Lab: 7

Fahid Imran

Roll No: 23i-0061

COAL

Instructor

Mr. Sulaman Saboor

Fast NUCES Islamabad Campus

Task1:

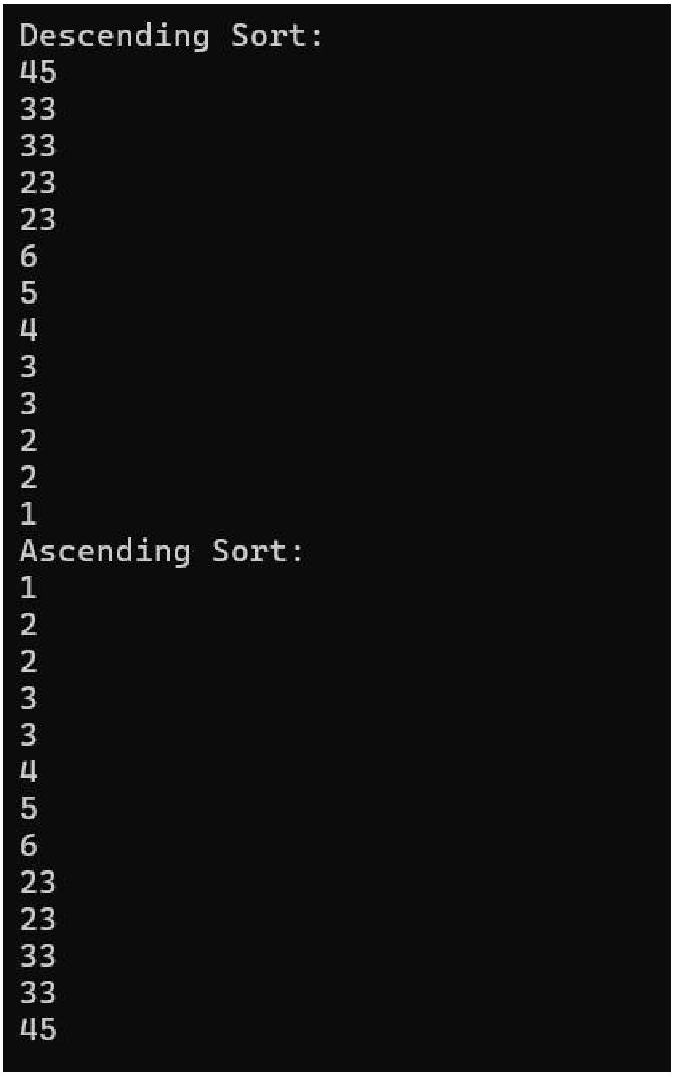
Code:

```
include irvine32.inc
.386
.model flat,stdcall
.stack 4096

.data
arr byte 1,3,2,4,3,5,6,23,2,33,45,33,23
var dword ?
len dword ?
len2 dword ?
msg1 Byte 'Ascending Sort: ', 0
msg2 Byte 'Descending Sort: ', 0

.code
main PROC

mov len, lengthof arr
mov eax, len
mov len2, eax
dec len2

mov ecx, 0
.while ecx < len
  mov ebx, 0
  .while ebx < len2
         mov al, arr[ebx]
         mov ah, arr[ebx+1]
         cmp al, ah
         jl swap
         jmp endLoop
         swap:
         mov arr[ebx], ah
         mov arr[ebx+1], al
  endLoop:
  inc ebx
  .endw
inc ecx
.endw
```

```
mov edx , offset msg2
call writestring
call crlf

mov ecx, 0
.while ecx < len
mov eax, 0
mov al, arr[ecx]
call writedec
call crlf
inc ecx
.endw

mov ecx, 0
.while ecx < len
  mov ebx, 0
  .while ebx < len2
         mov al, arr[ebx]
         mov ah, arr[ebx+1]
         cmp al, ah
         jg swap2
         jmp endLoop2
         swap2:
         mov arr[ebx], ah
         mov arr[ebx+1], al
  endLoop2:
  inc ebx
  .endw
inc ecx
.endw

mov edx , offset msg1
call writestring
call crlf

mov ecx, 0
.while ecx < len
mov eax, 0
mov al, arr[ecx]
call writedec
call crlf
```

```
inc ecx
.endw
exit
main endp
END main
```

Output:



```
Descending Sort:
45
33
33
23
23
6
5
4
3
3
2
2
1
Ascending Sort:
1
2
2
3
3
4
5
6
23
23
33
33
45
```

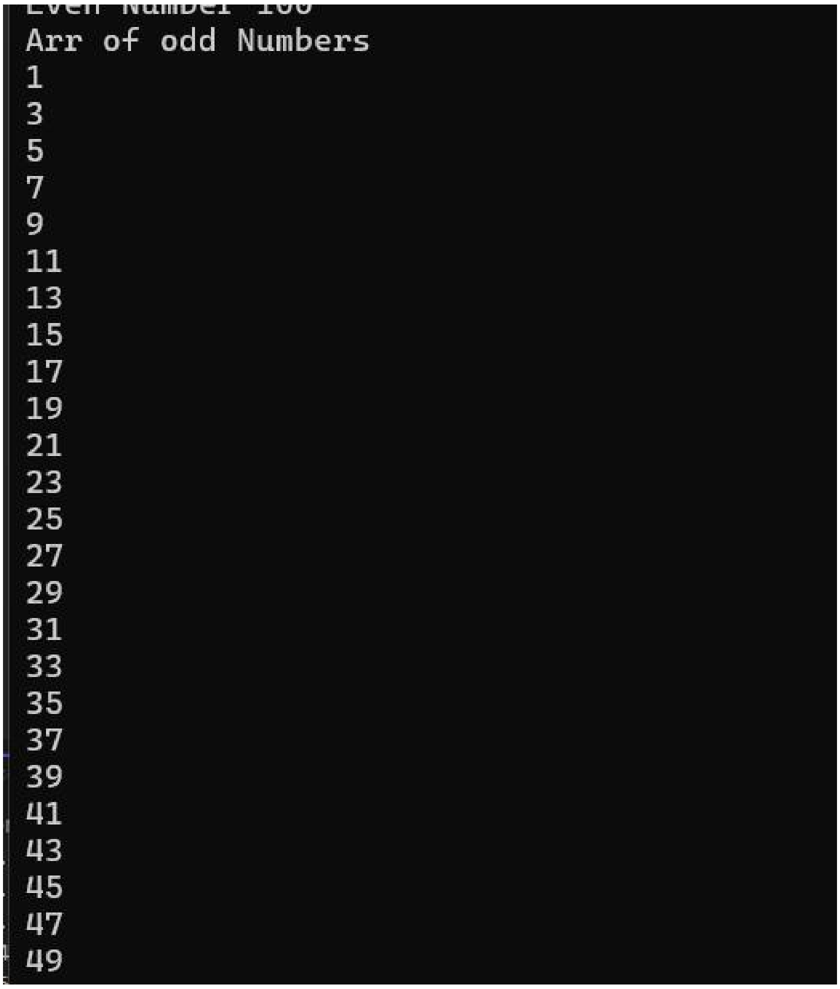
Task2:

Code:

```
Include irvine32.inc
.386
.model flat, stdcall
.stack 4096
```

```
.data
arr Dword 50 Dup (0)
msg Byte 'Even Number ', 0
msg2 Byte 'Arr of odd Numbers ', 0
var dword 2
.code
main PROC

mov ecx , 1
mov ebx, 0
.while ecx < 101
  mov eax, ecx
  mov edx, 0
  div var

  cmp edx, 0
  je label1
  jne label2
  label1:
        mov edx, offset msg
        call writestring
        mov eax, ecx
        call writedec
        call crlf
        inc ecx
  label2:
        mov arr[ebx], ecx
        add ebx, 4
        inc ecx


.endw

mov ecx , lengthof arr
mov edx, offset msg2
call writestring
call crlf

mov ecx , 0
mov var, sizeof arr
.while ecx<var
    mov eax, 0
    mov eax,arr[ecx]
    call writedec
    call crlf
  add ecx,4

.endw

exit
main endp
end main
```

Output:



```
Even Number 100
Arr of odd Numbers
1
3
5
7
9
11
13
15
17
19
21
23
25
27
29
31
33
35
37
39
41
43
45
47
49
```

Task3:

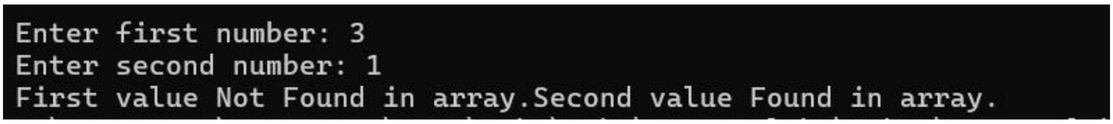
Code:

```
include irvine32.inc
.386
```

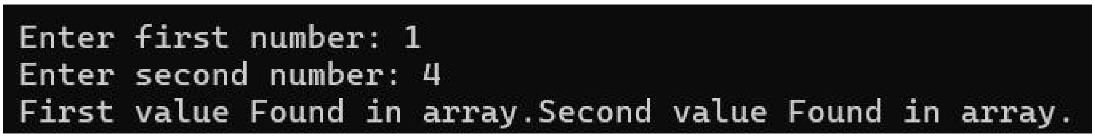
```
.model flat, stdcall
.stack 4096

.data
arr word 1,4,55,67,89,23,45,12,34,56
var1 word ?
var2 word ?
msg1 byte 15 Dup (0)
msg2 byte 15 Dup (0)
msg3 byte 'Enter first number: ', 0
msg4 byte 'Enter second number: ', 0
msg5 byte 'First value ' , 0
msg6 byte 'Second value ' , 0
msg7 byte ' in array.', 0
len dword ?
.code
main PROC
mov edx , offset msg3
call writestring
call readint
mov var1, ax

mov edx , offset msg4
call writestring
call readint
mov var2, ax

mov ecx , 0
mov len , sizeof arr
.while ecx < len
  mov ax, arr[ecx]
  cmp ax, var1
  je label1
  jmp label4

  label1:

  mov msg1[0], 'F'
  mov msg1[1], 'o'
  mov msg1[2], 'u'
  mov msg1[3], 'n'
  mov msg1[4], 'd'
  jmp label4


  label4:
  add ecx, 2



.endw

 mov al, msg1[0]
 cmp al, 0
 jne goto1

    mov msg1[0], 'N'
    mov msg1[1], 'o'
    mov msg1[2], 't'
    mov msg1[3], ' '
    mov msg1[4], 'F'
    mov msg1[5], 'o'
    mov msg1[6], 'u'
    mov msg1[7], 'n'
    mov msg1[8], 'd'

goto1:
mov ecx , 0
mov len , sizeof arr
.while ecx < len
  mov ax, arr[ecx]
  cmp ax, var2
  je label2
  jmp label5

  label2:

  mov msg2[0], 'F'
  mov msg2[1], 'o'
  mov msg2[2], 'u'
  mov msg2[3], 'n'
  mov msg2[4], 'd'
  jmp label5
```

```
      label5:
      add ecx, 2



    .endw

    mov al, msg2[0]
     cmp al, 0
     jne goto2

        mov msg2[0], 'N'
        mov msg2[1], 'o'
        mov msg2[2], 't'
        mov msg2[3], ' '
        mov msg2[4], 'F'
        mov msg2[5], 'o'
        mov msg2[6], 'u'
        mov msg2[7], 'n'
        mov msg2[8], 'd'

    goto2:
    mov edx , offset msg5
    call writestring
    mov edx , offset msg1
    call writestring
    mov edx , offset msg7
    call writestring

    mov edx , offset msg6
    call writestring
    mov edx , offset msg2
    call writestring
    mov edx , offset msg7
    call writestring
    exit
    main endp
end main
```

Output:



```
Enter first number: 3
Enter second number: 1
First value Not Found in array.Second value Found in array.
```



```
Enter first number: 1
Enter second number: 4
First value Found in array.Second value Found in array.
```

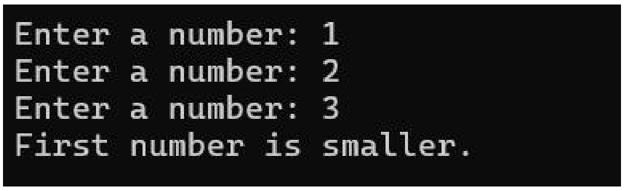
Task4:

Code:

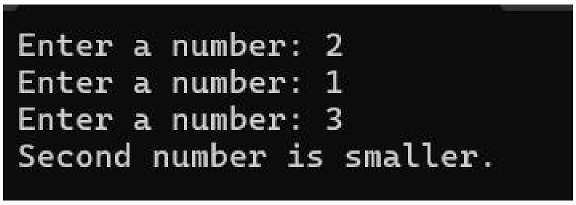
```
include irvine32.inc
.386
.model flat, stdcall
.stack 4096

.data
msg byte 'Enter a number: ', 0
var1 word ?
var2 word ?
var3 word ?
msg1 byte 'First number is smaller.', 0
msg2 byte 'Second number is smaller.', 0
msg3 byte 'Third number is smaller.', 0
.code
main PROC

mov edx , offset msg
call writestring
call readint
mov var1, ax

mov edx , offset msg
call writestring
call readint
mov var2, ax

mov edx , offset msg
call writestring
call readint
```

```
    mov var3, ax

    mov ax, var1
    cmp ax, var2
    jg point_1
      cmp ax, var3
      jg point_2
      mov edx, offset msg1
      call writestring
      call crlf
      jmp endProg

    point_1:
      mov ax, var2
      cmp ax, var3
      jg point_2
      mov edx, offset msg2
      call writestring
      call crlf
      jmp endProg
    point_2:
    mov edx, offset msg3
    call writestring
    call crlf
    endProg:

    exit
    main endp
end main
```

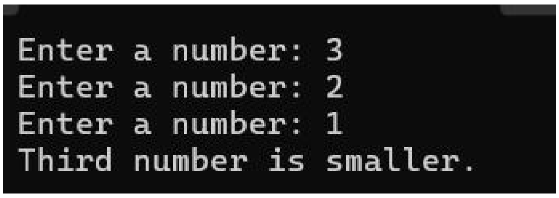
Output:



```
Enter a number: 1
Enter a number: 2
Enter a number: 3
First number is smaller.
```



```
Enter a number: 2
Enter a number: 1
Enter a number: 3
Second number is smaller.
```



```
Enter a number: 3
Enter a number: 2
Enter a number: 1
Third number is smaller.
```

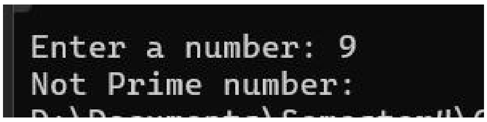
Task5:

Code:

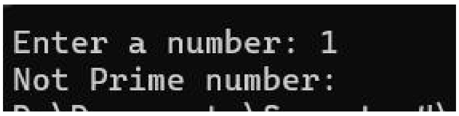
```
include irvine32.inc
.386
.model flat, stdcall
.stack 4096

.data

msg Byte 'Enter a number: ', 0
msg1 Byte 'Prime number: ', 0
msg2 Byte 'Not Prime number: ', 0
var word ?
.code
main PROC

mov edx, offset msg
call writestring
call readint
mov var , ax

cmp ax, 2
```

```
    jl point_1

    mov cx, 2
    .while cx < var
      mov ax, var
      mov edx, 0
      div cx
      inc cx

      cmp edx, 0
      je point_1

    .endw

    point_2:
    mov edx , offset msg1
    call writestring
    jmp endProg

    point_1:
    mov edx , offset msg2
    call writestring
    jmp endProg



    endProg:
    exit
    main endp
end main
```

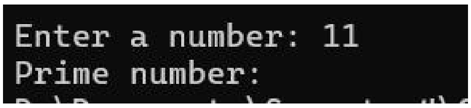
Output:



```
Enter a number: 9
Not Prime number:
```



```
Enter a number: 1
Not Prime number:
```



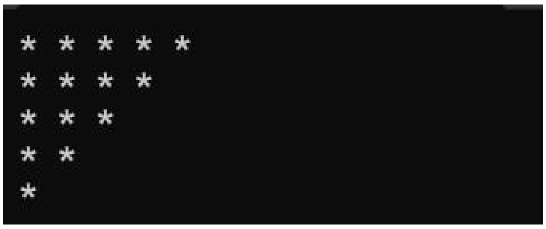
```
Enter a number: 11
Prime number:
```

Task6:

Code:

```
include irvine32.inc
.386
.model flat, stdcall
.stack 4096

.data
msg byte '* ', 0
.code
main PROC
mov ecx , 1

.while ecx <= 5
  mov ebx, ecx
  .while ebx <= 5
        mov edx , offset msg
        call writestring
        inc ebx
  .endw
  inc ecx
  call crlf
.endw
exit
main endp
end main
```

Output:



```
* * * * *
* * * *
* * *
* *
*
```