## Task#1

‘Equal sign, =’ directive associates a symbol with an integer expression.

Name = expression # count = 100

Any symbol defined with the equal sign directive can be redefined in the code later.

‘equ’ directive associates a symbolic name with an integer expression or some arbitrary

text.

name equ expression # a equ 2+3

name equ symbol # a equ var (var defined by = or equ)

name equ &lt;text&gt; # a equ &lt;2+3&gt;

Any symbol defined with the equ directive cannot be redefined in the same source code

file.

The **‘textequ’** directive creates a text macro. There are three different formats:

var1 = 5

name textequ text # count textequ %(var1 \* 2)

name textequ &lt;text macro&gt; # move textequ &lt;mov&gt;

name textequ %constant\_expression # Copy\_AL textequ &lt;move al,count&gt;

Any symbol defined with the directive textequ can be redefined at any time.

## Task#5

dosseg

.model small

.stack 100h

.data

var db 50h

.code

main proc

mov si, offset var

mov al,[si]

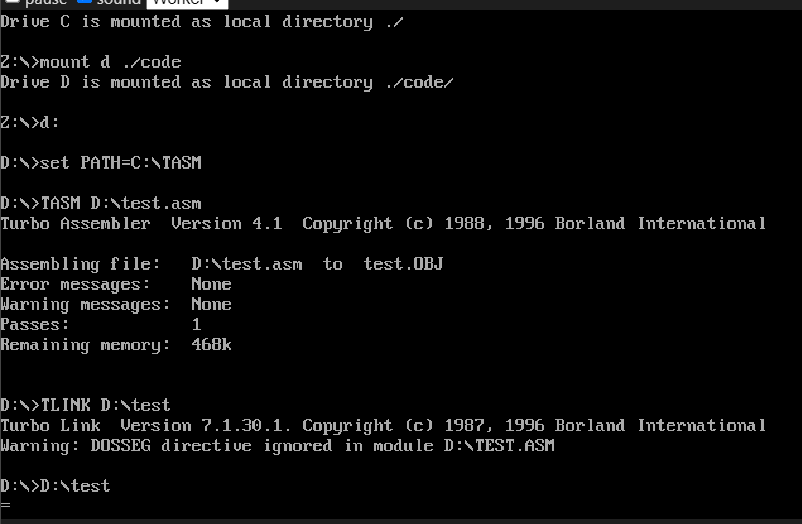
mov ah, 02h

mov dl, al

int 21h

main endp

end main



## Task#3

A)

dosseg

.model small

.stack 100h

.data

array1 db 'a','b','c','d','e','f'

array1Size = ($-array1)

array1Type db ?

array1Length dw ?

array1Offset dw ?

.code

main proc

;TYPE

mov dl,'T'

mov ah,02h

int 21h

mov dl,'Y'

mov ah,02h

int 21h

mov dl,'P'

mov ah,02h

int 21h

mov dl,'E'

mov ah,02h

int 21h

mov dl,':'

mov ah,02h

int 21h

;printing and saving array1 type

mov array1Type, TYPE array1

mov dl, array1Type

add dl,48 ;printing array1 type

mov ah,02h

int 21h

mov dl,10; endline char

mov ah,02h

int 21h

;Length

mov dl,'L'

mov ah,02h

int 21h

mov dl,'E'

mov ah,02h

int 21h

mov dl,'N'

mov ah,02h

int 21h

mov dl,'G'

mov ah,02h

int 21h

mov dl,'T'

mov ah,02h

int 21h

mov dl,'H'

mov ah,02h

int 21h

mov dl,':'

mov ah,02h

int 21h

mov bl,array1Type

mov al,array1Size

mov ah,0

div bl

mov dl,al

add dl,'0'

mov ah,02h

int 21h

mov dl,10

mov ah,02h

int 21h

mov dl,'S'

mov ah,02h

int 21h

mov dl,'I'

mov ah,02h

int 21h

mov dl,'Z'

mov ah,02h

int 21h

mov dl,'E'

mov ah,02h

int 21h

mov dl,':'

mov ah,02h

int 21

mov si, OFFSET array1

mov array1Offset,si

mov dl,array1Size

add dl,'0'

mov ah,02h

int 21h

mov dl,10; endline char

mov ah,02h

int 21h

mov ah,4ch

int 21h

main endp

end main

A screenshot of a computer

Description automatically generated

B)

dosseg

.model small

.stack 100h

.data

array1 dw 'a','b','c','d'

array1Size = ($-array1)

array1Type db ?

array1Length dw ?

array1Offset dw ?

.code

main proc

;TYPE

mov dl,'T'

mov ah,02h

int 21h

mov dl,'Y'

mov ah,02h

int 21h

mov dl,'P'

mov ah,02h

int 21h

mov dl,'E'

mov ah,02h

int 21h

mov dl,':'

mov ah,02h

int 21h

;printing and saving array1 type

mov array1Type, TYPE array1

mov dl, array1Type

add dl,48 ;printing array1 type

mov ah,02h

int 21h

mov dl,10; endline char

mov ah,02h

int 21h

;Length

mov dl,'L'

mov ah,02h

int 21h

mov dl,'E'

mov ah,02h

int 21h

mov dl,'N'

mov ah,02h

int 21h

mov dl,'G'

mov ah,02h

int 21h

mov dl,'T'

mov ah,02h

int 21h

mov dl,'H'

mov ah,02h

int 21h

mov dl,':'

mov ah,02h

int 21h

mov bl,array1Type

mov al,array1Size

mov ah,0

div bl

mov dl,al

add dl,'0'

mov ah,02h

int 21h

mov dl,10

mov ah,02h

int 21h

;SIZE

mov dl,'S'

mov ah,02h

int 21h

mov dl,'I'

mov ah,02h

int 21h

mov dl,'Z'

mov ah,02h

int 21h

mov dl,'E'

mov ah,02h

int 21h

mov dl,':'

mov ah,02h

int 21h

mov dl,array1Size

add dl,'0'

mov ah,02h

int 21h

mov dl,10; endline char

mov ah,02h

int 21h

mov si, OFFSET array1

mov array1Offset,si

mov dl,10

mov ah,02

int 21h

mov ah,4ch

int 21h

main endp

end main

A screenshot of a computer

Description automatically generated

## Task#4

dosseg

.model small

.stack 100h

.data

array1 db 1,2,3,4

array1Size = ($-array1)

num db 3

.code

main proc

mov ax, @data

mov ds, ax

;offset

mov si, OFFSET array1

mov cl,array1Size

;loop

l1:

mov dl, [si] ; Load the value at current index

add dl, num

mov [si], dl ; Store the updated value back in the array

mov dl, [si] ; Load the updated value ; Print the updated value

add si, 2 ; Move to the next odd index

loop l1

mov dl,array1[0]

add dl,'0'

mov ah,02

int 21h

mov dl,array1[1]

add dl,'0'

mov ah,02

int 21h

mov dl,array1[2]

add dl,'0'

mov ah,02

int 21h

mov dl,array1[3]

add dl,'0'

mov ah,02

int 21h

mov dl,10

mov ah,02

int 21h

mov ah,4ch

int 21h

main endp

end main

A screenshot of a computer

Description automatically generated

## Task#2

A computer screen shot of a blue screen

Description automatically generated