Lab Assignment-1

ROLL: 2005535 | NAME: SAHIL SINGH | DATE: 27/07/21

QUES 1: WAP to find out the smallest and largest element stored in an array of n integers

SOLUTION:

```c
#include <stdio.h>
int main()
{
    // NAME: Sahil Singh
    // ROLL: 2005535
    int n, i, small, large;
    printf("Enter the array size: ");
    scanf("%d", &n);

    int array[n];
    for (i = 0; i < n; i++)
    {
        printf("Enter element %d: ", i + 1);
        scanf("%d", &array[i]);
    }
    small = array[0];
    large = array[0];
    for (i = 1; i < n; i++)
    {
        if (array[i] < small)
        {
            small = array[i];
        }
        if (array[i] > large)
        {
            large = array[i];
        }
    }

    printf("\nLargest element is: %d", large);
    printf("\nSmallest element is: %d", small);
    return 0;
}
```

OUTPUT:

```
Enter the array size: 5
Enter element 1: 12
Enter element 2: 10
Enter element 3: 4
Enter element 4: 6
Enter element 5: 38
```

```
Largest element is: 38
Smallest element is: 4
```

QUES 2: WAP to reverse the contents of a dynamic array of n elements.

SOLUTION:

```c
#include <stdio.h>
void reverse(int array[], int start, int end, int size)
{
    // NAME: Sahil Singh
    // ROLL: 2005535
    int temp;
    while (start < end)
    {
        temp = array[start];
        array[start] = array[end];
        array[end] = temp;
        start++;
        end--;
    }
    int i;
    for (i = 0; i < size; i++)
    {
        printf("%d ", array[i]);
    }
    printf("\n");
}
int main()
{
    int array[] = {10, 20, 30, 40, 50, 60, 70, 80, 90, 100};
    int n = sizeof(array) / sizeof(array[0]);
    printf("Reversed array: ");
    reverse(array, 0, n - 1, n);
    return 0;
}
```

OUTPUT:

```
Reversed array: 100 90 80 70 60 50 40 30 20 10
```

QUES 3: WAP to search an element in a dynamic array of n numbers.

SOLUTION:

```c
#include <stdio.h>
int main()
{
    // NAME: Sahil Singh
    // ROLL: 2005535
    int i, n, key;
    printf("Enter size of the array: ");
    scanf("%d", &n);

    int a[n];
    printf("Enter elements in array: ");
    for (i = 0; i < n; i++)
    {
        scanf("%d", &a[i]);
    }
    printf("Enter the element to search for: ");
    scanf("%d", &key);
    for (i = 0; i < n; i++)
    {
        if (a[i] == key)
        {
            printf("Element found at index %d\n", i);
            return 0;
        }
    }
    printf("Element not found!!!\n");
    return 0;
}
```

OUTPUT:

```
Enter size of the array: 10
Enter elements in array: 25 47 23 34 56 78 93 12 32 54
Enter the element to search for: 56
Element found at index 4
```

QUES 4: WAP to sort a dynamic array of n numbers.

SOLUTION:

```c
#include <stdio.h>
void swap(int *xp, int *yp)
{
    // NAME: Sahil Singh
    // ROLL: 2005535
    int temp = *xp;
    *xp = *yp;
    *yp = temp;
}
void bubbleSort(int arr[], int n, int size)
{
    int i, j, k;
    for (i = 0; i < n - 1; i++)
        for (j = 0; j < n - i - 1; j++)
            if (arr[j] > arr[j + 1])
                swap(&arr[j], &arr[j + 1]);
    for (k = 0; k < size; k++)
        printf("%d ", arr[k]);
    printf("\n");
}
int main()
{
    int arr[] = {10, 8, 9, 7, 4, 5, 3, 6, 2, 1};
    int n = sizeof(arr) / sizeof(arr[0]);
    printf("Sorted array: ");
    bubbleSort(arr, n, n);
    return 0;
}
```

OUTPUT:

```
Sorted array: 1 2 3 4 5 6 7 8 9 10
```

QUES 5: Given an unsorted dynamic array of size n, WAP to find and display the number of elements between two elements a and b (both inclusive).

SOLUTION:

```c
#include <stdio.h>
int main()
{
    // NAME: Sahil Singh
    // ROLL: 2005535
    int arr[10] = {1, 2, 2, 7, 5, 4}, a, b, ctr = 0, count = 0;

    printf("Array: ");
    for (int i = 0; i < 10; i++)
    {
        printf("%d ", arr[i]);
    }

    printf("\nEnter a and b: ");
    scanf("%d %d", &a, &b);

    for (int i = 0; i < 10; i++)
    {
        if (arr[i] == a)
        {
            count = 1;
        }
        if (arr[i] == b)
        {
            break;
        }
        if (count = 1)
        {
            ctr++;
        }
    }

    printf("\nOutput: %d", ctr);
    return 0;
}
```

OUTPUT:

```
Array: 1 2 2 7 5 4 0 0 0 0
Enter a and b: 2 5
Output: 4
```

QUES 6: Given a dynamic array, WAP to print the next greater element (NGE) for every element. The next greater element for an element x is the first greater element on the right side of x in array. Elements for which no greater element exist, consider next greater element as -1.

SOLUTION:

```
#include <stdio.h>
void nxt_grt_ele(int arr[], int n)
{
    // NAME: Sahil Singh
    // ROLL: 2005535
    int next, i, j;
    for (i = 0; i < n; i++)
    {
        next = -1;
        for (j = i + 1; j < n; j++)
        {
            if (arr[i] < arr[j])
            {
                next = arr[j];
                break;
            }
        }
        printf("%d -- %d\n", arr[i], next);
    }
}
int main()
{
    int arr[] = {2, 5, 3, 9, 7};
    int n = sizeof(arr) / sizeof(arr[0]);
    nxt_grt_ele(arr, n);
    return 0;
}
```

OUTPUT:

```
2 -- 5
5 -- 9
3 -- 9
9 -- -1
7 -- -1
```

QUES 7: Let A be nXn square dynamic matrix. WAP by using appropriate user defined functions for the following: a) Find the number of nonzero elements in A. b) Find the sum of the elements above the leading diagonal. c) Display the elements below the minor diagonal. d) Find the product of the diagonal elements.

SOLUTION:

```c
#include <stdio.h>
int main()
{
    // NAME: Sahil Singh
    // ROLL: 2005535
    int i, j, r1, c1, c = 0, p = 1, s = 0;
    printf("Enter number of rows: ");
    scanf("%d", &r1);
    printf("Enter number of columns: ");
    scanf("%d", &c1);

    int a[r1][c1];
    printf("Enter Matrix:\n");
    for (i = 0; i < r1; i++)
    {
        for (j = 0; j < c1; j++)
        {
            scanf("%d", &a[i][j]);
            if (a[i][j] > 0 || a[i][j] < 0)
                c = c + 1;
        }
    }
    for (i = 0; i < r1; i++)
    {
        for (j = 0; j < c1; j++)
        {
            if (i == j)
                p = p * a[i][j];
        }
    }
    for (i = 0; i < r1; i++)
    {
        for (j = 0; j < c1; j++)
        {
            if (j > i)
                s = s + a[i][j];
        }
    }
    printf("\nMatrix Formed:\n");
    for (i = 0; i < r1; i++)
    {
        for (j = 0; j < c1; j++)
        {
            printf("%d ", a[i][j]);
            if (j == c1 - 1)
                printf("\n");
        }
    }
    printf("\nNumber of Non Zero Elements: %d\n", c);
    printf("Sum of Elements above Leading Diagonal: %d\n", s);
    printf("Elements Below Leading Diagonal: ");
    for (i = 0; i < r1; i++)
    {
        for (j = 0; j < c1; j++)
        {
            if (j < i)
                printf("%d ", a[i][j]);
        }
    }
    printf("\nProduct of Diagonal: %d\n", p);
    return 0;
}
```

OUTPUT:

```
Enter number of rows: 3
Enter number of columns: 3
Enter Matrix:
1 1 1
2 2 2
3 3 3

Matrix Formed:
1 1 1
2 2 2
3 3 3

Number of Non Zero Elements: 9
Sum of Elements above Leading Diagonal: 4
Elements Below Leading Diagonal: 2 3 3
Product of Diagonal: 6
```