

COAL LAB#011 TASKS

Name: Saman Khan

ID: 19K-0354

Section: H

TASK#01:

TITLE My First Program (Test.asm)

INCLUDE Irvine32.inc

.data

msg1 BYTE "Enter three numbers : ",0

msg2 BYTE "The product of these three numbers is : ",0

.code

main PROC

mov edx, offset msg1

call writestring

mov ecx, 3

L1:

call readint

push eax

LOOP L1

call ThreeProd

pop eax

pop eax

pop eax

exit

main ENDP

ThreeProd PROC

enter 0,1

mov eax, [ebp+8]

mov ebx, [ebp+12]

imul ebx

mov ebx, [ebp+16]

imul ebx

mov edx, offset msg2

call writestring

call writeint

call crlf

leave

ret

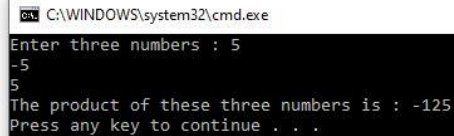
ThreeProd ENDP

END main

```
msg2 BYTE "The product of these three numbers is : ",0
```

```
.code  
main PROC  
mov edx, offset msg1  
call writestring  
mov ecx, 3  
L1:  
call readint  
push eax  
LOOP L1  
call ThreeProd  
pop eax  
pop eax  
pop eax  
exit  
main ENDP
```

```
ThreeProd PROC  
enter 0,1  
mov eax, [ebp+8]  
mov ebx, [ebp+12]  
imul ebx  
mov ebx, [ebp+16]  
imul ebx  
mov edx, offset msg2  
call writestring  
call writeint  
call crlf  
leave  
ret  
ThreeProd ENDP  
END main
```



```
C:\WINDOWS\system32\cmd.exe  
Enter three numbers : 5  
-5  
5  
The product of these three numbers is : -125  
Press any key to continue . . .
```

TASK#02:

TITLE My First Program (Test.asm)

INCLUDE Irvine32.inc

```
.data  
arr SDWORD 20 dup(?)  
min SDWORD ?  
max SDWORD ?  
msg1 BYTE "Enter 20 numbers in the array : ",0  
msg2 BYTE "MIN = ",0  
msg3 BYTE "MAX = ",0
```

MinMaxArray PROTO, X: PTR DWORD

```
.code  
main PROC  
push ebp  
mov ebp, esp  
mov edx, offset msg1  
call writestring  
mov ecx, lengthof arr  
mov esi, 0  
L1:  
call readint  
mov arr[esi], eax  
add esi, 4
```

```
Loop L1
mov esi, offset arr
INVOKE MinMaxArray, addr arr
pop ebp
add esp, 4
ret
exit
main ENDP
```

```
MinMaxArray PROC, X: PTR DWORD
push ebp
mov ebp, esp
push esi
mov ecx, lengthof arr
mov eax, [esi]
L1:
add esi, 4
mov ebx, [esi]
cmp eax, ebx
jg next
mov eax, ebx
next:
Loop L1
mov edx, offset msg3
call writestring
call writeint
mov max, eax
call crlf
pop esi
mov ecx, lengthof arr
mov eax, [esi]
L2:
add esi, 4
mov ebx, [esi]
cmp eax, ebx
jl find
mov eax, ebx
find:
Loop L2
mov edx, offset msg2
call writestring
call writeint
mov min, eax
call crlf
pop ebp
add esp, 8
ret
MinMaxArray ENDP
```

END main

```
mov eax, ebx
next:
Loop L1
mov edx, offset m
call writestring
call writeint
mov max, eax
call crlf
pop esi
mov ecx, lengthof
mov eax, [esi]
L2:
add esi, 4
mov ebx, [esi]
cmp eax, ebx
j1 find
t
output from: Build
Creating "Debug\c11
NASM:
Assembling [Input
est.asm(74): warni
ink:
LINK : C:\Users\B
coal.vcxproj -> C
inalizeBuildStatus
Deleting file "De
Touching "Debug\c
uild succeeded.
```

C:\WINDOWS\system32\cmd.exe

Enter 20 numbers in the array : 3

50

1

0

-9

23

100

-66

200

4

7

10

22

55

30

11

15

20

16

17

MAX = +200

MIN = -66

Press any key to continue . . .

TASK#03:

TITLE My First Program (Test.asm)

INCLUDE Irvine32.inc

.data

msg1 BYTE "Enter a number : ",0

msg2 BYTE "Square of the number entered is : ",0

.code

main PROC

call LocalSquare

exit

main ENDP

LocalSquare PROC

LOCAL temp : SDWORD

enter 4,1

mov edx, offset msg1

call writestring

call readint

mov temp, eax

imul temp

mov edx, offset msg2

```

call writestring
call writeint
call crlf
leave
ret
LocalSquare ENDP
end main

```

The screenshot shows a Visual Studio IDE with a file named 'test.asm' open. The code in the editor is as follows:

```

TITLE My First Program (Test.asm)
INCLUDE Irvine32.inc

.data
msg1 BYTE "Enter a number : ",0
msg2 BYTE "Square of the number entered is : ",0

.code
main PROC
call LocalSquare
exit
main ENDP

LocalSquare PROC
LOCAL temp : SDWORD
enter 4,1
mov edx, offset msg1
call writestring
call readint
mov temp, eax
imul temp
mov edx, offset msg2
call writestring
call writeint
call crlf
leave
ret
LocalSquare ENDP
end main

```

To the right of the code editor, a command prompt window titled 'C:\WINDOWS\system32\cmd.exe' is open, showing the output of the program:

```

Enter a number : -7
Square of the number entered is : +49
Press any key to continue . . .

```

TASK#04:

```

TITLE My First Program (Test.asm)
INCLUDE Irvine32.inc

```

```

.data
time DWORD ?
msg1 BYTE "Enter a number : ",0
msg2 BYTE "Factorial : ",0
msg3 BYTE "Time taken using recursive procedure (in seconds): ",0

```

```

Factorial PROTO, N: DWORD

```

```

.code
main PROC
call GetMSeconds
mov time, eax
mov edx, OFFSET msg1
call WriteString

```

```

call readint
mov ebx, eax
INVOKE Factorial, eax
mov edx, OFFSET msg2
call WriteString
call writeint
call crlf
call GetMSeconds
sub eax, time
mov edx, OFFSET msg3
call WriteString
call writeint
call crlf
exit
main ENDP

```

```

Factorial PROC, N: DWORD
push ebp
mov ebp, esp
dec ebx
cmp ebx, 0
jz last
mul ebx
INVOKE Factorial, ebx
last:
pop ebp
add esp, 8
ret
Factorial endp
end main

```

The screenshot shows a Visual Studio IDE with two windows. The left window displays an assembly file named 'Test.asm' with the following code:

```

TITLE My First Program (Test.asm)
INCLUDE Irvine32.inc

.data
time DWORD ?
msg1 BYTE "Enter a number : ",0
msg2 BYTE "Factorial : ",0
msg3 BYTE "Time taken using recursive procedure (in seconds): ",0

Factorial PROTO, N: DWORD

.code
main PROC
call GetMSeconds
mov time, eax
mov edx, OFFSET msg1

```

The right window shows the command prompt output of the program:

```

C:\WINDOWS\system32\cmd.exe
Enter a number : 5
Factorial : +120
Time taken using recursive procedure (in seconds): +712
Press any key to continue . . .

```

The IDE interface includes a taskbar at the top with several 'TASK1.DLL' instances and a 'Test.asm' window. The 'Output' window at the bottom left shows 'Show output from: Build'.

TASK#05:

TITLE My First Program (Test.asm)

INCLUDE Irvine32.inc

.data

time DWORD ?

msg1 BYTE "Enter a number : ",0

msg2 BYTE "Factorial : ",0

msg3 BYTE "Time taken using iterative procedure (in seconds): ",0

Factorial PROTO, N: DWORD

.code

main PROC

call GetMSeconds

mov time, eax

mov edx, OFFSET msg1

call WriteString

call readint

mov ecx, eax

INVOKE Factorial, eax

mov edx, OFFSET msg2

call WriteString

call writeint

call crlf

call GetMSeconds

sub eax, time

mov edx, OFFSET msg3

call WriteString

call writeint

call crlf

exit

main ENDP

Factorial PROC, N: DWORD

push ebp

mov ebp, esp

dec ecx

L1:

mul ecx

Loop L1

pop ebp

add esp, 8

ret

Factorial endp

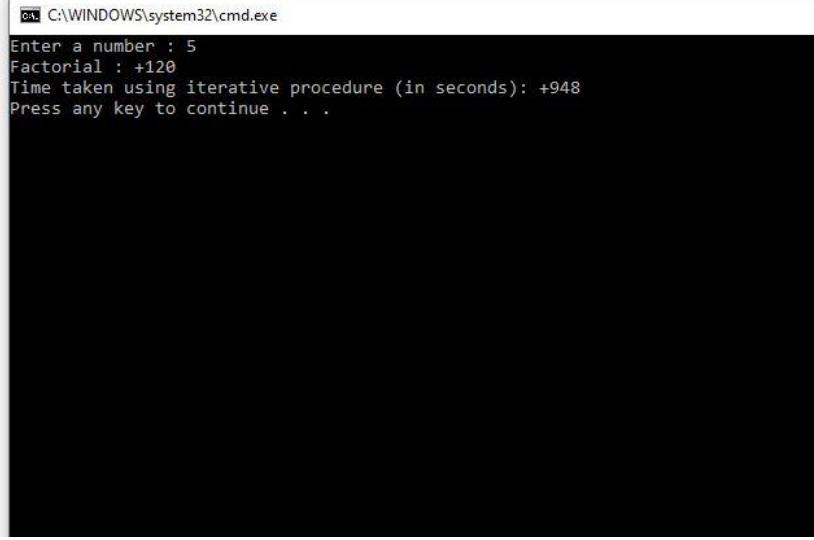
end main

```
msg1 BYTE "Enter a number : ",0
msg2 BYTE "Factorial : ",0
msg3 BYTE "Time taken using iterative procedure (in seconds): ",0
```

```
Factorial PROTO, N: DWORD
```

```
.code
main PROC
call GetMSeconds
mov time, eax
mov edx, OFFSET msg1
call WriteString
call readint
mov ecx, eax
INVOKE Factorial, eax
mov edx, OFFSET msg2
call WriteString
call writeint
call crlf
call GetMSeconds
sub eax, time
mov edx, OFFSET msg3
call WriteString
call writeint
call crlf
exit
main ENDP
```

```
Factorial PROC, N: DWORD
push ebp
mov ebp, esp
```



```
C:\WINDOWS\system32\cmd.exe
Enter a number : 5
Factorial : +120
Time taken using iterative procedure (in seconds): +948
Press any key to continue . . .
```

PS: After comparing the time taken by both iterative and recursive procedures it can be clearly seen that a recursive solution takes less time as compared to an iterative procedure.

TASK#06:

```
TITLE My First Program (Test.asm)
INCLUDE Irvine32.inc
```

```
.data
var DWORD 4 dup(?)
temp DWORD 2h
max DWORD ?
msg1 BYTE "Enter 4 number : ",0
msg2 BYTE "All 4 numbers are not prime so the LargestPrime procedure will not be executed",0
msg3 BYTE "All numbers are prime ",0
msg4 BYTE "Largest Prime = ",0
```

```
CheckPrime PROTO, X: PTR DWORD
LargestPrime PROTO, Y: PTR DWORD
```

```
.code
main PROC
mov edx, offset msg1
call writestring
mov ecx, 4
mov esi, 0
L1:
call readint
mov var[esi], eax
add esi, 4
```



```
Loop L1
mov esi, offset var
INVOKE CheckPrime, esi
next:
exit
main ENDP
```

```
CheckPrime PROC, X: PTR DWORD
push ebp
mov ebp, esp
mov ecx, lengthof var
L1:
mov eax, [esi]
cmp eax, 2
je prime
mov ebx, 2
top:
mov edx, 0
mov eax, [esi]
div temp
cmp ebx, eax
jae prime
mov edx, 0
mov eax, [esi]
div ebx
inc ebx
cmp edx, 0
je S1
jmp top
prime:
add esi, 4
Loop L1
mov edx, offset msg3
call writestring
call crlf
mov esi, offset var
INVOKE LargestPrime, esi
jmp final
S1:
mov edx, offset msg2
call writestring
call crlf
final:
pop ebp
add esp, 8
ret
CheckPrime ENDP
```

LargestPrime PROC, Y: PTR DWORD

push ebp

mov ebp, esp

mov ecx, lengthof var

mov eax, [esi]

L1:

add esi, 4

mov ebx, [esi]

cmp eax, ebx

ja next

mov eax, ebx

next:

Loop L1

mov max, eax

mov edx, offset msg4

call writestring

call writeint

call crlf

pop ebp

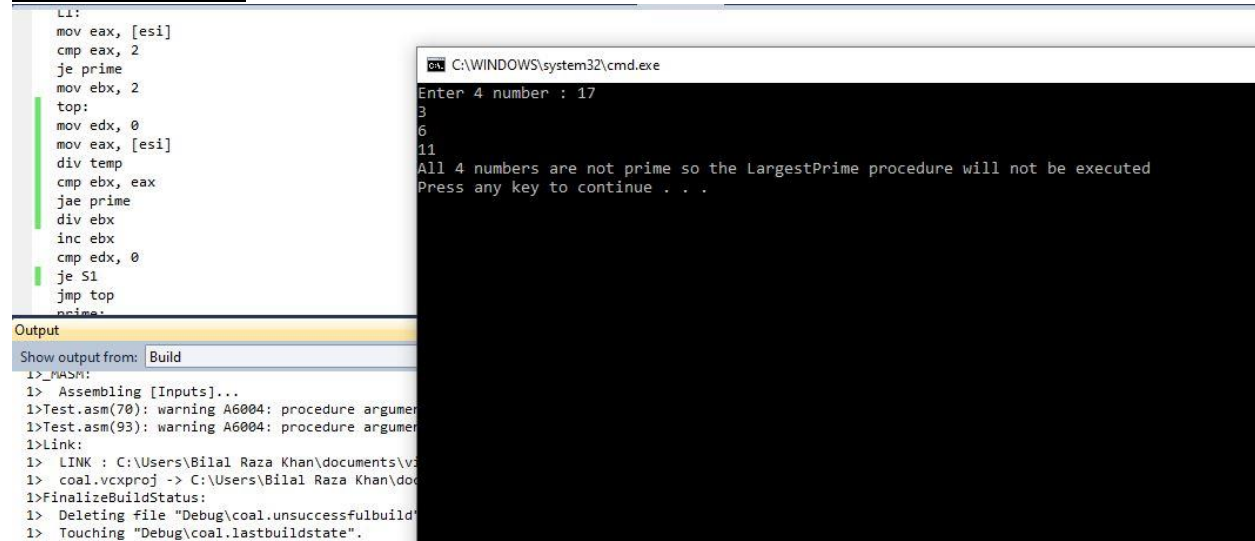
add esp, 8

ret

LargestPrime ENDP

END main

OUTPUT IN CASE 1:



The screenshot displays the assembly code for the LargestPrime procedure and its execution output. On the left, the assembly code is shown with line numbers 11 through 27. The code includes a loop L1 that iterates through an array of numbers, comparing each to the current maximum. The output window on the right shows the program's execution, where the user enters 17, and the program outputs 3, 6, and 11, followed by a message stating that all 4 numbers are not prime and the LargestPrime procedure will not be executed. The command prompt window shows the user's input and the program's output.

```
11: mov eax, [esi]
12: cmp eax, 2
13: je prime
14: mov ebx, 2
15: top:
16: mov edx, 0
17: mov eax, [esi]
18: div temp
19: cmp ebx, eax
20: jae prime
21: div ebx
22: inc ebx
23: cmp edx, 0
24: je $1
25: jmp top
26: prime:
27: 
```

Output

Show output from: Build

```
1> _NMAKE:
1> Assembling [Inputs]...
1> Test.asm(70): warning A6004: procedure argument not declared
1> Test.asm(93): warning A6004: procedure argument not declared
1> Link:
1> LINK : C:\Users\Bilal Raza Khan\documents\vc\
1> coal.vcxproj -> C:\Users\Bilal Raza Khan\doc
1> FinalizeBuildStatus:
1> Deleting file "Debug\coal.unsuccessfulbuild"
1> Touching "Debug\coal.lastbuildstate".
```

C:\WINDOWS\system32\cmd.exe

Enter 4 number : 17

3

6

11

All 4 numbers are not prime so the LargestPrime procedure will not be executed

Press any key to continue . . .

OUTPUT IN CASE 2:

```

mov ebp, esp
mov ecx, lengthof var
L1:
mov eax, [esi]
cmp eax, 2
je prime
mov ebx, 2
top:
mov edx, 0
mov eax, [esi]
div temp
cmp ebx, eax
jae prime
mov edx, 0
mov eax, [esi]
div ebx

```

Output

Show output from: Build

```

1>_NASM:
1> Assembling [Inputs]...
1>Test.asm(72): warning A6004: procedure argument o
1>Test.asm(95): warning A6004: procedure argument o
1>Link:
1> LINK : C:\Users\Bilal Raza Khan\documents\visua
1> coal.vcxproj -> C:\Users\Bilal Raza Khan\docume
1>FinalizeBuildStatus:
1> Deleting file "Debug\coal.unsuccessfulbuild".
1> Touching "Debug\coal.lastbuildstate".
1>
1>

```

C:\WINDOWS\system32\cmd.exe

```

Enter 4 number : 19
2
7
13
All numbers are prime
Largest Prime = +19
Press any key to continue . . .

```