# Bisakh Mondal
# Roll: 001810501079 (A3)
# BCSE-III
# System Programming
# Assignment-I

# INDEX

Q1. Write and test a MASM program to Display your name and program title on the output screen. 3

Q2. Write and test a MASM program to convert a letter from uppercase lowercase 4

Q3. Write and test a MASM program to add two Hexadecimal Numbers. 5

Q4. Write and test a MASM program to find the second max and second min from an array. 7

Q5. Write and test a MASM program to display a terminating message. 11

Q6. Write and test a MASM program to Take a character from the keyboard and print it. 12

Q7. Write and test a MASM program to validate second numbers is less than the first. 14

Q8. Write and test a MASM program to find maximum and minimum from an array. 16

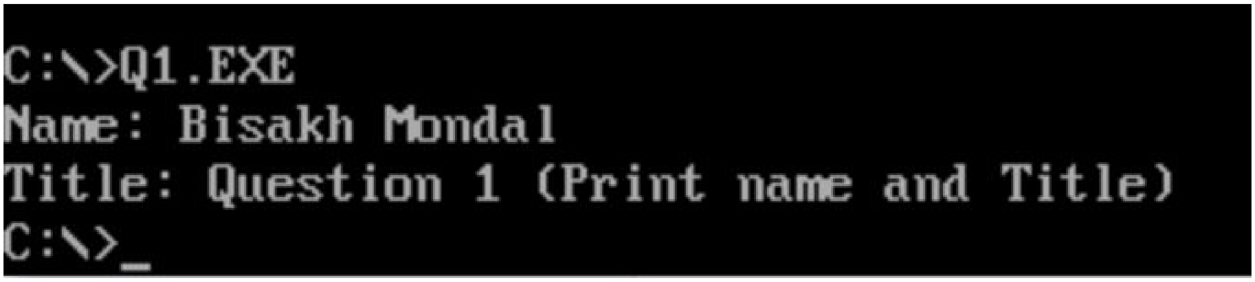
Q9. Write and test a MASM program to loop until the user decides to quit. 18

Q10. Write and test a MASM program to print all the characters from A-Z. 20

## Q1. Write and test a MASM program to Display your name and program title on the output screen.

Code

```
.model SMALL

.stack 100H

.data

nameP DB "Name: Bisakh Mondal", 10, 13, "$"
titleP DB "Title: Question 1 (Print name and Title)$"

.code
main proc
    MOV AX, @DATA
    MOV DS, AX

    MOV AH, 09H ; output a string
    LEA DX, nameP ; print name

    INT 21H
    LEA DX, titleP ; print title
    INT 21H

    MOV AH, 4CH ; for terminating the program
    INT 21H

main endp
end main
```



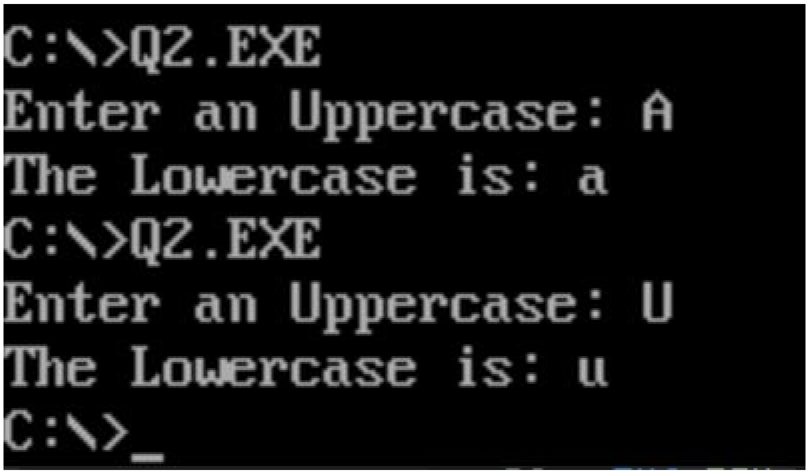
```
C:\>Q1.EXE
Name: Bisakh Mondal
Title: Question 1 (Print name and Title)
C:\>_
```

## Q2. Write and test a MASM program to convert a letter from uppercase lowercase

**Code.**

```
.model SMALL
.stack 100H

.data
msg1 db "Enter an Uppercase: $"
msg2 db 10d, 13d, "The Lowercase is: $"

.code

main proc
mov ax, @DATA
mov ds, ax

mov ah, 09h ; print preinput message
lea dx, msg1
int 21h

mov ah, 01h
int 21h ; take intput


mov ah, 09h ; print preoutput message
lea dx, msg2
int 21h

mov dl, al ; add 32 to ascii conversion
add dl, 32

mov ah, 02h ; print lowercase
int 21h

mov ah, 4ch
int 21h

main endp
```

```
end main
```



```
C:\>Q2.EXE
Enter an Uppercase: A
The Lowercase is: a
C:\>Q2.EXE
Enter an Uppercase: U
The Lowercase is: u
C:\>_
```

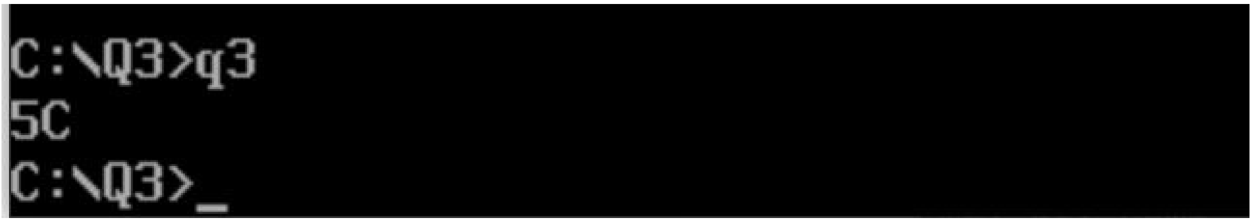
## Q3. Write and test a MASM program to add two Hexadecimal Numbers.

**Code.**

```
.model small
.stack 100h

.data

num1 db 1Ah
num2 db 42h

```

```
interrupt macro inttype
    mov ah, inttype
    int 21h
endm

.code

main proc

mov ax, @data
mov ds, ax

mov al, num1
add al, num2 ;performing addition

mov bl, al ; moving sum to lower register of bx
mov bh, 0 ; making upper byte 0
mov cl, 4 ; making it 4 for shift operation

shl bx, cl; shift bx to 4 bit left, so in bh the first bit of
addition will be stored

mov dl, bh; first print the higher byte
add dl, 48 ; converting to char

cmp dl, 59 ; if it is > '9' -> then next char will be 'A' so adding 7
jle notchar
add dl, 7
interrupt 02h

secbyte: mov dl, bl; print the lower byte
shr dl, cl; but the lower byte is shifted left to 4 byte before
add dl, 48

cmp dl, 59; ; if it is > '9' -> then next char will be 'A' so adding
7
jle notchatsec
add dl, 7
interrupt 02h
```

```
done:
interrupt 4ch


notchar: interrupt 02h
        jmp secbyte
notchatsec: interrupt 02h
            jmp done
main endp

end main
```

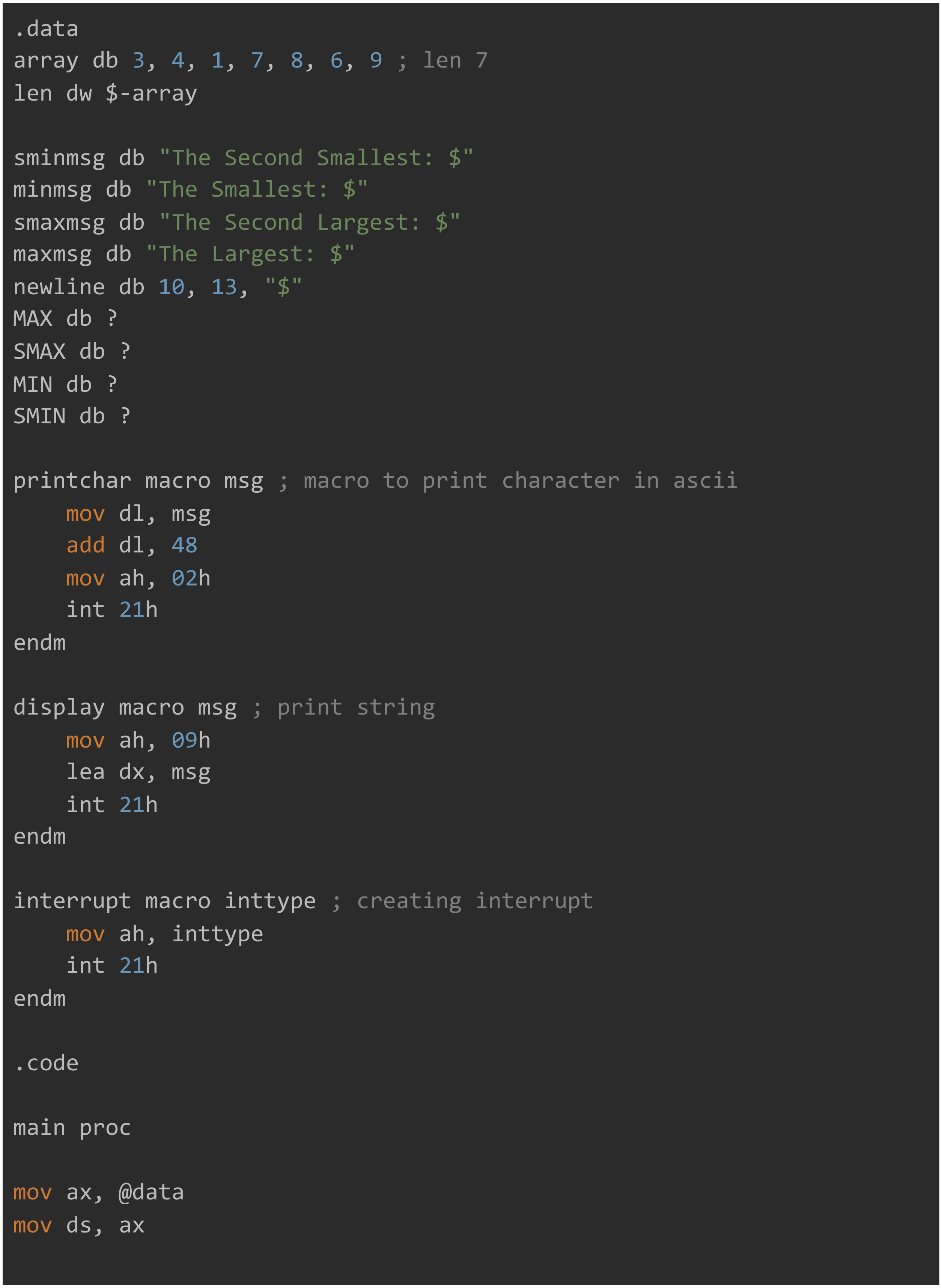


```
C:\Q3>q3
5C
C:\Q3>_
```

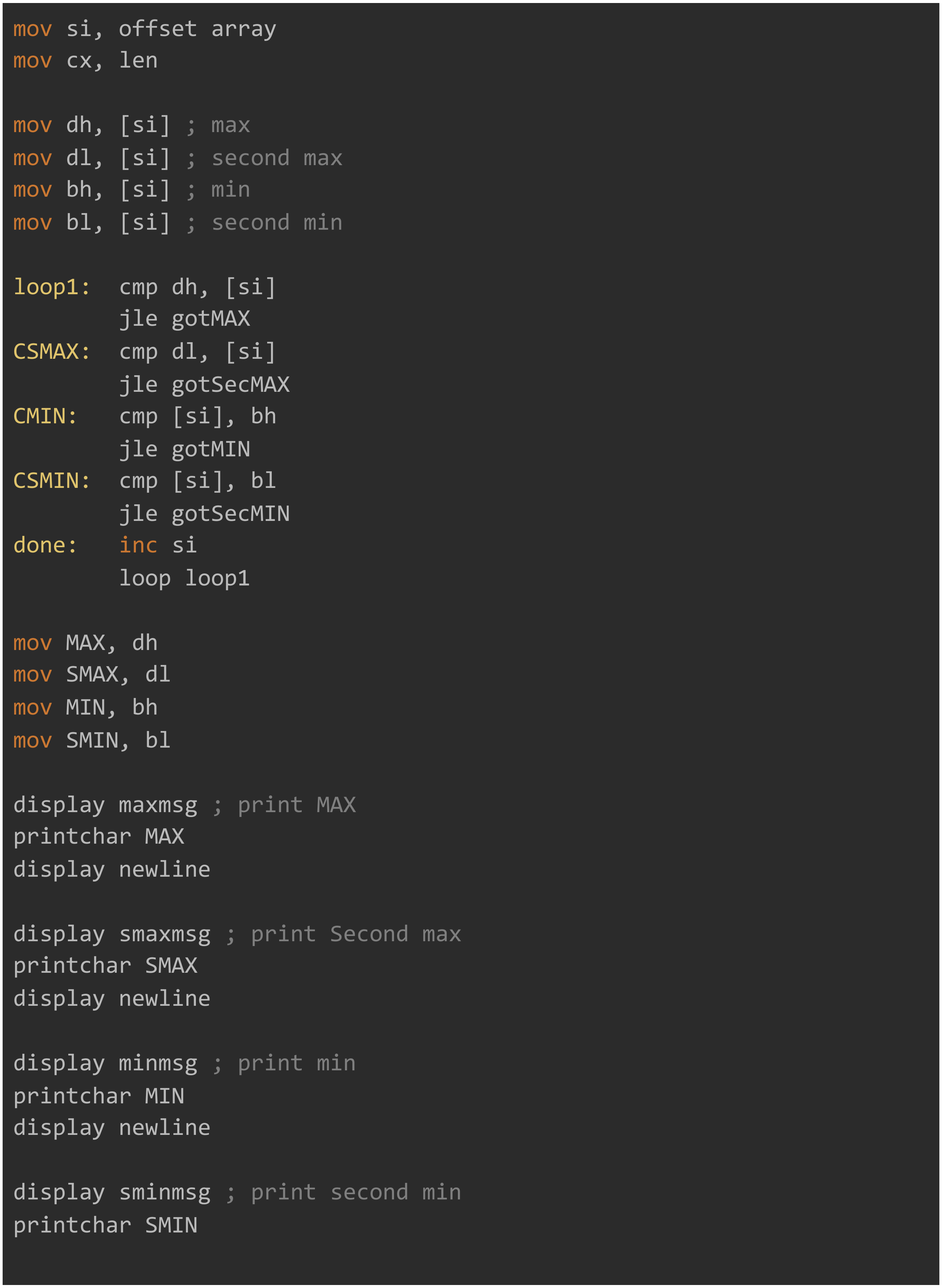
## Q4. Write and test a MASM program to find the second max and second min from an array.

**Code.**

```
.model small
.386
.stack 100h

```

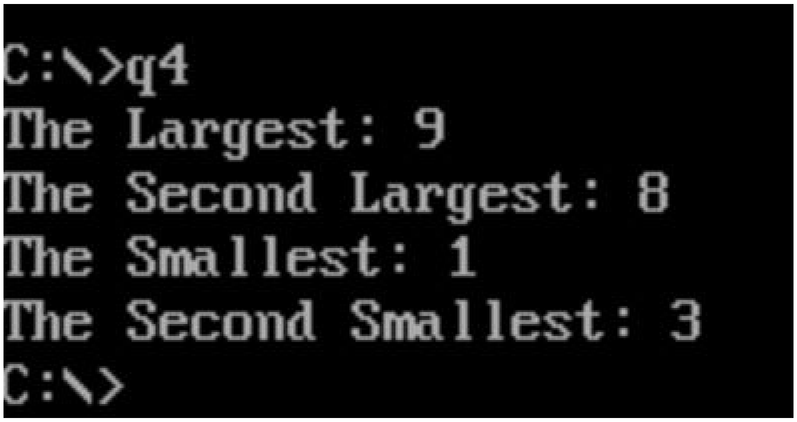


```
.data
array db 3, 4, 1, 7, 8, 6, 9 ; len 7
len dw $-array

sminmsg db "The Second Smallest: $"
minmsg db "The Smallest: $"
smaxmsg db "The Second Largest: $"
maxmsg db "The Largest: $"
newline db 10, 13, "$"
MAX db ?
SMAX db ?
MIN db ?
SMIN db ?

printchar macro msg ; macro to print character in ascii
    mov dl, msg
    add dl, 48
    mov ah, 02h
    int 21h
endm

display macro msg ; print string
    mov ah, 09h
    lea dx, msg
    int 21h
endm

interrupt macro inttype ; creating interrupt
    mov ah, inttype
    int 21h
endm

.code

main proc

mov ax, @data
mov ds, ax
```



```
mov si, offset array
mov cx, len

mov dh, [si] ; max
mov dl, [si] ; second max
mov bh, [si] ; min
mov bl, [si] ; second min

loop1:  cmp dh, [si]
        jle gotMAX
CSMAX:  cmp dl, [si]
        jle gotSecMAX
CMIN:   cmp [si], bh
        jle gotMIN
CSMIN:  cmp [si], bl
        jle gotSecMIN
done:   inc si
        loop loop1

mov MAX, dh
mov SMAX, dl
mov MIN, bh
mov SMIN, bl

display maxmsg ; print MAX
printchar MAX
display newline

display smaxmsg ; print Second max
printchar SMAX
display newline

display minmsg ; print min
printchar MIN
display newline

display sminmsg ; print second min
printchar SMIN
```

```
interrupt 4ch

gotMAX: mov dl, dh
        mov dh, [si]
        jmp CMIN

gotMIN: mov bl, bh
        mov bh, [si]
        jmp done

gotSecMIN: mov bl, [si]
           jmp done

gotSecMAX: mov dl, [si]
           jmp CMIN

main endp
end main
```

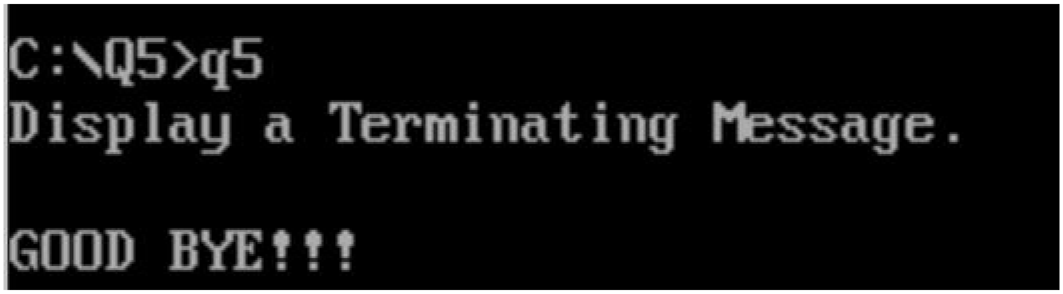


```
C:\>q4
The Largest: 9
The Second Largest: 8
The Smallest: 1
The Second Smallest: 3
C:\>
```

## Q5. Write and test a MASM program to display a terminating message.

**Code.**

```
.model small
.stack 100h

.data
PRINTSTR db "Display a Terminating Message.", 10, 13, "$"
GOODBYE db "GOOD BYE!!!$"
interrupt macro inttype ; creating interrupt
    mov ah, inttype
    int 21h
endm

display macro msg ; print string
    mov ah, 09h
    lea dx, msg
    int 21h
endm

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

    display PRINTSTR
    interrupt 01h
    display GOODBYE; print terminating message

    interrupt 4ch; exitting program

main endp
end main
```

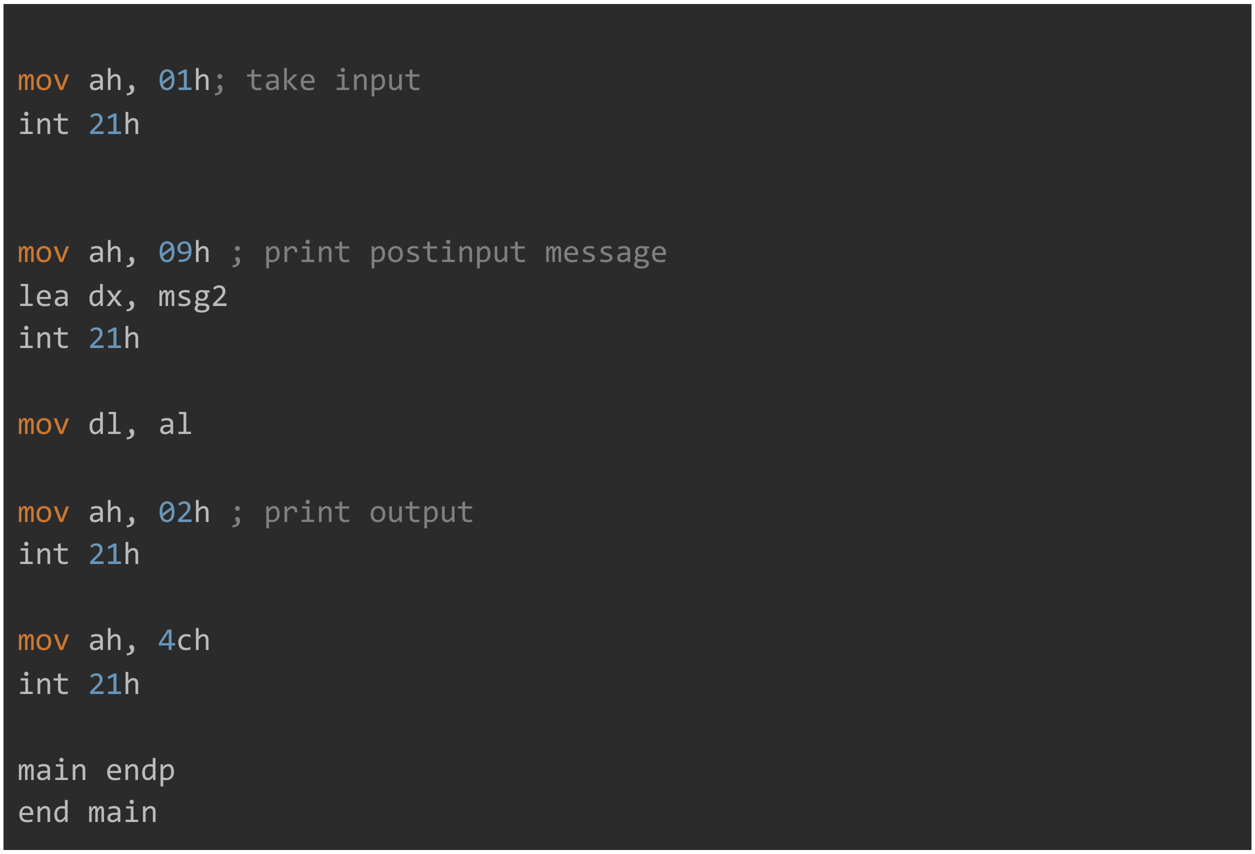


```
C:\Q5>q5
Display a Terminating Message.

GOOD BYE!!!
```

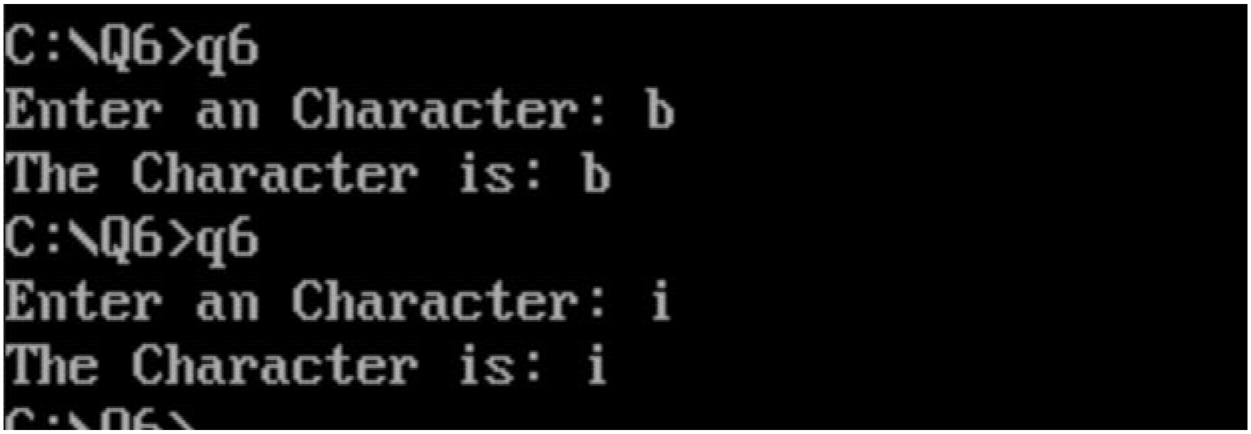
### Q6. Write and test a MASM program to Take a character from the keyboard and print it.

**Code.**

```
.model SMALL
.stack 100H

.data
msg1 db "Enter an Character: $"
msg2 db 10d, 13d, "The Character is: $"

.code

main proc
mov ax, @DATA
mov ds, ax

mov ah, 09h ; print preinput message
lea dx, msg1
int 21h
```



```
mov ah, 01h; take input
int 21h


mov ah, 09h ; print postinput message
lea dx, msg2
int 21h

mov dl, al

mov ah, 02h ; print output
int 21h

mov ah, 4ch
int 21h

main endp
end main
```

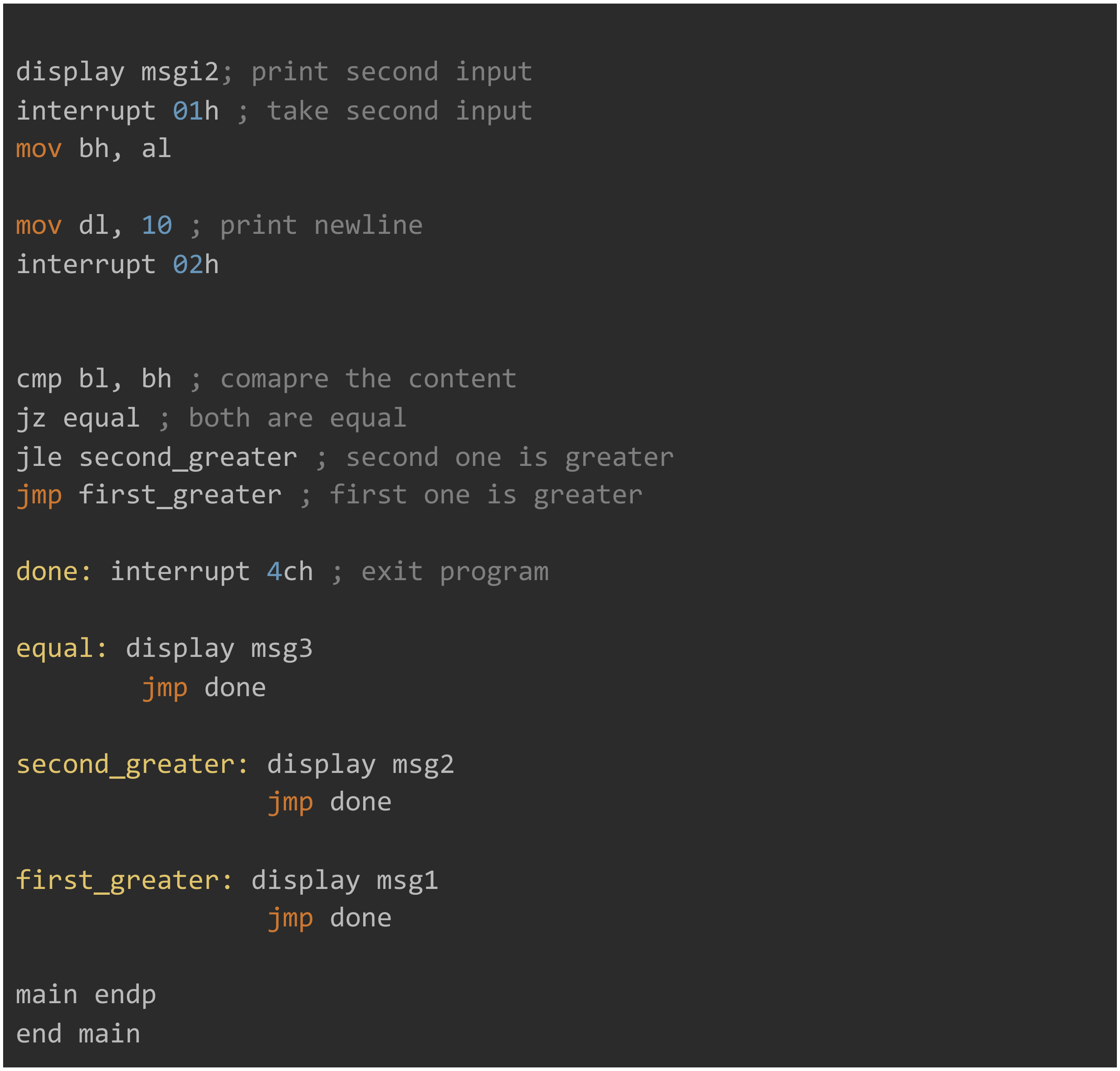


```
C:\Q6>q6
Enter an Character: b
The Character is: b
C:\Q6>q6
Enter an Character: i
The Character is: i
C:\Q6>
```

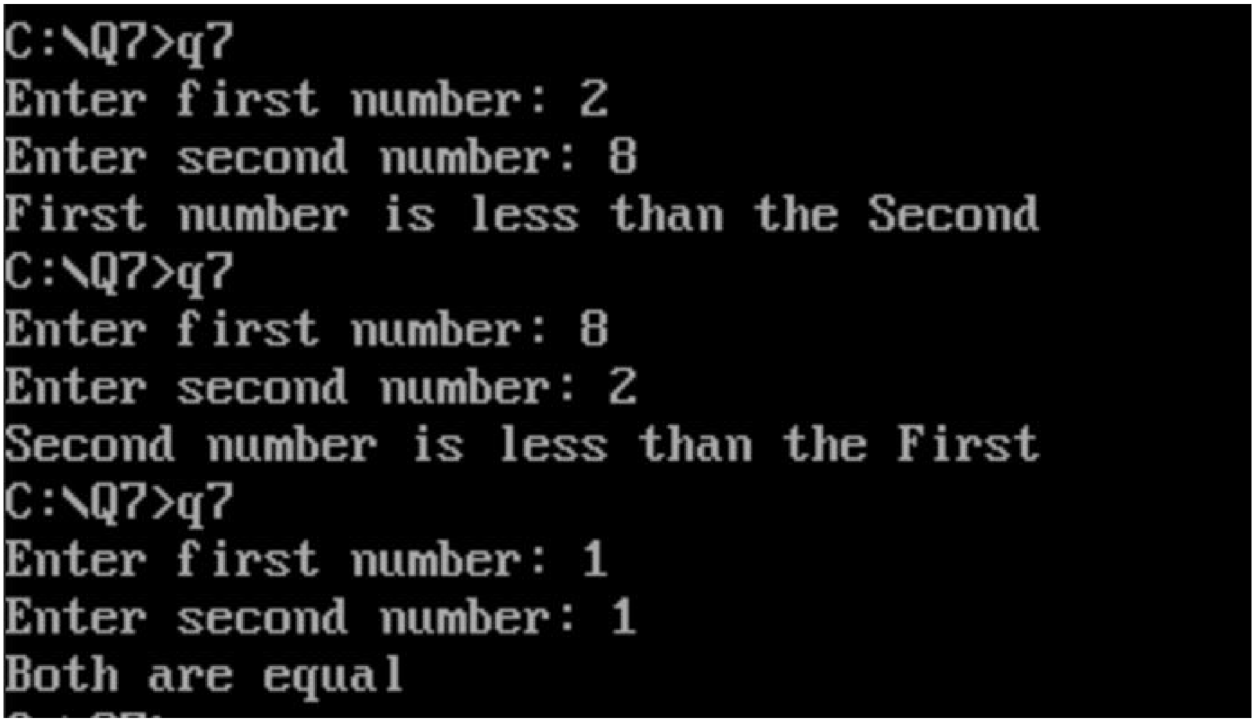
## Q7. Write and test a MASM program to validate second numbers is less than the first.

**Code.**

```
.model small
.stack 100h

.data
msgi1 db "Enter first number: $"
msgi2 db "Enter second number: $"
msg1 db "Second number is less than the First$"
msg2 db "First number is less than the Second$"
msg3 db "Both are equal$"

display macro msg ; macro for print message
    mov ah, 09h
    lea dx, msg
    int 21h
endm

interrupt macro inttype ; for creating interrupt
    mov ah, inttype
    int 21h
endm

.code

main proc
mov ax, @data
mov ds, ax

display msgi1 ; print input message
interrupt 01h ; take first input

mov bl, al

mov dl, 10 ; print newline
interrupt 02h
```



```
display msgi2; print second input
interrupt 01h ; take second input
mov bh, al

mov dl, 10 ; print newline
interrupt 02h


cmp bl, bh ; comapre the content
jz equal ; both are equal
jle second_greater ; second one is greater
jmp first_greater ; first one is greater

done: interrupt 4ch ; exit program

equal: display msg3
        jmp done

second_greater: display msg2
                jmp done

first_greater: display msg1
                jmp done

main endp
end main
```



```
C:\Q7>q7
Enter first number: 2
Enter second number: 8
First number is less than the Second
C:\Q7>q7
Enter first number: 8
Enter second number: 2
Second number is less than the First
C:\Q7>q7
Enter first number: 1
Enter second number: 1
Both are equal
```

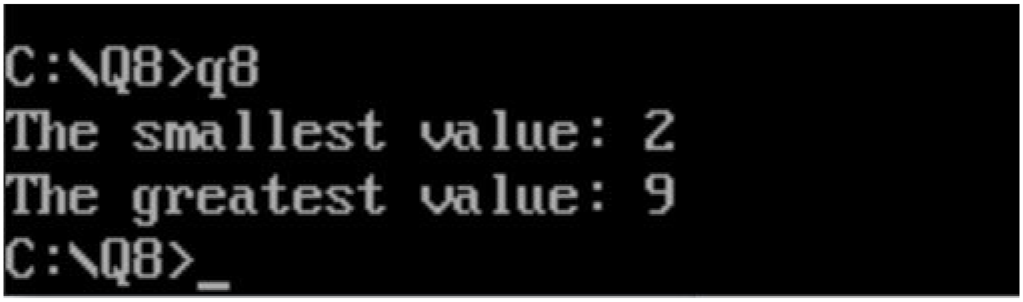
### Q8. Write and test a MASM program to find maximum and minimum from an array.

**Code.**

```
.model small
.stack 100h

.data
array db 3,2,5,9,6,3,4
len dw $-array
msg1 db "The smallest value: $"
msg2 db "The greatest value: $"

display macro msg
```

```
    mov ah, 09h
    lea dx, msg
    int 21h
endm

.code

main proc
mov ax, @data
mov ds, ax

mov si, offset array ; initialize source index with inital address of array
mov cx, len       ; array length
mov bh, [si] ; contains minimum
mov bl, [si] ; contains maximum

loop1:      cmp [si], bh
            jle updatemin
maxcheck:   cmp bl, [si]
            jle updatemax
resume:     inc si
            loop loop1

add bh, 48 ; to convert to ascii
add bl, 48 ; ascii

display msg1 ; display maximum message and print maximum
mov ah, 02h
mov dl, bh
int 21h

mov dl, 10 ; to print a newline
int 21h

display msg2 ; display minimum message and print minimum
mov ah, 02h
mov dl, bl
int 21h
```

```
mov ah, 4ch
int 21h

updatemin: mov bh, [si]
            jmp maxcheck
updatemax: mov bl, [si]
            jmp resume

main endp
end main
```



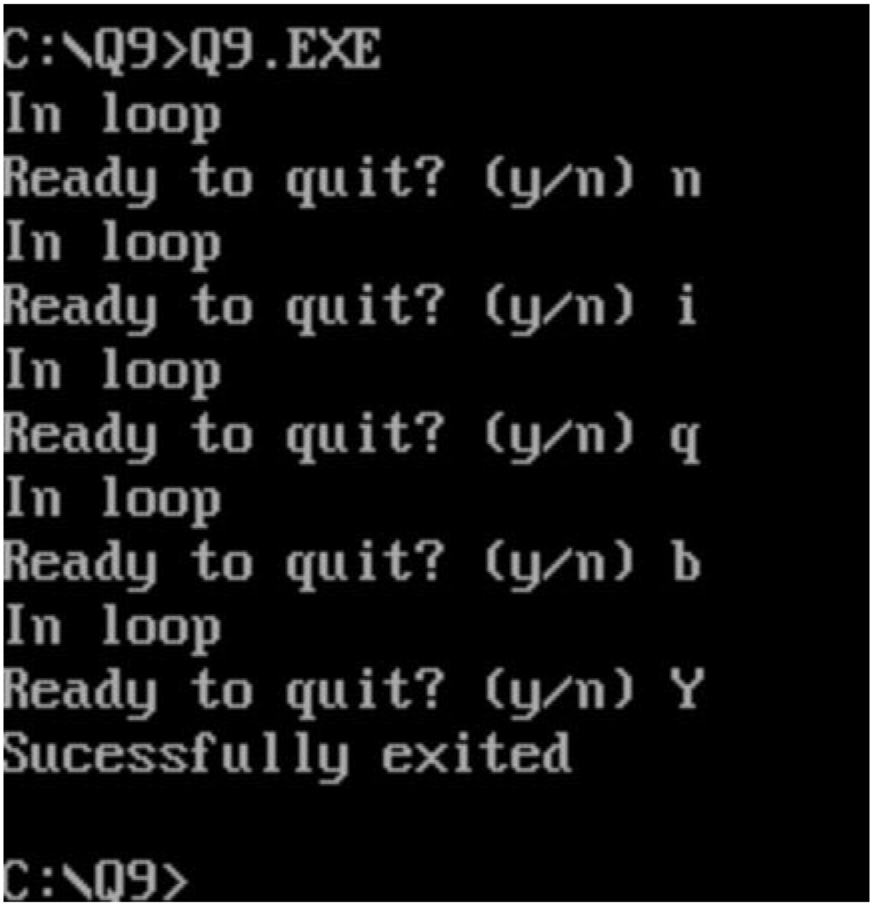
```
C:\Q8>q8
The smallest value: 2
The greatest value: 9
C:\Q8>_
```

### Q9. Write and test a MASM program to loop until the user decides to quit.

Code.

```
.model small
.stack 100h

.data
quitmsg db "Ready to quit? (y/n) $"
inloop db "In loop", 10, 13, "$"
outloop db "Sucessfully exited", 10, 13, "$"
```

```
.code

main proc

mov ax, @DATA
mov ds, ax
;creating a loop
reloop: mov ah, 09h
lea dx, inloop ; print inloop message
int 21h

lea dx, quitmsg; asking if user wants to quit
int 21h

mov ah , 01h ; taking input
int 21h
mov dh, al

mov dl, 10 ; print newline
mov ah, 02h
int 21h

cmp dh, 'y' ; checking if it is 'y' or 'Y'
jz exitloop
cmp dh, 'Y'
jz exitloop

jmp reloop ; if not reloop

exitloop: mov ah, 09h
lea dx, outloop ; print when exited from loop
int 21h

mov ah, 4ch; exit program
int 21h

main endp
end main
```



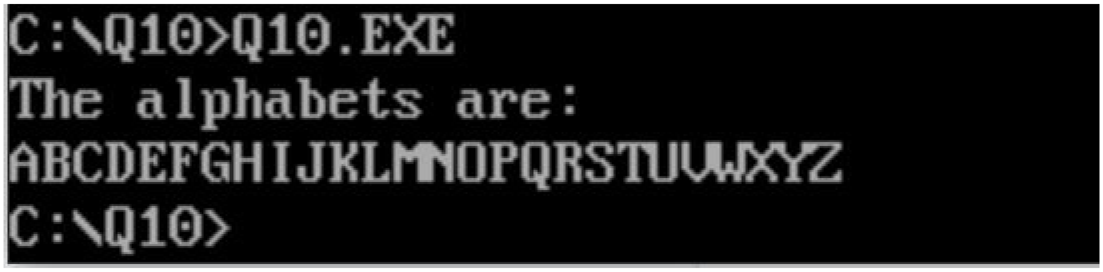
```
C:\Q9>Q9.EXE
In loop
Ready to quit? (y/n) n
In loop
Ready to quit? (y/n) i
In loop
Ready to quit? (y/n) q
In loop
Ready to quit? (y/n) b
In loop
Ready to quit? (y/n) Y
Sucessfully exited

C:\Q9>
```

### Q10. Write and test a MASM program to print all the characters from A-Z.

Code.

```
.model SMALL
.stack 100H

.data
msg1 db "The alphabets are: ", 10, 13, "$"

.code
```

```
main proc

mov ax, @DATA
mov ds, ax

mov ah, 09h
lea dx, msg1 ; print alphabets message
int 21h

mov cx, 26 ; initialize counter
mov dl, 'A' ; initializa with 'A'
mov ah, 02h

loop1:  int 21h ; print character
        add dl, 01h ; increment loop
        loop loop1 ; reloop

mov ah, 4ch ; exit program
int 21h

main endp
end main
```



```
C:\Q10>Q10.EXE
The alphabets are:
ABCDEFGHIJKLMNOPQRSTUVWXYZ
C:\Q10>
```