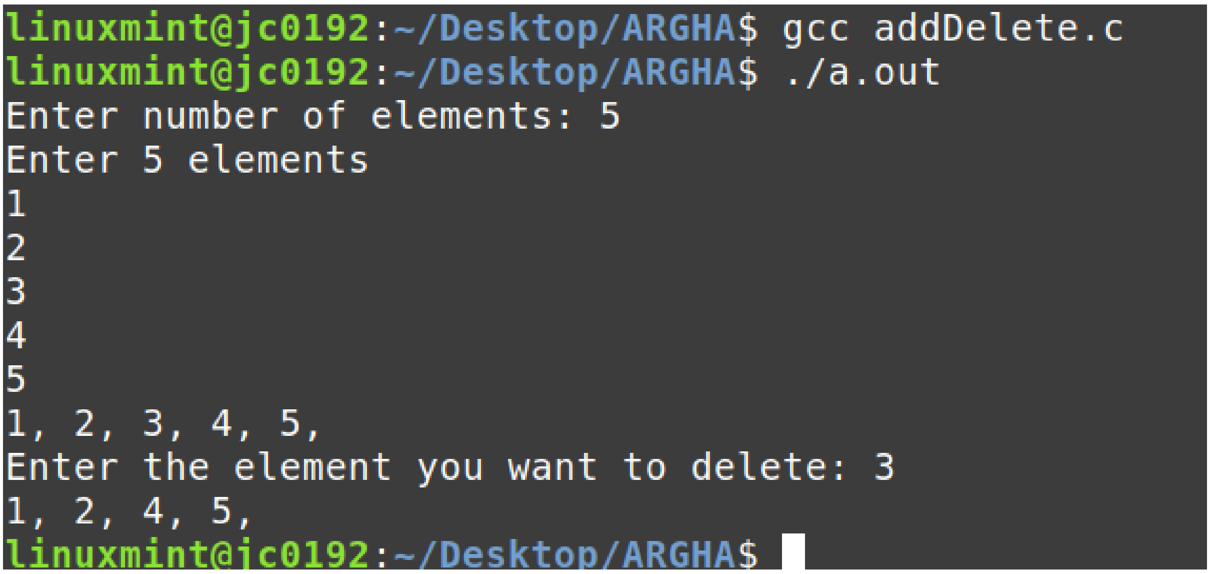
LAB 1 ASSIGNMENT – ARRAY

ARGHA MALLICK - 11500122014

1. WACP to insert and delete elements from a 1D array

```
#include<stdio.h>
void printArray(int arr[], int n){
    for(int i=0; i<n; i++) {
         printf("%d, ", arr[i]);
    }
        printf("\n");
}
void deleteElement(int arr[], int n, int d) {
        for(int i=0; i<n; i++){
                if(arr[i] == d) {
                        for(int j=i; j<n-1; j++) {
                                arr[j] = arr[j+1];
                        }
                }
        }
}
int main() {
        int n, d;
        printf("Enter number of elements: ");
        scanf("%d", &n);
        int arr[n];
        printf("Enter %d elements\n", n);
        for(int i=0; i<n; i++) {
                scanf("%d", &arr[i]);
        }
        printArray(arr, n);
```

```
            printf("Enter the element you want to delete: ");

            scanf("%d", &d);

            deleteElement(arr, n, d);

            printArray(arr, n-1);

}
```

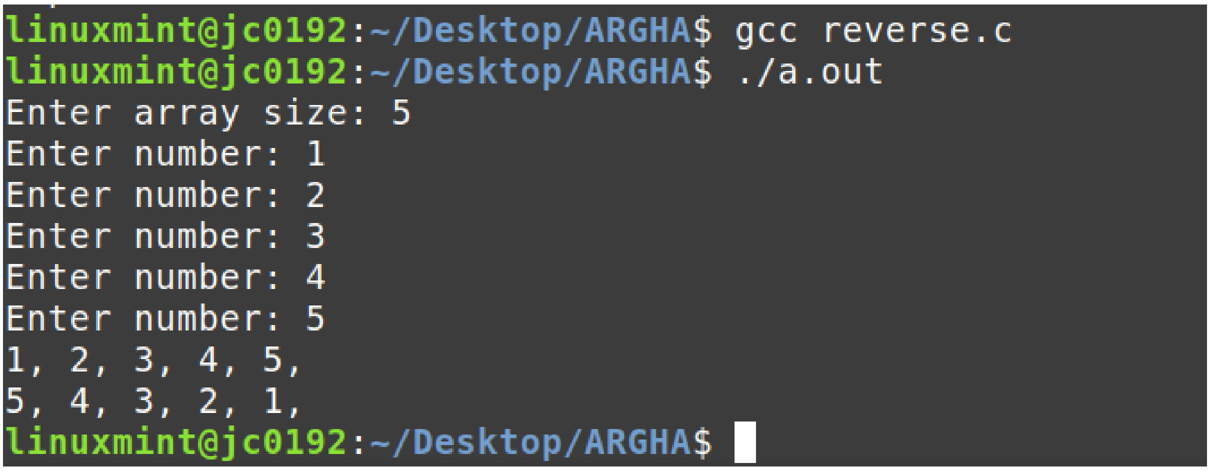


```
linuxmint@jc0192:~/Desktop/ARGHA$ gcc addDelete.c
linuxmint@jc0192:~/Desktop/ARGHA$ ./a.out
Enter number of elements: 5
Enter 5 elements
1
2
3
4
5
1, 2, 3, 4, 5,
Enter the element you want to delete: 3
1, 2, 4, 5,
linuxmint@jc0192:~/Desktop/ARGHA$
```

2. Reverse the elements in an array

```
#include<stdio.h>

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

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

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

       }

       printf("\n");

}


int main(){

            int n;

            printf("Enter array size: ");

            scanf("%d",&n);

            int arr[n];
```

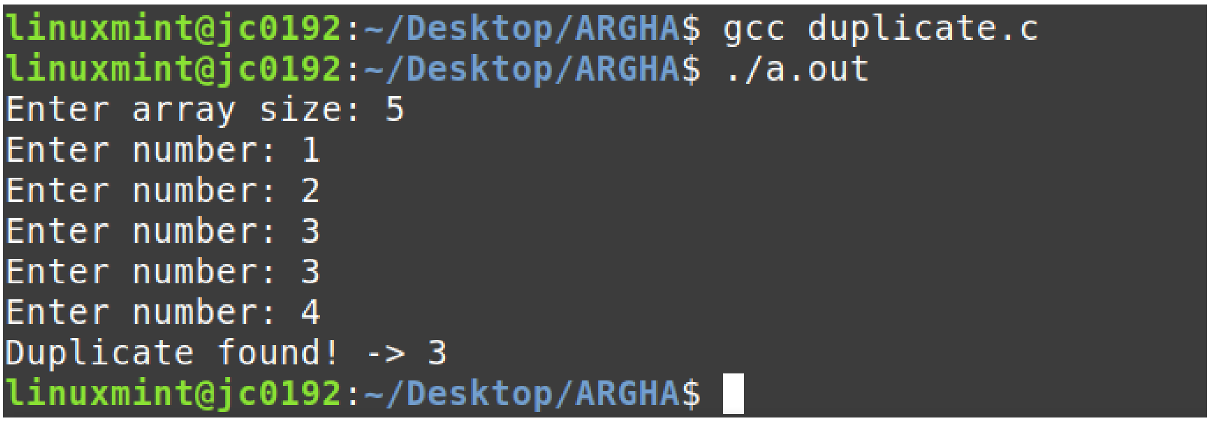
```
        for(int i=0; i<n; i++) {
                printf("Enter number: ");
                scanf("%d",&arr[i]);
        }
        printArray(arr, n);
        for(int i=0; i<n/2; i++) {
                int temp = arr[i];
                arr[i] = arr[n-i-1];
                arr[n-i-1] = temp;
        }
        printArray(arr, n);
}
```



```
linuxmint@jc0192:~/Desktop/ARGHA$ gcc reverse.c
linuxmint@jc0192:~/Desktop/ARGHA$ ./a.out
Enter array size: 5
Enter number: 1
Enter number: 2
Enter number: 3
Enter number: 4
Enter number: 5
1, 2, 3, 4, 5,
5, 4, 3, 2, 1,
linuxmint@jc0192:~/Desktop/ARGHA$ 
```

3. Delete duplicate elements in an array

```
#include<stdio.h>
int main() {
        int n;
    printf("Enter array size: ");
    scanf("%d",&n);
    int arr[n];
    for(int i=0; i<n; i++) {
        printf("Enter number: ");

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

    }

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

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

                     if(arr[i] == arr[j]){

                            printf("Duplicate found! -> %d\n", arr[j]);

                     }

              }

       }

}
```

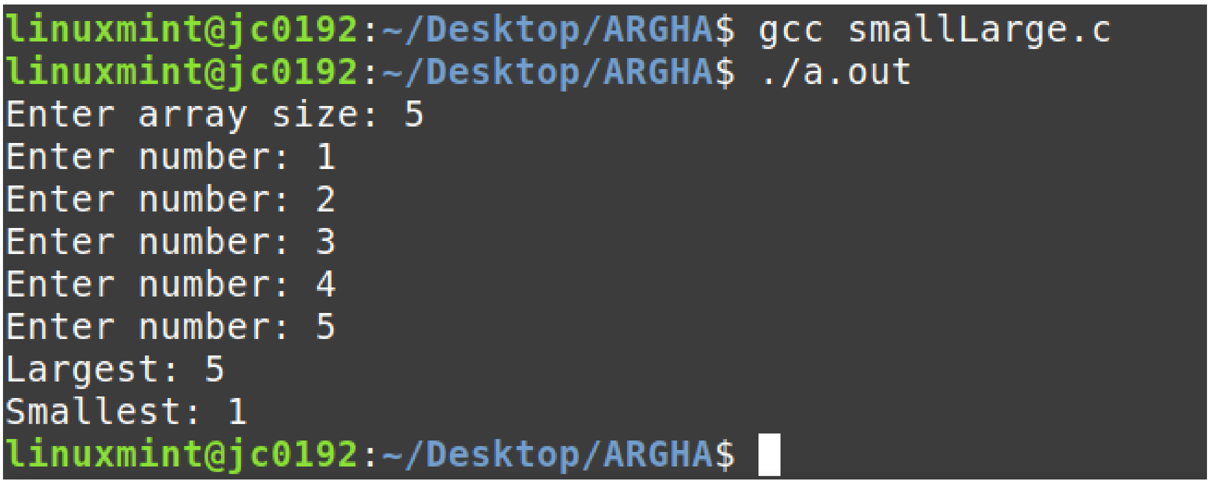


```
linuxmint@jc0192:~/Desktop/ARGHA$ gcc duplicate.c
linuxmint@jc0192:~/Desktop/ARGHA$ ./a.out
Enter array size: 5
Enter number: 1
Enter number: 2
Enter number: 3
Enter number: 3
Enter number: 4
Duplicate found! -> 3
linuxmint@jc0192:~/Desktop/ARGHA$ 
```

4. Find the largest and smallest element in an array

```
#include<stdio.h>

int main() {

    int n;

    printf("Enter array size: ");

    scanf("%d",&n);

    int arr[n];

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

        printf("Enter number: ");

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

    }
```

```
        int small = arr[0];

        int large = arr[0];

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

                small = arr[i] < small ? arr[i] : small;

                large = arr[i] > large ? arr[i] : large;

        }

        printf("Largest: %d\n", large);

        printf("Smallest: %d\n", small);

}
```

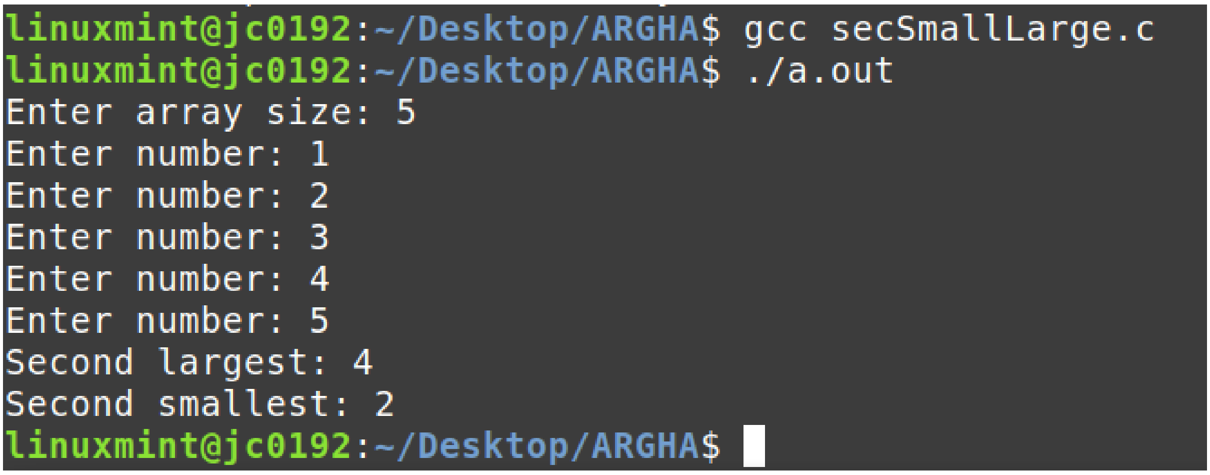


```
linuxmint@jc0192:~/Desktop/ARGHA$ gcc smallLarge.c
linuxmint@jc0192:~/Desktop/ARGHA$ ./a.out
Enter array size: 5
Enter number: 1
Enter number: 2
Enter number: 3
Enter number: 4
Enter number: 5
Largest: 5
Smallest: 1
linuxmint@jc0192:~/Desktop/ARGHA$ 
```

5. Find the second largest and second smallest element in an array

```
#include<stdio.h>

void bubbleSort(int arr[], int n) {

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

                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;

                        }
```

```
                    }
            }
}
int main() {
    int n;
    printf("Enter array size: ");
    scanf("%d",&n);
    int arr[n];
    for(int i=0; i<n; i++) {
        printf("Enter number: ");
        scanf("%d",&arr[i]);
    }
        bubbleSort(arr, n);
        printf("Second largest: %d\n", arr[n-2]);
        printf("Second smallest: %d\n", arr[1]);
}
```

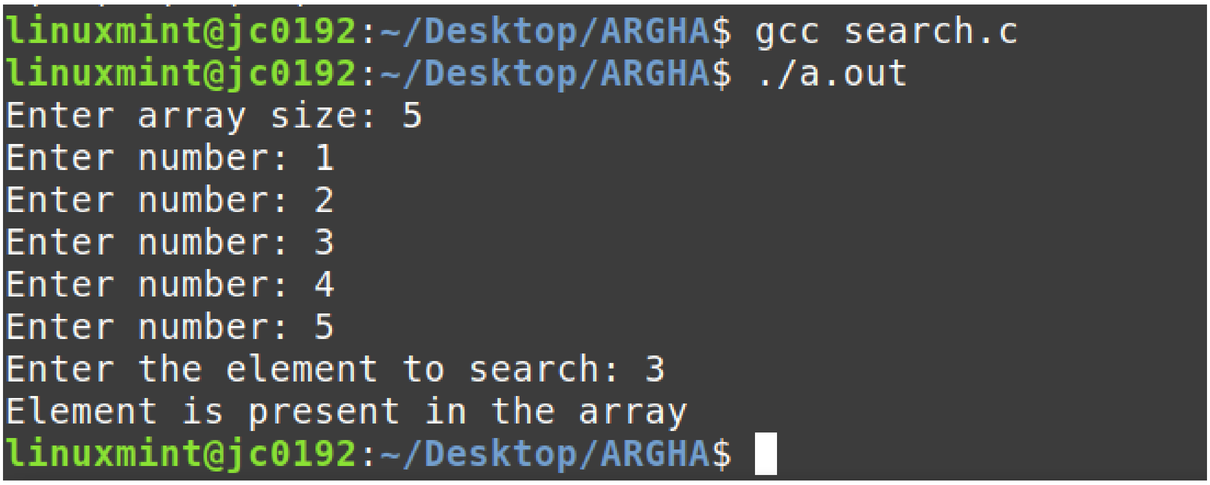


```
linuxmint@jc0192:~/Desktop/ARGHA$ gcc secSmallLarge.c
linuxmint@jc0192:~/Desktop/ARGHA$ ./a.out
Enter array size: 5
Enter number: 1
Enter number: 2
Enter number: 3
Enter number: 4
Enter number: 5
Second largest: 4
Second smallest: 2
linuxmint@jc0192:~/Desktop/ARGHA$ 
```

6. Implement linear and binary search - iterative and recursive

```
#include<stdio.h>
void linearSearch(int arr[], int n, int key){
        for(int i=0; i<n; i++){
                if(arr[i] == key) {
                        printf("Element is present in the array\n");
                        return;
                }
        }
        printf("Element is not present in the array!\n");
}
void RLinearSearch(int arr[], int n, int i, int key) {
        if(i == n) {
                printf("Element is not present in the array!\n");
                return;
        }
        if(arr[i] == key) {
                printf("Element is present in the array\n");
                return;
        } else RLinearSearch(arr, n, i+1, key);
}
void binarySearch(int arr[], int left, int right, int key){
        while(left <= right) {
                int mid = left + (right-1) / 2;
                if(arr[mid] == key) { printf("Element is present in the array\n"); return; }
                if(key > arr[mid]) left=mid+1;
                else right=mid-1;
        }
        printf("Element is not present in the array!\n");
        return;
}
void RBinarySearch(int arr[], int left, int right, int key){
        if(left <= right) {
                int mid = left + (right-1) / 2;
                if(key == arr[mid]) {
```

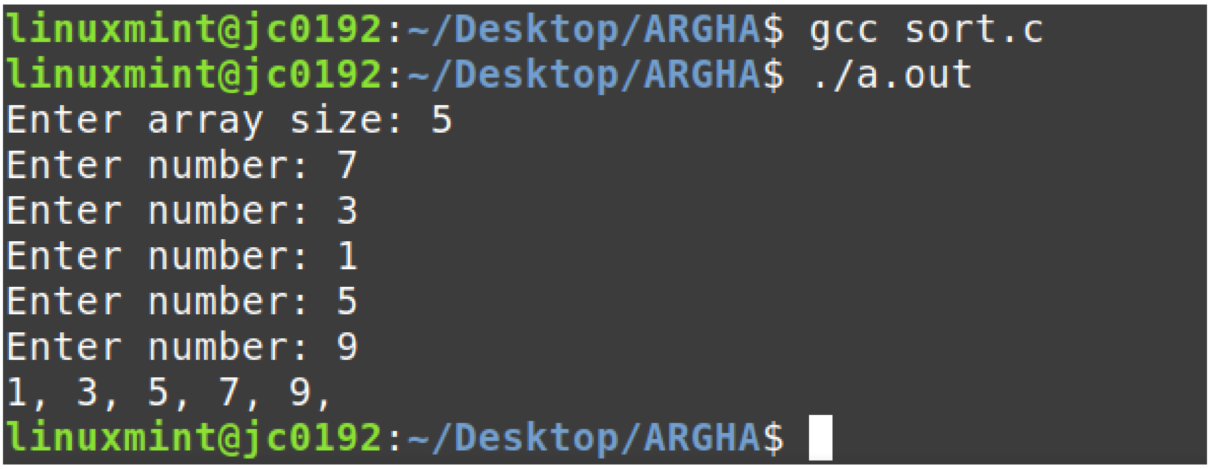
```
                printf("Element is present in the array\n");
                        return;
                }
                if(key < arr[mid]) return RBinarySearch(arr, left, mid-1, key);
                else return RBinarySearch(arr, mid+1, right, key);
        }
        printf("Element is not present in the array!\n");
    return;
}
int main() {
    int n, key;
    printf("Enter array size: ");
    scanf("%d",&n);
    int arr[n];
    for(int i=0; i<n; i++) {
         printf("Enter number: ");
         scanf("%d",&arr[i]);
    }
        printf("Enter the element to search: ");
        scanf("%d", &key);
        // linearSearch(arr, n, key);
        // binarySearch(arr, 0, n-1, key);
        // RLinearSearch(arr, n, 0, key);
        RBinarySearch(arr, 0, n-1, key);
}
```



```
linuxmint@jc0192:~/Desktop/ARGHA$ gcc search.c
linuxmint@jc0192:~/Desktop/ARGHA$ ./a.out
Enter array size: 5
Enter number: 1
Enter number: 2
Enter number: 3
Enter number: 4
Enter number: 5
Enter the element to search: 3
Element is present in the array
linuxmint@jc0192:~/Desktop/ARGHA$ 
```

7. Implement bubble sort on an array

```
#include<stdio.h>

void bubbleSort(int arr[], int n) {

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

        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;

            }

        }

    }

}

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

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

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

    }

    printf("\n");
}


int main(){
        int n;
        printf("Enter array size: ");
    scanf("%d",&n);
    int arr[n];
    for(int i=0; i<n; i++) {
         printf("Enter number: ");
         scanf("%d",&arr[i]);
    }
        bubbleSort(arr, n);
        printArray(arr, n);
}
```



```
linuxmint@jc0192:~/Desktop/ARGHA$ gcc sort.c
linuxmint@jc0192:~/Desktop/ARGHA$ ./a.out
Enter array size: 5
Enter number: 7
Enter number: 3
Enter number: 1
Enter number: 5
Enter number: 9
1, 3, 5, 7, 9,
linuxmint@jc0192:~/Desktop/ARGHA$
```