23K-0019

Date: ___/___/___

Q1:

```
.data
        dividend    DWORD    0D4Ah
        divisor     DWORD    019h
.code
main    PROC
        mov  eax, dividend
        call RecursiveDiv
        exit
main    ENDP
RecursiveDiv  PROC
        cmp  eax, 5
        jbe  Done
        xor  edx, edx
        mov  ebx, divisor
        div  edx
        call RecursiveDiv
        Done:
            ret
        RecursiveDive ENDP
END  main.
```

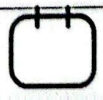
Q2:

```
.data
    msg1    BYTE   "Value found at index: ", 0
    msg2    BYTE   "Value not found in the array!", 0
    arr     DWORD  1, 2, 3, 4, 5, 6, 7, 8, 9, 10
    arrsize DWORD  10
    val     DWORD  ?
    index   DWORD  -1
    msg3    ~~Byte~~ BYTE  "Enter integer value: ", 0
```

Date: ___/___/_

```
.code
main PROC
    mov edx, OFFSET msg3
    call ~~WRITESTRING~~ WriteString
    call ~~READINT~~ ReadInt
    mov val, eax
    INVOKE Search, OFFSET arr, val, 0
    cmp index, -1
    jne found
    mov edx, OFFSET msg2
    call WriteString
    jmp exitProgramm

    found:
        mov edx, OFFSET msg1
        call WriteString
        mov eax, index
        call Crlf
    exitProgramm:
        exit
        main ENDP
Search PROC ~~USES~~ USES eax ebx ecx, edx esi
        push ebp
        mov ebp, esp
        mov esi, [ebp+12]
        mov eax, [ebp+8]
        mov ebx, [ebp+4]
        cmp esi, arrsize
        jge End
        mov edx, [ebx+esi*4]
        cmp edx, eax
        ~~jge~~ jne notfound
        mov index, esi
        jmp END.
```

Date: ___/___/___

```
notfound:
    inc esi
    INVOKE search, ebx, eax, esi
END:
    pop ebp
    ret
search ENDP.
END main.
```

①	②	③
ret	ret	ret
ebp	ebp	ebp
index = 0	index = 1	index = 2
val = [illegible]3	val = [illegible]3	val = [illegible]3
arr ptr	arr ptr	arr ptr assign

↓

arr[2] = 3

Q3:

```
.data
    found   BYTE [illegible] 0
    msg1    BYTE "Target String: ", 0
    msg2    BYTE "Source String: ", 0
    trgt    BYTE 100 DUP(0)
    len     DWORD ?
    index   DWORD 0
.code
main PROC
    mov edx, OFFSET msg2
```

Date: ___/___/___

Q3:

```
.data
    srcStr  BYTE  "This is the source string",0
    targetStr BYTE  100 DUP(0)
    targetLen DWORD 0
.code
main PROC
    mov  esi, OFFSET  srcStr
    mov  edi, OFFSET  targetStr
    nextChar:
        mov al, [esi]
        cmp al, 0
        je  done
        push esi
        mov  ecx, targetLen
        mov  esi, OFFSET  targetStr
        mov  ebx, 0
    searchLoop:
        cmp  ecx,0
        je   checkfound
        mov  dl, [esi]
        cmp  al, dl
        je   found
        inc  esi
        dec  ecx
        jmp  searchLoop
    found:
        mov  ebx, 1
    checkfound:
        pop esi
        cmp ebx, 1
        je  skipChar
        mov edi, [al]
        inc edi
        inc targetLen.
```

Date: ___/___/___

```
skipChar:
        inc  esi
        jmp  nextChar
done:
        mov  BYTE PTR [edi],0
        mov  edx, OFFSET srcStr
        call WriteString
        call Crlf
        mov  edx, OFFSET targetStr
        call WriteString
        call Crlf
        exit
main ENDP
END main.
```

Q4:

```
.data
        msg1    BYTE  "Enter a string: ",0
        inputStr BYTE 256 DUP(0)
        vowels  DWORD 5 DUP(0)
        labels  BYTE  "aA", "eE", "iI","oO", "uU",0
        msg2    BYTE  "vowel count: ",0
        msg3    BYTE  " = ",0
        msg4    BYTE  " or ",0
.code
main PROC
        mov  edx, OFFSET msg1
        call WriteString
        mov  edx, OFFSET inputStr
        mov  ecx, SIZEOF inputStr
        call ReadString.
        mov  esi, OFFSET inputStr
        mov  ecx,0
```

Date: ___/___

```
countVowels:
        mov al, [esi]
        cmp al, 0
        je displayRes
        mov edi, OFFSET labels
        mov ebx, 0
checkVowel:
        mov dl, [edi]
        cmp dl, 0
        je nextChar
        cmp al,dl
        je foundVowel
        inc edi
        mov edi, dl, [edi]
        cmp al,dl
        je foundVowel
        inc edi
        inc ebx
        jmp checkVowel
foundVowel:
        inc DWORD PTR vowels[ebx*4]
nextChar:
        inc esi
        jmp countVowels
displayRes:
        mov edx, OFFSET msg2
        call WriteString
        call Crlf
        mov ecx, 5
        mov esi, 0
        mov edi, OFFSET labels
```

Date: ___/___/___

```
L1:
    mov  al, [edi]
    call WriteChar
    mov  edx, OFFSET msg4
    call WriteString
    inc  edi
    mov  al, [edi]
    call WriteChar
    mov  edx, OFFSET msg3
    call WriteString
    mov  eax, vowels[esi*4]
    call WriteDec
    call Crlf
    add  edi, 1
    inc  esi
    loop L1

    exit
main  ENDP
END   main.
```