Домашняя работа

1. Дан массив положительных чисел, отвечать на запросы: «Какое максимальное число элементов из начала массива можно взять, чтобы их сумма была не больше X?».

#include <iostream>

#include <vector>

#include <queue>

using namespace std;

int main()

{

    int n, x;

    cin >> n >> x;

    queue <int> arr;

    int ch=0;

    for (int i=0; i<n; i++)

    {

        cin >> ch;

        arr.push(ch);

    }

    ch=0;

    while ((ch+=arr.front())<x)

        arr.pop();

    ch-=arr.front();

    cout << ch;

    return 0;

}

2. Задан массив, полученный приписыванием отсортированного по убыванию массива в конец отсортированному по возрастанию. Все элементы массива различны. Требуется за O(log n) найти в нем заданный элемент.

#include <iostream>

#include <fstream>

#include <vector>

using namespace std;

int main()

{

    ifstream fin ("binsearch.in");

    ofstream fout ("binsearch.out");

    int n, N;

    int x;

    fin >> n;

    vector <int> arr (n);

    for (int i=0; i<n; i++)

        fin >> arr[i];

    int l=0; int r=n-1;

    int mid;

    bool flag = false;

    while ((l <= r) && (flag != true))

    {

        mid = (l + r) / 2;

        if (arr[mid] == x) flag = true;

        if (arr[mid] > x) r = mid - 1;

        else l = mid + 1;

    }

    if (flag)

    {

        int j=mid;

        while (j>0 && arr[j-1]==arr[mid])

            j--;

        fout << j+1;

        j=mid;

        while (j<n && arr[j+1]==arr[mid])

            j++;

        fout << " " << j+1 << endl;

    }

    else

        fout << "-1 -1" << endl;

    return 0;

}

3. Задан массив, полученный циклическим сдвигом из отсортированного по возрастанию. Пусть элементы массива могут повторяться. Можно ли в таком массиве найти заданный элемент за O(log n)?

Да, можно. Тот же код, только мы присваиваем переменной “l” и “r” сдвиг нашего массива.

4. Добавьте в стек и очередь операцию getSum(), возвращающую сумму элементов в стеке/очереди. Время работы O(1). Дополнительная память O(1).

5. Научитесь вычислять выражения в инфиксной записи со скобками (обычные выражения). Для простоты можно считать, что в выражении проставлены все скобки (то есть внутри скобок вычисляется только один оператор, например так можно: (4 − ((1 + 2) ∗ 3)), а так — нет: (1 + 2 + 3).

#include <iostream>

#include <string>

#include <stack>

using namespace std;

int main()

{

string f;

stack <int> integer;

stack <char> znak;

cin >> f;

int i = 0;

while (i != f.size())

{

int ch = 0;

if (f[i] == '+' || f[i] == '-' || f[i] == '\*' || f[i] == '/')

znak.push(f[i]);

else if (f[i] == ')')

{

if (znak.top() == '+')

{

ch += integer.top();

integer.pop();

ch += integer.top();

integer.pop();

integer.push(ch);

}

else if (znak.top() == '-')

{

ch -= integer.top();

integer.pop();

ch += integer.top();

integer.pop();

integer.push(ch);

}

else if (znak.top() == '\*')

{

ch += integer.top();

integer.pop();

ch \*= integer.top();

integer.pop();

integer.push(ch);

cout << znak.top() << endl;

}

else if (znak.top() == '/')

{

ch += integer.top();

integer.pop();

ch = integer.top() / ch;

integer.pop();

integer.push(ch);

cout << znak.top() << endl;

}

znak.pop();

}

else if (f[i]!='(')

integer.push(f[i]-48);

i++;

}

cin >> f;

return 0;

}

6. Научитесь по выражению в инфиксной записи строить выражение в постфиксной записи.

7. Как развернуть односвязный список за время O(n) с O(1) дополнительной памяти?

8. Дан набор из n элементов, в каждом есть ссылка на какой-то другой. Проверьте, правда ли эти элементы образуют один большой кольцевой список (менять ссылки нельзя). (Время O(n), память O(1))

9. Дан набор из n элементов, в каждом есть ссылка на следующий и предыдущий. Проверьте, правда ли эти элементы образуют несколько кольцевых списков (менять ссылки нельзя). (Время O(n), память O(1))

10. Слить два отсортированных односвязных списка в один за время O(n) с O(1) дополнительной памяти.