

INDEX

|  |  |  |  |
| --- | --- | --- | --- |
| ***S No.*** | ***Objective*** | ***Date*** | ***Sign*** |
| **1** | **Write a program for linear search and binary search.** | **07/11/2023** |  |

**PRACTICAL-3**

Q3. Write a program for linear search and binary search?

# Linear Search Code:

.model small

.486

.data

str1 db 10,13, "Enter the size of the array(MAX 9) : $" str2 db 10,13, "----------ARRAY 1 $"

str3 db 10,13, "Enter the element to be searched (byte sized) : $" str4 db 10,13, "Array Input : $"

str5 db 10,13, "Element Found at index : $" str6 db 10,13, "Element not found! $" num1 db 9 dup(0)

s db 00h e db 00h

.stack 200h

.code

.startup

lea dx,str1

call prints ;prints string call getSize

lea dx,str2 call prints call getVal1 lea dx,str3 call prints call getVal2 call linear

.exit

prints proc near

mov ah,9 int 21h ret

prints endp

getSize proc near

mov ah,1 int 21h sub al,30h mov s,al ret

getSize endp

getVal1 proc near

mov si,0 mov cl,s lea dx,str4 ab1:

call prints mov ah,1 int 21h sub al,30h

.if al > 9h

sub al,7h

.endif

mov num1[si],al shl num1[si],4 mov ah,1

int 21h sub al,30h

.if al > 9h

sub al,7h

.endif

add num1[si],al inc si

loop ab1 ret

getVal1 endp

getVal2 proc near

mov ah,1 int 21h sub al,30h

.if al > 9h

sub al,7h

.endif

mov bl,al shl bl,4 mov ah,1 int 21h sub al,30h

.if al > 9h

sub al,7h

.endif add bl,al mov e,bl ret

getVal2 endp

linear proc near

mov si,0 mov cl,s mov bl,e ab:

.if num1[si] == bl jmp success

.endif inc si

loop ab lea dx,str6 call prints jmp fail success:

lea dx,str5 call prints mov dx,si add dl,30h mov ah,2 int 21h

fail: ret

linear endp end

# OU TPU T:

**Binary Search Code:**

.model small

.486

.data

str1 db 10,13, "Enter the size of the array(MAX 9) : $" str2 db 10,13, "----------ARRAY 1 $"

str3 db 10,13, "Enter the element to be searched (byte sized) : $" str4 db 10,13, "Array Input : $"

str5 db 10,13, "Element Found at index : $" str6 db 10,13, "Element not found! $" num1 db 9 dup(0)

s db 00h e db 00h

.stack 200h

.code

.startup

lea dx,str1

call prints ;prints string call getSize

lea dx,str2 call prints call getVal1 lea dx,str3 call prints call getVal2 call binary

.exit

prints proc near

mov ah,9 int 21h

ret prints endp

getSize proc near

mov ah,1 int 21h sub al,30h mov s,al ret

getSize endp

getVal1 proc near

mov si,0 mov cl,s lea dx,str4 ab1:

call prints mov ah,1 int 21h sub al,30h

.if al > 9h

sub al,7h

.endif

mov num1[si],al shl num1[si],4 mov ah,1

int 21h sub al,30h

.if al > 9h

sub al,7h

.endif

add num1[si],al inc si

loop ab1 ret

getVal1 endp

getVal2 proc near

mov ah,1 int 21h sub al,30h

.if al > 9h

sub al,7h

.endif

mov bl,al shl bl,4 mov ah,1 int 21h sub al,30h

.if al > 9h

sub al,7h

.endif add bl,al mov e,bl ret

getVal2 endp

binary proc near

mov cl,s mov si,cx dec si

shr si,1 ;divide by 2 mov bl,e

ab:

.if num1[si] == bl jmp success

.elseif num1[si] > bl dec si

shr si,1

.else

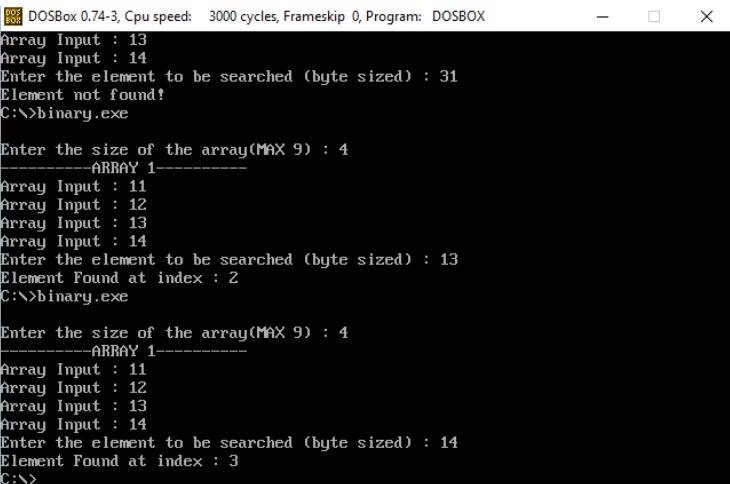
mov dx,si add dl,s mov si,dx shr si,1

.endif

loop ab lea dx,str6 call prints jmp fail success:

lea dx,str5

fail: ret

binary endp end **OUTPUT:**

call prints mov dx,si add dl,30h mov ah,2 int 21h