**COAL THEORY ASSIGNMENT 2**

**Q1)**

Include Irvine32.inc

.data

str1 BYTE "Enter k: ",0

str2 BYTE "Enter j: ",0

str3 BYTE "SUM: ",0

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

k DWORD 0

j DWORD 0

.code

main proc

mov eax,0

mov edx,OFFSET str1

call writestring

call readdec

push eax

mov eax,0

mov edx,OFFSET str2

call writestring

call readdec

push eax

push lengthof array

call sum

mov edx,OFFSET str3

call writeint

exit

main endp

sum proc

push ebp

mov ebp,esp

mov eax,0

mov esi,0

mov ecx,[ebp+8]

L1:

mov edx,[ebp+12]

cmp [array+esi],edx

JL back

mov edx,[ebp+16]

cmp [array+esi],edx

JG back

add eax,[array+esi]

back:

add esi,4

loop L1

pop ebp

ret 3

sum endp

end Main

**Q2)**

selectionSort PROC

LOCAL largest:DWORD,i:DWORD,j:DWORD

mov ecx,LENGTHOF arr

mov largest,0

dec ecx

mov i,ecx

mov j,ecx

outerLoop:

mov ebx,i

mov largest,ebx

push ecx

mov edx,i

mov j,edx

innerLoop:

dec j

mov edx,j

mov eax,[arr + edx \* 4]

mov edx,largest

mov ebx,[arr + edx \* 4]

cmp eax,ebx

jg markNewMax

jmp continueLoop

markNewMax:

mov edx,j

mov largest,edx

continueLoop:

loop innerLoop

push i

push largest

call SWAP

pop ecx

dec i

loop outerLoop

ret

selectionSort ENDP

**Q3)**

Include Irvine32.inc

.data

array dword 10, 9, 8, 7, 6, 5, 4, 3, 2, 1, 0

.code

main PROC

call BubbleSort

exit

main endp

BubbleSort PROC

mov eax,0

mov ebx,0

mov ecx,0

mov edx,0

mov esi,0

mov edi,0

mov ecx,11

L1:

mov edx,ecx

mov ecx,10

L2:

mov eax,0

mov ebx,0

mov eax,array[edi\*type array]

inc edi

mov ebx,array[edi\*type array]

dec edi

cmp eax,ebx

JG call\_swap

back\_after\_swap:

inc edi

loop L2

mov edi,0

inc esi

mov ecx,edx

loop L1

call print\_array

ret

call\_swap:

call swap

jmp back\_after\_swap

BubbleSort endp

swap PROC

xchg eax,ebx

mov array[edi\*type array],eax

inc edi

mov array[edi\*type array],ebx

dec edi

mov eax,0

mov ebx,0

ret

swap endp

print\_array PROC

mov eax,0

mov esi,0

mov ecx,11

mov edx,0

L3:

mov eax,array[edx\*type array]

call writedec

call crlf

inc edx

loop L3

ret

print\_array endp

end main

**Q4)**

INCLUDE Irvine32.inc

.data

msg BYTE "Factorial : ",0

msg2 BYTE "Enter the number for factorial: ",0

.code

main PROC

mov edx,offset msg2

call writestring

call readint

push eax

call Factorial

mov edx, offset msg

call WriteString

call WriteDec

call Crlf

exit

main ENDP

Factorial PROC

push ebp

mov ebp,esp

;mov eax,[ebp+4]

mov eax,[ebp+8]

cmp eax,0

ja L1

mov eax,1

jmp L2

L1:

;inc eax

dec eax

push eax

call Factorial

ReturnFact:

mov ebx,[ebp+8]

mul ebx

L2:

pop ebp

ret 4

Factorial ENDP

END main

**Q5)**

Include Irvine32.inc

.data

str1 BYTE "Enter a character to check its ascii: ",0

str2 BYTE "ASCI: ",0

str3 BYTE "Number of 1's: ",0

.code

main proc

mov eax,0

mov edx,OFFSET str1

call writestring

call readchar

call writechar

call crlf

mov bl,00000000b

and ah,bl

mov edx,OFFSET str2

call writestring

call writebin

call crlf

mov ecx,8

mov ebx,0

L1:

SHR al,1

JNC back

add ebx,1

back:

loop L1

mov edx,OFFSET str3

call writestring

mov eax,ebx

call writedec

exit

main endp

end Main

**Q6)**

Include Irvine32.inc

.data

array1 sdword 1,2,3,4,5

array2 sdword 9,8,3,6,5

str\_result byte 'Total Matches Found : ',0

;CountMatches PROTO

.code

main PROC

mov eax,0

mov ebx,0

mov edx,0

mov ecx,0

mov esi,0

mov edi,0

mov esi,offset array1

mov edi,offset array2

mov edx,lengthof array1

push esi

push edi

push edx

mov esi,0

mov edi,0

mov ecx,0

call CountMatches

mov edx,offset str\_result

call writestring

call writedec

exit

main endp

CountMatches PROC

Enter 0,0

mov eax,0

mov esi,[ebp+16]

mov edi,[ebp+12]

mov ecx,[ebp+8]

L1:

mov ebx,[esi]

mov edx,[edi]

cmp ebx,edx

JE add\_count

back:

add esi,4

add edi,4

loop L1

Leave

ret 12

add\_count:

inc eax

jmp back

CountMatches ENDP

end main

**Q7)**

Include Irvine32.inc

.data

arr1 DWORD 100,250000,36000

arr2 DWORD 1000,2000,300

result DWORD 3 DUP(?)

.code

main proc

call extended\_sub

exit

main endp

extended\_sub proc

mov ecx,3

mov esi,0

L1:

mov eax,[arr1+esi]

add eax,[arr2+esi]

call writedec

call crlf

add esi,4

loop L1

ret

extended\_sub endp

end Main

**Q8)**

Include Irvine32.inc

.data

array1 DWORD 36000,250000,100

array2 DWORD 2000,100,300

.code

main proc

call extSub

exit

main endp

extSub proc

mov ecx,3

mov esi,0

L1:

mov eax,[array1+esi]

add eax,[array2+esi]

call writedec

call crlf

add esi,4

loop L1

ret

extSub endp

end Main

**Q9)**

Include Irvine32.inc

GCD PROTO , a:dword , b:dword

.code

main PROC

mov eax,0

invoke GCD , 5d , 20d

mov eax,edx

call writedec

call crlf

invoke GCD , 24d , 18d

mov eax,edx

call writedec

call crlf

invoke GCD , 432d , 226d

mov eax,edx

call writedec

call crlf

exit

main endp

GCD PROC , a:dword , b:dword

mov eax,a

mov ebx,b

cmp a,0

JE bb

cmp b,0

JE aa

cmp eax,ebx

JE aa

cmp eax,ebx

JG ab

mov eax,a

mov ebx,b

sub ebx,eax

mov b,ebx

invoke GCD , a , b

ret

ab:

mov eax,a

mov ebx,b

sub eax,ebx

mov a,eax

invoke GCD , a , b

ret

bb:

mov edx,b

ret

aa:

mov edx,a

ret

GCD ENDP

end main  
**Q10)**

INCLUDE Irvine32.inc

CountNearMatches PROTO, ptrArray1:PTR SDWORD, ptrArray2:PTR SDWORD, szArray:DWORD, diff:DWORD

.data

array1a SDWORD 1,2,3,4,5,6,7,8

array1b SDWORD 5,7,8,7,5,75,43,4

array2a SDWORD 5,44,7,65,8,7,23,4

array2b SDWORD 8,2,5,46,55,65,35,4

cnt DWORD ?,0

dif1 DWORD 11

dif2 DWORD 0

.code

main PROC

INVOKE CountNearMatches, ADDR array1a, ADDR array1b, LENGTHOF array1a, dif1

call WriteInt

call Crlf

INVOKE CountNearMatches, ADDR array2a, ADDR array2b, LENGTHOF array2a, dif2

call WriteInt

call Crlf

exit

main ENDP

CountNearMatches PROC USES edx ebx edi esi ecx, ptrArray1:PTR SDWORD, ptrArray2:PTR SDWORD, szArray:DWORD, diff:DWORD

mov esi,ptrArray1

mov edi,ptrArray2

mov ecx,szArray

; Loop

L1:

; compare sets

mov ebx,0

mov ebx,[esi]

mov edx,0

mov edx,[edi]

cmp ebx,edx

JNG elsepart

mov eax,ebx

sub eax,edx

JMP Lableee

elsepart:

mov eax,edx

sub eax,ebx

Lableee:

cmp eax,diff

JG lableee2

inc cnt

lableee2:

add esi, SIZEOF SDWORD

add edi, SIZEOF SDWORD

loop L1

; increment count

mov eax,0

mov eax,cnt

mov cnt,0

ret

CountNearMatches ENDP

END main