RAM LAL ANAND COLLEGE

UNIVERSITY OF DELHI

DEPARTMENT OF COMPUTER SCIENCE

SESSION: July-December 2022

PRACTICAL FILE

Submitted to: **Dr. Vandana Gandotra**

Program Name: B.Sc(H) Computer Science

Semester: V

Title of the paper: Microprocessors

Unique Paper code: 32347504

Name of the Student: Md Ghulam Hussain

Examination Roll No.: 20058570021

Class roll no.: 4021

Q1 Write a program for 32-bit binary Addition ,Subtraction, Division ,and Multiplication .

```
.model small; contain two segment data and code
.386
.stack 100h ; tells the assembler to reserve storage
.data;start of data segment
DATA1 DD 00000000H;initialize memory with double word
DATA2 DD 00000000H;initialize memory with double word
PROD1 dd ?; set double word variable
PROD2 dd ?; set double word variable
REM dd ?; set double word variable
QUO dd ?; set double word variable

msg1 db 10,13,"Enter a number(A): $";10 is the ascii control code for line fed while 13 is the
code for carriage return
msg2 db 10,13,"Enter another number(B): $"
msg3 db 10,13,"Press 1 to ADD.$"
msg4 db 10,13,"Press 2 to subtract.$"
msg5 db 10,13,"Press 3 to multiply .$"
msg6 db 10,13,"Press 4 to division.$"
msg7 db 10,13,"Enter your choice: $"
msg8 db 10,13,"A - B = $"
msg9 db 10,13,"A + B = $"
msg10 db 10,13,"A * B = $"
msg32 db 10,13,"The Remainder is :: $"
msg33 db 10,13,"The Quotient is :: $"

.code ; start of code segment
.startup; Generates program start-up code

mov BL,00H

mov ah,09
mov dx, offset msg3
int 21h; Output a string terminated by '$’ stored in DX, as AH =9

mov ah,09
mov dx, offset msg4
int 21h

mov ah,09
mov dx, offset msg5
int 21h
```

```
mov ah,09
mov dx, offset msg6
int 21h

mov ah,09
mov dx, offset msg7
int 21h; Output a string terminated by '$' stored in DX, as AH =9

mov ah, 01
int 21h;input from user

sub al,30h

cmp al,01h
je addition

cmp al,02h
je subtraction

cmp al,03h
je multiply

cmp al,04h
je division

addition:
    MOV AH,09
    MOV DX,OFFSET msg1
    INT 21H
    MOV EBX,0
    MOV CX,8
    AGAIN:
        MOV AH,01 ;1ST NO. ENTERED
        INT 21H
        CMP AL,'A'
        JGE L5; jump to the lebel
        SUB AL,30H
        JMP L6 ; jump to the lebel
    L5: SUB AL,37H
    L6:
        SHL EBX,4
        ADD BL,AL
        LOOP AGAIN; goto to the lebel
        MOV DATA1,EBX
        MOV AH,09
        MOV DX,OFFSET msg2
        INT 21H
        MOV EBX,0
        MOV CX,8
    AGAIN1:
        MOV AH,01 ;2nd NO. ENTERED
        INT 21H
        CMP AL,'A'
        JGE L7; jump to the lebel
        SUB AL,30H
        JMP L8; jump to the lebel
    L7: SUB AL,37H
    L8: SHL EBX,4
    ADD BL,AL
    LOOP AGAIN1; goto to the lebel
    ADD EBX,DATA1 ;ADDITION
    MOV AH,09
    MOV DX,OFFSET msg9
    INT 21H
    MOV CX,8
    AGAIN2:
        ROL EBX,4
        MOV DL,BL
        AND DL,0FH
        CMP DL,09
        JG L1 ; to o/p given no.
        ADD DL,30H
        JMP PRINT
    L1: ADD DL,37H
    PRINT: MOV AH,02
    INT 21H
    LOOP AGAIN2
    JMP ENDF

subtraction:
    MOV AH,09
    MOV DX,OFFSET msg1
    INT 21H
    MOV EBX,0
    MOV CX,8
    AGAIN21:
        MOV AH,01 ;1ST NO. ENTERED
        INT 21H
        CMP AL,'A'
        JGE L15; jump to the lebel
        SUB AL,30H
        JMP L16; jump to the lebel
```

```
  L15: SUB AL,37H
  L16:
    SHL EBX,4
    ADD BL,AL
    LOOP AGAIN21; goto to the lebel
    MOV DATA1,EBX
    MOV AH,09
    MOV DX,OFFSET msg2
    INT 21H
    MOV EBX,0
    MOV CX,8
  AGAIN3:
    MOV AH,01 ;2nd NO. ENTERED
    INT 21H
    CMP AL,'A'
    JGE L17
    SUB AL,30H
    JMP L18; jump to the lebel
  L17: SUB AL,37H
  L18:
    SHL EBX,4
    ADD BL,AL
    LOOP AGAIN3
    MOV DATA2, EBX
    MOV EBX, DATA1
    SUB EBX,DATA2 ;SUBTRACTION
    MOV AH,09
    MOV DX,OFFSET msg8
    INT 21H
    MOV CX,8
  AGAIN4:
    ROL EBX,4
    MOV DL,BL
    AND DL,0FH
    CMP DL,09
    JG L11 ; to o/p given no.
    ADD DL,30H
    JMP PRINT1
  L11: ADD DL,37H
  PRINT1:
    MOV AH,02
    INT 21H
    LOOP AGAIN4; goto to the lebel
  JMP ENDF

multiply:
```

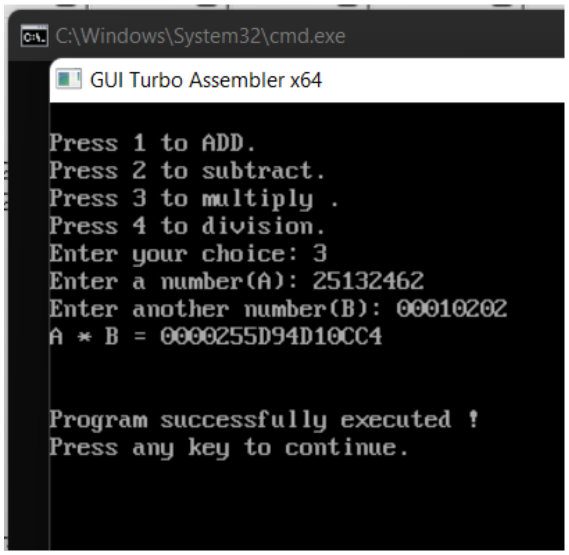
```
MOV AH,09
MOV DX,OFFSET msg1
INT 21H
MOV EBX,0
MOV CX,8
AGAIN5:
   MOV AH,01 ;1ST NO. ENTERED
   INT 21H
   CMP AL,'A'
   JGE L25
   SUB AL,30H
   JMP L26; jump to the lebel
L25: SUB AL,37H
L26:
   SHL EBX,4
   ADD BL,AL
   LOOP AGAIN5; goto to the lebel
   MOV DATA1,EBX
   MOV AH,09
   MOV DX,OFFSET msg2
   INT 21H
   MOV EBX,0
   MOV CX,8

AGAIN6:
   MOV AH,01 ;2nd NO. ENTERED
   INT 21H
   CMP AL,'A'
   JGE L27
   SUB AL,30H
   JMP L28
L27: SUB AL,37H
L28:
   SHL EBX,4
   ADD BL,AL
   LOOP AGAIN6
   MOV DATA2,EBX
   MOV EBX,0
   MOV EDX,0
   MOV EAX,0
   MOV EAX,DATA1
   MOV EBX,DATA2
   MUL EBX
   MOV PROD1,EDX
   MOV PROD2,EAX
   MOV AH,09
```

```
        MOV DX,OFFSET msg10
        INT 21H
        MOV EBX,PROD1
        MOV CX,8
    AGAIN7:
        ROL EBX,4
        MOV DL,BL
        AND DL,0FH ; to o/p the result
        CMP DL,9
        JBE L21; jump to the lebel
        ADD DL,37H
        MOV AH,02
        INT 21H
        JMP L22
    L21:
        ADD DL,30H
        MOV AH,02
        INT 21H
    L22:
        LOOP AGAIN7; goto to the lebel
        MOV EBX,PROD2
        MOV CX,8
    AGAIN8:
        ROL EBX,4
        MOV DL,BL
        AND DL,0FH ; to o/p the result
        CMP DL,9
        JBE L23
        ADD DL,37H
        MOV AH,02
        INT 21H
        JMP L24; jump to the lebel
    L23:
        ADD DL,30H
        MOV AH,02
        INT 21H
    L24:
        LOOP AGAIN8
        MOV AH,4CH
        INT 21H
    JMP ENDF ; JUMP TO ENDF LEVEL

division:
    MOV AH,09
    MOV DX,OFFSET msg1
    INT 21H
```

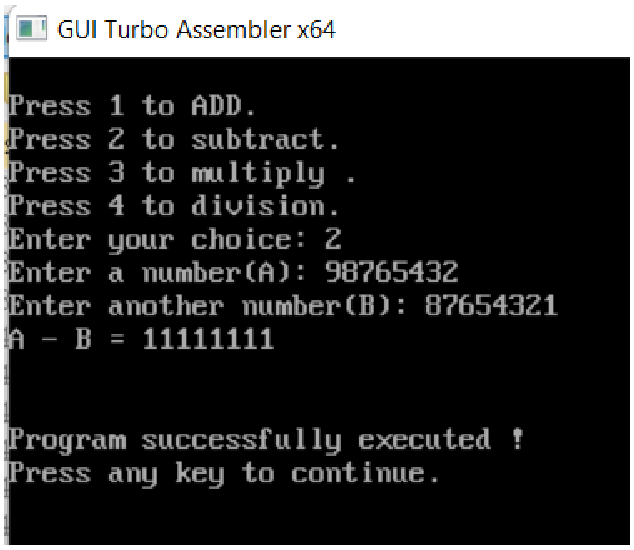
```
MOV EBX,0
MOV CX,8
AGAIN9:
    MOV AH,01 ;1ST NO. ENTERED
    INT 21H
    CMP AL,'A'
    JGE L35
    JMP L36
L35: SUB AL,37H
L36:
    SUB AL,30H
    SHL EBX,4
    ADD BL,AL
    LOOP AGAIN9; goto to the lebel
    MOV DATA1,EBX
    MOV AH,09
    MOV DX,OFFSET msg2
    INT 21H
    MOV EBX,0
    MOV CX,8
AGAIN10:
    MOV AH,01 ;2nd NO. ENTERED
    INT 21H
    CMP AL,'A'
    JGE L37
    SUB AL,30H
    JMP L38
L37: SUB AL,37H
L38:
    SHL EBX,4
    ADD BL,AL
    LOOP AGAIN10
    MOV DATA2,EBX
    MOV EBX,0
    MOV EDX,0
    MOV EAX,0
    MOV EAX,DATA1
    MOV EBX,DATA2
    DIV EBX
    MOV REM,EDX ;REM=REMAINDER
    MOV QUO,EAX ;QUO=QUOTIENT
    MOV AH,09
    MOV DX,OFFSET msg32
    INT 21H
    MOV EBX,REM
    MOV CX,8
```

```
   AGAIN11:
      ROL EBX,4
      MOV DL,BL
      AND DL,0FH ; to o/p the result in rem
      CMP DL,9
      JBE L31
      ADD DL,37H
      MOV AH,02
      INT 21H
      JMP L32; jump to the lebel
   L31:
      ADD DL,30H
      MOV AH,02
      INT 21H
   L32: LOOP AGAIN11
      MOV AH,09
      MOV DX,OFFSET msg33
      INT 21H
      MOV EBX,QUO
      MOV CX,8
   AGAIN12:
      ROL EBX,4
      MOV DL,BL
      AND DL,0FH ; to o/p the result in quo
      CMP DL,9
      JBE L33
      ADD DL,37H
      MOV AH,02
      INT 21H
      JMP L34; jump to the lebel
   L33:
      ADD DL,30H
      MOV AH,02
      INT 21H
   L34:
      LOOP AGAIN12; goto to the lebel
      MOV AH,4CH
      INT 21H
ENDF: .exit
end
```

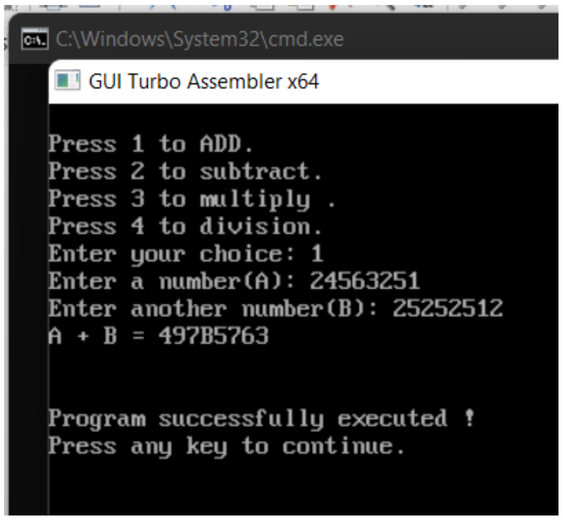
OUTPUT:



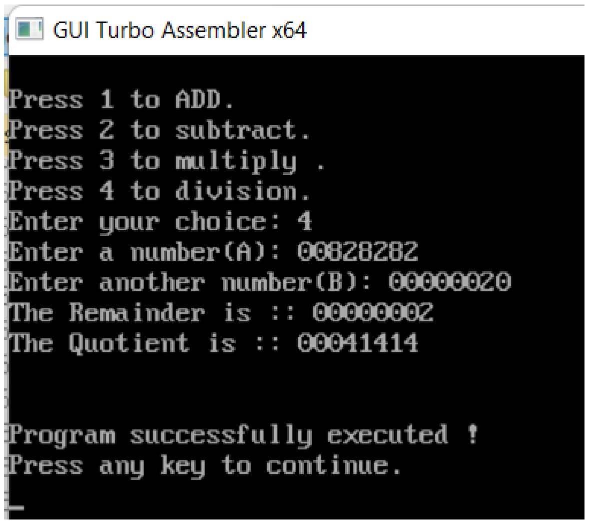
```
Press 1 to ADD.
Press 2 to subtract.
Press 3 to multiply .
Press 4 to division.
Enter your choice: 3
Enter a number(A): 25132462
Enter another number(B): 00010202
A * B = 0000255D94D10CC4


Program successfully executed !
Press any key to continue.
```



```
Press 1 to ADD.
Press 2 to subtract.
Press 3 to multiply .
Press 4 to division.
Enter your choice: 2
Enter a number(A): 98765432
Enter another number(B): 87654321
A - B = 11111111


Program successfully executed !
Press any key to continue.
```



```
Press 1 to ADD.
Press 2 to subtract.
Press 3 to multiply .
Press 4 to division.
Enter your choice: 1
Enter a number(A): 24563251
Enter another number(B): 25252512
A + B = 497B5763


Program successfully executed !
Press any key to continue.
```



```
Press 1 to ADD.
Press 2 to subtract.
Press 3 to multiply .
Press 4 to division.
Enter your choice: 4
Enter a number(A): 00828282
Enter another number(B): 00000020
The Remainder is :: 00000002
The Quotient is :: 00041414


Program successfully executed !
Press any key to continue.
_
```

Q2.-Write a program for 32-Bit BCD Addtion and Subtraction.

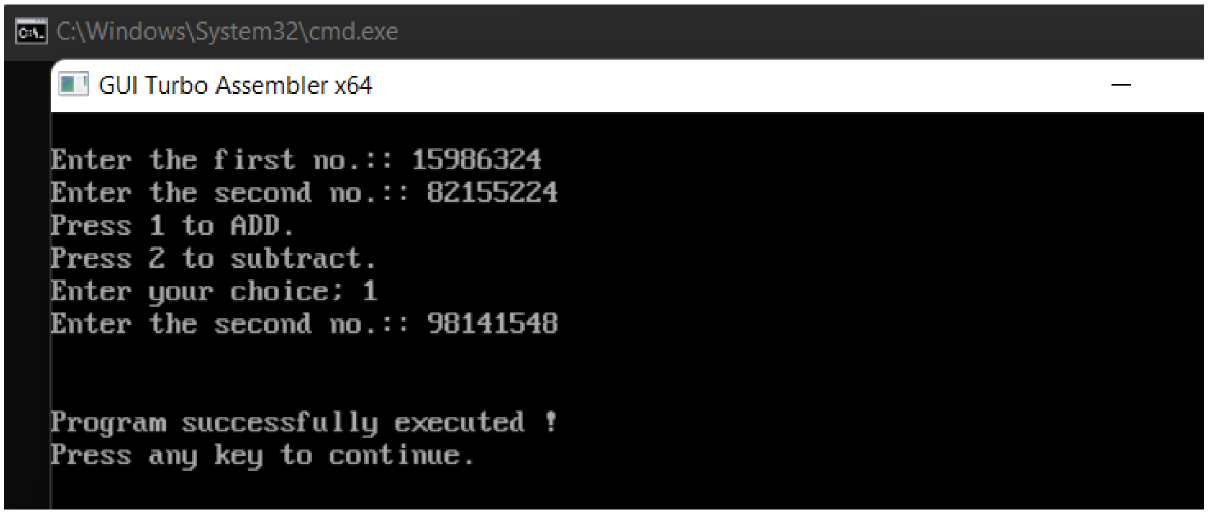
```
.model small; contain two segment data and code
   .386;instruction for 80386
   .data;data segment start
   num1 DD 00000000H;initialize memory with double word
   num2 DD 00000000H;initialize memory with double word
   num3 DD 00000000H;initialize memory with double word
   msg1 db 10,13,"Enter the first no.:: $"
   msg2 db 10,13,"Enter the second no.:: $"
   msg3 db 10,13,"Press 1 to ADD.$"
   msg4 db 10,13,"Press 2 to subtract.$"
   msg5 db 10,13,"Enter your choice; $"
   msg6 db 10,13,"A + B = $"
```

```
    msg7 db 10,13,"A - B = $"
.code ; start of code segmen
.startup; Generates program start-up code
    MOV AH,09
    MOV DX,OFFSET msg1
    INT 21H; Output a string terminated by '$’ stored in DX, as AH =9
    MOV EBX,0
    MOV CX,8
AGAIN:
    MOV AH,01 ;1ST NO. ENTERED
    INT 21H;input from user
    CMP AL,'A'
    JGE L2
    SUB AL,30H
    SHL EBX,4
    ADD BL,AL
    LOOP AGAIN
    MOV num1,EBX

    MOV AH,09
    MOV DX,OFFSET msg2
    INT 21H; Output a string terminated by '$’ stored in DX, as AH =9
    MOV EBX,0
    MOV CX,8
AGAIN1:
    MOV AH,01 ;2nd NO. ENTERED
    INT 21H;input from user
    CMP AL,'A'
    JGE L2
    SUB AL,30H
    SHL EBX,4
    ADD BL,AL
    LOOP AGAIN1
    MOV num2, EBX

mov ah,09
mov dx,offset msg3
int 21h; Output a string terminated by '$’ stored in DX, as AH =9

mov ah,09
mov dx,offset msg4
int 21h; Output a string terminated by '$’ stored in DX, as AH =9

mov ah,09
mov dx,offset msg5
int 21h; Output a string terminated by '$’ stored in DX, as AH =9
```

```
mov ah,01
int 21h; input from user

sub al,30h

cmp al,01h
je addition;jump to label addition

cmp al,02h
je subtraction;jump to label subtraction


addition:
   mov ax, word ptr num1
   mov dx, word ptr num2
   add al,dl
   daa
   mov bl,al

   mov al,ah
   adc al,dh
   daa
   mov bh,al

   mov word ptr num3,bx

   mov ax, word ptr num1+2
   mov dx, word ptr num2+2
   adc al,dl
   daa
   mov bl,al

   mov al,ah
   adc al,dh
   daa
   mov bh,al

   mov word ptr num3+2,bx
   mov ebx,num3



   mov ah, 09h
   mov dx, offset msg2
   int 21h; Output a string terminated by '$' stored in DX, as AH =9
```

```
    jnc l6
    mov ah, 02h
    mov dl, "1"
    int 21h

subtraction:
    mov ax, word ptr num1
    mov dx, word ptr num2
    sub al,dl
    daa
    mov bl,al

    mov al,ah
    sub al,dh
    daa
    mov bh,al

    mov word ptr num3,bx

    mov ax, word ptr num1+2
    mov dx, word ptr num2+2
    sub al,dl
    daa
    mov bl,al

    mov al,ah
    sub al,dh
    daa
    mov bh,al

    mov word ptr num3+2,bx
    mov ebx,num3



    mov ah, 09h
    mov dx, offset msg2
    int 21h; Output a string terminated by '$' stored in DX, as AH =9
    jnc l6
    mov ah, 02h
    mov dl, "1"
    int 21h

l6: MOV CX,8

AGAIN2:
    ROL EBX,4
    MOV DL,BL
    AND DL,0FH
    ADD DL,30H
    MOV AH,02
    INT 21H
    LOOP AGAIN2;go to label again2
L2: .EXIT
END
```

Output:



C:\Windows\System32\cmd.exe

GUI Turbo Assembler x64

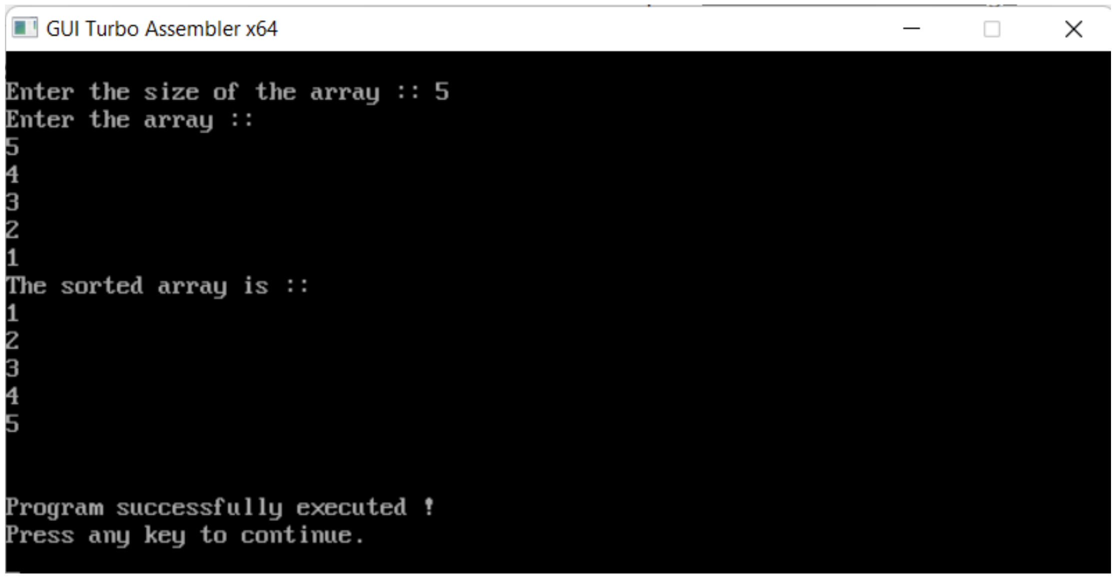
```
Enter the first no.:: 15986324
Enter the second no.:: 82155224
Press 1 to ADD.
Press 2 to subtract.
Enter your choice: 1
Enter the second no.:: 98141548


Program successfully executed !
Press any key to continue.
```

Q3.Write a program for Sorting.

```
.model small; contain two segment data and code
    .386;instruction for 80386
    .data;data segment start
        ARRAY DW 20 DUP (?);declaring aray with garbage
        DATA1 dw 0000H;initialize memory with word
NUMB DW 0000H
msg db 10,13,"Enter the size of the array :: $"
msg2 db 10,13,"Enter the array :: $"
msg3 db 10,13,"The sorted array is :: $"
.code ; start of code segmen
.startup; Generates program start-up code
MOV AH,09
MOV DX,OFFSET msg
INT 21H; Output a string terminated by '$' stored in DX, as AH =9
MOV AH,01
INT 21H
SUB AL,30H
MOV AH,0
MOV CX,AX
MOV DATA1,AX
MOV AH,09
MOV DX,OFFSET msg2
INT 21H; Output a string terminated by '$' stored in DX, as AH =9
MOV AH,0
MOV SI, 0
MOV BX, OFFSET ARRAY
L1: MOV DL, 0AH ; jump onto next line
MOV AH, 02H
INT 21H
MOV DX, SI ; input element of the array
MOV AH, 01H
INT 21H
SUB AL,30H; convert the hexadecimal digits into its equivalent ASCI
MOV SI, DX
MOV [BX + SI], AX ; store at memory location addressed by DS[BX+SI]
INC SI
LOOP L1

MOV CX, DATA1
MOV BX, OFFSET ARRAY ; store the offset address of array
MOV DI,CX
L2: MOV CX, DATA1
MOV NUMB, CX     ; Change1
DEC NUMB         ; Change2
MOV CX, NUMB     ; change3
MOV SI, 0
L3: MOV AL, [BX + SI]
CMP AL, [BX + SI + 1]; compare the value of content in AL and at DS[BX+SI+1]
JL L4
XCHG AL,[BX + SI + 1]; exchange the value of content in AL and at DS[BX+SI+1]
MOV [BX + SI],AL
L4: INC SI
LOOP L3
DEC DI
JNZ L2

MOV CX, DATA1
MOV SI, 0
MOV BX, OFFSET ARRAY
MOV AH,09
MOV DX,OFFSET msg3
INT 21H; Output a string terminated by '$' stored in DX, as AH =9
L5: MOV DL, 0AH ; jump onto next line
MOV AH, 02H
INT 21H
```

```
MOV DX, [BX + SI]
INC SI
ADD DL, 30H
MOV AH, 02
INT 21H ; Output a character present in DL , as AH value is 2
LOOP L5
.EXIT
END
```

Output:



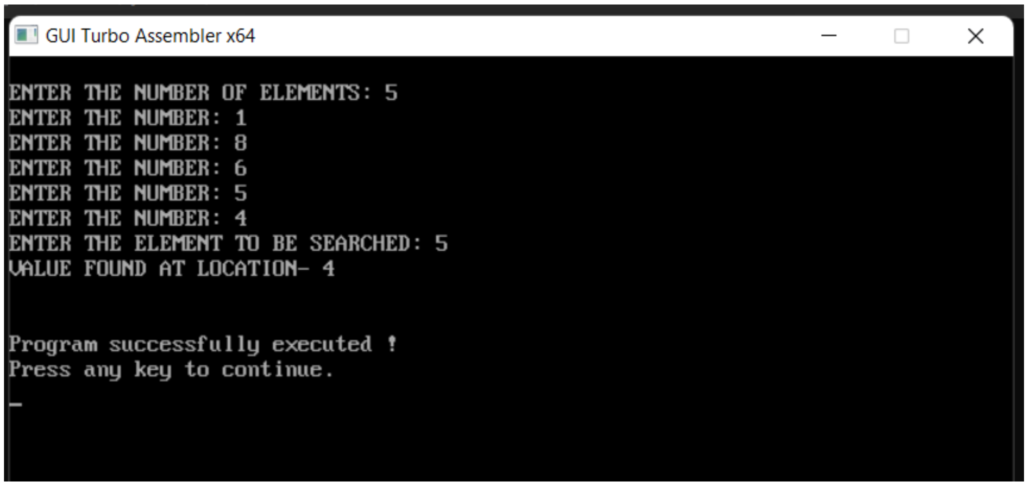
```
Enter the size of the array :: 5
Enter the array ::
5
4
3
2
1
The sorted array is ::
1
2
3
4
5


Program successfully executed !
Press any key to continue.
```

Q4.Write a program for Linear search and Binary Search.

```
.model small; contain two segment data and code
.stack ; tells the assembler to reserve storage
.386;instructions for the 80386 processor
.data ; start of data segment
ARRAY DB 10 DUP(?) ; Declaring an array with garbage
MESS0 DB 13,10,"ENTER THE NUMBER: $"
MESS1 DB 13,10,"ENTER THE NUMBER OF ELEMENTS: $"
MESS2 DB 13,10,"ENTER THE ELEMENT TO BE SEARCHED: $"
MESS3 DB 13,10,"VALUE FOUND AT LOCATION- $"
MESS4 DB 13,10,"VALUE NOT FOUND!!!$"
ErrMess DB 13,10,"ERROR IN INPUT DIGIT$"
DAT DB ? ; set byte size variable
number dw ? ; set double word variable

.code ; start of code segment
.startup; Generates program start-up code
MOV DX,OFFSET MESS1
MOV AH,09
INT 21H; Output a string terminated by '$’ stored in DX, as AH =9
```

```
MOV AH,01
INT 21H ; input from user
cmp al,39h
jbe abc; jump to abc , if al == 39h
MOV DX,OFFSET ErrMess
MOV AH,09
INT 21H; Output a string terminated by '$' stored in DX, as AH =9
jmp myexit; jump to myexit
abc:
and al,0fh
mov ah,0
mov number,ax; move data ax to number
MOV CX,AX
MOV DI,0
MYLOOP:
MOV DX,OFFSET MESS0
MOV AH,09
INT 21H; Output a string terminated by '$' stored in DX, as AH =9
MOV AH,01
INT 21H ; input from user
cmp al,39h
jbe abc2 ; jump if below or equal to
MOV DX,OFFSET ErrMess
MOV AH,09
INT 21H; Output a string terminated by '$' stored in DX, as AH =9
jmp myexit
abc2:
and al,0fh
MOV ARRAY[DI],AL
INC DI
LOOP MYLOOP
MOV DX,OFFSET MESS2
MOV AH,09
INT 21H; Output a string terminated by '$' stored in DX, as AH =9
MOV AH,01
INT 21H ; input from user
cmp al,39h
jbe abc3 ; jump if below or equal to
MOV DX,OFFSET ErrMess
MOV AH,09
INT 21H; Output a string terminated by '$' stored in DX, as AH =9
jmp myexit
abc3:
and al,0fh
MOV DAT,AL
mov ax,ds
mov es,ax
mov al,dat
CLD
mov cx,number
INC CX
mov DI, offset ARRAY
repne SCASB
CMP CX,0
JE NTFOUND
MOV DX,OFFSET MESS3
MOV AH,09
INT 21H; Output a string terminated by '$' stored in DX, as AH =9
SUB NUMBER,CX ;FIND ELEMENT LOCATION
ADD NUMBER,30H
MOV DX,NUMBER
INC DX
MOV AH,02
INT 21H
JMP myexit
NTFOUND:
MOV DX,OFFSET MESS4
MOV AH,09
INT 21H; Output a string terminated by '$' stored in DX, as AH =9
myexit:
MOV AH,4CH
INT 21H
END
```

Output:



```
ENTER THE NUMBER OF ELEMENTS: 5
ENTER THE NUMBER: 1
ENTER THE NUMBER: 8
ENTER THE NUMBER: 6
ENTER THE NUMBER: 5
ENTER THE NUMBER: 4
ENTER THE ELEMENT TO BE SEARCHED: 5
VALUE FOUND AT LOCATION- 4


Program successfully executed !
Press any key to continue.
_
```

Binary Search

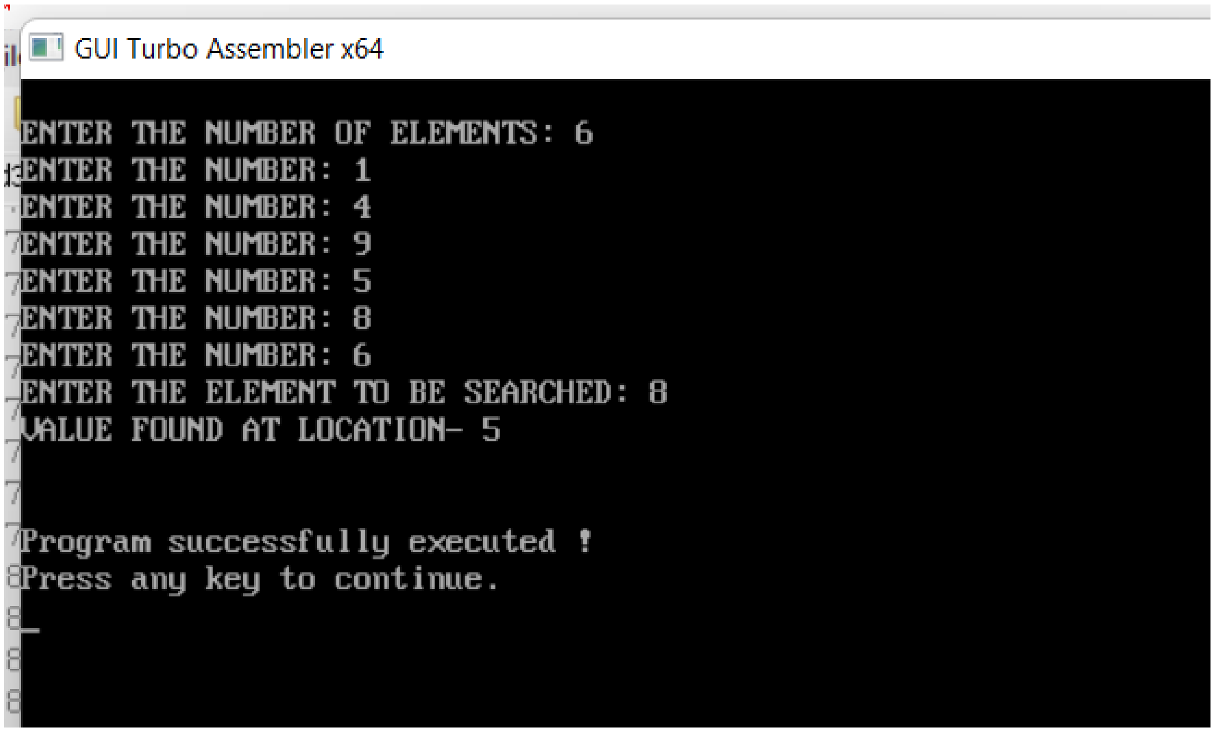
```
.model small
.stack
```

```
.386
.data
   ARRAY DB 10 DUP(?)
   MESS0 DB 13,10,"ENTER THE NUMBER: $"
   MESS1 DB 13,10,"ENTER THE NUMBER OF ELEMENTS: $"
   MESS2 DB 13,10,"ENTER THE ELEMENT TO BE SEARCHED: $"
   MESS3 DB 13,10,"VALUE FOUND AT LOCATION- $"
   MESS4 DB 13,10,"VALUE NOT FOUND!!!$"
   ErrMess DB 13,10,"ERROR IN INPUT DIGIT$"
   DAT DB ?
   number dw ?

.code
.startup
   MOV DX,OFFSET MESS1
   MOV AH,09
   INT 21H
   MOV AH,01
   INT 21H
   cmp al,39h
   jbe abc
   MOV DX,OFFSET ErrMess
   MOV AH,09
   INT 21H
   jmp myexit

abc:
and al,0fh
mov ah,0
mov number,ax
MOV CX,AX
MOV DI,0

MYLOOP:
   MOV DX,OFFSET MESS0
   MOV AH,09
   INT 21H
   MOV AH,01
   INT 21H
   cmp al,39h
   jbe abc2
   MOV DX,OFFSET ErrMess
   MOV AH,09
   INT 21H
   jmp myexit
```

```
abc2:
   and al,0fh
   MOV ARRAY[DI],AL
   INC DI
   LOOP MYLOOP
   MOV DX,OFFSET MESS2
   MOV AH,09
   INT 21H
   MOV AH,01
   INT 21H
   cmp al,39h
   jbe abc3
   MOV DX,OFFSET ErrMess
   MOV AH,09
   INT 21H
   jmp myexit

abc3:
   and al,0fh
   MOV DAT,AL
   mov ax,ds
   mov es,ax
   mov al,dat
   CLD
   mov cx,number
   INC CX
   mov DI, offset ARRAY
   repne SCASB
   CMP CX,0
   JE NTFOUND
   MOV DX,OFFSET MESS3
   MOV AH,09
   INT 21H

   SUB NUMBER,CX ;FIND ELEMENT LOCATION
   ADD NUMBER,30H
   MOV DX,NUMBER
   INC DX
   MOV AH,02
   INT 21H
   JMP myexit

NTFOUND:
   MOV DX,OFFSET MESS4
   MOV AH,09
   INT 21H
```

```
myexit:
   MOV AH,4CH
   INT 21H
END;
```

ouput



GUI Turbo Assembler x64

```
ENTER THE NUMBER OF ELEMENTS: 6
ENTER THE NUMBER: 1
ENTER THE NUMBER: 4
ENTER THE NUMBER: 9
ENTER THE NUMBER: 5
ENTER THE NUMBER: 8
ENTER THE NUMBER: 6
ENTER THE ELEMENT TO BE SEARCHED: 8
VALUE FOUND AT LOCATION- 5


Program successfully executed !
Press any key to continue.
_
```

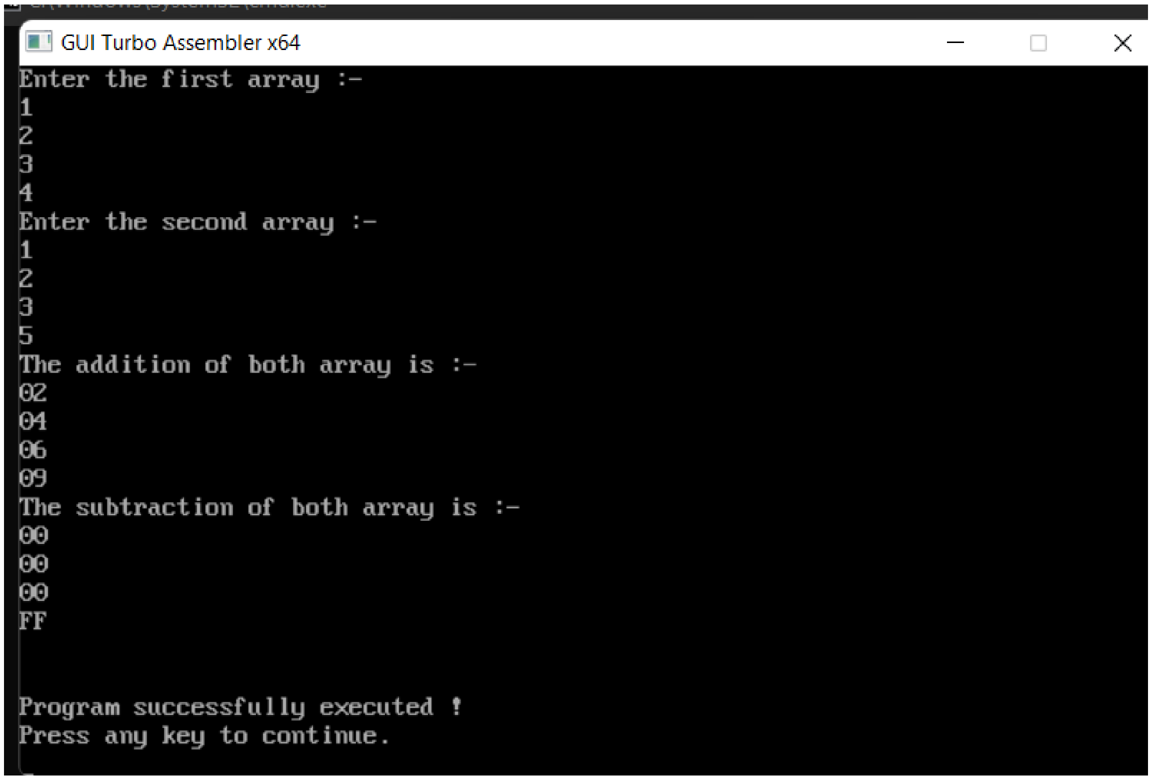
Q5.- Write a program to add and subtract two array.

```
.model small; contain two segment data and code
.386
.data;start of data segment
   A1 DB 20 DUP (?);declaring array
A2 DB 20 DUP (?)
A3 DB 20 DUP (?)
DATA1 dw 0000H
DATA2 DW 0000H
msg db 10,13,"Enter the size of the array one :- $"
msg2 db 10,13,"Enter the first array :- $"

msg4 db 10,13, "Enter the second array :-$"
msg5 db 10,13, "The addition of both array is :-$"
msg6 db 10,13, "The subtraction of both array is :-$"
.code;start code segment
.startup
MOV AH,09
MOV DX,OFFSET msg
INT 21H

MOV CX, 2
L4: MOV AH,01
INT 21H
CMP AL,'A'
JGE L9
SUB AL,30H
JMP L8
L9: SUB AL,37H
L8: SHL BX, 4
ADD BL, AL
LOOP L4
MOV AL, BL
MOV CL, AL
MOV AH, 0
MOV DATA1, AX
MOV CX, DATA1


MOV AH,09
MOV DX,OFFSET msg2
INT 21H
;MOV AH,0

MOV CX, DATA1
LEA SI, A1
L1: MOV DL, 0AH ; jump onto next line
MOV AH, 02H
INT 21H
MOV AH, 01H
INT 21H
SUB AL,30H
MOV [SI], AL
INC SI
LOOP L1


MOV CX, DATA1

MOV AH,09
MOV DX,OFFSET msg4
INT 21H
MOV AH,0
```

```
LEA DI, A2
L3: MOV DL, 0AH ; jump onto next line
MOV AH, 02H
INT 21H

MOV AH, 01H
INT 21H
SUB AL,30H

MOV [DI], AL
INC DI
LOOP L3


LEA SI, A3
LEA DI, A1

MOV CX, DATA1
CPYA: MOV AL, [DI]
MOV [SI], AL
INC DI
INC SI
LOOP CPYA

LEA SI, A1
LEA DI, A2

MOV CX, DATA1
ADDA: MOV AL, [SI]
ADD AL, [DI]
MOV [SI], AL
INC DI
INC SI
LOOP ADDA

MOV AH, 09H
MOV DX, OFFSET MSG5
INT 21H

MOV CX, DATA1
LEA SI, A1
L5:mov ah, 02h
mov dl, 0ah
int 21h
MOV DATA2, CX
MOV CX, 2
MOV BL, [SI]
ADDA1: ROL BL, 4; rotates the bits within the destination operand to the left
MOV DL, BL
AND DL, 0FH
CMP Dl, 9
JA L6
ADD DL, 30h
JMP L7
L6: ADD DL, 37H
L7: MOV AH, 02
INT 21H
LOOP ADDA1
MOV CX, DATA2
INC SI
LOOP L5


LEA SI, A3
LEA DI, A2

MOV CX, DATA1
SUBA: MOV AL, [SI]
SUB AL, [DI]
MOV [SI], AL
INC DI
INC SI
LOOP SUBA

MOV AH, 09H
MOV DX, OFFSET MSG6
INT 21H

MOV CX, DATA1
LEA SI, A3
L18:mov ah, 02h
mov dl, 0ah
int 21h
MOV DATA2, CX
MOV CX, 2
MOV BL, [SI]
SUBA1: ROL BL, 4; rotates the bits within the destination operand to the left
MOV DL, BL
AND DL, 0FH
CMP Dl, 9
JA L19
ADD DL, 30h
```

```
JMP L20
L19: ADD DL, 37H
L20: MOV AH, 02
INT 21H
LOOP SUBA1
MOV CX, DATA2
INC SI
LOOP L18

.EXIT
END
```

output:

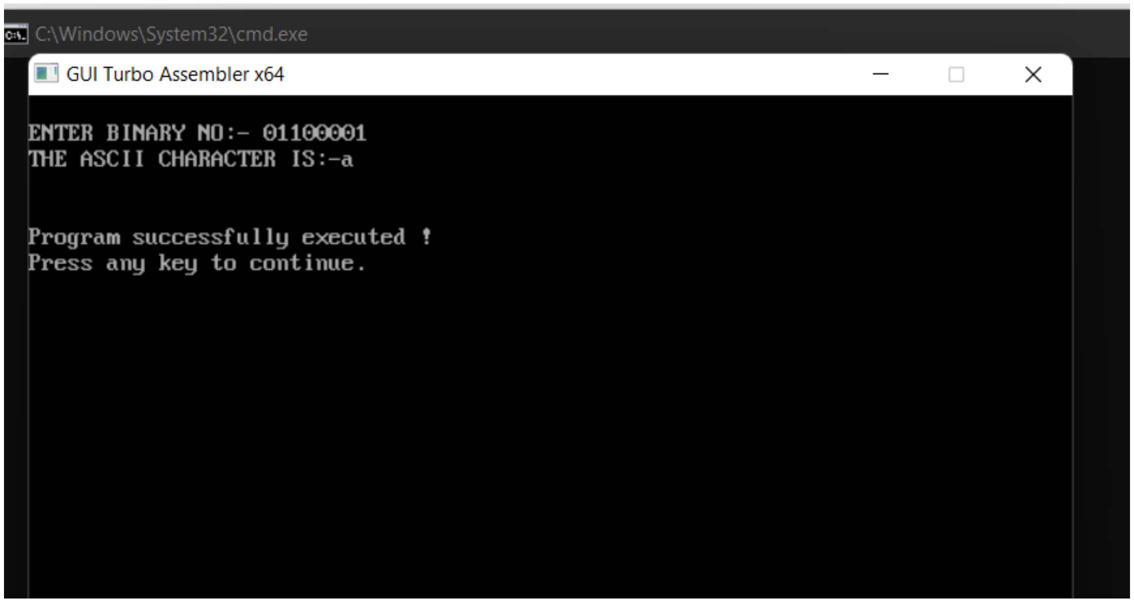


```
Enter the first array :-
1
2
3
4
Enter the second array :-
1
2
3
5
The addition of both array is :-
02
04
06
09
The subtraction of both array is :-
00
00
00
FF


Program successfully executed !
Press any key to continue.
```

Q6.Write a program for binary to ascii conversion.

```
.MODEL SMALL;contain two segment data and code
   .DATA;start of data segment
      INPUT DB 10,13 , 'ENTER BINARY NO:- $';10 is the ascii control code for line fed while 13
;is the code for carriage return
OUTPUT DB 10,13, 'THE ASCII CHARACTER IS:-$'

.CODE;start of code segment
   .STARTUP;generates program start up code
MOV AH,09H
MOV DX,OFFSET INPUT
INT 21H; Output a string terminated by '$' stored in DX, as AH =9
MOV BL, 00H
MOV CL,08H
INPUT1: MOV AH,01H
INT 21H;input from user
SUB AL,30H
SHL BL,1
ADD BL,AL
LOOP INPUT1;go to label input
MOV AH,09H
LEA DX,OUTPUT
INT 21H; Output a string terminated by '$' stored in DX, as AH =9
MOV AH,02H
MOV DL,BL
INT 21H ; Output a character present in DL , as AH value is 2
.EXIT
END
```

Output:



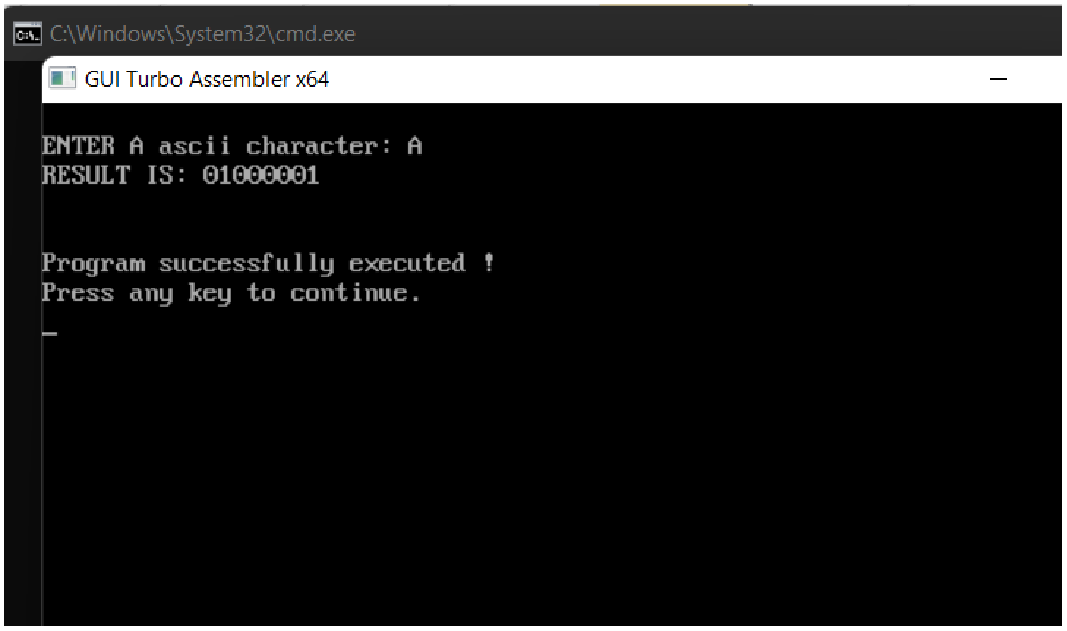
```
ENTER BINARY NO:- 01100001
THE ASCII CHARACTER IS:-a


Program successfully executed !
Press any key to continue.
```

Q7.Write a program for ascii to binary conversion

```
.MODEL SMALL; contain two segment data and code
.DATA; start of data segment
MESG DB 10,13, 'ENTER A ascii character: $'
RESULT DB 10,13, 'RESULT IS: $'; 10 is the ASCII control code for line feed while 13 is the
code for
;carriage return
```

```
.CODE; start of code segment
.STARTUP

MOV DX,OFFSET MESG;loading the offset address
MOV AH,09H
INT 21H

MOV AH,01H
INT 21H;input from user
MOV BL,AL

MOV DX,OFFSET RESULT
MOV AH,09H
INT 21H ;Output a string terminated by '$' stored in DX, as AH =


MOV CL,08H
MOV AH,00H
MOV AL,BL
L1: SHL AL, 01H
MOV BL,AL
MOV AL,00H
ADC AL,30H
MOV DL,AL
MOV AH,02H
INT 21H ;Output a string terminated by '$' stored in DX, as AH value is 2

MOV AL,BL
LOOP L1
.EXIT
END
```

Output:



C:\Windows\System32\cmd.exe

GUI Turbo Assembler x64

```
ENTER A ascii character: A
RESULT IS: 01000001


Program successfully executed !
Press any key to continue.
_
```