL **SMALL**  .**STACK** 100H  **.DATA**  A **DB** 50 DUP**(**'?'**)**  .**CODE**  MAIN PROC  **MOV** **AX**, @**DATA**  **MOV** **DS**,**AX**  **LEA** **SI**,A  **MOV** **BX**,0  INPUT:  **MOV** **AH**,1  **INT** 21H  **MOV** A**[SI]**,**AL**  **INC** **SI**  **INC** **BX**  **CALL** SELECT  **MOV** **AH**,2  **MOV** **DL**,0AH  **INT** 21H  **MOV** **DL**,0DH  **INT** 21H  **CALL** OUTPUT  **MOV** **AH**,2  **MOV** **DL**,0AH  **INT** 21H  **MOV** **DL**,0DH  **INT** 21H  **JMP** INPUT  **MOV** **AH**,4CH  **INT** 21H  MAIN ENDP  SELECT PROC  **PUSH** **BX**  **PUSH** **CX**  **PUSH** **DX**  **PUSH** **SI**  **DEC** **BX**  **JE** END\_SORT  **LEA** **SI**,A  **JNL** NEXT  **MOV** **DI**,**SI**  **MOV** **AL**,**[DI]**  NEXT:  **LOOP** FIND\_BIG  **CALL** SWAP  **DEC** **BX**  **JNE** SORT\_LOOP  END\_SORT:  **POP** **SI**  **POP** **DX**  **POP** **CX**  **POP** **BX**  **RET**  SELECT ENDP  SWAP PROC  **PUSH** **AX**  **MOV** **AL**,**[SI]**  **XCHG** **AL**,**[DI]**  **MOV** **[SI]**,**AL**  **POP** **AX**  **RET**  SWAP ENDP  **MOV** **DX**,**SI**  SORT\_LOOP:  **MOV** **SI**,**DX**  **MOV** **CX**,**BX**  **MOV** **DI**,**SI**  **MOV** **AL**,**[DI]**  FIND\_BIG:  **INC** **SI**  **CMP** **[SI]**,**AL**  OUTPUT PROC  **LEA** **SI**,A  **MOV** **CX**,**BX**  PRINT:  **MOV** **AH**,2  **MOV** **DL**,**[SI]**  **CMP** **DL**,5BH  **JL** PRI  **INT** 21H  **ADD** **SI**,1  **DEC** **CX**  **CMP** **CX**,0  **JE** EXIT  **CMP** **CX**,0  **JG** PRINT  PRI:  **MOV** **AH**,2  **MOV** **DL**,0AH  **INT** 21H  **MOV** **DL**,0DH  **INT** 21H  PRINT1:  **MOV** **AH**,2  **MOV** **DL**,**[SI]**  **CMP** **DL**,3AH  **JL** PRI1  **INT** 21H  **ADD** **SI**,1  **DEC** **CX**  **CMP** **CX**,0  **JE** EXIT  **CMP** **CX**,0  **JG** PRINT1  PRI1:  **MOV** **AH**,2  **MOV** **DL**,0AH  **INT** 21H  **MOV** **DL**,0DH  **INT** 21H  PRINT2:  **MOV** **AH**,2  **MOV** **DL**,**[SI]**  **INT** 21H  **ADD** **SI**,1  **DEC** **CX**  **CMP** **CX**,0  **JE** EXIT  **CMP** **CX**,0  **JG** PRINT2  EXIT:  **RET**  OUTPUT ENDP  END MAIN |

**Output:**



**Discussion :**

The array is sorted on the base of ascii codes. In this program, selection sort is used for sorting the array in descending order. The array contains small letter, capital letter and digit .We can use another sorting technique for better performance .