	PUNE INSTITUTE OF COMPUTER TECHNOLOGY PUNE - 411043	
	Department of Electronics & Telecommunication	
	ASSESMENT YEAR: 2024-2025	CLASS: SE
	SUBJECT: DATA STRUCTURES	
EXPT No:	LAB Ref: SE/2024-25/	Starting date:
	Roll No: 22181	Submission date:

Title:	**Array Operations**
Problem Statement	Perform various operations on array such as 1. Create 2. Display, 3. Sort, 4. Search, 5. Modify, 6. Insert, 7. Delete. A. Without Pointers B. using pointer (Beyond Syllabus)

Programmer Name: Toshit Warke
Batch: H-5

Code:

```
#include <stdio.h>

void insert(int arr[], int n) {
   int pos, element;
   printf("Enter the position where the element must be inserted (1 to %d): ", n + 1);
   scanf("%d", &pos);

   if (pos > n + 1 || pos < 1) {
      printf("Invalid position!\n");
      return;
   }

   printf("Enter the element to be inserted: ");
   scanf("%d", &element);

   for(int k = n; k >= pos; k--) {
      arr[k] = arr[k-1];
   }

   arr[pos - 1] = element;

   printf("Array after insertion:\n");
   for(int m = 0; m <= n; m++) {
      printf("%d ", arr[m]);
```

```
    }
    printf("\n");
}

void delete(int arr[], int n) {
    int pos;
    printf("Enter the position of the element to be deleted (1 to %d): ", n);
    scanf("%d", &pos);

    if (pos > n || pos < 1) {
        printf("Invalid position!\n");
        return;
    }

    for(int k = pos - 1; k < n - 1; k++) {
        arr[k] = arr[k+1];
    }

    printf("Array after deletion:\n");
    for(int m = 0; m < n - 1; m++) {
        printf("%d ", arr[m]);
    }
    printf("\n");
}

void modify(int arr[], int n) {
    int pos, newValue;
    printf("Enter the position of the element to be modified (1 to %d): ", n);
    scanf("%d", &pos);

    if (pos > n || pos < 1) {
        printf("Invalid position!\n");
        return;
    }

    printf("Enter the new value: ");
```

```
    scanf("%d", &newValue);

    arr[pos - 1] = newValue;

    printf("Array after modification:\n");
    for(int m = 0; m < n; m++) {
        printf("%d ", arr[m]);
    }
    printf("\n");
}

void bubbleSort(int arr[], int n) {
    for (int i = 0; i < n-1; i++) {
        printf("Pass %d: ", i+1);
        for (int j = 0; j < n-i-1; j++) {
            if (arr[j] > arr[j+1]) {
                int temp = arr[j];
                arr[j] = arr[j+1];
                arr[j+1] = temp;
            }
        }
        for (int k = 0; k < n; k++)
            printf("%d ", arr[k]);
        printf("\n");
    }
    printf("\nSorted array: ");
    for (int i = 0; i < n; i++)
        printf("%d ", arr[i]);
    printf("\n");
}

void insertionSort(int arr[], int n) {
    for (int i = 1; i < n; i++) {
        int key = arr[i];
        int j = i - 1;
        while (j >= 0 && arr[j] > key) {
```

```
            arr[j + 1] = arr[j];
            j--;
        }
        arr[j + 1] = key;
        printf("Pass %d: ", i);
        for (int k = 0; k < n; k++)
            printf("%d ", arr[k]);
        printf("\n");
    }
    printf("\nSorted array: ");
    for (int i = 0; i < n; i++)
        printf("%d ", arr[i]);
    printf("\n");
}

void selectionSort(int arr[], int n) {
    for (int i = 0; i < n-1; i++) {
        int min_idx = i;
        for (int j = i+1; j < n; j++) {
            if (arr[j] < arr[min_idx])
                min_idx = j;
        }
        int temp = arr[min_idx];
        arr[min_idx] = arr[i];
        arr[i] = temp;
        printf("Pass %d: ", i+1);
        for (int k = 0; k < n; k++)
            printf("%d ", arr[k]);
        printf("\n");
    }
    printf("\nSorted array: ");
    for (int i = 0; i < n; i++)
        printf("%d ", arr[i]);
    printf("\n");
}
```

```
int linearSearch(int arr[], int n, int key) {
   for(int i = 0; i < n; i++) {
      if(arr[i] == key) {
         return i;
      }
   }
   return -1;
}

int binarySearch(int arr[], int n, int key) {
   int left = 0, right = n - 1;

   while(left <= right) {
      int mid = left + (right - left) / 2;

      if(arr[mid] == key) {
         return mid;
      }
      if(arr[mid] < key) {
         left = mid + 1;
      } else {
         right = mid - 1;
      }
   }
   return -1;
}

int main() {
   int n;
   printf("Enter the length of the array: ");
   scanf("%d", &n);
   int arr[100];
   int original_arr[100];
   for(int i = 0; i < n; i++) {
      printf("Enter integer %d in the array: ", i + 1);
      scanf("%d", &arr[i]);
```

```
    original_arr[i] = arr[i];
}

int choice = 0;
int key, result;

while(choice != 9) {
    for(int i = 0; i < n; i++) {
        arr[i] = original_arr[i];
    }

    printf("\nYour array is:\n");
    for(int j = 0; j < n; j++) {
        printf("%d ", arr[j]);
    }
    printf("\n\nSelect an operation to perform:\n");
    printf("1) Insert an integer\n");
    printf("2) Delete an integer\n");
    printf("3) Modify an integer\n");
    printf("4) Bubble Sort\n");
    printf("5) Insertion Sort\n");
    printf("6) Selection Sort\n");
    printf("7) Linear Search\n");
    printf("8) Binary Search\n");
    printf("9) Exit\n");
    printf("Enter your choice: ");
    scanf("%d", &choice);

    if(choice == 1) {
        insert(arr, n);
    } else if(choice == 2) {
        delete(arr, n);
    } else if(choice == 3) {
        modify(arr, n);
    } else if(choice == 4) {
        bubbleSort(arr, n);
```

```
        } else if(choice == 5) {
            insertionSort(arr, n);
        } else if(choice == 6) {
            selectionSort(arr, n);
        } else if(choice == 7) {
            printf("Enter the element to search: ");
            scanf("%d", &key);
            result = linearSearch(arr, n, key);
            if(result != -1) {
                printf("Element found at index %d using linear search.\n", result);
            } else {
                printf("Element not found using linear search.\n");
            }
        } else if(choice == 8) {

            insertionSort(arr, n);
            printf("Enter the element to search: ");
            scanf("%d", &key);
            result = binarySearch(arr, n, key);
            if(result != -1) {
                printf("Element found at index %d using binary search.\n", result);
            } else {
                printf("Element not found using binary search.\n");
            }
        } else if(choice == 9) {
            printf("Exiting program.\n");
        } else {
            printf("Invalid choice! Please select a valid operation.\n");
        }
    }

    return 0;
}
```

Output:

Enter the length of the array: 5
Enter integer 1 in the array: 7
Enter integer 2 in the array: 6
Enter integer 3 in the array: 5
Enter integer 4 in the array: 4
Enter integer 5 in the array: 3

Your array is:
7 6 5 4 3

Select an operation to perform:
1) Insert an integer
2) Delete an integer
3) Modify an integer
4) Bubble Sort
5) Insertion Sort
6) Selection Sort
7) Linear Search
8) Binary Search
9) Exit
Enter your choice: 1
Enter the position where the element must be inserted (1 to 6): 3
Enter the element to be inserted: 81
Array after insertion:
7 6 81 5 4 3

Your array is:
7 6 5 4 3

Select an operation to perform:
1) Insert an integer
2) Delete an integer
3) Modify an integer
4) Bubble Sort
5) Insertion Sort

6) Selection Sort
7) Linear Search
8) Binary Search
9) Exit
Enter your choice: 2
Enter the position of the element to be deleted (1 to 5): 3
Array after deletion:
7 6 4 3

Your array is:
7 6 5 4 3

Select an operation to perform:
1) Insert an integer
2) Delete an integer
3) Modify an integer
4) Bubble Sort
5) Insertion Sort
6) Selection Sort
7) Linear Search
8) Binary Search
9) Exit
Enter your choice: 3
Enter the position of the element to be modified (1 to 5): 3
Enter the new value: 81
Array after modification:
7 6 81 4 3

Your array is:
7 6 5 4 3

Select an operation to perform:
1) Insert an integer
2) Delete an integer
3) Modify an integer
4) Bubble Sort

5) Insertion Sort
6) Selection Sort
7) Linear Search
8) Binary Search
9) Exit
Enter your choice: 4
Pass 1: 6 5 4 3 7
Pass 2: 5 4 3 6 7
Pass 3: 4 3 5 6 7
Pass 4: 3 4 5 6 7

Sorted array: 3 4 5 6 7

Your array is:
7 6 5 4 3

Select an operation to perform:
1) Insert an integer
2) Delete an integer
3) Modify an integer
4) Bubble Sort
5) Insertion Sort
6) Selection Sort
7) Linear Search
8) Binary Search
9) Exit
Enter your choice: 5
Pass 1: 6 7 5 4 3
Pass 2: 5 6 7 4 3
Pass 3: 4 5 6 7 3
Pass 4: 3 4 5 6 7

Sorted array: 3 4 5 6 7

Your array is:
7 6 5 4 3

Select an operation to perform:
1) Insert an integer
2) Delete an integer
3) Modify an integer
4) Bubble Sort
5) Insertion Sort
6) Selection Sort
7) Linear Search
8) Binary Search
9) Exit
Enter your choice: 6
Pass 1: 3 6 5 4 7
Pass 2: 3 4 5 6 7
Pass 3: 3 4 5 6 7
Pass 4: 3 4 5 6 7

Sorted array: 3 4 5 6 7

Your array is:
7 6 5 4 3

Select an operation to perform:
1) Insert an integer
2) Delete an integer
3) Modify an integer
4) Bubble Sort
5) Insertion Sort
6) Selection Sort
7) Linear Search
8) Binary Search
9) Exit
Enter your choice: 7
Enter the element to search: 6
Element found at index 1 using linear search.

Your array is:
7 6 5 4 3

Select an operation to perform:
1) Insert an integer
2) Delete an integer
3) Modify an integer
4) Bubble Sort
5) Insertion Sort
6) Selection Sort
7) Linear Search
8) Binary Search
9) Exit
Enter your choice: 8
Pass 1: 6 7 5 4 3
Pass 2: 5 6 7 4 3
Pass 3: 4 5 6 7 3
Pass 4: 3 4 5 6 7

Sorted array: 3 4 5 6 7
Enter the element to search: 7
Element found at index 4 using binary search.