Passing Array to Function, Searching and Sorting

1. Searching Algorithms

Linear Search

In a linear search, we go through each element of the array one by one to find the target value.

Example Code:

```
#include <stdio.h>
int linearSearch(int arr[], int size, int target) {
   for (int i = 0; i < size; i++) {
      if (arr[i] == target)
         return i;
   }
   return -1;
}
int main() {
   int arr[] = {2, 4, 6, 8, 10};
   int target = 6;
   int size = sizeof(arr) / sizeof(arr[0]);
   int result = linearSearch(arr, size, target);
   if (result != -1)
      printf("Element found at index: %d\n", result);
   else
      printf("Element not found\n");
   return 0;
}
```

Binary Search

Binary search works on sorted arrays. It repeatedly divides the array in half to find the target.

Example Code:

```
#include <stdio.h>
int binarySearch(int arr[], int size, int target) {
   int low = 0, high = size - 1;
```

```
    while (low <= high) {
        int mid = (low + high) / 2;
        if (arr[mid] == target)
            return mid;
        else if (arr[mid] < target)
            low = mid + 1;
        else
            high = mid - 1;
    }
    return -1;
}
int main() {
    int arr[] = {1, 3, 5, 7, 9};
    int target = 5;
    int size = sizeof(arr) / sizeof(arr[0]);
    int result = binarySearch(arr, size, target);

    if (result != -1)
        printf("Element found at index: %d\n", result);
    else
        printf("Element not found\n");
    return 0;
}
```

2. Sorting Algorithms

Bubble Sort

Bubble sort compares adjacent elements and swaps them if they are in the wrong order. It repeatedly passes through the array.

Example Code:

```
#include <stdio.h>
void bubbleSort(int arr[], int size) {
    for (int i = 0; i < size - 1; i++) {
```

```
        for (int j = 0; j < size - i - 1; j++) {
            if (arr[j] > arr[j + 1]) {
                // Swap elements
                int temp = arr[j];
                arr[j] = arr[j + 1];
                arr[j + 1] = temp;
            }
        }
    }
}
int main() {
    int arr[] = {64, 34, 25, 12, 22};
    int size = sizeof(arr) / sizeof(arr[0]);
    bubbleSort(arr, size);
    printf("Sorted array: ");
    for (int i = 0; i < size; i++)
        printf("%d ", arr[i]);
    return 0;
}
```

Selection Sort

Selection sort repeatedly finds the minimum element from the unsorted part of the array and places it at the beginning.

Example Code:

```
#include <stdio.h>
void selectionSort(int arr[], int size) {
    for (int i = 0; i < size - 1; i++) {
        int minIdx = i;
        for (int j = i + 1; j < size; j++) {
            if (arr[j] < arr[minIdx])
                minIdx = j;
        }
```

```
        int temp = arr[minIdx];
        arr[minIdx] = arr[i];
        arr[i] = temp;
    }
}
int main() {
    int arr[] = {29, 10, 14, 37, 13};
    int size = sizeof(arr) / sizeof(arr[0]);
    selectionSort(arr, size);
    printf("Sorted array: ");
    for (int i = 0; i < size; i++)
        printf("%d ", arr[i]);
    return 0;
}
```

3.Passing Array and Single Value to Functions

Passing a Single Value

You can pass individual values (e.g., integers) directly to a function.

Example Code:

```
#include <stdio.h>
void displayValue(int num) {
    printf("The value is: %d\n", num);
}
int main() {
    int value = 5;
    displayValue(value);  // Pass single value to the function
    return 0;
}
```

Passing an Entire Array

Example Code:

```
#include <stdio.h>
void displayArray(int arr[], int size) {
```

```
    printf("Array elements: ");
    for (int i = 0; i < size; i++)
        printf("%d ", arr[i]);
    printf("\n");
}
int main() {
    int arr[] = {10, 20, 30, 40};
    int size = sizeof(arr) / sizeof(arr[0]);
    displayArray(arr, size);  // Pass array to the function
    return 0;
}
```