|  |  |  |
| --- | --- | --- |
| Assignment Code | : | C.S.P0043 |
| Assignment Name | : | Array Manipulations |
| Student Name | : | Le Thi Thanh Nhan |
| Time/Date | : | 15h00,31/10/2019 |

Approach

1. Add: add length of array 1 and input the value for the last element.
2. Search: input value to search, for loop, if a element equal input value then print out it.
3. Removefirst: input value to remove, for loop, if a element equal input value then it equal NULL, break loop and create loop to find element equal NULL and remove it.
4. Removeall: input value to remove, for loop, if a element equal input value then it equal NULL and create loop to find elements equal NULL and remove them.
5. Print: print out all element in array.
6. Sort Ascending: create a array and copy all element from the first array to it. Sort the new array order by ascending.
7. Sort Descending: create a array and copy all element from the first array to it. Sort the new array order by descending.

Source code

#include <stdio.h>

#include <conio.h>

#include <stdlib.h>

#define MAX 100

int print(int arr[], int \*n)

{

    int i;

    for (i = 0; i < \*n; i++)

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

}

int add(int arr[], int \*n)

{

    int new, i;

    printf("\nPlease enter a new value: ");

    scanf("%d", &new);

    (\*n)++;

    arr[(\*n) - 1] = new;

    printf("The value is added!");

}

int search(int arr[], int \*n)

{

    int a;

    printf("Enter value to search: ");

    scanf("%d", &a);

    int i, count = 0;

    for (i = 0; i <= \*n - 1; i++)

        if (arr[i] == a)

        {

            printf("Found! a[%d]=%d\n", i, arr[i]);

            count++;

        }

    if (count = 0)

        printf("This array don't have any number equal %d", n);

}

int removefirst(int arr[], int \*n)

{

    int a;

    printf("Enter value to remove: ");

    scanf("%d", &a);

    int i, count = 0;

    for (i = 0; i <= (\*n); i++)

        if (arr[i] == a)

        {

            arr[i] = NULL;

            break;

        }

    for (i = 0; i < (\*n) - count; i++)

    {

        arr[i] = arr[i + count];

        if (arr[i] == NULL)

        {

            count++;

            i--;

        }

    }

    (\*n) -= count;

    printf("The first existence of %d is removed!", a);

}

int removeall(int arr[], int \*n)

{

    int a;

    printf("Enter value to remove: ");

    scanf("%d", &a);

    int i, count = 0;

    for (i = 0; i <= (\*n); i++)

        if (arr[i] == a)

            arr[i] = NULL;

    for (i = 0; i < (\*n) - count; i++)

    {

        arr[i] = arr[i + count];

        if (arr[i] == NULL)

        {

            count++;

            i--;

        }

    }

    (\*n) -= count;

    printf("All existence of %d are removed!", a);

}

int ascending(int arr[], int n)

{

    int arrtmp[MAX];

    int i, j, tmp;

    for (i = 0; i < n; i++)

        arrtmp[i] = arr[i];

    for (i = 0; i < n; i++)

        for (j = i; j < n; j++)

            if (arrtmp[i] > arrtmp[j])

            {

                tmp = arrtmp[i];

                arrtmp[i] = arrtmp[j];

                arrtmp[j] = tmp;

            }

    printf("Array after sort:\n");

    for (i = 0; i < n; i++)

        printf("%d ", arrtmp[i]);

}

int descending(int arr[], int n)

{

    int arrtmp[MAX];

    int i, j, tmp;

    for (i = 0; i < n; i++)

        arrtmp[i] = arr[i];

    for (i = 0; i < n; i++)

        for (j = i; j < n; j++)

            if (arrtmp[i] < arrtmp[j])

            {

                tmp = arrtmp[i];

                arrtmp[i] = arrtmp[j];

                arrtmp[j] = tmp;

            }

    printf("Array after sort:\n");

    for (i = 0; i < n; i++)

        printf("%d ", arrtmp[i]);

}

int main()

{

    int op, i, length = 0;

    int a[MAX];

    int x;

    printf("input length of array: ");

    scanf("%d", &length);

    for (i = 0; i < length; i++)

    {

        printf("a[%d]=", i);

        scanf("%d", &a[i]);

    }

    do

    {

        system("cls");

        printf("\n1- Add  a value");

        printf("\n2- Search a value");

        printf("\n3- Remove the first existence of a value");

        printf("\n4- Remove all existences of a value");

        printf("\n5- Print out the array ");

        printf("\n6- Sort the array in ascending order (positions of elements are preserved)");

        printf("\n7- Sort the array in descending order (positions of elements are preserved)");

        printf("\nOthers- Quit");

        printf("\nSelect an operation:");

        scanf("%d", &op);

        switch (op)

        {

        case 1:

            add(a, &length);

            break;

        case 2:

            search(a, &length);

            break;

        case 3:

            removefirst(a, &length);

            break;

        case 4:

            removeall(a, &length);

            break;

        case 5:

            print(a, &length);

            break;

        case 6:

            ascending(a, length);

            break;

        case 7:

            descending(a, length);

        }

        printf("\n");

        system("pause");

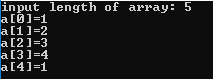
    } while ((op > 0) && (op < 8));

    getch();

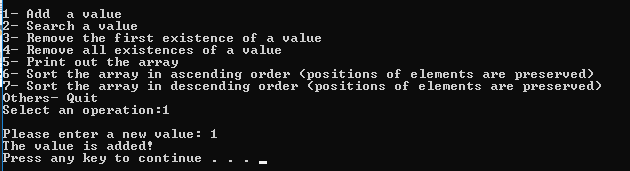
    return 0;

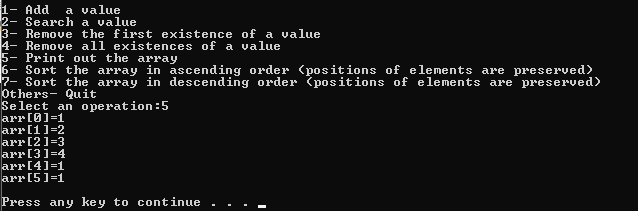
}

Result

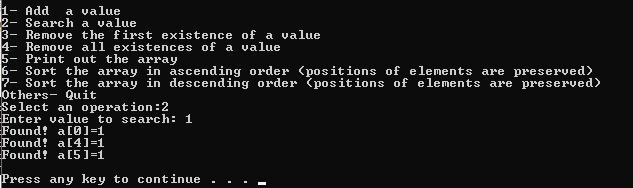


1----

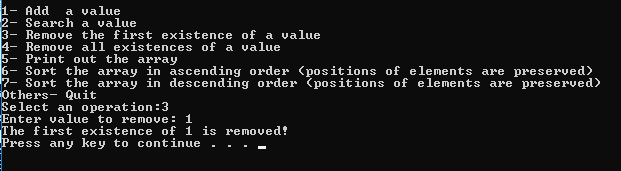


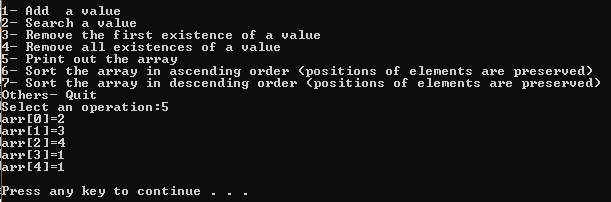


2----

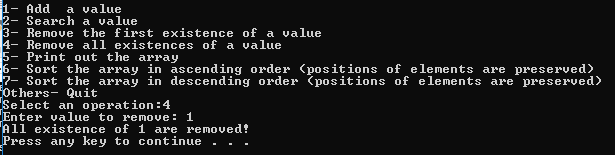


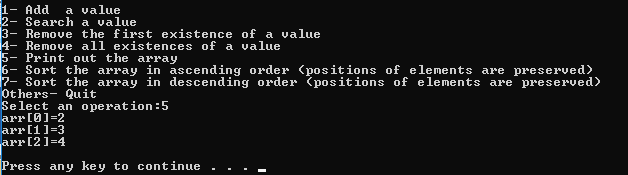
3----





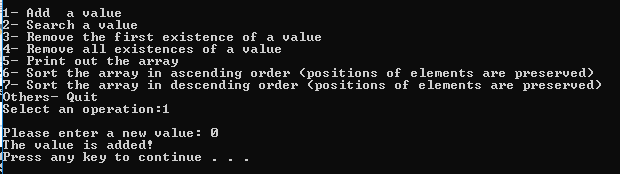
4----

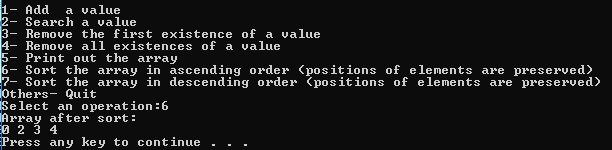




6---

Add a value 0;





7----

