0. Write an assembly language program that will take an alphabet as an input (A~Z) or (a~z) and also take a given value N as input (0 to 9). Now, the output will show/display the N number character from the alphabet order starting from starting input alphabet (in the same case upper/lower as the given input).
Sample Input / Output:
Input Alphabet: B
Given Value N: 3
Output: E

```
org 100h

.data
character db "Enter the character: $"
number db "Enter the number: $"
counter db 0

.code

mov ax, @data
mov ds, ax

  lea dx, character     ; showing message
  mov ah, 09h
  int 21h


  mov ah, 01 ;taking character input
  int 21h

  mov bl, al   ; storing character in bl

  ; go to a new line with carriage return
  MOV AH, 2
  MOV DL, 0DH
  INT 21h
  MOV DL, 0AH
  INT 21h

  lea dx, number      ; showing message
  mov ah, 09h
  int 21h

  mov ah, 01 ; taking number input
  int 21h
```

```
    mov bh, al ; storing the number in bh

    sub bh, 48       ; considering ascii value
    add bh, bl       ; adding values to get output character

    ; go to a new line with carriage return
    MOV AH, 2
    MOV DL, 0DH
    INT 21h
    MOV DL, 0AH
    INT 21h


    mov ah, 02    ; printing the output
    mov dl, bh
    int 21h

ret
```

Write an assembly language program that will accept an input string of 5 (five) letters in LOWERCASE from the keyboard and displays the string in reverse order in UPPERCASE in a new line.
Sample Input / Output:
Input: abcdef
Output: FEDCBA

```
.data
lowercase db 5 dup(0)
uppercase db 5 dup(0)

.code

main proc

    mov ax, @data

    mov cl, 5
    mov si, 0

    loop1:
    mov ah, 01h
    int 21h

    mov lowercase[si], al
    add si,1
    loop loop1

    mov cl,5
    mov si,0

    loop2:
    mov di,cx
    sub di,1
    mov al,lowercase[si]
    mov uppercase[di],al
    add si,1

    loop loop2

    mov cx,5
    mov si,0

    loop3:
    xor ax,ax
    xor dx,dx

    mov ah,02h
    mov dl,uppercase[si]
    sub dl,32
    int 21h

    add si,1

    loop loop3

    ret

main endp
```

Write an assembly language program that will accept an input string of 5 (five) letters in UPPERCASE from the keyboard and displays the string in reverse order in LOWERCASE in a new line.
Sample Input / Output:
Input: ABCDEF
Output: fedcba

```
.model small
.stack 100h
.data
```

```
.code
main proc

    include 'emu8086.inc'
    printn "enter string size& Input: "
    MOV ah,01h
    INT 21h

    printn

    MOV cl,al
    SUB cl,48
    MOV bl,0

    MOV ah,01h

    input:

    INT 21h
    MOV [bx],al
    INC bl
    CMP bl,cl
    JL input


    finish:

    printn
    printn "OUTPUT : "
    MOV ah,02h
    MOV bl,cl
    DEC bl

    rev:
    MOV dx,[bx]
    ADD dx,32
    INT 21h
    DEC bl
    CMP bl,0
    JGE rev


    MOV ax, 4c00h
    INT 21h
    RET

MAIN ENDP
END MAIN
```

Write an assembly language program that will accept an input digit N (0 to 9) from the keyboard and finds the even digits up to that input and displays those in new lines.
Sample Input / Output:
Input: 9
Output: Even Digits: 0 2 4 6 8

```
org 0100h

.data
n db ?
arr db ?
print dw   'Even digits :','$'

.code

main proc

    mov ax,@data
    mov ds,ax
    mov si,offset print

    ;taking input from user
    mov ah,01h
    int 21h
    sub al,48
    mov n,al
    mov cl,al


    xor cl,cl
    mov cl,n
    mov bl,0


    mov dl,0dh
    mov ah,02h
    int 21h
    mov dl,0ah
    int 21h

    mov ah,09h
    lea dx,print
    int 21h


     even:
         mov dl,0h
         mov ah,02h
         int 21h


         mov dl,bl
         add dl,48
         mov ah,02h
         int 21h
         add bl,2
         cmp bl,cl
         jle even




 main endp
end main
ret
```

Write an assembly language program that will accept an input digit N (0 to 9) from the keyboard and finds the odd digits up to that input and displays those in new lines.
Sample Input / Output:
Input: 9
Output: Odd Digits: 1 3 5 7 9

```
ORG 0100h

.DATA

            MSG DB 'Odd Digits : $'

            COUNT DB ?
```

```
.CODE

MAIN PROC
    MOV AX, @DATA
    MOV DS, AX

    MOV AH, 1               ; TAKING THE INPUT
    INT 21H

    SUB AL, 48              ; CONVERTING ASCII TO INTEGER
    MOV COUNT, AL           ; STORING AL INTO COUNT VARIABLE

    MOV AH, 2
    MOV DL, 0AH             ; PRINTING NEWLINE
    INT 21H
    MOV DL, 0DH             ; PRINTING CARRIAGE RETURN
    INT 21H

    MOV AH, 9               ; PRINTING MSG
    MOV DX, OFFSET MSG
    INT 21H

    MOV AH, 2               ; STARTED PRINTING OUTPUT

    MOV BL, 1

LOOP0:
    CMP BL, COUNT
    JG  EXIT
    MOV DL, BL
    ADD DL, 48
    INT 21H
    MOV DL, ' '
    INT 21H
    ADD BL, 2
    JMP LOOP0

EXIT:

RET

MAIN ENDP
```

Write an assembly language program that will accept an input of 5 (five) digits (0 to 9)

from the keyboard randomly and rearrange them in ascending order.
Sample Input / Output:
Input: 2 4 5 3 2
Output: Ascending: 2 2 3 4 5

```
include 'emu8086.inc'
org 100h

.data

arr db 5 dup(?)
input_array db 'Input:$'
output_array db 'Ascending:$'

.code
main proc

mov ah,9
lea dx,input_array ; output of the input string
int 21h

mov cx,5 ; takes the value 5 for the input loop
lea bx, arr; load effective address of arr taken in bx
mov ah,1

input:
int 21h
mov [bx],al; input taken into the arr array
inc bx
loop input


mov cx,5
dec cx

outerloop:
mov bx, cx
mov si,0

comploop:
mov al,arr[si]
mov dl,arr[si+1]
cmp al,dl

jc noSwap
```

```
mov arr[si],dl
mov arr[si+1],al

noSwap:
inc si
dec bx
jnz comploop

loop outerloop

mov ah,2
mov dl,10
int 21h

mov dl,13
int 21h


mov ah,9
lea dx,output_array
int 21h

mov cx,5
mov bx, offset arr

Outputs:
mov dl,[bx]
mov ah,2
int 21h

inc bx
loop Outputs

main endp
ret
```

Write an assembly language program that will accept an input of 5 (five) digits (0 to 9) from the keyboard randomly and sort-out the odd and even digits from them.
Sample Input / Output:
Input: 2 4 5 3 2
Output: Odd: 3 5
Even: 2 2 4

```
ORG 0100H
.DATA
```

```
input DB 'Input: $' ; define string to display
odd DB  0DH, 0AH,'Odd: $'
even DB  0DH, 0AH,'Even: $'
str DB 5 dup(0)
.CODE
MAIN PROC
mov ax, @DATA
mov ds, ax
xor ax,ax
mov si, OFFSET input

mov ah, 09h
int 21h

mov cl,5
mov si,0
mov ah, 01h
takeinput: int 21h
        sub al, 48
        mov str[si], al
        inc si
        loop takeinput

mov ah, 09h
lea dx,odd
int 21h


mov cl,5
mov si,0

l1: xor ax,ax
   call oddnum ; call procedure
   inc si
   loop l1


mov ah, 09h
lea dx,even
int 21h


mov cl,5
mov si,0
```

```
l3: xor ax,ax
    call evennum ; call procedure
    inc si
    loop l3

mov ah, 4Ch
mov al, 00h ; a code after procedure call and return
int 21h ; exit to DOS
MAIN ENDP

oddnum proc ; declare a procedure named oddnum

mov al,str[si]
mov bl,2
div bl
cmp ah, 0
jz exit
mov ah, 02h
mov dl, str[si]
add dl,48
int 21h
ret

exit: ret

oddnum ENDP ; end of procedure oddnum

evennum proc ; declare a procedure named evennum
mov al,str[si]
mov bl,2
div bl
cmp ah, 0
jnz exit2
mov ah, 02h
mov dl, str[si]
add dl,48
int 21h
ret

exit2: ret

evennum ENDP ; end of procedure evennum
```

```
END MAIN ; end of program
```

Write an assembly language program that will accept an input of 8 (eight) digits (0 to 9) from the keyboard randomly and sort-out the odd digits in ascending order.
Sample Input / Output:
Input: 2 4 7 5 6 1 8 3
Output: Odd Ascending: 1 3 5 7

```
; You may customize this and other start-up templates;
; The location of this template is c:\emu8086\inc\0_com_template.txt

org 100h
.model small
.data

arr db 8 dup(?)

odd_arr db 8 dup (?)
output_message db 'Output: Odd Ascending:$'
input db 'Input:$'
.code

main proc
   mov ax, @data
   mov ds,ax

   mov cx,8
   mov bx,offset arr

   mov dx,offset input
   mov ah,09h
   int 21h
   xor dx,dx
   mov ah,1

   inputs:
   int 21h


   mov [bx],al
   inc bx

   Loop inputs
```

```
mov cx,8

dec cx

Outerloop:
mov bx,cx
mov si,0
Comploop:
mov al,arr[si]
mov dl,arr[si+1]
cmp al,dl

jc noSwap

mov arr[si],dl
mov arr[si+1],al

noSwap:
inc si
dec bx
jnz Comploop

loop Outerloop

mov ah,2
mov dl,10
int 21h
mov dl,13
int 21h
  mov dx,offset output_message

mov ah,09h

int 21h

xor si,si
xor dx,dx


mov cx,8
```

```
    mov bx,offset arr
    xor dx,dx
    mov si,9
    xor ax,ax


    Outputs:
    dec si
    xor ax,ax
    mov cx,si
    mov al,[bx]
    sub al,48
    mov cl,02h
    div cl
    cmp ah,0

    je continue

    mov dl,[bx]

    mov ah,2
    int 21h

    mov dl,' '
    mov ah,2
    int 21h

    continue:
    inc bx

    mov cx,si
    loop Outputs

 main endp
ret
```

Write an assembly language program that will accept an input of 8 (eight) digits (0 to 9) from the keyboard randomly and sort-out the odd digits in descending order.
Sample Input / Output:
Input: 2 4 7 5 6 1 8 3
Output: Odd Descending: 7 5 3 1

```
INCLUDE 'emu8086.inc'
org 100h
```

```
.data
arr db 8 dup(?)

.code
main proc
   mov ax, @DATA
   mov ds, ax

   mov cx, 8
   lea bx, arr
   mov ah, 1

   ; Taking inputs
   inputs:
      int 21h
      mov [bx], al
      inc bx
      loop inputs


   mov cx, 8
   dec cx

   traverse:
      mov bx, cx
      mov si, 0

   compare:
      mov al, arr[si]
      mov dl, arr[si+1]
      cmp dl, al

   jc no
   mov arr[si], dl
   mov arr[si+1], al

   no:
      inc si
      dec bx
      jnz compare

   loop traverse
   mov ah, 2
   mov dl, 10
   int 21h

   mov dl, 13
   int 21h

  ;Printing the sorted odds
   mov cx, 8
   lea si, arr

   PRINT "Odd Descending: "

   Outputs:
      mov al, [si]
      test al, 1
      jz no_print

      mov dl, [si]
      mov ah, 2
      int 21h

      mov dl, 32
      mov ah, 2
      int 21h

      no_print:
         inc si
   loop Outputs

main endp
ret
```

Write an assembly language program that will accept an input of 8 (eight) digits (0 to 9) from the keyboard randomly and sort-out the even digits in ascending order.
Sample Input / Output:
Input: 2 4 7 5 6 1 8 3
Output: Even Ascending: 2 4 6 8

```
INCLUDE 'emu8086.inc'
ORG 100H


.DATA
```

```
    array DB 8 DUP(?)

.CODE
MAIN PROC
    PRINT "Enter 8 digits: "
    MOV AH, 02H
    MOV DL, 0DH
    INT 21H
    MOV DL, 0AH
    INT 21H

    MOV SI, OFFSET array
    MOV CX, 8
    MOV BX, 0H

    LOOP1:
    MOV AH, 01H
    INT 21H
    MOV [SI], AL
    MOV DL, 32
    MOV AH, 02H
    INT 21H
    INC SI
    LOOP LOOP1

    MOV AH, 02H
    MOV DL, 0DH
    INT 21H
    MOV DL, 0AH
    INT 21H

    MOV CX, 7

    LOOP2:
    MOV BX, CX
    MOV SI, 0

    LOOP3:
    MOV AL, array[SI]
    MOV DL, array[SI+1]
    CMP AL, DL

    JC NoSWAP
    MOV array[SI], DL
    MOV array[SI+1], AL

    NoSWAP:
    INC SI
    DEC BX
    JNZ LOOP3

    LOOP LOOP2

    MOV CX, 8
    MOV SI, OFFSET array

    PRINT "Even Ascending: "

    LOOP4:
    MOV AL, [SI]
    TEST AL,1

    JNZ EVEN

    MOV DL,[SI]
    MOV AH, 02H
    INT 21H
    MOV DL, 32
    MOV AH, 02H
    INT 21H

    EVEN:
    INC SI
    LOOP LOOP4


MAIN ENDP

RET
```

Write an assembly language program that will accept an input of 8 (eight) digits (0 to 9) from the keyboard randomly and sort-out the even digits in descending order.
Sample Input / Output:
Input: 2 4 7 5 6 1 8 3
Output: Even Descending: 8 6 4 2

```
include 'emu8086.inc'
org 100h
```

```
.data

arr db 8 dup(?)
even_checker db  2

.code
main proc




print "Enter 8 number in array:"

mov cx,8
mov bx,offset arr
mov ah,1

inputs:
   int 21h
   mov [bx],al
   inc bx
   loop inputs


mov cx,8
dec cx

outerloop:
   mov bx, cx
   mov si,0

comploop:
   mov al,arr[si]
   mov dl,arr[si+1]
   cmp dl,al

   jc noSwap
   mov arr[si],dl
   mov arr[si+1],al


noSwap:
    inc si
    dec bx
    jnz comploop

    loop outerloop

mov ah,2
mov dl,10
int 21h

mov dl,13
int 21h

print "After Sorting Array: "

mov cx,8
mov si, offset arr


Outputs:
    mov al,[si]
    test al,1
    jnz even




    mov dl,[si]
    mov ah,2
    int 21h

    mov dl,32
    mov ah,2
    int 21h
    even:
        inc si
        loop Outputs

main endp
ret
```

Write an assembly language program that will accept an input of 8 (eight) digits (0 to 9) from the keyboard randomly and finds the prime digits in descending order.
Sample Input / Output:

Input: 2 4 7 5 6 1 8 3
Output: Primes Digits Descending: 7 5 3 2

```
ORG 0100H
.DATA
input DB 'Input: $'
outputp DB 'Prime Digits Descending: $'
arr DB 8 dup(0)
.CODE
MAIN PROC
mov ax,@DATA
mov ds,ax
xor ax,ax
mov dx,OFFSET input
mov ah,09h
int 21h
mov cx,8
mov si,0
mov ah,01h
l1: int 21h
    sub al,48
    mov arr[si],al
    inc si
    loop l1
mov ah,2
mov dl,0Ah
int 21h
mov dl,0Dh
int 21h
mov cx,8
dec cx
continue:
mov bx,cx
mov si,0
compare:
mov al,arr[si]
mov dl,arr[si+1]
cmp al,dl
jg unchanged
mov arr[si],dl
mov arr[si+1],al
unchanged:
inc si
dec bx
```

```
jnz compare
loop continue
mov bx,offset arr
mov ah,09h
lea dx,outputp
int 21h
mov cl,8
mov si,0
l2: xor ax,ax
   call prime
   inc si
   loop l2
MOV AH,4Ch
MOV AL,00h
INT 21h
MAIN ENDP
prime proc
cmp arr[si],0
jz exit
cmp arr[si],1
jz exit
cmp arr[si],2
jz exit2
cmp arr[si],3
jz exit2
cmp arr[si],4
jz exit
cmp arr[si],5
jz exit2
cmp arr[si],6
jz exit
cmp arr[si],7
jz exit2
cmp arr[si],8
jz exit
cmp arr[si],9
jz exit
exit2: mov ah,02h
      mov dl,arr[si]
      add dl,48
      int 21h
      ret
exit: ret
prime ENDP
```

```
END MAIN
```

Write an assembly language program that will accept an input of 8 (eight) digits (0 to 9) from the keyboard randomly and sort-out the odd prime digits in ascending order.
Sample Input / Output:
Input: 2 4 7 5 6 1 8 3
Output: Odd Primes Ascending: 3 5 7

```
; Done by Nahian_180041136

org 0100h

.data
InputMsg DB 'Input: $' ; Showing Input message => 'Input: '
OutputMsg DB 'Output: Odd Primes Ascending: $' ; Showing Output message => 'Output: Odd
Primes Ascending: '
OddPrimeMsg DB 'Prime: $' ; Showing Odd Prime message => 'Prime: '

OddPrimeArr DB 8 DUP(0,20h),'$' ; Initializing array for prime digits
                                ;20h is inserted in array for space

count_3 db 0
count_5 db 0
count_7 db 0


.code
main proc

lea dx, InputMsg ; set DX to point to 1st element of string array InputMsg
mov ah, 09h
int 21h

xor cl, cl  ; clearing for counter
mov cl, 8   ; intializing counter value 5 to take 5 inputs

Loop_1:
   mov ah, 01h       ; taking singe-key input
   int 21h

   ; only 3, 5, 7 are odd prime from 0~9

   cmp al, 33h
   jz inc_3_call
```

```
    cmp al, 35h
    jz inc_5_call

    cmp al, 37h
    jz inc_7_call


    jmp NextLoop

    inc_3_call:
        call inc_3        ; calling inc_3 procesure procedure to assign prime digits
        jmp NextLoop      ; skipping other procedure call

    inc_5_call:
        call inc_5
        jmp NextLoop

    inc_7_call:
        call inc_7
        jmp NextLoop

 NextLoop:
    mov ah, 2       ; display character function
    mov dl, ' '     ; character is a space
    int 21h
    loop Loop_1

mov ah, 2
mov dl, 0DH ; carriage return (start of a line)
int 21h
mov dl, 0AH ; line feed (new line)
int 21h

lea dx, OutputMsg ; set DX to point to 1st element of string array OutputMsg
mov ah, 09h
int 21h

 xor cx,cx
 cmp count_3,0
 jz work_with_5
 mov cl, count_3

 Loop_print_3:
    mov ah, 2       ; display character function
    mov dl, '3'     ; character is a space
    int 21h
    mov ah, 2       ; display character function
    mov dl, ' '     ; character is a space
    int 21h
    loop Loop_print_3

 work_with_5:
    cmp count_5,0
    jz work_with_7
    mov cl, count_5

 Loop_print_5:
    mov ah, 2       ; display character function
    mov dl, '5'     ; character is a space
    int 21h
    mov ah, 2       ; display character function
    mov dl, ' '     ; character is a space
    int 21h
    loop Loop_print_5

 work_with_7:
    cmp count_7,0
    jz work_done
    mov cl, count_7

 Loop_print_7:
    mov ah, 2       ; display character function
    mov dl, '7'     ; character is a space
    int 21h
    mov ah, 2       ; display character function
    mov dl, ' '     ; character is a space
    int 21h
    loop Loop_print_7

work_done:
    mov ah, 4ch
    mov al, 00h ; a code after procedure call and return
    int 21h

main endp

inc_3 proc
```

```
    inc count_3     ; if input digit is equal to 3, increasing count_3 counter value
    ret
inc_3 endp

inc_5 proc
    inc count_5     ; if input digit is equal to 5, increasing count_5 counter value
    ret
inc_5 endp

inc_7 proc
    inc count_7     ; if input digit is equal to 7, increasing count_7 counter value
    ret
inc_7 endp
```

Write an assembly language program that will accept an input of 8 (eight) digits (0 to 9) from the keyboard randomly and sorts them in ascending order.
Sample Input / Output:
Input: 2 4 3 5 6 1 8 3
Output: 1 2 3 3 4 5 6 8

```
include 'emu8086.inc'
org 100h

.data

arr db 5 dup(?)

.code
main proc

print "Enter 5 number in array:"

mov cx,5
mov bx,offset arr
mov ah,1

inputs:
int 21h
mov [bx],al
inc bx
loop inputs


mov cx,5
dec cx
```

```
outerloop:
mov bx, cx
mov si,0

comploop:
mov al,arr[si]
mov dl,arr[si+1]
cmp al,dl

jc noSwap
mov arr[si],dl
mov arr[si+1],al

noSwap:
inc si
dec bx
jnz comploop

loop outerloop

mov ah,2
mov dl,10
int 21h

mov dl,13
int 21h

print "After Sorting Array: "

mov cx,5
mov bx, offset arr

Outputs:
mov dl,[bx]
mov ah,2
int 21h

mov dl,32
mov ah,2
int 21h

inc bx
loop Outputs
```

```
main endp
ret
```

Write an assembly language program that will accept an input of 8 (eight) digits (0 to 9) from the keyboard randomly and sorts them in descending order.
Sample Input / Output:
Input: 2 4 3 5 6 1 8 3
Output: 8 6 5 4 3 3 2 1

```
include 'emu8086.inc'
org 100h
.model small
.data

   arr db 8 dup(?)

.code
main proc
   mov ax, @data
   mov ds, ax

   print "Input:"

   mov cx,8
   mov bx, offset arr
   mov ah,1

   inputs:
   int 21h
   mov [bx],al
   inc bx
   Loop inputs

   mov cx, 8
   dec cx

   OuterLoop:
   mov bx,cx
   mov si, 0

   CompLoop:
   mov al, arr[si]
```

```
    mov dl, arr[si+1]
    cmp al,dl

    jnc noSwap

    mov arr[si], dl
    mov arr[si+1], al

    noSwap:
    inc si
    dec bx
    jnz CompLoop

    loop OuterLoop


    mov ah,2
    mov dl,10
    int 21h

    mov dl,13
    int 21h

    print "Output:"
     mov cx,8
     mov bx, offset arr

    Outputs:

    mov dl, [bx]
    mov ah,2
    int 21h

    inc bx
    loop Outputs

main endp
ret
```

Write an assembly language program that will accept an input of 5 (five) digits (0 to 9) from the keyboard and finds the summation of the digits and displays the result in HEX digit.
Sample Input / Output:
Input: 1 2 3 4 5

Output: Sum: F

```
ORG 100h


.DATA                                          ; Data segment starts
sum DB 0
flag DB 33 DUP(0)
hexdigit1 DB ?
hexdigit2 DB ?
inputMessage DB 'Enter 5 decimal digits: $'
outputMessage DB 'Hex Sum: $'

.CODE                                          ; Code segment starts

MAIN PROC

   MOV AX, @DATA
   MOV DS, AX
   XOR AX, AX
                                          ; Print Message to enter values
   MOV DX, OFFSET inputMessage
   MOV AH, 09h
   INT 21h

   CALL TAKE_INPUT
   CALL CALC_SUM
   CALL DISPLAY_OUTPUT


   RET

MAIN ENDP



TAKE_INPUT PROC

   XOR CX, CX                                  ; Clear count register
   MOV CL, 5                                   ; To take 5 inputs
   MOV SI, 0                                   ; Initialize SI

   input_digits:
      MOV AH, 01h
```

```
        INT 21h

        SUB AL, 48                          ; To get decimal value of the inputs
        ADD sum, AL
        INC SI
    LOOP input_digits

    RET

TAKE_INPUT ENDP



CALC_SUM PROC

    CMP sum,32
    JL less32
    MOV flag[32],1
    SUB sum,32

    less32:
    CMP sum,16
    JL less16
    MOV flag[16],1
    SUB sum,16

    less16:
    CMP sum,8
    JL less8
    MOV flag[8],1
    SUB sum,8

    less8:
    CMP sum,4
    JL less4
    MOV flag[4],1
    SUB sum,4

    less4:
    CMP sum,2
    JL less2
    MOV flag[2],1
    SUB sum,2
```

```
less2:
CMP sum,1
JL less1
MOV flag[1],1
SUB sum,1

less1:

XOR AX,AX
MOV AL,2
MUL flag[32]
ADD hexdigit1,AL

MOV AL,1
MUL flag[16]
ADD hexdigit1,AL

MOV AL,8
MUL flag[8]
ADD hexdigit2,AL

MOV AL,4
MUL flag[4]
ADD hexdigit2,AL

MOV AL,2
MUL flag[2]
ADD hexdigit2,AL

MOV AL,1
MUL flag[1]
ADD hexdigit2,AL

CMP hexdigit1,10
JL notChar1
ADD hexdigit1,7

notChar1:
CMP hexdigit2,10
JL notChar2
ADD hexdigit2,7

notChar2:
    RET

CALC_SUM ENDP


DISPLAY_OUTPUT PROC
    MOV AH, 02h
    MOV DL, 0Ah                          ; to print new line
    INT 21h
    MOV DL, 0Dh                          ; carriage return
    INT 21h

    MOV DX, OFFSET outputMessage
    MOV AH, 09h
    INT 21h

    XOR DX,DX
    XOR AX,AX

    MOV AH, 02h
    MOV DL, hexdigit1
    CMP DL, 0
    JE dontPrint:
    ADD DL, 48
    INT 21h

    dontPrint:
    MOV DL, hexdigit2
    ADD DL, 48
    INT 21h


    RET

DISPLAY_OUTPUT ENDP
```

Write an assembly language program that will display an array inputted string “Islamic University of Technology” and it’s reverse string one after another for 10 times each (forward and reverse print) in total 20 (twenty times) in different lines with line feed..

```
ORG 0100h
.DATA

Forward DW 'ISLAMIC UNIVERSITY OF TECHNOLOGY $'
```

```
Reverse DW 'YGOLONHCET FO YTISREVINU CIMALSI $'



.CODE
MAIN PROC

MOV CX,10  ;each string for 10 times

Level:

MOV AH,9
MOV DX,OFFSET Forward ; for geetting the forward string
INT 21H

MOV AH, 2
MOV DL, 0DH
INT 21h
MOV DL, 0AH
INT 21h

MOV AH,9
MOV DX,OFFSET Reverse ; then for getting the reverse string next
INT 21H

MOV AH, 2
MOV DL, 0DH
INT 21h
MOV DL, 0AH
INT 21h


LOOP Level ; for repeating the 2 strings again in the same order


MAIN ENDP
END MAIN
RET

END MAIN
```

Write an assembly language program that will display all the ASCII characters at REVERSE ORDER.

```
.DATA
Message  DB  '128 ASCII char in reverse order:',13,10,'$'

.CODE
MAIN PROC

    MOV AX, @DATA
    MOV DS, AX

    LEA DX, Message
    MOV AH, 9
    INT 21H

    MOV CX, 128                  ; initialize CX

    MOV AH, 2
    MOV DL, 127                  ; initialize DL with last ASCII character

    top:                     ; loop label
      INT 21H                    ; print ASCII character

      DEC DL                     ; decrement DL to next ASCII character
      DEC CX                     ; decrement CX
      JNZ top                    ; jump to label @LOOP if CX is 0

    MOV AH, 4CH
    INT 21H

MAIN ENDP
END MAIN
```

Write an assembly language program that will display only the ASCII characters of all DIGITS, ALPHABETS (upper and lower) at FORWARD and REVERSE ORDER (in different lines).

```
.MODEL SMALL
 .STACK 100H

 .DATA
   PROMPT  DB  'The ASCII values are : $'

 .CODE
  MAIN PROC
```

```
MOV AX, @DATA
MOV DS, AX

LEA DX, PROMPT
MOV AH, 9
INT 21H

MOV CX, 10

MOV AH, 2
MOV DL, 48

myLOOP_1:
  INT 21H

  INC DL
  DEC CX
JNZ myLoop_1



MOV CX, 26

MOV AH, 2
MOV DL, 65


myLOOP_2:
  INT 21H

  INC DL
  DEC CX
JNZ myLoop_2



MOV CX, 26

MOV AH, 2
MOV DL, 97


myLOOP_3:
  INT 21H
  INC DL
  DEC CX
JNZ myLoop_3


MOV CX, 10

MOV AH, 2
MOV DL, 57


myLOOP_4:
  INT 21H

  DEC DL
  DEC CX
JNZ myLoop_4


MOV CX, 26

MOV AH, 2
MOV DL, 90


myLOOP_5:
  INT 21H

  DEC DL
  DEC CX
JNZ myLoop_5


MOV CX, 26

MOV AH, 2
MOV DL, 122


myLOOP_6:
  INT 21H

  DEC DL
```

```
      DEC CX
     JNZ myLoop_6


     MOV AH, 4CH
     INT 21H
   MAIN ENDP
 END MAIN
```

Write an assembly language program that will accept an input of 5 (five) digits (0 to 9) from the keyboard randomly and rearrange them in descending order.
Sample Input / Output:
Input: 2 4 5 3 2
Output: Descending: 5 4 3 2 2

```
;180041128

org 100h
.DATA
        input db 5 dup(0)
        prompt db 10,13,'Enter 5 digits to be sorted in DESCENDING order: $'
        answer db 10,13,'Descending: $'
        SPACE db ' $'

.CODE
        MAIN PROC
                mov ax, @DATA
                mov ds, ax
                xor ax, ax

                mov ah, 09h                                     ;Character string output
                lea dx, prompt                          ;asking for 5 digits to be sorted
                int 21h

                mov ah, 01h
                mov cx, 5
                mov si, OFFSET input                    ;input array address

                inploop:
                        int 21h
                        mov [si], al
                        inc si
                Loop inpLoop
```

```
		Call Sort					;calling the sorting procedure.

		mov ah, 09h
		lea dx, answer					;outputting "Descending"
		int 21h

		mov cx, 5
		mov si, OFFSET input

		outloop:
			mov ah, 02h				;for Character output
			mov dx, [si]
			int 21h
			mov ah, 09h				;for String output to print SPACE
			lea dx, SPACE
			int 21h
			inc si
		Loop outLoop

		mov ah, 4ch					;return
		mov ah, 00h
		int 21h
	MAIN ENDP

	; the sort is a bubble sort algorithm
	Sort PROC
		mov cx, 5
		dec cx						;The outer loop will spin 4 times as
after each loop, the last element of the array will be the smallest.

		outerloop:
			mov bx, cx				;bx will be used as the counter for
the inner loop
			mov si,0

			compLoop:				;inner loop for comparison
				mov al, input[si]
				mov dl, input[si+1]
				cmp al,dl

				jnc noSwap			;if there is no carry in after
comparison, no swap will happen

				mov input[si], dl
```

```
                                        mov input[si+1], al

                                        noSwap:
                                                  inc si
                                                  dec bx
                                        jnz comploop            ;the comparison loop will repeat if bx is not
0
                    Loop outerLoop
                    ret                                                   ;return to main
          Sort ENDP
END MAIN
```

Write an assembly language program that will accept an input of 2 (two) digits (0 to 9) from the keyboard randomly and finds the Greatest Common Divisor of two digits.
Sample Input / Output:
Input: 2 3 Input: 2 4
Output: GCD: 1 Output: GCD: 2

```
ORG 0100H
.DATA


A DB ?
B DB ?
PROMPT_1 DB 'Input 2 Numbers: ', '$'
PROMPT_2 DB 'GCD: ', '$'


.CODE
MAIN PROC

LEA DX, PROMPT_1
MOV AH, 9
INT 21h

MOV AH, 1
INT 21H
MOV A, AL
MOV AH, 2
MOV DL, ' '
INT 21H

MOV AH, 1
INT 21H
MOV B, AL
MOV AH, 2
MOV DL, ' '
INT 21H

SUB A, 48
SUB B, 48

MOV AL, A
MOV BL, B

CMP AL, BL
JG GREATER
JMP LESS

GREATER:
   XOR CX, CX
   MOV CL, B
   JMP EXIT

LESS:
   XOR CX, CX
   MOV CL, A

EXIT:


LOOP0:
   XOR AX, AX
   MOV AL, A
   DIV CL
   CMP AH, 0
   JNE NO
   MOV AL, B
   DIV CL
   CMP AH, 0
   JNE NO
   JMP PRINT

   NO:
   LOOP LOOP0

PRINT:
   MOV AH, 2
   mov dl,0Ah  ;carriage return
```

```
    int 21h
    mov dl,0Dh  ;new line
    int 21h

    LEA DX, PROMPT_2
    MOV AH, 9
    INT 21h

    MOV AH, 2
    MOV DL, CL
    ADD DL, 48
    INT 21H

RET
MAIN ENDP
END MAIN
```