**Assignment: - 04/ *1D Arrays: searching, sorting:***

1 **) Write a C program to find out the largest and smallest element from a 1D array**.

#include<stdio.h>

int main()

{

int a[50], size, i, big, small;

printf("\nEnter the size of the array: ");

scanf("%d", &size);

printf("\n\nEnter the %d elements of the array: \n\n", size);

for(i = 0; i < size; i++)

scanf("%d", &a[i]);

big = a[0];

for(i = 1; i < size; i++)

{

if(big < a[i])

{

big = a[i];

}

}

printf("\n\nThe largest element is: %d", big);

small = a[0];

for(i = 1; i < size; i++)

{

if(small>a[i])

{

small = a[i];

}

}

printf("\n\nThe smallest element is: %d", small);

return 0;

}

Output,

Enter the size of the array: 5

Enter the 5 elements of the array:

1 5 8 7 3

The largest element is: 8

The smallest element is: 1

--------------------------------

Process exited after 9.5 seconds with return value 0

Press any key to continue . . .

**2) Write a C program to search an element from a 1D integer array using linear search technique.**

#include <stdio.h>

int main()

{

int array[100], search, c, n;

printf("Enter number of elements in array\n");

scanf("%d", &n);

printf("Enter %d integer(s)\n", n);

for (c = 0; c < n; c++)

scanf("%d", &array[c]);

printf("Enter a number to search\n");

scanf("%d", &search);

for (c = 0; c < n; c++)

{

if (array[c] == search) /\* If required element is found \*/

{

printf("%d is present at location %d.\n", search, c+1);

break;

}

}

if (c == n)

printf("%d isn't present in the array.\n", search);

return 0;

}

Output

Enter number of elements in array

5

Enter 5 integer(s)

4

6

9

8

7

Enter a number to search

9

9 is present at location 3.

--------------------------------

Process exited after 16 seconds with return value 0

Press any key to continue . . .

**3) Write a C program to search an element from a 1D integer array using binary search technique.**

#include <stdio.h>

int main()

{

int c, first, last, middle, n, search, array[100];

printf("Enter number of elements\n");

scanf("%d", &n);

printf("Enter %d integers\n", n);

for (c = 0; c < n; c++)

scanf("%d", &array[c]);

printf("Enter value to find\n");

scanf("%d", &search);

first = 0;

last = n - 1;

middle = (first+last)/2;

while (first <= last) {

if (array[middle] < search)

first = middle + 1;

else if (array[middle] == search) {

printf("%d found at location %d.\n", search, middle+1);

break;

}

else

last = middle - 1;

middle = (first + last)/2;

}

if (first > last)

printf("Not found! %d isn't present in the list.\n", search);

return 0;

}

Output

Enter number of elements

6

Enter 6 integers

4

5

6

9

8

7

Enter value to find

5

5 found at location 2.

--------------------------------

Process exited after 10.52 seconds with return value 0

Press any key to continue . . .

**4) Write a C program to sort a list on n numbers using *any one* of the following sorting technique:**

1. Bubble Sort.
2. Selection sort.
3. #include <stdio.h>

int main()

{

int arr[10], n,i,j,k,temp;

printf("Enter number of elements\n");

scanf("%d", &n);

for (i = 0; i < n; i++)

{

printf("Enter arr[%d] element: ",i);

scanf("%d",&arr[i]);

}

for (j = 0 ; j< n - 1; j++)

{

for (k= 0 ; k < n - j- 1; k++)

{

if (arr[k] > arr[k+1])

{

temp = arr[k];

arr[k] = arr[k+1];

arr[k+1] = temp;

}

}

}

printf("Sorted list in ascending order:\n");

for (i = 0; i < n; i++)

printf("%d\n", arr[i]);

}

Output

Enter number of elements

6

Enter arr[0] element: 6

Enter arr[1] element: 9

Enter arr[2] element: 8

Enter arr[3] element: 2

Enter arr[4] element: 3

Enter arr[5] element: 2

Sorted list in ascending order:

2

2

3

6

8

9

--------------------------------

Process exited after 9.571 seconds with return value 0

Press any key to continue . . .

1. #include <stdio.h>

int main()

{

int array[100], n, c, d, position, t;

printf("Enter number of elements\n");

scanf("%d", &n);

printf("Enter %d integers\n", n);

for (c = 0; c < n; c++)

scanf("%d", &array[c]);

for (c = 0; c < (n - 1); c++) // finding minimum element (n-1) times

{

position = c;

for (d = c + 1; d < n; d++)

{

if (array[position] > array[d])

position = d;

}

if (position != c)

{

t = array[c];

array[c] = array[position];

array[position] = t;

}

}

printf("Sorted list in ascending order:\n");

for (c = 0; c < n; c++)

printf("%d\n", array[c]);

return 0;

}

Output

Enter number of elements

6

Enter 6 integers

8

5

4

2

7

3

Sorted list in ascending order:

2

3

4

5

7

8

--------------------------------

Process exited after 10.87 seconds with return value 0

Press any key to continue . . .