C program to implement linear search in unsorted array

#include<stdio.h>

// Function to find the element for deletion

// where array[] is the array from which element needs to be deleted

// size is the size of the array

// keyToFind is the element to be deleted from the array

int findElement(int array[], int size, int keyToBeSearched)

{

int i;

// Finding & returning the position of the element

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

if (array[i] == keyToBeSearched)

return i;

return - 1;

}

// Main Function

int main()

{

int array[] = { 31, 27, 3, 54, 67, 31 };

int size = sizeof(array) / sizeof(array[0]);

int keyToBeSearched = 67;

// Calling the function to delete an element from the array

int pos = findElement(array, size, keyToBeSearched);

if(pos==-1){

printf("n Element %d not found", keyToBeSearched);

}

else{

printf("n Position of %d: %d", keyToBeSearched ,pos+1);

}

return 0;

}

## **C Program to Insert An Element In An Array**

**Methode 1**

#include<stdio.h>

//Function to insert an element in an array.

//arr[] = array, elements = number of elements present in the array

//keyToBeInserted = element to be inserted in the array

// size of the array

int insertElement(int arr[], int elements, int keyToBeInserted, int size)

{

// Check if the capacity of the array is already full

if (elements >= size)

return elements;

//If not then the element is inserted at the last index

//and the new array size is returned

arr[elements] = keyToBeInserted;

return (elements + 1);

}

// Main Function

int main()

{

int array[20] = { 31, 27, 3, 54, 67, 31 };

int size = sizeof(array) / sizeof(array[0]);

int elements = 6;

int i, keyToBeInserted = 32;

printf("n Before Insertion: ");

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

printf("%d  ", array[i]);

// Calling the function to insert the element in the array

elements = insertElement(array, elements, keyToBeInserted, size);

printf("n After Insertion: ");

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

printf("%d  ",array[i]);

return 0;

}

**Methode 2**

**/\* C program to insert an element in an array \*/**

**#include <stdio.h>**

**int main()**

**{**

**int n;**

**scanf(“%d”,&n);**

**int arr[n];**

**int i;**

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

**{**

**scanf(“%d”,&arr[i]);**

**}**

**int pos;**

**scanf(“%d”,&pos);**

**int ele;**

**scanf(“%d”,&ele);**

**if(pos > n)**

**printf(“Invalid Input”);**

**else**

**{**

**for (i = n – 1; i >= pos – 1; i–)**

**arr[i+1] = arr[i];**

**arr[pos-1] = ele;**

**printf(“Array after insertion is:\n”);**

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

**printf(“%d\n”, arr[i]);**

**}**

**return 0;**

**}**

**Methode 3**

**/ Solution using C pointers**

#include<stdio.h>#include<stdlib.h>

void insert(int n1, int \*a, int l, int e)

{

int i;

printf(“Array after insertion is:\n”);

for(i=0;i<l-1;i++)

{

printf(“%d\n”,\*(a+i));

}

printf(“%d\n”,e);

for(i=l-1;i<n1;i++)

{

printf(“%d\n”,\*(a+i));

}

}

int main()

{

int \*a,n1,i,l,e;

scanf(“%d”,&n1);

a=(int\*)malloc(n1\*sizeof(int));

for(i=0;i<n1;i++)

{

scanf(“%d”,a+i);

}

scanf(“%d”,&l);

if(l<=n1)

{

scanf(“%d”,&e);

insert(n1,a,l,e);

}

else

{

printf(“Invalid Input”);

}

return 0;

}

## **C Function to Delete An Element From An Array**

**Methode 1**

#include<stdio.h>

// Function to delete an element

// where array[] is the array from which element needs to be deleted

// size is the size of the array

// keyToBeDeleted is the element to be deleted from the array

int deleteElement(int array[], int size, int keyToBeDeleted)

{

// Calling findElement function to get the position of the element which needs to be deleted

int pos = findElement(array, size, keyToBeDeleted);

// If element is not found then it prints Element not found

if (pos == - 1)

{

printf("Element not found");

return size;

}

// Otherwise it deletes the element & moves rest of the element by one position

int i;

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

array[i] = array[i + 1];

return size - 1;

}

// Function to find the element for deletion

int findElement(int array[], int size, int keyToBeDeleted)

{

int i;

// Finding & returning the position of the element

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

if (array[i] == keyToBeDeleted)

return i;

return - 1;

}

// Main Function

int main()

{

int array[] = { 31, 27, 3, 54, 67, 31 };

int size = sizeof(array) / sizeof(array[0]);

int i, keyToBeDeleted = 67;

printf("n Before Deletion: ");

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

printf("%d  ", array[i]);

// Calling the function to delete an element from the array

size = deleteElement(array, size, keyToBeDeleted);

printf("n After Deletion: ");

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

printf("%d  ",array[i]);

return 0;

}

**Methode 2**

#include <stdio.h>

int main()

{

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

printf(“Enter the number of elements of the array : “);

scanf(“%d”, &n);

printf(“\nInput the array elements : “);

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

scanf(“%d”, &array[c]);

printf(“\nEnter the position : “);

scanf(“%d”, &position);

if (position >= n+1)

printf(“\nDeletion not possible.\n”);

else

{

for (c = position – 1; c < n – 1; c++)

array[c] = array[c+1];

printf(“\nArray after deletion : “);

for (c = 0; c < n – 1; c++)

printf(“%d\n”, array[c]);

}

return 0;

}

**Methode 3**

**// Solution using C pointers**

#include<stdio.h>#include<stdlib.h>

void delete(int n,int \*a,int l);

int main()

{

int \*a,n,i,l;

scanf(“%d”,&n);

a=(int\*)malloc(sizeof(int)\*n);

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

{

scanf(“%d”,(a+i));

}

scanf(“%d”,&l);

delete(n,a,l);

return 0;

}

void delete(int n,int \*a,int l)

{

int i,j;

if(l<=n)

{

for(i=l-1;i<n;i++)

{

j=i+1;

\*(a+i)=\*(a+j);

}

printf(“Array after deletion is:\n”);

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

{

printf(“%d\n”,(\*(a+i)));

}

}

else

{

printf(“Invalid Input”);

}

}