1. Design and develop an assembly language program to search a key element "X" in a list of 'n' 16-bit numbers. Adopt *Binary search* algorithm in your program for searching.

```
.model small

initds macro
    mov ax,@data        ; Initializing the Data Segment
    mov ds,ax           ; it is ds, not dx
endm

printf macro msg
    lea dx,msg          ; Load the Effective Address to DX
    mov ah,9            ; Function Number is 9
    int 21h             ; Using DOS interrupt 21h
endm

putchar macro char
    mov dl,char       ; load the printable character's HEX value in DL
    mov ah,2            ; Function Number is 9
    int 21h             ; Using DOS interrupt 21h
endm

exit macro
    mov ah,4ch          ; to terminate
    int 21h
endm

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
.data

    array dw 1122h,2345h,3333h,4455h,6666h       ; 16 bit array
    len dw ($-array)/2          ; len = (last_index - first_index)/2

    search equ 2345h            ; key to Search

    foundmsg db 'Element found at position : $'
    position db  0              ; now it's 0, later we shall put

    notfoundmsg db 'Element not found $'

.code

    initds              ; Initializing Data Segment (call that macro)

    mov bx,1                ; low
    mov dx,len              ; high
    mov cx,search           ; key
    again:

        cmp bx,dx           ; while(low<high)
        ja failure          ; if (low>high) then its not found case.

        mov ax,bx
        add ax,dx           ; low+high
        shr ax,1            ; (low+high) /2
        mov si,ax           ; have an index
```

```
        dec si                  ; adjust the index (pointing to the mid)
        add si,si               ; for 16 bit data
        cmp cx,array[si]        ; if(key==array[mid])
        jae bigger              ; search in the RIGHT part of the array

        dec ax      ; dec high (search in the LEFT part of the array)
        mov dx,ax               ; make this as new high
        jmp again               ; continue searching

    bigger:

        je success              ; found case
        inc ax                  ; inc low
        mov bx,ax               ; make this as new low
        jmp again               ; continue searching

    success:

        add al,30h              ; add 30h (or '0') to the position(AL)
                                ; (just to convert to ascii)
        mov position,al         ; move the position to our variable

        printf foundmsg         ; printing found message
        putchar position        ; printing found position
        exit                    ; you are done, so bye bye!

    failure:

        printf notfoundmsg      ; printing not found message
        exit                    ; bye!

end
```