**LAB 04**

**TASK 1**

If the array is already sorted, we don’t want to continue with the comparisons. This can be  achieved with modified bubble sort. Update the code in example 02 to have a modified  bubble sort function.

**SOURCE CODE:**

#include <iostream>

using namespace std;

class Adil\_Lab04 {

    int n;

    int \*arr;

public:

    Adil\_Lab04(int no) {

        n = no;

        arr = new int[n];

    }

    void inputArray() {

        cout << "Enter the elements of the array:" << endl;

        for (int i = 0; i < n; i++) {

            cin >> arr[i];

        }

    }

    void bubblesort() {

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

            bool ans = false;

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

                if (arr[j] > arr[j + 1]) {

                    ans = true;

                    swap(arr[j], arr[j + 1]);

                }

            }

            if (!ans) {

                break;

            }

        }

    }

    void display() {

        cout << "The sorted array is:" << endl;

        for (int i = 0; i < n; i++) {

            cout << arr[i] << " ";

        }

        cout << endl;

    }

    ~Adil\_Lab04() {

        delete[] arr;

    }

};

int main() {

    int n;

    cout << "Enter number of elements: ";

    cin >> n;

    Adil\_Lab04 sort(n);

    sort.inputArray();

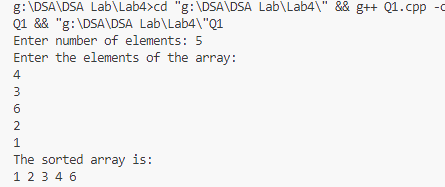
    sort.bubblesort();

    sort.display();

    return 0;

}

**OUTPUT:**



**TASK 2**

Given an array **arr[ ]** of length **N** consisting cost of **N** toys and an integer **K** the amount with  you. The task is to find maximum number of toys you can buy with **K** amount. **Test Case: Input:** N = 7, K = 50, arr[] = {1, 12, 5, 111, 200, 1000, 10}, **Output:** 4 **Explanation:** The costs of the toys. You can buy are 1, 12, 5 and 10.

**SOURCE CODE:**

#include <iostream>

using namespace std;

class Adil\_Lab04 {

    int N;

    int k;

    int\* arr;

public:

    Adil\_Lab04(int No, int amount) {

        N = No;

        k = amount;

        arr = new int[N];

    }

    ~Adil\_Lab04() {

        delete[] arr;

    }

    void inputdata() {

        cout << "Enter the cost of toys:" << endl;

        for (int i = 0; i < N; i++) {

            cout << "Enter price for toy " << i + 1 << " : ";

            cin >> arr[i];

        }

    }

    void bubblesort() {

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

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

                if (arr[j] > arr[j + 1]) {

                    swap(arr[j], arr[j + 1]);

                }

            }

        }

    }

    void maxtoys() {

        bubblesort();

        int count = 0;

        for (int i = 0; i < N; i++) {

            if (k >= arr[i]) {

                k -= arr[i];

                count++;

            } else {

                break;

            }

        }

        cout << "The maximum number of toys you can buy is: " << count << endl;

    }

};

int main() {

    int n, k;

    cout << "Enter number of toys: ";

    cin >> n;

    cout << "Enter the total amount: ";

    cin >> k;

    Adil\_Lab04 toy(n, k);

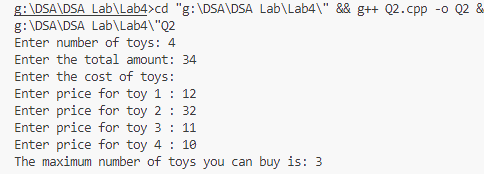
    toy.inputdata();

    toy.maxtoys();

    return 0;

}

**OUTPUT:**



**TASK 3**

Create a single class Sort, which will provide the user the option to choose between all 3 sorting techniques. The class should have following capabilities:

- Take an array and a string (indicating the user choice for sorting technique) as input and  perform the desired sorting.

- Should allow the user to perform analysis on a randomly generated array. The analysis  provides number of comparisons and number of swaps performed for each technique.  - After printing all the results in the main program, highlight the best and worst techniques.

**SOURCE CODE:**

#include <iostream>

#include <ctime>

#include <cstdlib>

#include <string>

using namespace std;

class Adil\_Lab04 {

    int N;

    int comparisons;

    int swaps;

    int\* arr;

public:

    Adil\_Lab04(int no) {

        N = no;

        arr = new int[N];

    }

    ~Adil\_Lab04() {

        delete[] arr;

    }

    void randomarray() {

        srand(time(0));

        for (int i = 0; i < N; i++) {

            arr[i] = rand() % 100;

        }

    }

    void selectionsort() {

        comparisons = swaps = 0;

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

            int minIndex = i;

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

                comparisons++;

                if (arr[j] < arr[minIndex]) {

                    minIndex = j;

                }

            }

            if (minIndex != i) {

                swap(arr[i], arr[minIndex]);

                swaps++;

            }

        }

    }

    void insertionsort() {

        comparisons = swaps = 0;

        for (int i = 1; i < N; i++) {

            int key = arr[i];

            int j = i - 1;

            comparisons++;

            while (j >= 0 && arr[j] > key) {

                arr[j + 1] = arr[j];

                j--;

                swaps++;

                comparisons++;

            }

            arr[j + 1] = key;

        }

    }

    void bubblesort() {

        comparisons = swaps = 0;

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

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

                comparisons++;

                if (arr[j] > arr[j + 1]) {

                    swap(arr[j], arr[j + 1]);

                    swaps++;

                }

            }

        }

    }

    void display() {

        cout << "Array: ";

        for (int i = 0; i < N; i++) {

            cout << arr[i] << " ";

        }

        cout << endl;

    }

    void performanalysis(string sort) {

        cout << "The type of sorting is: " << sort << endl;

        cout << "The number of comparisons: " << comparisons << endl;

        cout << "The number of swaps: " << swaps << endl;

    }

};

int main() {

    int n;

    cout<<"Enter number of elements:";

    cin>>n;

    Adil\_Lab04 analysis(n);

    string type;

    cout << "Enter the sorting type (bubble 'b'/selection 's'/insertion 'i'): ";

    cin >> type;

    analysis.randomarray();

    cout << "Before sorting:" << endl;

    analysis.display();

    if (type == "b") {

        analysis.bubblesort();

        analysis.performanalysis("bubble sort");

    }

    else if (type == "s") {

        analysis.selectionsort();

        analysis.performanalysis("selection sort");

    }

    else if (type == "i") {

        analysis.insertionsort();

        analysis.performanalysis("insertion sort");

    }

    else {

        cout << "Invalid sort type" << endl;

    }

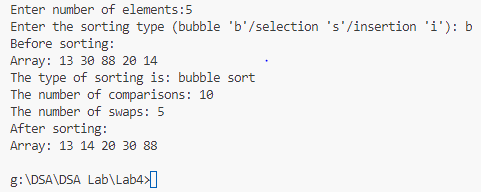
    cout << "After sorting:" << endl;

    analysis.display();

    return 0;

}

**OUTPUT:**



**TASK 4**

Given an array of integers arr, sort the array by performing a series of **pancake flips**. In one  pancake flip we do the following steps:

- Choose an integer k where 1 <= k <= arr.length.

- Reverse the sub-array arr[0...k-1] (**0-indexed**).

For example, if arr = [3,2,1,4] and we performed a pancake flip choosing k = 3, we reverse  the sub-array [3,2,1], so arr = [1,2,3,4] after the pancake flip at k = 3. Return *an array of  the* k*-values corresponding to a sequence of pancake flips that sort* arr. Any valid answer that  sorts the array within 10 \* arr.length flips will be judged as correct.

**Example 1: Input:** arr = [3,2,4,1], **Output:** [4,2,4,3]

**Explanation:** We perform 4 pancake flips, with k values 4, 2, 4, and 3.

Starting state: arr = [3, 2, 4, 1]

After 1st flip (k = 4): arr = [1, 4, 2, 3]

After 2nd flip (k = 2): arr = [4, 1, 2, 3]

After 3rd flip (k = 4): arr = [3, 2, 1, 4]

After 4th flip (k = 3): arr = [1, 2, 3, 4], which is sorted.

**SOURCE CODE:**

#include <iostream>

#include <vector>

#include <algorithm>

using namespace std;

class Adi\_Lab04 {

    int n;

    vector<int> arr;

public:

    Adi\_Lab04(int no) : n(no) {

        arr.resize(n);

        cout << "Enter the elements of the array: ";

        for (int i = 0; i < n; i++) {

            cin >> arr[i];

        }

    }

    void flip(int k) {

        reverse(arr.begin(), arr.begin() + k + 1);

    }

    int max\_index(int currentSize) {

        int ind = 0;

        for (int i = 1; i < currentSize; i++) {

            if (arr[i] > arr[ind]) {

                ind = i;

            }

        }

        return ind;

    }

    vector<int> pancake() {

        vector<int> answer;

        for (int i = n; i > 1; i--) {

            int maxIdx = max\_index(i);

            if (maxIdx != i - 1) {

                if (maxIdx != 0) {

                    answer.push\_back(maxIdx + 1);

                    flip(maxIdx);

                }

                answer.push\_back(i);

                flip(i - 1);

            }

        }

        return answer;

    }

    void display() {

        cout << "Sorted Array: ";

        for (int x : arr) {

            cout << x << " ";

        }

        cout << endl;

    }

    void displayFlips(const vector<int>& flips) {

        cout << "Sequence of flips: ";

        for (int k : flips) {

            cout << k << " ";

        }

        cout << endl;

    }

};

int main() {

    int n;

    cout << "Enter number of elements: ";

    cin >> n;

    Adi\_Lab04 pancakeflips(n);

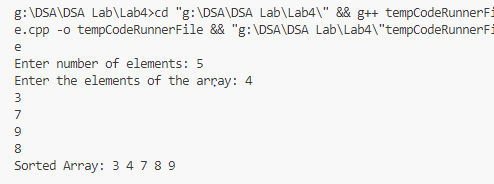
    vector<int> flips = pancakeflips.pancake();

    pancakeflips.display();

    return 0;

}

**OUTPUT:**



**TASK 5**

 Given an array nums with n objects colored red, white, or blue, sort them inplace so that  objects of the same color are adjacent, with the colors in the order red, white, and blue. We  will use the integers 0, 1, and 2 to represent the color red, white, and blue, respectively. You  must solve this problem by writing a sort function.

**Example 1: Input:** nums = [2,0,2,1,1,0], **Output:** [0,0,1,1,2,2]

**Example 2: Input:** nums = [2,0,1], **Output:** [0,1,2]

**SOURCE CODE:**

#include <iostream>

using namespace std;

class Adil\_Lab04 {

    int N;

    int\* arr;

public:

    Adil\_Lab04(int No) {

        N = No;

        arr = new int[N];

    }

    ~Adil\_Lab04() {

        delete[] arr;

    }

    void inputdata() {

        cout << "Enter array nums for red as 0, white as 1 and blue as 2:" << endl;

        for (int i = 0; i < N; i++) {

            int temp;

            cout << "Enter data for " << i + 1 << " object: ";

            cin >> temp;

            while (temp < 0 || temp > 2) {

                cout << "Invalid value. Please enter again (0, 1, or 2): ";

                cin >> temp;

            }

            arr[i] = temp;

        }

    }

    void bubblesort() {

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

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

                if (arr[j] > arr[j + 1]) {

                    swap(arr[j], arr[j + 1]);

                }

            }

        }

    }

    void display() const {

        cout << "The array is: " << endl;

        for (int i = 0; i < N; i++) {

            cout << arr[i] << " ";

        }

        cout << endl;

    }

};

int main() {

    int n;

    cout<<"Enter the number of objects:";

    cin>>n;

    Adil\_Lab04 sort(n);

    sort.inputdata();

    sort.bubblesort();

    sort.display();

    return 0;

}

**OUTPUT:**

