LAB 7

1.Find Smallest and largest number in the array of words

2.Linear search in an array of 10 unsigned words

3.Selection sort (Descending)

4.Bubble sort (Ascending)

5.Insertion sort (Ascending)

6.)Binary Search on sorted array

1.) ;Find minimum and maximum elements in an array of words

```
min macro         ;Macro for finding minimum word in an array
mov dx,[si]       ;make a copy of number pointed by si in dx
dec cx            ;set count value in cx for comparison
UP:cmp dx,[si+2]  ;compare two adjacent numbers
   jb continue    ;if first number is smaller,go to continue
   mov dx,[si+2]  ;if first number is greater, move smaller number to dx
continue:add si,2 ;move to next number for comparison
   dec cx         ;decrement cx to check if all no.s are compared
   jnz up         ;if no,continue to compare
endm

max macro      ;Macro for finding maximum word in an array
mov bx,[si]    ;make a copy of number pointed by si in bx
dec cx         ;set count value in cx for comparison
UP1:cmp bx,[si+2]  ;compare two adjacent numbers
   ja continue1    ;if first number is greater,go to continue
   mov bx,[si+2]   ;if first number is smaller, move smaller number to dx
continue1:add si,2 ;move to next number for comparison
   dec cx          ;decrement cx to check if all no.s are compared
   jnz up1         ;if no,continue to compare
```

```
endm

data segment
array dw 1234h,2345h,3456h,1267h,0099h
arr_size dw 0005h
minimum dw ?
maximum dw ?
data ends

code segment
assume cs:code,ds:data
start:mov ax,data
    mov ds,ax
    mov cx,arr_size   ;load cx with number of data words in array
    lea si,array      ;make si point to base address of array
    min               ;call macro for mainimum dataword
    mov minimum,dx    ;store smallest element in data segment
    mov cx,arr_size   ;load cx with number of data words in array
    lea si,array      ;make si point to base address of array
    max               ;call macro for maximum dataword
    mov maximum,bx    ;store largest element in data segment
    mov ah,4ch
    int 21h
    code ends
    end start
```

2.);Linear search of an element in an array of words

```
prtstr macro msg
lea dx,msg
```

```
mov,ah,09h

int 21h

endm


DATA SEGMENT

array dw 0001h,0002h,0003h,0004h,0005h,0704h,0976h,1233h,1237h,000ah

arr_size dw $-array  ;size of array

s_key dw 0704h       ;element to be found

msg db 'Found$'

msg1 db 'Not found$'

DATA ENDS


CODE SEGMENT

ASSUME CS:CODE,DS:DATA

START: mov ax,data

       mov ds,ax

      lea si,array     ;Load effective address of array to si

      mov cx,arr_size  ;move size of array to cx

      mov ax,s_key     ;move element to be found to ax


Label1:cmp ax,[si]     ;compare ax contents with elements of array

       jz label2       ;element found if z=0

       add si,2        ;element not found, move to next element of array

       loop label1     ;go to label1

       prtstr msg1     ;print not found message

       jmp exit        ;jump to exit

label2:prtstr msg      ;print found message


exit: mov ah,4ch
```

```
     int 21h
     code ends
     end start
```

3.);Selection sort in descending order

```
DATA SEGMENT
array db 98h,09h,12h,32h,21h,89h,01h,05h,67h,62h
arr_size db $ - array ;size of the array
nc db ?       ;to keep count of no. of comparisons to be made in an iteration
DATA ENDS

CODE SEGMENT
assume cs:code,ds:data
START: mov ax,data
      mov ds,ax
      mov dh,0h
      mov dl,arr_size    ;dx=000ah
      dec dx             ;dx=0009h=no. of passes needed to complete sorting

OUTER: mov cx,dx          ;no. of comparisons to be made in a pass
      lea si,array       ;si contains position of an element in array
      mov ah,[si]        ;move first element of unsorted part of array to ah
      mov bx,si          ;move value of si to bx

INNER: inc si            ;increment si to point to next element
      inc nc             ;no. of comparisons made
      cmp ah,[si]        ;compare ah and element pointed by si
      jb go_on           ;if it's less, go to label go_on
      mov ah,[si]        ;if it's greater then move element to ah
      mov bx,si          ;ah contains smallest element so far and bx, its position
```

```
GO_ON: loop inner        ;decrement cx using loop

      xchg ah,[si]      ;exchange last element of unsorted array with smallest element of unsorted part
      mov [bx],ah
      dec dx            ;if dx=0, sorting is complete
      jnz outer         ;go to outer if dx is not 0
      mov ah,4ch
      int 21h
      code ends
      end start
```

4.) ;Bubble sort in ascending order

```
DATA SEGMENT
array db 09h,08h,07h,06h,05h,04h,03h,02h,01h,0ah
arr_size db $ - array
DATA ENDS
CODE SEGMENT
ASSUME CS:CODE,DS:DATA
START: mov ax,data
      mov ds,ax
      mov bh,0h
      mov bl,arr_size ;move array size to bx
      dec bx          ;decrement bx to get number of passes to complete sorting
OUTER: mov cx,bx      ;cx=no. of comparison in a pass
      lea si,array    ;make si point to base address of array
INNER: mov al,[si]    ;move the element of array to al
      inc si          ;makesi point to next element
      cmp al,[si]     ;compare two consecutive elements
      jb GO_ON        ;if first element is lesser, continue loop
```

```
       xchg al,[si]    ;if first element is smaller,
       mov [si-1],al   ;exchange with next element position
GO_ON: loop INNER      ;go to inner label if cx not equals 0
       dec bx          ;decrese bx
       jnz OUTER       ;if bx=0, the sorting is over, else continue sorting
       mov ah,4ch
       int 21h
code ends
end start
```

5.);Insertion sort in ascending order

```
data segment
array dw 0009h,0008h,0001h,0034h,2345h,1234h
arr_size dw 0006h    ;size of array
data ends

code segment
assume cs:code,ds:data
start:mov ax,data
      mov ds,ax
      mov cx,2    ;to insert second element in proper position
outer:mov dx,cx
      dec dx      ;maximum no. of comparisons needed to insert an element
      mov si,dx   ;load si with dx
      add si,si   ;word elements are located at an offset of 0,2,4.. etc
      mov ax,array[si] ;ax=no. to be inserted in proper position
inner:cmp array[si-2],ax ;if ax has smaller no., the correct position of insertion is obtained
      jbe inexit        ;exit inner loop
      mov di,array[si-2]  ;mov contents to insert ax contents
      mov array[si],di
```

```
     dec si
     dec si          ;di points to previous word now
     dec dx
     jnz inner       ;if dx is not zero, go to inner loop again
inexit:mov array[si],ax  ;mov ax to array[si]
      inc cx           ;increment cx to insert next element in proper position
      cmp cx,arr_size
      jbe outer       ;if cx<=array size, go to outer loop
mov ah,4ch
int 21h
code ends
end start
```

6.) ;Binary search on a sorted array

```
prtstr macro str
lea dx,str
mov ah,09h
int 21h
endm
data segment
array dw 1122h,2345h,3344h,4455h,5566h
len dw 0005h    ;length of array
msg db 'Found$'
msg1 db 'Not Found$'
skey dw 2345h   ;element to be found in the array
data ends
code segment
assume cs:code,ds:data
start:mov ax,data
```

```
      mov ds,ax
      mov bx,0001h    ;move 1 to bx
      mov dx,len      ;move lenght of array to dx
      mov cx,skey     ;move element to be found to cx
again:cmp bx,dx       ;compare first and last element of array
      ja failure      ;if bx>dx, search unsuccessful
      mov ax,bx
      add ax,dx
      shr ax,1
      mov si,ax
      dec si
      add si,si
      cmp cx,array[si]
      jae bigger
      dec ax
      mov dx,ax
      jmp again
bigger:je success
       inc ax
       mov bx,ax
       jmp again
success:prtstr msg ;display found message
        jmp exit
failure:prtstr msg1;display not found message
exit:mov ah,4ch
     int 21h
     code ends
     end start
```