COAL Theory Assignment 3:

# Name: Owais Ali Khan

# Section: 3-F

# Roll no: 21K-3298

Question # 01:

TITLE Question 1

INCLUDE Irvine32.inc

INCLUDE Macros.inc

.data

intArray SDWORD 30, -40, 20, 65, 80, 45

count DWORD LENGTHOF intArray

.code

main PROC

mWrite "Array values:"

mov ecx, count

mov esi, OFFSET intArray

PRINT:

mov eax, [esi]

call WriteInt

mWrite " "

add esi, TYPE intArray

loop PRINT

call Crlf

call Crlf

mWrite "j = 20 and k = 50 for 1st call"

call Crlf

push 50 ; K

push 20 ; J

push count

push OFFSET intArray

call SumArray

mWrite "Sum = "

call WriteInt

call Crlf

call Crlf

mWrite "j = 35 and k = 90 for 1st call"

call Crlf

push 90 ; K

push 35 ; J

push count

push OFFSET intArray

call SumArray

mWrite "Sum = "

call WriteInt

call Crlf

exit

main ENDP

SumArray PROC USES ebx ecx edx esi, array:PTR DWORD, array\_size:DWORD, J:DWORD, K:DWORD

mov esi, array

mov ecx, array\_size

mov eax, 0

L1:

mov ebx, [esi]

cmp ebx, J

JL NEXT

cmp ebx, K

JG NEXT

add eax, ebx

NEXT:

add esi, 4

loop L1

ret

SumArray ENDP

END main

Question # 02:

TITLE Question 2

INCLUDE Irvine32.inc

INCLUDE Macros.inc

.data

intArray DWORD 60, 4, 17, 45, 7

count DWORD 5

.code

main PROC

mWrite "Array values: "

mov ecx, count

mov esi, OFFSET intArray

PRINT:

mov eax, [esi]

call WriteInt

mWrite " "

add esi, 4

loop PRINT

call Crlf

call Crlf

push count

push OFFSET intArray

call SelectSort

mWrite "After SelectSort: "

mov ecx, count

mov esi, OFFSET intArray

PRINT2:

mov eax, [esi]

call WriteInt

mWrite " "

add esi, 4

loop PRINT2

call Crlf

exit

main ENDP

SelectSort PROC, array:PTR DWORD, array\_size:DWORD

mov esi, array

mov ecx, array\_size

mov eax, 0

L1:

push ecx

push esi

mov eax, [esi]

mov edi, array

L2:

mov ebx, [edi]

cmp eax, ebx

JAE NotSwapped

mov eax, ebx

mov esi, edi

NotSwapped:

add edi, 4

loop L2

sub edi, 4

push edi

push esi

mWrite "Before calling SWAP. [esi] = "

mov eax, [esi]

call WriteDec

call Crlf

mWrite "Before calling SWAP. [edi] = "

mov eax, [edi]

call WriteDec

call Crlf

call Crlf

call SWAP

pop esi

pop ecx

loop L1

ret

SelectSort ENDP

SWAP PROC

push ebp

mov ebp, esp

mov esi, [ebp + 8]

mov edi, [ebp + 12]

mov eax, [esi]

mov ebx, [edi]

xchg eax, ebx

mov [esi], eax

mov [edi], ebx

mov [ebp + 8], esi

mov [ebp + 12], edi

mov esp, ebp

pop ebp

ret 8

SWAP ENDP

END main

Question # 03:

TITLE Question 3

INCLUDE Irvine32.inc

INCLUDE Macros.inc

.data

intArray BYTE 60, 4, 17, 45, 7, 69, 21, 33, 96, 81

count DWORD 10

.code

main PROC

mWrite "Enter 10 single-digit integer values:"

call Crlf

mov esi, OFFSET intArray

mov ecx, count

mov ebx, 0

INPUT:

mWrite "Enter single-digit integer "

mov eax, ebx

call WriteDec

mWrite ": "

call ReadDec

cmp eax, 10

jl GOOD\_INPUT

mWrite "Invalid number of digits. Try Again."

call Crlf

jmp INPUT

GOOD\_INPUT:

mov [esi], al

add esi, 1

add ebx, 1

loop INPUT

mWrite "Array values: "

mov eax, 0

mov ecx, count

mov esi, OFFSET intArray

PRINT:

mov al, [esi]

call WriteInt

mWrite " "

add esi, 1

loop PRINT

call Crlf

call Crlf

mov ebx, count

mov esi, OFFSET intArray

call Bubble

mWrite "After Bubble: "

mov ecx, count

mov esi, OFFSET intArray

PRINT2:

mov al, [esi]

call WriteInt

mWrite " "

add esi, 1

loop PRINT2

call Crlf

exit

main ENDP

Bubble PROC

mov ecx, ebx

mov eax, 0

mov ebx, 0

L1:

push ecx

push esi

L2:

cmp ecx, 1

JBE END\_LOOP2

mov edi, esi

add edi, 1

mov al, [esi]

mov bl, [edi]

cmp al, bl

JBE NotSwapped

push edi

push esi

call SWAP

pop esi

pop edi

NotSwapped:

add esi, 1

add edi, 1

loop L2

END\_LOOP2:

pop esi

pop ecx

loop L1

ret

Bubble ENDP

SWAP PROC

push ebp

mov ebp, esp

mov esi, [ebp + 8]

mov edi, [ebp + 12]

mov al, [esi]

mov bl, [edi]

xchg al, bl

mov [esi], al

mov [edi], bl

mov [ebp + 8], esi

mov [ebp + 12], edi

mov esp, ebp

pop ebp

ret

SWAP ENDP

END main

Question # 04:

TITLE Question 4

INCLUDE Irvine32.inc

INCLUDE Macros.inc

.data

N DWORD 5

.code

main PROC

mWrite "Enter a positive integer: "

call ReadDec

mov N, eax

call Crlf

mWrite "N = "

mov eax, N

call WriteDec

call Crlf

mov ecx, N

call FACTORIAL

mWrite "Factorial of N = "

call WriteDec

call Crlf

exit

main ENDP

FACTORIAL PROC ; receives N in ecx and returns N! in eax

mov eax, 1

mov edx, 0

cmp ecx, 0

JBE END\_LOOP

L1:

mov edx, 0

mul ecx

loop L1

END\_LOOP:

ret

FACTORIAL ENDP

END main

Question # 05:

TITLE Question 5

INCLUDE Irvine32.inc

INCLUDE Macros.inc

.data

char DWORD " "

countBits DWORD 0

.code

main PROC

mov eax, 0

mWrite "TYPE A CHARACTER: "

call ReadChar

call Crlf

and eax, 000000FFh

mov char, eax

mWrite "THE ASCII CODE OF A IN BINARY IS "

call WriteBin

call Crlf

L1:

shr eax, 1

adc countBits, 0

or eax, eax

jnz L1

mWrite "THE NUMBER OF 1 BITS IS "

mov eax, countBits

call WriteDec

call Crlf

exit

main ENDP

END main

Question # 06:

TITLE Question 6

INCLUDE Irvine32.inc

INCLUDE Macros.inc

.data

arrayA SDWORD 0, 1, 2, 3, 4, 5, 6, 7, 8, 9

arrayB SDWORD 1, 2, 3, 4, 5, 6, 7, 8, 9, 9

arraySize DWORD 10

count DWORD 0

.code

main PROC

CountMatches PROTO, arrA:PTR DWORD, arrB:PTR DWORD, arrSize:DWORD

INVOKE CountMatches, ADDR arrayA, ADDR arrayB, arraySize

mov eax, count

mWrite "The total count of same elements is: "

call WriteDec

exit

main ENDP

CountMatches PROC USES esi edi, arrA:PTR DWORD, arrB:PTR DWORD, arrSize:DWORD

mov esi, arrA

mov edi, arrB

mov ecx, arrSize

L1:

mov eax, [esi]

cmp eax, [edi]

jne No\_increment

inc count

No\_increment:

add esi, 4

add edi, 4

loop L1

ret

CountMatches ENDP

END main

Question # 07:

TITLE Question 7

INCLUDE Irvine32.inc

INCLUDE Macros.inc

.data

;first DWORD 0FFFFFFFFh, 0FFFFFFFFh, 0FFFFFFFFh, 0000FFFFh

;second DWORD 0EEEEEEEEh, 0EEEEEEEEh, 0EEEEEEEEh, 0000EEEEh

first DWORD 00000000h, 00000000h, 00000000h, 00009999h

second DWORD 00000000h, 00000000h, 00000000h, 00008888h

len DWORD 4 ; 32 bits = 4, 4x4 = 16

diff DWORD 5 DUP(0)

.code

main PROC

push len

push OFFSET second

push OFFSET first

call Extended\_Sub

mov esi, OFFSET diff

mov ecx, len

L2:

mov eax, [esi]

;call WriteDec

call WriteHex

mWrite " "

add esi, 4

loop L2

call Crlf

exit

main ENDP

Extended\_Sub PROC, num1:PTR DWORD, num2:PTR DWORD, count:DWORD ; Subtracts num2 from num1

mov esi, num1

mov edi, num2

mov ebx, OFFSET diff

mov ecx, count

clc

L1:

mov eax, [esi]

sbb eax, [edi]

pushfd

mov [ebx], eax

add esi, 4

add edi, 4

add ebx, 4

popfd

loop L1

sbb DWORD PTR [ebx], 0

ret

Extended\_Sub ENDP

END main

Question # 08:

TITLE Question 8

INCLUDE Irvine32.inc

INCLUDE Macros.inc

.data

;first DWORD 0FFFFFFFFh, 0FFFFFFFFh, 0FFFFFFFFh, 0000FFFFh

;second DWORD 0EEEEEEEEh, 0EEEEEEEEh, 0EEEEEEEEh, 0000EEEEh

first DWORD 00000000h, 00000000h, 00000000h, 00001111h

second DWORD 00000000h, 00000000h, 00000000h, 00008888h

len DWORD 4 ; 32 bits = 4, 4x4 = 16

sum DWORD 5 DUP(0)

.code

main PROC

push len

push OFFSET second

push OFFSET first

call Extended\_Add

mov esi, OFFSET sum

mov ecx, len

L2:

mov eax, [esi]

call WriteHex

mWrite " "

add esi, 4

loop L2

call Crlf

exit

main ENDP

Extended\_Add PROC, num1:PTR DWORD, num2:PTR DWORD, count:DWORD ; Subtracts num2 from num1

mov esi, num1

mov edi, num2

mov ebx, OFFSET sum

mov ecx, count

clc

L1:

mov eax, [esi]

adc eax, [edi]

pushfd

mov [ebx], eax

add esi, 4

add edi, 4

add ebx, 4

popfd

loop L1

adc DWORD PTR [ebx], 0

ret

Extended\_Add ENDP

END main

Question # 09:

TITLE Question 9

INCLUDE Irvine32.inc

INCLUDE Macros.inc

.data

.code

main PROC

push 20

push 5

call GCD

mov eax, ecx

mWrite "The GCD of 5 and 20 is: "

call WriteDec

call Crlf

push 18

push 24

call GCD

mov eax, ecx

mWrite "The GCD of 24 and 18 is: "

call WriteDec

call Crlf

push 226

push 432

call GCD

mov eax, ecx

mWrite "The GCD of 432 and 226 is: "

call WriteDec

call Crlf

exit

main ENDP

GCD PROC ; Stores values of A and B in eax and ebx respectively. Returns answer in ecx

push ebp

mov ebp, esp

mov eax, [ebp + 8]

mov ebx, [ebp + 12]

CASE\_A:

cmp eax, 0

JNE CASE\_B

mov ecx, ebx

mov esp, ebp

pop ebp

ret 8

CASE\_B:

cmp ebx, 0

JNE CASE\_A\_EQUALS\_B

mov ecx, eax

mov esp, ebp

pop ebp

ret 8

CASE\_A\_EQUALS\_B:

cmp eax, ebx

JNE CASE\_A\_GREATER\_THAN\_B

mov ecx, eax

mov esp, ebp

pop ebp

ret 8

CASE\_A\_GREATER\_THAN\_B:

cmp eax, ebx

JBE CASE\_A\_LESS\_THAN\_B

sub eax, ebx

push ebx

push eax

call GCD

jmp ENDD

CASE\_A\_LESS\_THAN\_B:

sub ebx, eax

push ebx

push eax

call GCD

ENDD:

mov esp, ebp

pop ebp

ret 8

GCD ENDP

END main

Question # 10:

TITLE Question 10

INCLUDE Irvine32.inc

INCLUDE Macros.inc

.data

arr1 SDWORD 1, 9, 5, 23, 68, 6

arr2 SDWORD 0, 2, 3, 4, 5, 3

count DWORD 6

difference DWORD 5

.code

main PROC

CountNearMatches PROTO, array1:PTR SDWORD, array2:PTR SDWORD, arrSize:DWORD, diff:DWORD

INVOKE CountNearMatches, ADDR arr1, ADDR arr2, count, difference

mWrite "The count of near matches in given arrays is: "

call WriteDec

call Crlf

exit

main ENDP

CountNearMatches PROC USES esi edi ebx ecx, array1:PTR SDWORD, array2:PTR SDWORD, arrSize:DWORD, diff:DWORD

mov eax, 0

mov esi, array1

mov edi, array2

mov ecx, arrSize

L1:

mov ebx, [esi]

sub ebx, [edi]

cmp ebx, diff

JG NotIncremented

inc eax

NotIncremented:

add esi, 4

add edi, 4

loop L1

ret

CountNearMatches ENDP

END main