**ALGORITHMS:**

**Searching Algorithms:**

Searching algorithms are designed to check for an element or retrieve an element from any data structure where it is stored.

These are two types:

1. Sequential search
2. Interval Search---user for searching in sorted data structures.

**Ex:** binary search

LINEAR SEARCH:

**BINARY SEARCH:**

1.Iterative method

EX:

#include <stdio.h>

int binary\_search(int arr[], int n, int x ) {

int low = 0;

int high = n - 1;

while (low <= high) {

int mid = (low + high) / 2;

if (arr[mid] == x) {

return mid;

}

else if (x>arr[mid]) {

low=mid+1;

}

else {

high = mid - 1;

}

}

return -1;

}

int main() {

int arr[] = {1, 3, 5, 7, 9, 11, 13, 15, 17, 19};

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

int x;

printf("Enter the value to search:");

scanf("%d",&x);

int result = binary\_search(arr, n, x);

if (result != -1)

{

printf("Element %d is found at index %d.\n", x, result);

} else

{

printf("Element %d is not found in the array.\n", x);

}

return 0;

}

2.Recursive method

EX:

**BUBBLE SORT:**

Bubble sort is also referred as sinking sort.

We repeatedly compare each pair of adjacent items and swap them if they are in the wrong order.

After the bubble sort the last element is in the correct position.

**Selection Sort:**

#include <stdio.h>

int selectionSort(int arr[], int n) {

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

{

int min=i;

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

if (arr[j]>min) {

min=arr[j];

}

}

}

}

int main() {

int arr[] = {1, 3, 5, 7, 9, 11, 13, 15, 17, 19};

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

selectionSort(arr, n);

printf("Sorted array: ");

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

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

}

printf("\n");

return 0;

}

**write a program to merger two arrays to form a sorted list?**

#include <stdio.h>

int sort(int final[], int size)

{

for (int i = 0; i < size - 1; i++)

{

for (int j = 0; j < size - i - 1; j++)

{

if (final[j] > final[j + 1])

{

int temp = final[j];

final[j] = final[j + 1];

final[j + 1] = temp;

}

}

}

}

int main() {

int arr1[] = {1, 3, 2, 5, 4};

int size1 = sizeof(arr1) / sizeof(arr1[0]);

int arr2[] = {6, 9, 7, 10, 8};

int size2 = sizeof(arr2) / sizeof(arr2[0]);

int size = size1 + size2;

int final[size];

for (int i = 0; i < size1; i++) {

final[i] = arr1[i];

}

for (int j = 0; j < size2; j++) {

final[size1 + j] = arr2[j];

}

sort(final, size);

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

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

}

return 0;

}