**“CYCLE SORT”**



|  |
| --- |
| **Microprocessor and Assembly Language** |

**Presented to: Waheed Uddin Hyder**

**Presented by: Sheikh Haris Khalid**

**ID: SP17-BSSE-0046**

**Section: B(M)**

org 0x100

.data

array dw 29, 34, 58, 86, 96, 05, 77, 31

msg2 db 10,13, '$'

msg3 db 10,13, 'Cycle compelete.$'

x db 10

startingValue db 0

middleValue db 0

newValue db 0

.code

main proc

mov ax,@data

mov ds,ax

i\_loop db 0 ;i=0

t9\_index db 8 ;indexValue

t8\_index db 7 ;indexValue

plusTwo db 0

jmp Print ;Print the starting array

For:

mov startingValue,[array + plusTwo] ;move value of array

j\_loop db 0 ;j=0

count db 0 ;count=0

anyNumber db 2 ;for value '2'

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

cmp i\_loop, 0 ;if (i = 0)

je Add\_2\_for\_0 ;jump to Add\_2\_for\_0

Add\_2\_for\_anyNumber:

mov middleValue, [array] ;move array in arr2

mul anyNumber,i\_loop ;2 = 2 \* i

add middleValue,anyNumber ;middleValue + anyNumber

jmp Go ;jump to Go

Add\_2\_for\_0:

mov middleValue, [array] ;move array in arr2

add middleValue,2 ;middleValue + 2

jmp Go ;jump to Go

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

Go:

k db 7 ;for 7 iterations

mov bx, 2

For2:

mov value, [array + bx] ;move array in arr1

cmp startingValue, value

jge Increment ;startingValue > value

NotIncrement:

add bx, 2 ;arrNextValue

dec k ;k--

jnz For2

jmp Exit2 ;jump to Exit2

Increment:

add count,1 ;count++

add bx, 2 ;arrNextValue

dec k ;k--

jnz For2

Exit2:

add count,i\_loop ;count + i

cmp i\_loop,count ;if (i = count)

je NextValue ;ZF flag

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;; TO GET THE VALUE OF count AND newValue ;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

mul count,2 ;multiply the count by 2

mov bx, count ;move count in bx

mov newValue,[array + bx] ;move array in array to temp

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;; PERFORM SWAP ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;; BETWEEN startingValue AND newValue ;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

mov temp, newValue ;temp = $t4

mov newValue ,startingValue ;t4 = $t3

mov startingValue, temp ;$t3 = temp

mov [array + bx],newValue ;value move and store to array

mov [array + plusTwo],startingValue ;value move and store to array

jmp Print ;jump print

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;; GO TO NEXT VALUE ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

NextValue:

dec t8\_index ;indexValue - 1

add plusTwo,2 ;arrNextValue

lea dx, msg3 ;print 'Cycle compelete.'

mov ah, 9

int 21h

lea dx, msg2 ;print 'space'

mov ah, 9

int 21h

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;;;;;;;; TO PRINT THE ARRAY ;;;;;;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

Print:

m db 7 ;loop = 7

mov bx, 0

StartPrint:

mov ax,[array + bx] ;mov value to ax

div x ;to break them into two seperate numbers.

add bx, 2 ;add 2 into bx

add al,30h ;BCD to ASCII

add ah,30h ;BCD to ASCII

mov ch,ah ;save ah into ch for 2nd print

mov ah,2 ;print first value

mov dl,al

int 21h

mov ah,2 ;print second value

mov dl,ch

int 21h

mov dl, 09 ;print tab

int 21h

dec m ;m--

jnz StartPrint ;jump not zero = ZF flag

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;; FOR NEXT LINE ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

NextLine:

mov dl,0A ;newLine

int 21h

mov dl,OD ;create

int 21h

dec t9\_index ;indexValue - 1

jmp For

Exit:

ret ;exitCode