1. **Write a program to implement linear search.**

#include <stdio.h>

int main() {

int a[100];

int n,i;

printf("Enter the limit: ");

scanf("%d",&n);

printf("Enter the element: ");

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

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

}

printf("\nList is\n");

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

printf("%d ",a[i]);

}

int value,f;

printf("\nEnter the element to be searched: ");

scanf("%d",&value);

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

if(a[i]==value) {

printf("\nPosition = %d",i+1);

f=1;

}

}

if(f==0) {

printf("Value not in the list");

}

return 0;

}

**Output:**

Enter the limit: 3

Enter the element: 1022

36

20List is

1022 36 20

Enter the element to be searched: 36

Position = 2

1. **Write a program to implement binary search.**

#include<stdio.h>

int main() {

int a[100];

int i,n,j,t,low,high,mid,f=0,key;

printf("Enter the limit: ");

scanf("%d",&n);

printf("Enter elements: ");

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

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

}

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

for(j=i+1;j<n;j++) {

if(a[i]>a[j]) {

t=a[i];

a[i]=a[j];

a[j]=t;

}

}

}

printf("\nSorted list is: ");

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

printf("%d ",a[i]);

}

printf("\nEnter the element to be searched \n");

scanf("%d",&key);

low=0;

high=n-1;

mid=(low+high)/2;

while(low<=high){

if(a[mid]==key) {

printf("\nPosition =%d",mid+1);

f=1;

break;

}

else if(a[mid]>key) {

high=mid-1;

}

else if(a[mid]<key) {

low=mid+1;

}

mid=(low+high)/2;

}

if(f==0) {

printf("not found");

}

}

**Output:**

Enter elements: 22

33

11

66

44

Sorted list is: 11 22 33 44 66

Enter the element to be searched

22

Position =2

1. **Write a program to implement interpolation search.**

#include<stdio.h>

int main() {

int a[100];int n,i;

printf("Enter limit: ");

scanf("%d",&n);

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

printf("\nEnter the elements: ");

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

}

int key;

printf("\nEnter element to be searched: ");

scanf("%d",&key);

int low,high,mid,f=0;

while(low<=high) {

mid= low+(high-low)\*((key-a[low])/(a[high]-a[low]));

if(key==a[mid]){

printf("Position: %d",mid+1);

f=1;

break;

}

else if(key<a[mid]) {

high=mid-1;

}

else if(key>a[mid]) {

low=mid+1;

}

if(f==0) {

printf("\nNot found");

}

}

}

**Output:**

Enter limit: 3

Enter the elements: 65

Enter the elements: 12

Enter the elements: 32Enter element to be searched: 32

Position = 3

1. **Write a program to search a number occurring odd number of times.**

#include<stdio.h>

int main() {

int a[100];

int n,i,c=0;

printf("Enter limit: ");

scanf("%d",&n);

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

printf("\nEnter the elements: ");

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

}

int key;

printf("\nEnter element to be searched: ");

scanf("%d",&key);

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

if(a[i]==key) {

c++;

}

}

if(c%2!=0) {

printf("\nNumber of times: %d",c);

}

else {

printf("\nThere is no odd count");

}

}

**Output :**

Enter limit: 5

Enter the elements: 22Enter the elements: 33

Enter the elements: 44

Enter the elements: 33

Enter the elements: 33

Enter element to be searched: 33

Number of times: 3

**5.Write a program to implement bubble sort.**

#include<stdio.h>

int main() {

int a[100];

int n,i,c=0;

printf("Enter limit: ");

scanf("%d",&n);

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

printf("\nEnter the elements: ");

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

}

printf("\nList is\n");

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

printf("%d ",a[i]);

}

int j,t,k;

for(i=1;i<n;i++){

for(j=0;j<i;j++) {

if(a[j]>a[i]) {

t=a[j];

a[j]=a[i];

for(k=i;k>j;k--) {

a[k]=a[k-1];

}

a[k+1]=t;

}

}

}

printf("\nSorted List is\n");

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

printf("%d ",a[i]);

}

}

**Output :**

Enter limit: 5

Enter the elements: 45

Enter the elements: 66

Enter the elements: 10

Enter the elements: 32

Enter the elements: 20

List is

45 66 10 32 20

Sorted List is

10 20 32 45 66

**6.Write a program to implement selection sort.**

#include<stdio.h>

int main() {

int a[100];

int n,i,c=0;

printf("Enter limit: ");

scanf("%d",&n);

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

printf("\nEnter the elements: ");

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

}

printf("\nList is\n");

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

printf("%d ",a[i]);

}

int j,t;

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

for(j=i+1;j<n;j++) {

if(a[j]<a[j+1]) {

t=a[i];

a[i]=a[j];

a[j]=t;

}

}

}printf("\nSorted List is\n");

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

printf("%d ",a[i]);

}

}

**Output:**

Enter limit: 5

Enter the elements: 66

Enter the elements: 44

Enter the elements: 11

Enter the elements: 33

Enter the elements: 22

List is

66 44 11 33 22

Sorted List is

11 44 66 33 22

**8.Write a program to implement merge sort.**

#include <stdio.h>

void merge(int a[], int beg, int mid, int end) {

int i, j, k;int n1 = mid - beg + 1;

int n2 = end - mid;

int LeftArray[n1], RightArray[n2];

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

LeftArray[i] = a[beg + i];

for (int j = 0; j < n2; j++)

RightArray[j] = a[mid + 1 + j];

i = 0;

j = 0;

k = beg;

while (i < n1 && j < n2) {

if(LeftArray[i] <= RightArray[j]) {

a[k] = LeftArray[i];

i++;

}

else {

a[k] = RightArray[j];

j++;

}

k++;

}

while (i<n1) {

a[k] = LeftArray[i];

i++;

k++;

}

while (j<n2) {

a[k] = RightArray[j];

j++;

k++;

}

}void mergeSort(int a[], int beg, int end) {

if (beg < end) {

int mid = (beg + end) / 2;

mergeSort(a, beg, mid);

mergeSort(a, mid + 1, end);

merge(a, beg, mid, end);

}

}

void printArray(int a[], int n) {

int i;

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

printf("%d ", a[i]);

printf("\n");

}

int main() {

int a[] = { 12, 31, 25, 8, 32, 17, 40, 42 };

int n = sizeof(a) / sizeof(a[0]);

printf("Before sorting array elements are - \n");

printArray(a, n);

mergeSort(a, 0, n - 1);

printf("After sorting array elements are - \n");

printArray(a, n);

return 0;

}

**Output:**

Before sorting array elements are -

12 31 25 8 32 17 40 42

After sorting array elements are -

8 12 17 25 31 32 40 42

**9.Write a program to implement quick sort.**

#include <stdio.h>

void swap(int \*a, int \*b) {

int t = \*a;

\*a = \*b;

\*b = t;

}

int partition(int array[], int low, int high) {int pivot = array[high];

int i = (low - 1);

for (int j = low; j < high; j++) {

if (array[j] <= pivot) {

i++;

swap(&array[i], &array[j]);

}

}

swap(&array[i + 1], &array[high]);

return (i + 1);

}

void quickSort(int array[], int low, int high) {

if (low < high) {

int pi = partition(array, low, high);

quickSort(array, low, pi - 1);

quickSort(array, pi + 1, high);

}

}

void printArray(int array[], int size) {

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

printf("%d ", array[i]);

}

printf("\n");

}

int main() {

int data[] = {8, 7, 2, 1, 0, 9, 6};

int n = sizeof(data) / sizeof(data[0]);

printf("Unsorted Array\n");

printArray(data, n);

quickSort(data, 0, n - 1);

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

printArray(data, n);

}

**Output:**

Unsorted Array

8 7 2 1 0 9 6

Sorted array in ascending order:

0 1 2 6 7 8 9

**10. Write a program to find three elements in the array such that their sum is equal to given**

**element K.**

#include<stdio.h>

int main() {

int a[100];

int n,i;

printf("Enter limit: ");

scanf("%d",&n);

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

printf("Enter elements: ");

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

}

printf("\nList is:\n");

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

printf("%d ",a[i]);

}

int j,k,m;

printf("\nnter the value of k: ");

scanf("%d",&k);

printf("\nSum is\n");

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

for(j=i+1;j<n;j++) {

for(m=j+1;m<n;k++) {

if(a[i]+a[j]+a[m]==k) {

printf("That three elements that their sum is equal to the given element K\n%d, %d,

%d",a[i],a[j],a[m]);

break;

}

}

}

}

}

**Output:**

Enter limit: 5

Enter elements: 1

Enter elements: 2

Enter elements: 3

Enter elements: 4

Enter elements: 5

List is:

1 2 3 4 5

Enter the value of k: 5

That three elements that their sum is equal to the given element K

1 ,2 , 3