Lab: 12

Fahid Imran

Roll No: 23i-0061

COAL

Instructor

Mr. Sulaman Saboor

Fast NUCES Islamabad

Campus

Task1:

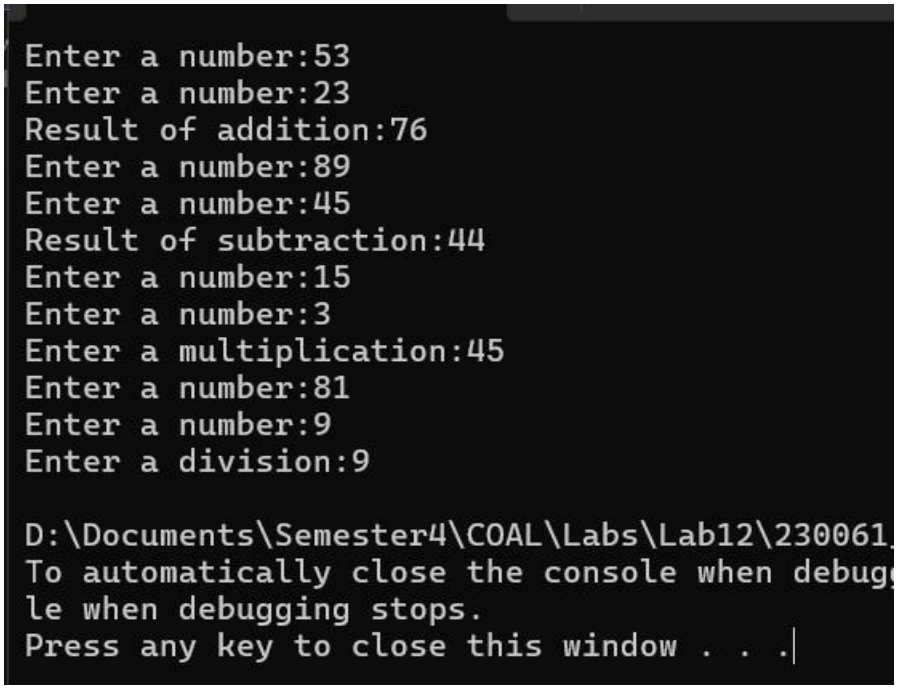
Code:

```
include Irvine32.inc
.386
.model flat, stdcall
.stack 4096

.data
  msg1 byte "Enter a number:", 0
  msg2 byte "Result of addition:", 0
  msg3 byte "Result of subtraction:", 0
  msg4 byte "Enter a multiplication:", 0
  msg5 byte "Enter a division:", 0
  addition word ?
  subtraction word ?
  multiplication word ?
  division word ?
  num1 word ?
  num2 word ?

.code

main PROC
  mov edx, offset msg1
  call writestring
  call readint
  mov num1, ax
  mov edx, offset msg1
  call writestring
  call readint
  mov num2, ax

  mov ax, num1
  add ax, num2
  mov addition, ax

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

```
mov edx, offset msg1
call writestring
call readint
mov num1, ax
mov edx, offset msg1
call writestring
call readint
mov num2, ax

mov ax, num1
sub ax, num2
mov subtraction, ax

mov edx, offset msg3
call writestring
call writedec
call crlf


mov edx, offset msg1
call writestring
call readint
mov num1, ax
mov edx, offset msg1
call writestring
call readint
mov num2, ax

mov ax, num1
mul num2
mov multiplication, ax

mov edx, offset msg4
call writestring
call writedec
call crlf

mov edx, offset msg1
call writestring
call readint
mov num1, ax
```

```
    mov edx, offset msg1
    call writestring
    call readint
    mov num2, ax

    mov edx, 0
    mov ax, num1
    div num2
    mov division, ax

    mov edx, offset msg5
    call writestring
    call writedec
    call crlf




exit
main endp
end main
```

Output:



Task2:

Code:

```
include Irvine32.inc
.386
.model flat, stdcall
.stack 4096

.data
  msg1 byte "Most frequent value is:", 0
  msg12 byte "* ", 0
  msg13 byte " ", 0
  arr byte 1, 2, 3, 1, 4, 2, 2

  counter dword 0

.code



func PROC

  push ecx
  mov counter , ecx

  outerloop:

        push ecx

        innerloop1:
              mov edx, offset msg13
              call writestring
        loop innerloop1
        pop ecx

        push ecx
        mov ebx, ecx
        dec ebx
```

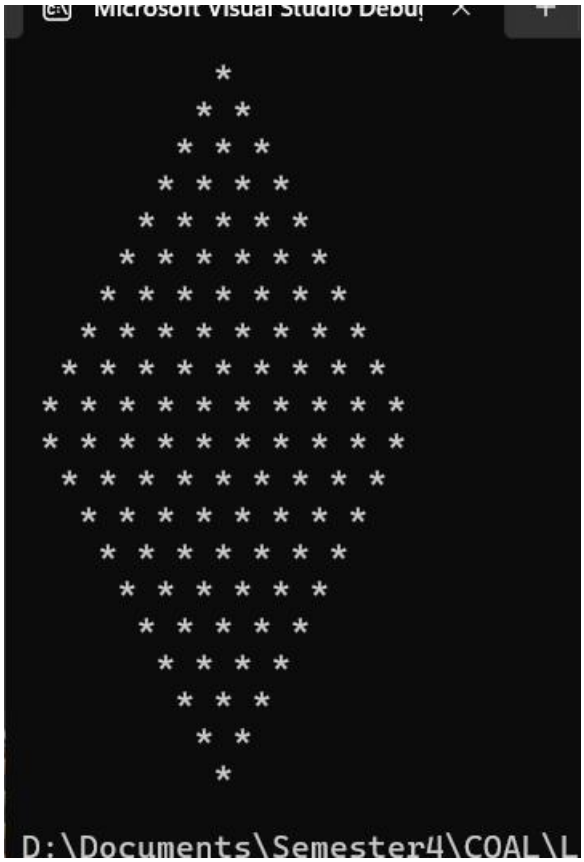
```
	mov ecx, counter
	sub ecx, ebx
	innerloop2:
		mov edx, offset msg12
		call writestring
	loop innerloop2
	pop ecx



	call crlf

loop outerloop


mov ecx, counter
outerloop2:

	push ecx
	mov ebx, ecx
	dec ebx
	mov ecx, counter
	sub ecx, ebx
	innerloop4:
		mov edx, offset msg13
		call writestring
	loop innerloop4
	pop ecx

	push ecx
	innerloop3:
		mov edx, offset msg12
		call writestring
	loop innerloop3
	pop ecx




	call crlf
```

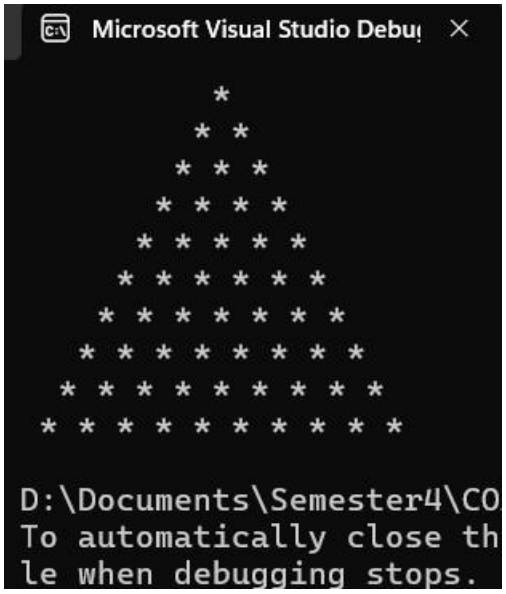
```
  loop outerloop2

  pop ecx
  ret
func endp

func2 PROC

  push ecx
  mov counter , ecx

  outerloop:

        push ecx

        innerloop1:
              mov edx, offset msg13
              call writestring
        loop innerloop1
        pop ecx

        push ecx
        mov ebx, ecx
        dec ebx
        mov ecx, counter
        sub ecx, ebx
        innerloop2:
              mov edx, offset msg12
              call writestring
        loop innerloop2
        pop ecx



        call crlf

  loop outerloop

  pop ecx
  ret
func2 endp
```

```
func3 PROC

  push ecx


  outerloop:

        push ecx

        innerloop1:
              mov edx, offset msg12
              call writestring
        loop innerloop1
        pop ecx
        call crlf

  loop outerloop

  pop ecx
  ret
func3 endp


main PROC


  mov ecx, 10
  call func
  call func2
  call func3


exit
main endp
end main
```

Output:


* * * * * * * * * * *
* * * * * * * * * *
* * * * * * * * *
* * * * * * *
* * * * * *
* * * * *
* * * *
* * *
* *
*
D:\Documents\Semester4\C
To automatically close t
le when debugging stops.


*
* *
* * *
* * * *
* * * * *
* * * * * *
* * * * * * *
* * * * * * * *
* * * * * * * * *
* * * * * * * * * *
* * * * * * * * * *
* * * * * * * * *
* * * * * * * *
* * * * * * *
* * * * * *
* * * * *
* * *
* *
*
D:\Documents\Semester4\COAL\L



Task3:

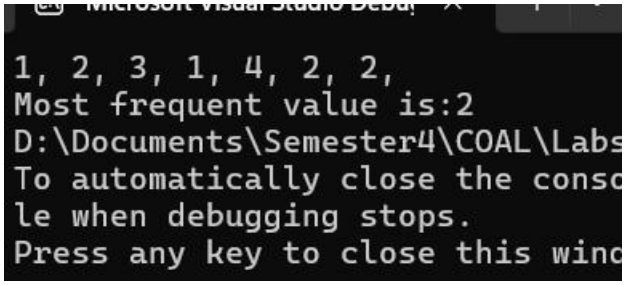
Code:

```
include Irvine32.inc
.386
.model flat, stdcall
.stack 4096

.data
  msg1 byte "Most frequent value is:", 0
  msg12 byte ", ", 0
  arr byte 1, 2, 3, 1, 4, 2, 2


  most byte 0
  mode byte 0
  count byte 0
  counter dword 0

.code
printArr PROC
  push ebx
  push ecx
  mov esi , 0
  label2:
        mov eax, 0
```

```
        mov al, byte ptr [ebx + esi]
        call writedec
        mov edx, offset msg12
        call writestring
        inc esi
  loop label2
  call crlf
  pop ecx
  pop ebx
  ret
printArr endp


func PROC
  push ebx
  push ecx
  mov esi , 0
  outerloop:
        mov count, 0
        push ecx
        push esi
        mov al, byte ptr [ebx + esi]
        mov edx, ebx
        mov ecx, [esp+8]
        mov esi, 0
        innerloop:
              cmp al, byte ptr [edx + esi]
              je jump1
              jmp endinner
              jump1:
                    inc count
              endinner:
              inc esi
        loop innerloop
        mov dl, count
        cmp dl, most
        jg jump2
        jmp endouter
        jump2:
              mov most, dl
              mov mode, al
        endouter:
```

```
        pop esi
        pop ecx

        inc esi
    loop outerloop

    mov edx, offset msg1
    call writestring
    mov al, mode
    call writedec

    pop ecx
    pop ebx

    ret
func endp

main PROC

    mov ebx, offset arr
    mov ecx, lengthof arr
    call printArr
    mov ebx, offset arr
    mov ecx, lengthof arr
    call func


exit
main endp
end main
```

Output:



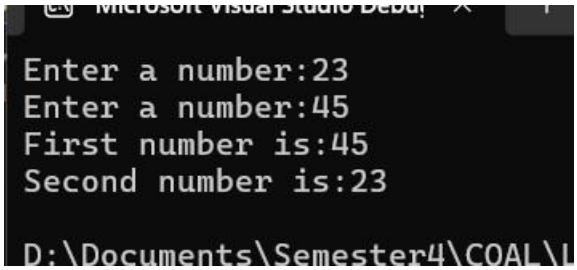
Task4:

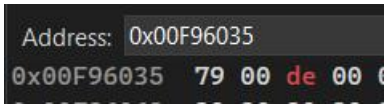
Code:

```
include Irvine32.inc
.386
.model flat, stdcall
.stack 4096

.data
  msg1 byte "Enter a number:", 0
  msg2 byte "First number is:", 0
  msg3 byte "Second number is:", 0

  temp word ?
  num1 word ?
  num2 word ?

.code

main PROC
  mov edx, offset msg1
  call writestring
  call readint
  mov num1, ax
  mov edx, offset msg1
  call writestring
  call readint
  mov num2, ax

  mov ax, num1
  mov bx, num2

  mov temp, ax
  mov ax, bx
  mov bx, temp

  mov num1, ax
  mov num2, bx

  mov edx, offset msg2
  call writestring
```

```
  mov ax, num1
  call writedec
  call crlf
  mov edx, offset msg3
  call writestring
  mov ax, num2
  call writedec
  call crlf



exit
main endp
end main
```

Output:





Task5:

Code:

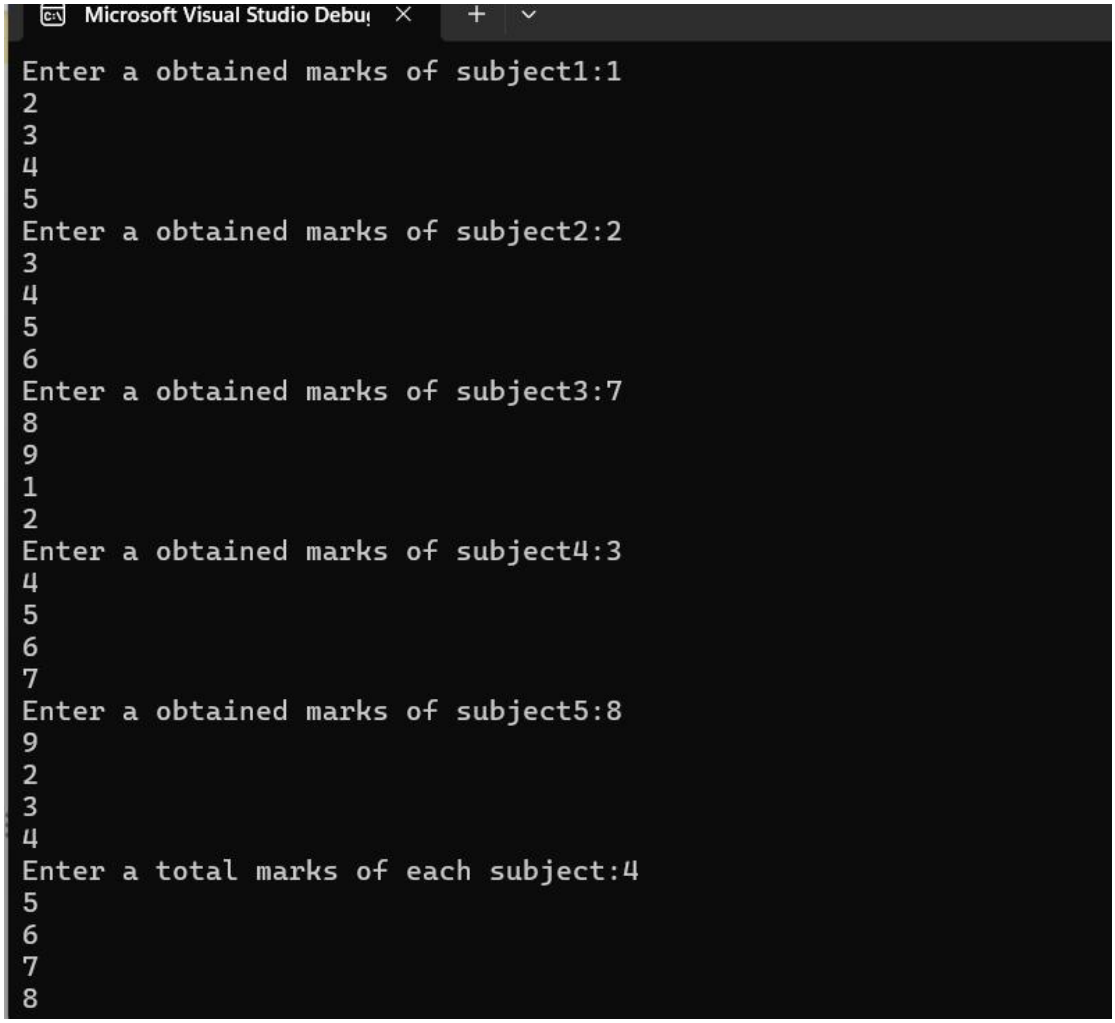
```
include Irvine32.inc
.386
.model flat, stdcall
.stack 4096

.data
  msg byte "Enter a marks:", 0
  msg1 byte "Enter a obtained marks of subject1:", 0
```

```
  msg2 byte "Enter a total marks of each subject:", 0
  msg3 byte "Enter a obtained marks of subject2:", 0
  msg4 byte "Enter a total marks of subject2:", 0
  msg5 byte "Enter a obtained marks of subject3:", 0
  msg6 byte "Enter a total marks of subject3:", 0
  msg7 byte "Enter a obtained marks of subject4:", 0
  msg8 byte "Enter a total marks of subject4:", 0
  msg9 byte "Enter a obtained marks of subject5:", 0
  msg10 byte "Enter a total marks of subject5:", 0
  msg11 byte "Array is:", 0
  msg12 byte ", ", 0



  sub1_marks byte 5 dup(?), 0
  sub_total byte 5 dup(?), 0
  sub2_marks byte 5 dup(?), 0
  sub2_total byte 5 dup(?), 0
  sub3_marks byte 5 dup(?), 0
  sub3_total byte 5 dup(?), 0
  sub4_marks byte 5 dup(?), 0
  sub4_total byte 5 dup(?), 0
  sub5_marks byte 5 dup(?), 0
  sub5_total byte 5 dup(?), 0
  percentage byte 5 dup(?), 0
  sums byte 5 dup(0), 0
  min byte ?
  max byte ?
  total_sum1 byte 0
  total_sum2 byte 0
  total_sum3 byte 0
  total_sum4 byte 0
  total_sum5 byte 0
  sum byte 0
  sum2 byte 0

  num1 word ?
  num2 word ?

.code
printArr PROC
  push ebx
```

```
  push ecx
  mov esi , 0
  mov edx, offset msg11
  label2:
        mov al, byte ptr [ebx + esi]
        call writedec
        mov edx, offset msg12
        call writestring
        inc esi
  loop label2
  call crlf
  pop ecx
  pop ebx
  ret
printArr endp

inputArr PROC
  push ebx
  push ecx
  mov esi , 0
  loop1:
        call readint
        mov byte ptr [ebx + esi], al
  loop loop1
  pop ecx
  pop ebx
  ret
inputArr endp

takeAllINPUT PROC
  mov edx, offset msg1
  call writestring
  mov ebx, offset sub1_marks
  mov ecx, 5
  call inputArr

  mov edx, offset msg3
  call writestring
  mov ebx, offset sub2_marks
  mov ecx, 5
  call inputArr
```

```
  mov edx, offset msg5
  call writestring
  mov ebx, offset sub3_marks
  mov ecx, 5
  call inputArr

  mov edx, offset msg7
  call writestring
  mov ebx, offset sub4_marks
  mov ecx, 5
  call inputArr

  mov edx, offset msg9
  call writestring
  mov ebx, offset sub5_marks
  mov ecx, 5
  call inputArr

  mov edx, offset msg2
  call writestring
  mov ebx, offset sub_total
  mov ecx, 5
  call inputArr
  ret
takeAllINPUT endp

main PROC
  call takeAllINPUT


exit
main endp
end main
```

Output:


Microsoft Visual Studio Debu
Enter a obtained marks of subject1:1
2
3
4
5
Enter a obtained marks of subject2:2
3
4
5
6
Enter a obtained marks of subject3:7
8
9
1
2
Enter a obtained marks of subject4:3
4
5
6
7
Enter a obtained marks of subject5:8
9
2
3
4
Enter a total marks of each subject:4
5
6
7
8