**COAL ASSIGNMNET # 3**

**BY Ali Zain 21K-4653**

**SECTION : K**

**QUESTION #1:**

INCLUDE Irvine32.inc

N=10

.data

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

j DWORD ?

k DWORD ?

.code

main PROC

call Clrscr

mov j, 20

mov k, 50

mov ESI, OFFSET array

mov ECX, lengthof array

call SUM

call WriteInt

call crlf

mov j, 35

mov k, 90

mov ESI, OFFSET array

mov ECX, lengthof array

call SUM

call WriteInt

call crlf

exit

main ENDP

SUM PROC USES esi ecx

mov eax, 0

l1:

mov ebx, [esi]

cmp ebx, j

jge true1

jmp next

true1:

cmp ebx, k

jle true2

jmp next

true2:

add eax, ebx

next:

add esi, 4

loop l1

ret

SUM ENDP

END main

**OUTPUT:**

Text

Description automatically generated

**QUESTION NO.2:**

INCLUDE IRVINE32.inc

.data

array dword 60,4,17,45,7

msg1 BYTE "Before: ",0

msg2 BYTE "After Sort: ",0

msg3 BYTE " ",0

count dword ?

x dword ?

.code

MAIN PROC

xor edx,edx

mov edx,OFFSET msg1

call writestring

call display

call crlf

call crlf

push OFFSET array

push LENGTHOF array

call selectionSort

mov edx,OFFSET msg2

call writestring

call display

call crlf

exit

MAIN ENDP

selectionSort PROC

ENTER 0,0

xor ebx,ebx

xor eax,eax

xor edx,edx

mov ecx,[ebp+8]

sub ecx,1

L1:

mov edx,eax

mov ebx,eax

add ebx,4

mov count,ecx

mov ecx,[ebp+8]

sub ecx,x

INNER:

mov esi,[ebp+12]

mov edi,[esi+edx]

cmp [esi+ebx],edi

jb JUMP

jmp OKAI

JUMP:

mov edx,ebx

OKAI:

add ebx,4

loop INNER

mov esi,[ebp+12]

call swap

add x,1

add eax,4

mov ecx,count

loop L1

LEAVE

ret 8

selectionSort ENDP

swap PROC

mov edi,[esi+eax]

xchg edi,[esi+edx]

mov [esi+eax],edi

ret

swap ENDP

display PROC

mov ecx,lengthof array

mov eax,0

mov esi,offset array

l1:

mov ebx,[esi]

mov eax,ebx

call writedec

mov edx,OFFSET msg3

call writestring

add esi,4

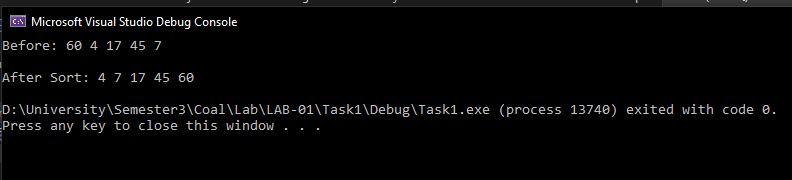
loop l1

ret

display ENDP

END MAIN

**OUTPUT:**

****

**QUESTION NO.3:**

INCLUDE Irvine32.inc

bubbleSort PROTO,

ptrArr:DWORD, len:DWORD

.data

msg1 byte "Enter 10 elements in the array: ",0

msg2 byte "Array in sorted order: ",0

arr DWORD 10 dup(?)

.code

main PROC

mov ecx, lengthof arr

mov esi, 0

mov edx, offset msg1

call writestring

call crlf

INPUT:

mov eax, 0

call readint

mov arr[esi\*type arr], eax

inc esi

loop INPUT

INVOKE bubbleSort, ADDR arr, lengthof arr

call crlf

mov edx, offset msg2

call writeString

call crlf

mov esi, 0

mov ecx, lengthof arr

DISPLAY:

mov eax, arr[esi\*type arr]

call writedec

call crlf

inc esi

loop DISPLAY

exit

main ENDP

bubbleSort PROC, ptrArr:DWORD, l:DWORD

mov esi, ptrArr

sub l, 1

mov ecx, l

BAHAR:

push ecx

mov ecx, l

mov edi, ptrArr

ANDAR:

mov eax, [edi]

mov ebx, [edi+4]

cmp eax, ebx

jle cont

xchg eax, ebx

mov [edi], eax

mov [edi+4], ebx

cont:

add edi,4

loop ANDAR

add esi, 4

pop ecx

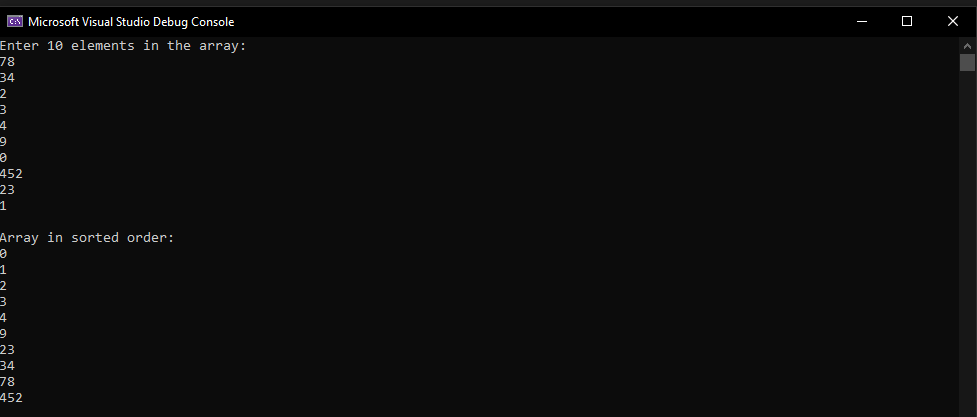
loop BAHAR

ret 8

bubbleSort ENDP

END main

**OUTPUT:**



**QUESTION NO.4:**

INCLUDE Irvine32.inc

.data

V1 dword ?

msg byte "factorial is: ",0

.code

main PROC

call readint

mov ecx,eax

mov eax, 1

call factorial

L1:

mov edx,OFFSET msg

call writestring

call WriteDec

exit

main ENDP

factorial PROC

cmp ecx, 0

jz L2

mul ecx

dec ecx

call factorial

L2:

ret

factorial ENDP

END main

**Text

Description automatically generated**

**QUESTION NO.5:**

INCLUDE IRVINE32.inc

.data

char BYTE ?

msg1 BYTE "Type a character:",0

msg2 BYTE "The Ascii is:",0

msg3 BYTE "The number of 1 bits is:",0

count dword 0

times dword 8

.code

main PROC

xor eax,eax

xor ebx,ebx

xor edx,edx

mov edx, OFFSET msg1

call writestring

call readchar

mov char,al

call writechar

mov ah,0

mov edx, OFFSET msg2

call crlf

call crlf

call writestring

call writebinb

mov bl,al

mov ecx,times

L1:

shr bl,1

jc milgaya

jmp nhimila

milgaya:

inc count

nhimila:

loop L1

call crlf

call crlf

mov edx, OFFSET msg3

call writestring

mov eax,count

call writedec

call crlf

EXIT

main ENDP

END main

**OUTPUT:**

**Text

Description automatically generated**

**Question no.6:**

INCLUDE IRVINE32.inc

COUNTMATCHING PROTO, array1: SDWORD, array2:SDWORD, len: DWORD

.data

array1 sdword 56,-69,32,-4,-35,11

array2 sdword 10,-69,-33,5,-35,10

msg2 BYTE "The MATCHED VALUES COUNT: ",0

.code

main PROC

xor eax,eax

xor edx,edx

INVOKE COUNTMATCHING, addr array1, addr array2, LENGTHOF array1

mov edx,OFFSET msg2

call writestring

call writedec

call crlf

exit

main ENDP

COUNTMATCHING PROC, arr1:SDWORD, arr2:SDWORD, l: DWORD

xor eax,eax

mov esi,arr1

mov edi,arr2

mov ecx,l

L1:

mov ebx,[esi]

cmp ebx,[edi]

je milgaya

jmp nhimila

milgaya:

inc eax

nhimila:

add esi,TYPE DWORD

add edi,TYPE DWORD

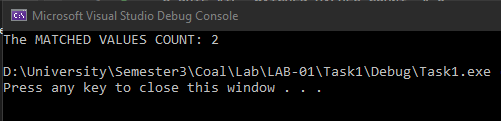
loop L1

ret 12

COUNTMATCHING ENDP

end main

**OUTPUT:**

****

**QUESTION NO.7:**

INCLUDE IRVINE32.inc

.data

VALUE1 qword 08010000000294502h

VALUE2 qword 0A2B2A40674901894h

VALUE3 qword 0C003801080090302h

VALUE4 qword 0124981934B2B3089h

ans dword 3 DUP(0)

.code

MAIN PROC

mov eax, DWORD PTR VALUE1

PUSH eax

mov eax, DWORD PTR VALUE1+4

PUSH eax

mov eax, DWORD PTR VALUE2

PUSH eax

mov eax, DWORD PTR VALUE2+4

PUSH eax

call ExtendedSub

call crlf

mov eax, DWORD PTR VALUE3

PUSH eax

mov eax, DWORD PTR VALUE3+4

PUSH eax

mov eax, DWORD PTR VALUE4

PUSH eax

mov eax, DWORD PTR VALUE4+4

PUSH eax

call ExtendedSub

call crlf

EXIT

MAIN ENDP

ExtendedSub PROC

PUSH ebp

mov ebp,esp

mov edx,0

mov ebx, [ebp+20]

sbb ebx, [ebp+12]

mov [ans+8],ebx

mov ebx, [ebp+16]

sbb ebx, [ebp+8]

mov [ans+4],ebx

sbb ans,0

mov esi, OFFSET ans

mov ecx, LENGTHOF ans

mov ebx, TYPE ans

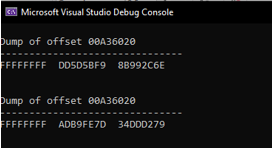
call dumpMem

POP ebp

ret 16

ExtendedSub ENDP

END MAIN

****

**QUESTION NO.8:**

INCLUDE IRVINE32.inc

.data

VALUE1 qword 08010000000294502h

VALUE2 qword 0A2B2A40674901894h

VALUE3 qword 0C003801080090302h

VALUE4 qword 0124981934B2B3089h

ans dword 3 DUP(0)

.code

MAIN PROC

mov eax, DWORD PTR VALUE1

PUSH eax

mov eax, DWORD PTR VALUE1+4

PUSH eax

mov eax, DWORD PTR VALUE2

PUSH eax

mov eax, DWORD PTR VALUE2+4

PUSH eax

call ExtendedAdd

call crlf

mov eax, DWORD PTR VALUE3

PUSH eax

mov eax, DWORD PTR VALUE3+4

PUSH eax

mov eax, DWORD PTR VALUE4

PUSH eax

mov eax, DWORD PTR VALUE4+4

PUSH eax

call ExtendedAdd

call crlf

EXIT

MAIN ENDP

ExtendedAdd PROC

PUSH ebp

mov ebp,esp

mov edx,0

mov ebx, [ebp+20]

adc ebx, [ebp+12]

mov [ans+8],ebx

mov ebx, [ebp+16]

adc ebx, [ebp+8]

mov [ans+4],ebx

adc ans,0

mov esi, OFFSET ans

mov ecx, LENGTHOF ans

mov ebx, TYPE ans

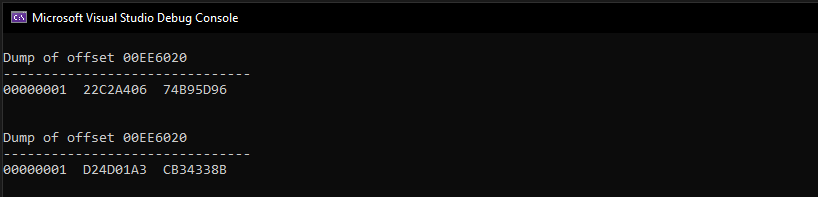
call dumpMem

POP ebp

ret 16

ExtendedAdd ENDP

END MAIN



**QUESTION NO.9:**

INCLUDE IRVINE32.inc

.data

a dword ?

b dword ?

msg1 BYTE "Enter NUM 1: ",0

msg2 BYTE "Enter NUM 2: ",0

msg3 BYTE "The GCD for NUM1 and NUM2 is: ",0

.code

main PROC

xor eax,eax

xor ecx,ecx

mov edx,OFFSET msg1

call writestring

call readdec

mov a,eax

call crlf

mov edx,OFFSET msg2

call writestring

call readdec

mov b,eax

call crlf

push a

push b

call GCD

mov edx,OFFSET msg3

call writestring

call writedec

call crlf

exit

main ENDP

GCD PROC

xor edx,edx

ENTER 0,0

cmp [ebp+8],edx

jz L1

cmp [ebp+12],edx

jz L2

mov ebx,[ebp+8]

cmp [ebp+12],ebx

je L1

cmp [ebp+12],ebx

ja FIRST

mov ebx,[ebp+12]

sub [ebp+8],ebx

push [ebp+12]

push [ebp+8]

call GCD

jmp ENDD

FIRST:

mov ebx,[ebp+12]

sub ebx,[ebp+8]

push ebx

push [ebp+8]

call GCD

jmp ENDD

L2:

mov eax,[ebp+8]

jmp ENDD

L1:

mov eax,[ebp+12]

ENDD:

LEAVE

ret 8

GCD ENDP

end main



**QUESTION NO.10:**

INCLUDE IRVINE32.inc

cnm PROTO, array1: SDWORD, array2:SDWORD, len: DWORD, diff: SDWORD

.data

array1 sdword 10,20,30,40,50

array2 sdword 5,16,28,31,49

msg1 BYTE "Enter the common difference:",0

msg2 BYTE "The count value is:",0

diff sdword ?

.code

main PROC

xor eax,eax

xor edx,edx

mov edx,OFFSET msg1

call writestring

call crlf

call readint

call crlf

mov diff,eax

INVOKE cnm, addr array1, addr array2, LENGTHOF array1, diff

mov edx,OFFSET msg2

call writestring

call writeDec

exit

main ENDP

cnm PROC, arr1:SDWORD, arr2:SDWORD, l: DWORD, dif: SDWORD

xor eax,eax

mov esi,arr1

mov edi,arr2

mov ecx,l

L1:

mov ebx,[esi]

sub ebx,[edi]

cmp ebx,dif

jle jump

jmp done

jump:

inc eax

done:

add esi, 4

add edi, 4

loop L1

ret

cnm ENDP

end main

