**quicksort:**

#include <iostream>

using namespace std;

int divizare(int tablou[], int indexInceput, int indexFinal) {

    int valoarePivot = tablou[indexFinal];

    int indexPivot = indexInceput;

    for (int i = indexInceput; i <= indexFinal - 1; i++)

    {

        if (tablou[i] <= valoarePivot) {

            int aux = tablou[indexPivot];

            tablou[indexPivot] = tablou[i];

            tablou[i] = aux;

            indexPivot++;

        }

    }

    tablou[indexFinal] = tablou[indexPivot];

    tablou[indexPivot] = valoarePivot;

    return indexPivot;

}

void quicksort(int tablou[], int indexInceput, int indexFinal) {

    if (indexInceput < indexFinal) {

        int indexPivot = divizare(tablou, indexInceput, indexFinal);

        quicksort(tablou, indexInceput, indexPivot - 1);

        quicksort(tablou, indexPivot + 1, indexFinal);

    }

}

int main()

{

    int tablou[]{ 3,6,4,8,1,9,12,3,10 };

    quicksort(tablou, 0, 8);

    for (int element : tablou) {

        cout << element << " ";

    }

}

binary search:

    int binarySearch(int tablou[], int indexInceput, int indexFinal, int valoareCautata) {

    if (indexInceput < indexFinal) {

        int indexMijloc = indexInceput + (indexFinal - indexInceput) / 2;

        if (tablou[indexMijloc] == valoareCautata) {

            return indexMijloc;

        }

        else if (tablou[indexMijloc] > valoareCautata)

        {

            return binarySearch(tablou, indexInceput, indexMijloc - 1, valoareCautata);

        }

        else

        {

            return binarySearch(tablou, indexMijloc + 1, indexFinal, valoareCautata);

        }

    }

    return -1;

}

bool metodaDeComparare(int element1, int element2) {

    return element1 > element2;

}

int main()

{

    int tablou[]{ 3,6,4,8,1,9,12,3,10 };

    vector<int> tablouVector(tablou, tablou + 9);

    quicksort(tablou, 0, 8);

    for (int element : tablou) {

        cout << element << " ";

    }

    sort(tablouVector.begin(), tablouVector.end(), metodaDeComparare);

    if (binary\_search(tablouVector.begin(), tablouVector.end(), 8, metodaDeComparare)) {

        cout << "L'am gasit";

    }

    else { cout << "Nu l'am gasit"; }

    cout << "\n" << binarySearch(tablou, 0, 8, 20);

}