COAL A3 23K-0842, KINZA

Date: M T W T F S S

Q1)
```
.data
dividend  DWORD  0D4A4h
divisor   DWORD  0Ah
quotient  DWORD ?
.code
main PROC
mov eax,0
mov ecx,0
mov eax, dividud
mov ecx, divisor
call DivIntegers
mov eax,quotient
call WriteDec
call Crlf
exit
main ENDP
```

```
DivIntegers  PROC
cmp eax ,5h
jle L2
xor edx ,edx
div ecx
mov quotient , eax
call DivIntegers
L2:
ret
DivIntegers ENDP
END  main
```

Q2.)
```
.data
  arr BYTE 10,20,30,40,50
msg BYTE "Enter a number you wants to find in array:",0
tosearch  BYTE ?
flagfound  BYTE 0
foundmsg BYTE " Value found at Index:",0
notfoundmsg BYTE "Value is not found",0
searchValue PROTO ptrArray:PTR BYTE, _size :DWORD ,value :BYTE
.code
  main PROC
mov eax ,0
mov edx, OFFSET msg
call WriteString
call ReadInt
mov tosearch ,al
mov esi, OFFSET arr
mov ecx, LENGTHOF arr
INVOKE searchValue, esi,ecx,al
cmp flagfound,1
je valueFound
mov edx, OFFSET notfoundmsg
call WriteString
jmp _exit
valueFound:
mov edx ,OFFSET foundmsg
call WriteString
call WriteDec
_exit:
exit
main ENDP
```

```
searchValue PROC ptrArray: PTR BYTE,
_size :DWORD, value :BYTE
 mov esi, PtrArray
 mov ecx, size
 mov al ,value
 cmp ecx,0
 je notfound
 mov bl, [esi]
 cmp bl, al
 je found
 inc esi
 dec ecx
 INVOKE searchValue ,esi,ecx ,al
 ret
found:
 sub esi ,OFFSET arr
 mov eax ,esi
 mov flagfound,1
 ret
 notfound:
 mov flagfound,0
 ret
searchValue ENDP
END main.
```

Q3) .data

```
string1 BYTE "This is the source string",0
string 2 BYTE 20 DUP (?)
msg1 BYTE "Copied String after removing
        duplicates ",0
.code
  main PROC
   mov esi, OFFSET string1
   mov edi, OFFSET string2
```

Date:
M T W T F S S

```
mov ecx,0
loop1:
 mov al,[esi]
 cmp al,0
 je done
  mov ebx,OFFSET string2
  mov edx,ecx
 check_duplicates:
  cmp al,[ebx]
  je found_duplicate
  inc ebx
  dec edx
  jnz check_duplicates
  mov [edi+ecx],al
   inc ecx
  found_duplicate:
  inc esi
  jmp loop1
  done:
   mov BYTE PTR [edi+ecx],0
   mov edx,OFFSET msg1
   call WriteString
   mov edx,OFFSET string2
   call WriteString
   call Crlf
  exit
   main ENDP
   END main
```

Q4) .data

```
string BYTE "Advanced Programming in
UNIX Environment",0
msg1 BYTE "Vowels Count: ",0
a_or_A DWORD 0
e_or_E DWORD 0
i_or_I DWORD 0
o_or_O DWORD 0
u_or_U DWORD 0
.code
  main PRO
  mov esi,OFFSET string
  mov ecx,SIZEOF string
loop1:
    cmp ecx,0
    je done
    mov al,[esi]
    cmp al,0
    je done
    cmp al,'a'
    je vowel_found_a
    cmp al,'e'
    je vowel_found_e
    cmp al,'i'
    je vowel_found_i
    cmp al,'o'
    je vowel_found_o
    cmp al,'u'
    je vowel_found_u
```

Date: M T W T F S S

```
cmp al 'A'
je vowel_found_a
cmp al, 'E'
je vowel_found_e
cmp al, 'I'
je vowel_found_i
cmp al, 'O'
je vowel_found_O
cmp al, 'U'
je vowel_found_u
inc esi
dec ecx
jmp loop1

vowel_found_a:
inc DWORD PTR [a_or_A]
inc esi
dec ecx
jmp loop1
vowel_found_e:
inc DWORD PTR [e_or_E]
inc esi
dec ecx
jmp loop1
vowel_found_i:
inc DWORD PTR [i_or_I]
inc esi
dec ecx
jmp loop1
vowel_found_o:
inc DWORD PTR [o_or_O]
inc esi
dec ecx
jmp loop1
vowel_found_u:
inc DWORD PTR [u_or_U]
inc esi
dec ecx
jmp loop1
exit
main ENDP
END main
```

Q5) .code

```
DifferentInputs PROC var1 : DWORD, var2 : DWORD,
var3 : DWORD
 mov eax, var1
 mov ebx, var2
 cmp eax, ebx
 je not_different
 cmp ebx, var3
 je not_different
 cmp eax, var3
 je not_different
 mov eax, 1
 ret
 not_different:
 mov eax, 0
 ret
DifferentInputs ENDP
```

Date: M T W T F S S

Qc) .data

```
original BYTE "original String:",0
changed BYTE "Changed String:",0
String BYTE "###ABC",0
.code
strtrim PROTO, string1: PTR BYTE ,
        removeChar : BYTE
main PROC
lea edx ,original
 call WriteString
 lea edx , string
 call WriteString
 call crlf

INVOKE strtrim , ADDR string ,'#'
 lea edx ,changed
 ~~call WriteString~~
 lea edx ,string
 call WriteString
 call crlf
exit
main ENDP
```

```
strtrim PROC uses esi edi, string1 :
 PTR BYTE, removeChar : BYTE
 mov esi, string1
 mov edi, string1
 mov al, remove Char
 trim_loop :
 mov dl, [esi]
 cmp dl, 0
 je end_trim
 cmp dl ,al
 jne copy-char
 inc esi
 jmp trim_loop
 copy-char :
 mov [edi] , dl
 inc esi
 inc edi
 jmp trim-loop
 end_trim :
 mov BYTE PTR [edi] ,0
 ret
 strtrim ENDP
 END main
```