KALINDI COLLEGE

UNIVERSITY OF DELHI

NAME: Prachi Aggarwal

COURSE: Bsc(H) Computer Science

ROLLNO: 21570015

Unique Paper Code: 32347504

University Rno: 21033570042

SUBJECT: Microprocessor

PRACTICAL FILE

ASSEMBLY LANGUAGE PROGRAMS

Q1. Write a program to take two single-digit numbers as input, and then display their single-digit sum and difference.

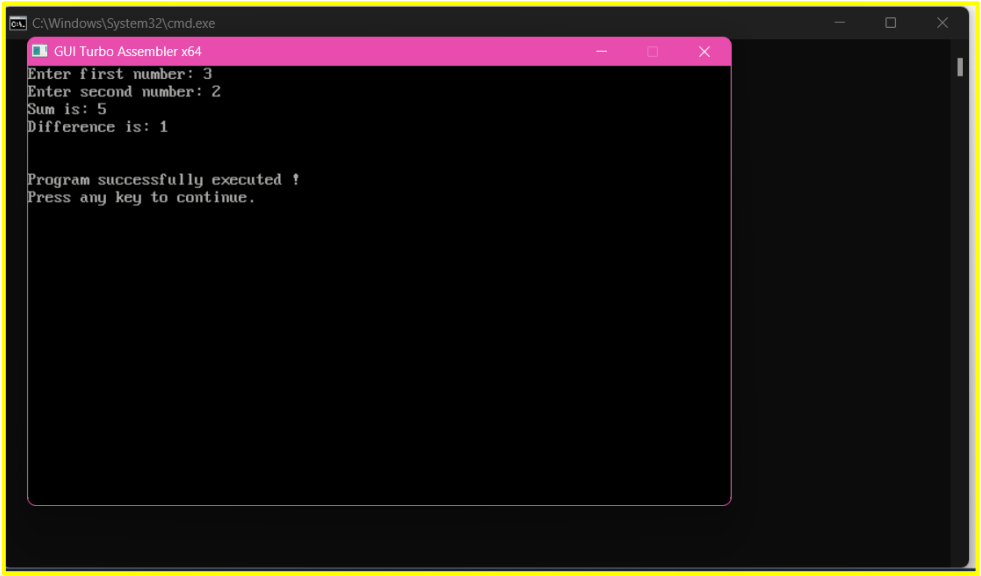
```
.model small
.data
   MSG1 db "Enter first number: $"
   MSG2 db 10,"Enter second number: $"
   MSG3 db 10,"Sum is: $"
   MSG4 db 10,"Difference is: $"
   N1 db ?
   N2 db ?

.code
.startup
   MOV DX,OFFSET MSG1
   MOV AH,09H
   INT 21H

   MOV AH,01H
   INT 21H

   MOV N1,AL

   MOV DX,OFFSET MSG2
   MOV AH,09H
   INT 21H
```

```
MOV AH,01H
INT 21H

MOV N2,AL

MOV DX,OFFSET MSG3
MOV AH,09H
INT 21H

MOV DL,N1
ADD DL,N2
SUB DL,30H

MOV AH,02H
INT 21H

MOV DX,OFFSET MSG4
MOV AH,09H
INT 21H

MOV DL,N1
SUB DL,N2
ADD DL,30H

MOV AH,02H
INT 21H
```

.exit

END

OUTPUT

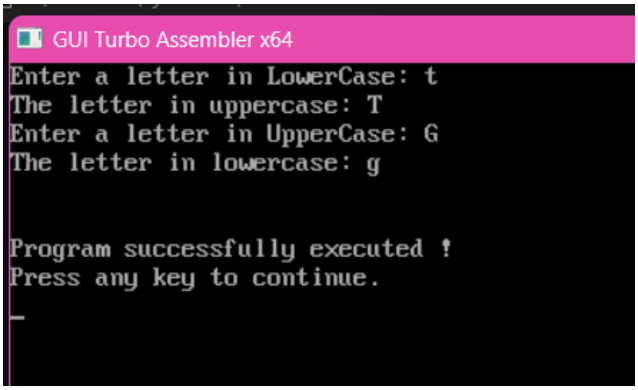


Q2. Write a program to take a character input in small case, and print the capital case of that character, and vice-versa.

```
.model small
.data
   MSG1 db "Enter a letter in LowerCase: $"
   MSG2 db 10,"Enter a letter in UpperCase: $"
   RSLT1 db 10,"The letter in uppercase: $"
   RSLT2 db 10,"The letter in lowercase: $"
.code
.startup
   MOV DX, OFFSET MSG1
   MOV AH,09H
   INT 21H

   MOV AH,01H
   INT 21H

   MOV DX,OFFSET RSLT1
   MOV AH,09H
   INT 21H

   MOV DL,AL
   SUB DL,20H

   MOV AH,02H
   INT 21H
```

```
    MOV DX, OFFSET MSG2
    MOV AH,09H
    INT 21H

    MOV AH,01H
    INT 21H

    MOV DX,OFFSET RSLT2
    MOV AH,09H
    INT 21H

    MOV DL,AL
    ADD DL,20H

    MOV AH,02H
    INT 21H

.exit
END
```

OUTPUT



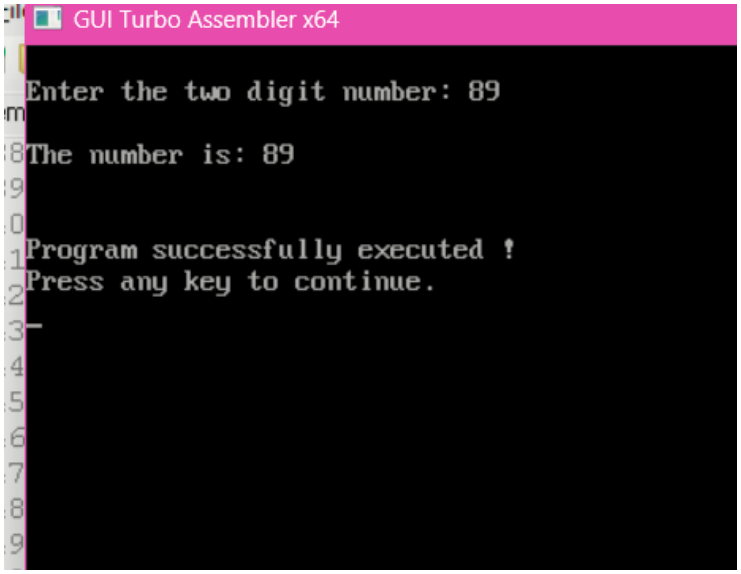
Q3. Write a program to take a two-digit number from the user and print it.

```
.model small
.data
   N dw ?
   MSG1 db 10,"Enter the two digit number: $"
   MSG2 db 10,"The number is: $"
.code
.startup
   MOV DX,OFFSET MSG1
   MOV AH,09H
   INT 21H

   MOV DI,OFFSET N
   CALL takeInput

   MOV DX,OFFSET MSG2
   MOV AH,09H
   INT 21H
   CALL PRINT
.exit
PRINT PROC NEAR
   MOV AX,N
   MOV CL,10
   DIV CL
   cmp AL,0
   JE L1
```

```
    PUSH AX
    MOV DL,AL
    ADD DL,30H
    MOV AH,02H
    INT 21H
    POP AX
    L1:
    MOV DL,AH
    ADD DL,30H
    MOV AH,02H
    INT 21H
    RET
PRINT ENDP
takeInput PROC NEAR
    MOV BX,0H
    ScanNum:
        MOV AH,01H
        INT 21H

        CMP AL,13
        JE EXIT

        SUB AL,30H
        MOV AH,0H

        PUSH AX
```

```
        MOV AL,BL
        MOV DL,10
        MUL DL

        POP DX

        ADD AX,DX
        MOV BX,AX

        JMP ScanNum

    EXIT : MOV [DI],BX
    RET
takeInput ENDP
END
```

OUTPUT



Q4. Write a program to take an array from the user and display it.

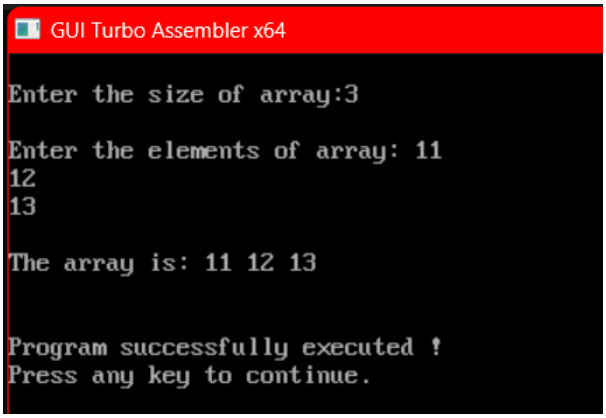
```
.model small
.data
        N1 dw ?
        ARRAY1 dw 60 DUP(?)
        MSG1 db 10,"Enter the size of array:$ "
        MSG2 db 10,"Enter the elements of array: $"
        MSG3 db 10,"The array is: $"
.code
.startup
        MOV DX,OFFSET MSG1
        MOV AH,09H
        INT 21H

        MOV DI,OFFSET N1
        CALL takeInput

        MOV DX,OFFSET MSG2
        MOV AH,09H
        INT 21H

        MOV DI,OFFSET ARRAY1
        MOV CX,N1
        CALL takeArrayInput

        MOV DX,OFFSET MSG3
```

```
        MOV AH,09H
        INT 21H

        MOV DI,OFFSET ARRAY1
        MOV CX,N1

        CALL arrayPrint
.exit
        takeArrayInput PROC NEAR
                L1: CALL takeInput
                        INC DI
                        INC DI
                        LOOP L1
                RET
        takeArrayInput ENDP

        arrayPrint PROC NEAR
                L2:CALL PRINT
                        MOV DL,' '
                        MOV AH,02H
                        INT 21H
                        INC DI
                        INC DI
                        LOOP L2
                RET
        arrayPrint ENDP
```

```
PRINT PROC NEAR
        MOV AX,[DI]
        MOV DL,10
        DIV DL
        PUSH AX
        MOV DL,AL
        ADD DL,30H
        MOV AH,02H
        INT 21H
        POP AX
        MOV DL,AH
        ADD DL,30H
        MOV AH,02H
        INT 21H
        RET
PRINT ENDP

takeInput PROC NEAR
        MOV BX,0H
        ScanNum:
                MOV AH,01H
                INT 21H

                CMP AL,13
                JE EXIT
```

```
                SUB AL,30H
                MOV AH,0H

                PUSH AX
                MOV AL,BL
                MOV DL,10
                MUL DL

                POP DX

                ADD AX,DX
                MOV BX,AX

                JMP ScanNum

        EXIT : MOV [DI],BX
        RET
    takeInput ENDP
END
```

OUTPUT



Q5. Write a program to take two arrays from the user and display their sum and difference.

```
.model small
.data
   N1 dw ?
   MSG1 db 10,"Enter the size of arrays:$ "
   MSG2 db 10,"Enter the elements of array1: $"
   MSG3 db 10,"The array1 is: $"
   MSG4 db 10,"The difference of the arrays is: $ "
   MSG5 db 10,"Enter the elements of array2: $"
   MSG6 db 10,"The array2 is: $"
   MSG7 db 10,"The sum of array is: $"
   ARRAY1 dW 60 DUP(?)
   ARRAY2 dw 60 DUP(?)
   SUM dw 60 DUP(?)
   SUBB dw 60 DUP(?)
.code
.startup
   MOV DX,OFFSET MSG1
   MOV AH,09H
   INT 21H

   MOV DI,OFFSET N1
   CALL takeInput

   MOV DX,OFFSET MSG2
   MOV AH,09H
```

```
INT 21H

MOV DI,OFFSET ARRAY1
MOV CX,N1
CALL takeArrayInput

MOV DX,OFFSET MSG3
MOV AH,09H
INT 21H

MOV DI,OFFSET ARRAY1
MOV CX,N1

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

MOV DI,OFFSET ARRAY2
MOV CX,N1
CALL takeArrayInput

MOV DI,OFFSET ARRAY2
MOV SI,OFFSET array1
```

```
MOV BP, OFFSET SUM
MOV CX,N1
CALL addArrays

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

MOV DI,OFFSET SUM
MOV CX,N1
CALL arrayPrint

MOV DI,OFFSET ARRAY2
MOV SI,OFFSET array1
MOV BP, OFFSET SUBB
MOV CX,N1
CALL subArrays

MOV DX,OFFSET MSG4
MOV AH,09H
INT 21H

MOV DI,OFFSET SUBB
MOV CX,N1
CALL arrayPrint
```

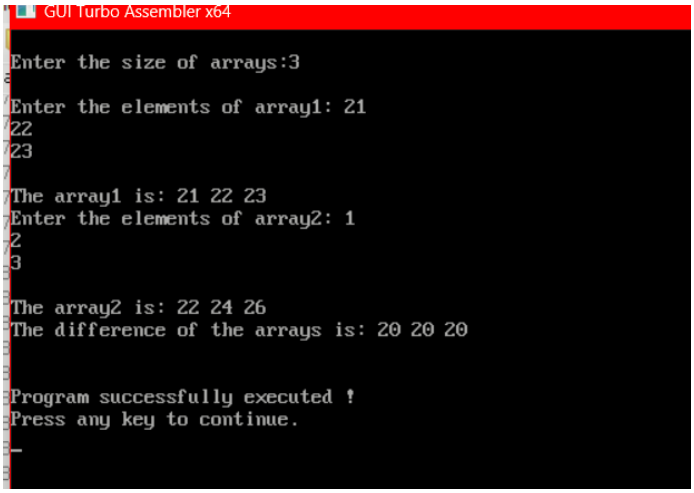
```
.exit
  subArrays PROC NEAR
    L4:
      MOV BX,[DI]
      MOV AX,[SI]
      SUB AX,BX
      mov [bp],ax
      INC SI
      INC SI
      INC DI
      INC DI
      INC BP
      INC BP
      LOOP L4
    RET
  subArrays ENDP
  addArrays PROC NEAR
    L3:
      MOV BX,[DI]
      MOV AX,[SI]
      ADD AX,BX
      mov [bp],ax
      INC SI
      INC SI
      INC DI
      INC DI
```

```
        inc bp
        inc bp
        LOOP L3
     RET
addArrays ENDP

 takeArrayInput PROC NEAR

    L1: CALL takeInput
        INC DI
        INC DI
        LOOP L1
    RET
takeArrayInput ENDP

arrayPrint PROC NEAR
    L2:CALL PRINT
        MOV DL,' '
        MOV AH,02H
        INT 21H
        INC DI
        INC DI
        LOOP L2
    RET

arrayPrint ENDP
```

```
PRINT PROC NEAR
   MOV AX,[DI]
   MOV DL,10
   DIV DL
   PUSH AX
   MOV DL,AL
   ADD DL,30H
   MOV AH,02H
   INT 21H
   POP AX
   MOV DL,AH
   ADD DL,30H
   MOV AH,02H
   INT 21H

   RET
PRINT ENDP
takeInput PROC NEAR
   MOV BX,0H
   ScanNum:

      MOV AH,01H
      INT 21H

      CMP AL,13
      JE EXIT
```

```
        SUB AL,30H
        MOV AH,0H

        PUSH AX

        MOV AL,BL
        MOV DL,10
        MUL DL

        POP DX

        ADD AX,DX
        MOV BX,AX

        JMP ScanNum

    EXIT : MOV [DI],BX
    RET
    takeInput ENDP
END
```

OUTPUT


GUI Turbo Assembler x64
Enter the size of arrays:3
Enter the elements of array1: 21
22
23
The array1 is: 21 22 23
Enter the elements of array2: 1
2
3
The array2 is: 22 24 26
The difference of the arrays is: 20 20 20
Program successfully executed !
Press any key to continue.

Q6. Write a program to take an array from the user and perform linear search on it.

```
.model small
 .data
   ARR1 dW 80 DUP(?)
   N db ?
   M dw ?
   P dw ?
   NUM db 10,"Enter number to be search: $"
   INP db 10,"Enter size of the array: $"
   INPARR dB 10,"Enter elements of array: $"
   MSG db 10,"Not Found: $"
   RSLT db 10,"Found at Index no.: $"
.code
.startup
   MOV DX,OFFSET INP
   MOV AH,09H
   INT 21H

   mov DI, OFFSET N
   CALL takeInput

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

   MOV DI,OFFSET ARR1
```

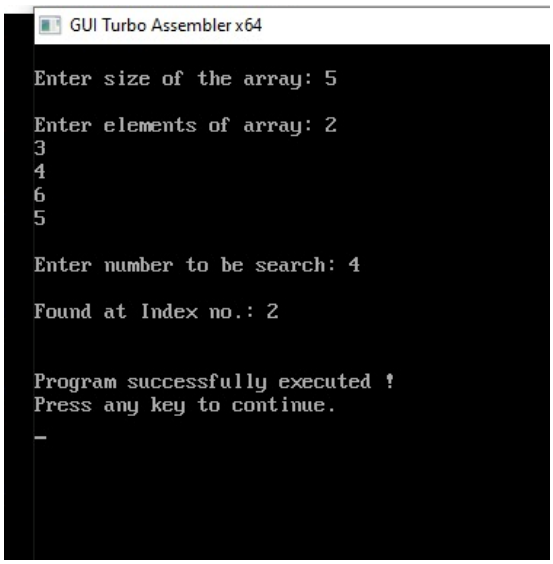
```
    MOV Cl,N
    mov ch,0
    CALL takeArrayInput

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

    MOV DI, OFFSET M
    CALL takeInput

    CALL search

    LAB: MOV DX,OFFSET RSLT
         MOV AH,09H
         INT 21H
         MOV DX,SI
         ADD DX,48
         MOV AH,02H
         INT 21H
         JMP EXIIT
   EXIIT:
.exit
    search PROC NEAR
       MOV BX,0
       MOV CL,N
```

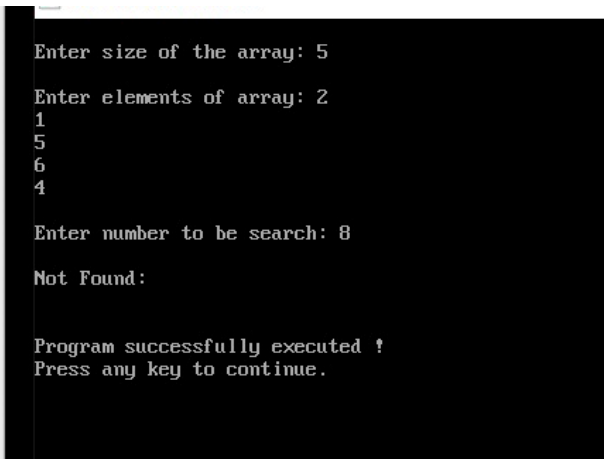
```
    MOV CH,0
    MOV SI,0
      L5:
        MOV AX,M
        MOV DX,ARR1[BX]
        CMP DX,AX
        JE LAB
        INC BX
        INC BX
        INC SI
        LOOP L5
        MOV DX,OFFSET MSG
        MOV AH,09H
        INT 21H
        JMP EXIIT

    RET
search ENDP

takeInput PROC NEAR
    MOV BX,0H
    ScanNum:

      MOV AH,01H
      INT 21H
```

```
    CMP AL,13
    JE EXIT

    SUB AL,30H
    MOV AH,0H

    PUSH AX

    MOV AL,BL
    MOV DL,10
    MUL DL

    POP DX

    ADD AX,DX
    MOV BX,AX

    JMP ScanNum

EXIT : MOV [DI],BX
RET
takeInput ENDP

takeArrayInput PROC NEAR

    L1: CALL takeInput
```

```
        INC DI
        INC DI
        LOOP L1
        RET
    takeArrayInput ENDP
END
```

OUTPUT





Q7: Write a program to take an array from the user and perform binary search on it.

```
.MODEL SMALL
.DATA
MSSG1 DB "ENTER THE SIZE OF ARRAY : $"
MSSG2 DB 10, 13 , "ENTER THE ELEMENT OF ARRAY (IN SORTED ORDER): $"
MSSG3 DB 10, 13, "ENTER THE ELEMENT TO BE SEARCHED : $"
MSSG4 DB 10, 13, "ELEMENT FOUND AT POSITION : $"
MSSG5 DB 10, 13, "ELEMENT NOT FOUND $"
ARR DB 20 DUP(?)
N DB ?
P DB ?
COUNT DW ?

.CODE
.STARTUP
MOV AH ,09H
MOV DX ,OFFSET MSSG1
INT 21H
MOV AH ,01H
INT 21H
SUB AL ,30H
MOV N , AL
MOV CH , 0
MOV SI , 0
MOV CL , N
MOV COUNT, CX
```

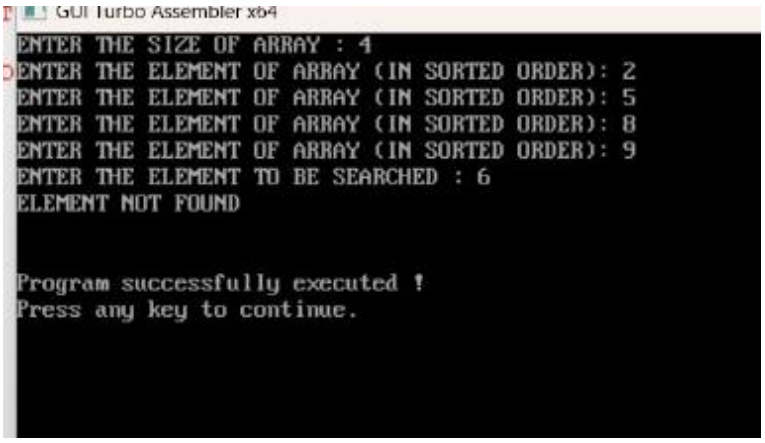
```
INPUT:
MOV AH , 09H
MOV DX , OFFSET MSSG2
INT 21H
MOV AH , 01H
INT 21H
MOV ARR[SI] , AL
INC SI
LOOP INPUT
MOV AH , 09H
MOV DX , OFFSET MSSG3
INT 21H
MOV AH , 01H
INT 21H
MOV P , AL
MOV BX , OFFSET ARR
MOV CH , 0
MOV Cl , n
MOV DI , 0
MOV AH , 0
MOV AL , 0
DEC SI
OUTPUT:
DEC COUNT
ADD AX , CX
MOV DL , 2
```

```
DIV DL
 MOV AH , 0
MOV DI , AX
MOV AL , BX[DI]
CMP COUNT , SI
JG NF
CMP P ,AL
JG GREATER
CMP P , AL
JL LESS
CMP P , AL
JE FOUND
NF:
MOV AH , 09H
MOV DX , OFFSET MSSG5
INT 21H
.EXIT

GREATER:
;INC DI
MOV AX , DI
JMP OUTPUT

LESS:
MOV AH , 0
MOV AL , 0
```

```
;DEC DI
MOV CX , DI
 JMP OUTPUT

FOUND:
MOV AH , 09H
MOV DX , OFFSET MSSG4
INT 21H
MOV DX , DI
INC DX
MOV AX,DX
CALL DISPH

.EXIT

DISPH PROC NEAR
MOV CL,4
MOV CH,4

DISPH1:
ROL AX,CL
PUSH AX
AND AL, 0FH
ADD AL,30H
CMP AL, '9'
JBE DISPH2
```

```
ADD AL,7
DISPH2:
MOV AH,2
 MOV DL,AL
INT 21H
POP AX
DEC CH
JNZ DISPH1
RET
DISPH ENDP

.EXIT
END
```

OUTPUT



Q8: Write a program to take an array from the user and perform bubble sort on it.

```
.model small
 .data
   ARR1 db 80 DUP(?)
   N Db ?
   NUM db 10,"Array after bubble sort: $"
   INP db 10,"Enter size of the array: $"
   INPARR dB 10,"Enter elements of array: $"

.code
.startup
   ;size of array message
   MOV DX,OFFSET INP
   MOV AH,09H
   INT 21H

   ;size of array
   mov DI, OFFSET N
   CALL takeInput

   ;array elements message
   MOV DX,OFFSET INPARR
   MOV AH,09H
   INT 21H

   ;array elements
```

```
    MOV DI,OFFSET ARR1
    mov cl,N
    mov ch,0
    CALL takeArrayInput

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

    CALL sort

        MOV DI,OFFSET ARR1
        mov cl,N
        mov ch,0
        CALL arrayPrint


.exit
    sort PROC NEAR
        mov cl,N
        mov ch,0
        dec cx
        outer:
            Push cx
            mov cl,N
            mov ch,0
```

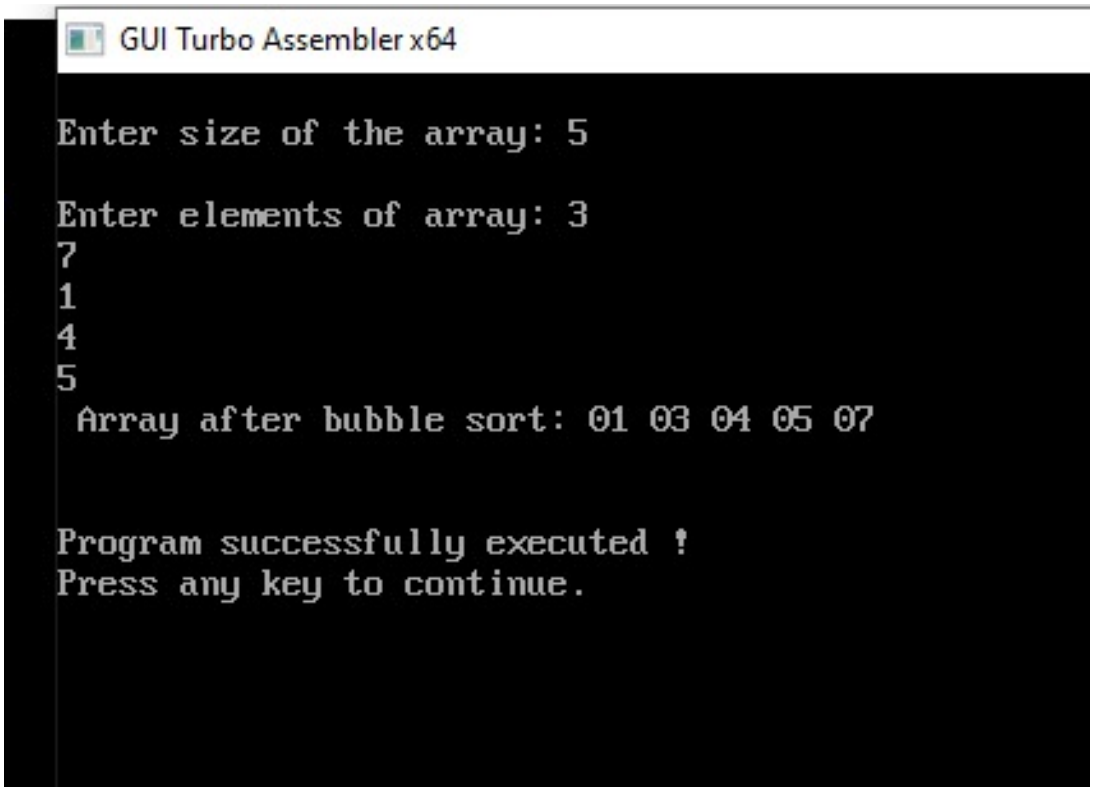
```
        dec cx
        inner:
            mov bx,cx
            mov al,ARR1[bx]
            dec bx
            mov dl,ARR1[bx]
            cmp al,dl
            ja noOp

            xchg al,dl
            mov ARR1[bx],dl
            inc bx
            mov ARR1[bx],al
            noOp:Loop inner
        Pop cx
        loop outer

    RET
sort ENDP

takeInput PROC NEAR
    MOV BX,0H
    ScanNum:

        MOV AH,01H
        INT 21H
```

```
    CMP AL,13
    JE EXIT

    SUB AL,30H
    MOV AH,0H

    PUSH AX

    MOV AL,BL
    MOV DL,10
    MUL DL

    POP DX

    ADD AX,DX
    MOV BX,AX

    JMP ScanNum

EXIT : MOV [DI],BX
RET
takeInput ENDP

takeArrayInput PROC NEAR
```

```
    L1: CALL takeInput
        INC DI
        LOOP L1
        RET
takeArrayInput ENDP
arrayPrint PROC NEAR
L2:CALL PRINT
    MOV DL,' '
    MOV AH,02H
    INT 21H
    INC DI
    LOOP L2
RET

arrayPrint ENDP

PRINT PROC NEAR
    MOV AL,[DI]
    mov ah,0
    MOV DL,10
    DIV DL
    PUSH AX
    MOV DL,AL
    ADD DL,30H
    MOV AH,02H
    INT 21H
```

```
        POP AX
        MOV DL,AH
        ADD DL,30H
        MOV AH,02H
        INT 21H

        RET
    PRINT ENDP

END
```

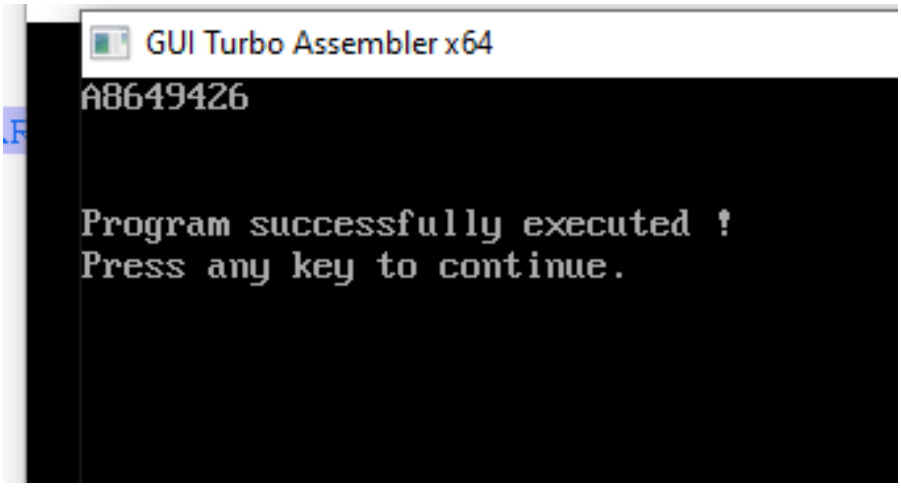
OUTPUT



Q9: Write a program for 32-bit BCD addition and subtraction.

```
.MODEL TINY
.CODE
.STARTUP
MOV AX,3214H
MOV BX,6212H
MOV CX,4321H
MOV DX,6543H

ADD AX,BX
ADC DX,CX

MOV BX,AX
MOV AX,DX

CALL DISPH

MOV AX,BX
CALL DISPH

.EXIT

DISPH PROC NEAR
MOV CL,4
MOV CH,4

DISPH1:
ROL AX,CL
PUSH AX
AND AL,0FH
ADD AL,30H
CMP AL,'9'
JBE DISPH2
ADD AL,7

DISPH2:
MOV AH,2
```

```
MOV DL,AL
INT 21H
POP AX
DEC CH
JNZ DISPH1
RET
DISPH ENDP
END
```



SUBTRACTION

```
.model small

.data

res db ?


first db 10,"The first number is : $"

second db 10,"The second number is : $"

msg1 db 10,"The sum of two numbers is : $"

msg2 db 10,"The differnce of two numbers is : $"

.code

.startup
```

```
; firstnum is : 67829876

; secondnum is : 12345678


mov dx, offset first

mov ah,9

int 21h


mov ax,6782h

call disph

mov ax , 9876h

call disph

mov dx, offset second

mov ah,9

int 21h


mov ax , 1234h

call disph


mov ax , 5678h

call disph
```

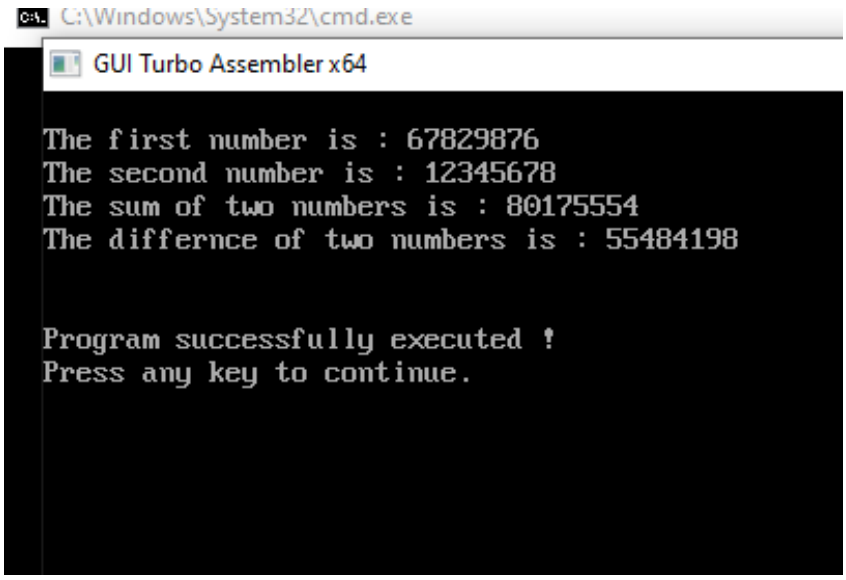
```
mov dx, offset msg1
mov ah,9
int 21h
call sum
mov dx, offset msg2
mov ah,9
int 21h
call sub_

.exit
sum proc near
mov ax , 9876h
mov bx, 6782h
mov cx , 5678h
mov dx , 1234h

add al , cl
daa
mov res , al

adc ah,ch
mov al,ah
```

```
daa
mov res+1 , al

adc bl,dl
mov al,bl
daa
mov res+2 , al
adc bh,dh
mov al,bh
daa
mov res+3 , al
mov ah,res+3

mov al,res+2
call disph
mov ah,res+1
mov al,res
call disph
ret
sum endp
sub_ proc near
```

```
mov ax , 9876h
mov bx, 6782h
mov cx , 5678h
mov dx , 1234h

sub al , cl
das
mov res , al

sbb ah,ch
mov al,ah
das
mov res+1 , al

sbb bl,dl
mov al,bl
das
mov res+2 , al
sbb bh,dh
mov al,bh
das
mov res+3 , al
```

```
mov res+3 , al
mov ah,res+3

mov al,res+2
call disph

mov ah,res+1
mov al,res
call disph

ret
sub_ endp
disph proc near
mov cl,4
mov ch , 4

disph1:
rol ax , cl
push ax
and al,0fh
add al,30h
```

```
cmp al,'9'
jbe disph2
add al,7h

disph2:
mov ah,2
mov dl,al
int 21h
pop ax
dec ch
jnz disph1

ret

disph endp

end
```


C:\Windows\System32\cmd.exe
GUI Turbo Assembler x64
The first number is : 67829876
The second number is : 12345678
The sum of two numbers is : 80175554
The differnce of two numbers is : 55484198
Program successfully executed !
Press any key to continue.

Q10: Write a program for 32-bit binary division and multiplication.

```
.MODEL SMALL
.386
.DATA
FIRST DB 10,"THE FIRST NUMBER IS : $"
SECOND DB 10,"THE SECOND  NUMBER IS : $"
M1 DB 10,"THE SUM OF NUMBERS IS : $"
M2 DB 10,"THE SUB OF NUMBERS IS : $"
M3 DB 10,"THE DIV OF NUMBERS IS : $"
M4 DB 10,"THE MUL OF NUMBERS IS : $"
.CODE
.STARTUP

;;;;FIRSTNUM 00000019H
;;;;SECONDNUM 00000005H

MOV DX,OFFSET FIRST
MOV AH,9
INT 21H

MOV EAX , 00000019H
CALL DISPH

MOV DX,OFFSET SECOND
MOV AH,9
INT 21H

MOV EAX , 00000005H
CALL DISPH

CALL SUM
CALL SUB_
CALL MUL_
CALL DIV_
.EXIT

SUM PROC NEAR
```

```
MOV DX,OFFSET M1
MOV AH,9
INT 21H
MOV EAX ,00000019H
MOV EDX , 00000005H

ADD EAX , EDX
CALL DISPH
RET
SUM ENDP

SUB_ PROC NEAR

MOV DX,OFFSET M2
MOV AH,9
INT 21H


MOV EAX , 00000019H
MOV EDX , 00000005H

SUB EAX , EDX
CALL DISPH
RET
SUB_ ENDP

DIV_  PROC NEAR
MOV DX,OFFSET M3
MOV AH,9
INT 21H

MOV EAX , 00000019H
MOV EDX, 0H
MOV ECX , 00000005H

DIV ECX
CALL DISPH
```

```
RET
DIV_  ENDP
MUL_ PROC NEAR

MOV DX,OFFSET M4
MOV AH,9
INT 21H


MOV EAX , 00000019H
MOV EDX , 00000005H

MUL EDX
MOV EBX,EAX
MOV EAX , EDX
CALL DISPH

MOV EAX , EBX
CALL DISPH
RET
MUL_ ENDP

DISPH PROC NEAR

MOV CL,4H
MOV CH,8H
DISPH1:
ROL EAX , CL
PUSH EAX
AND AL,0FH

ADD AL,30H
CMP AL,'9'
JBE DISPH2
ADD AL,7H

DISPH2 :
```

```
MOV AH,2
MOV DL,AL
INT 21H

POP EAX
DEC CH
JNZ DISPH1
RET
DISPH ENDP

END
```