МИНОБРНАУКИ РОССИИ

Федеральное государственное автономное
образовательное учреждение высшего образования
«Пермский государственный национальный
исследовательский университет»

Институт компьютерных наук и технологий

ОТЧЁТ
по индивидуальной работе №2
по дисциплине «Язык программирования C++»
Вариант 12

Работу выполнил
студент группы ИТ-11,12-2024 1 курса
Горбунов Глеб Алексеевич
«24» июня 2025 г.

Работу проверила
Кнутова Наталия Сергеевна
«26» июня 2025 г.

Пермь 2025

СОДЕРЖАНИЕ

Постановка задачи

Игрок выставляет в линию шарики разных цветов. Когда образуется непрерывная цепочка из трех и более шариков одного цвета, она удаляется из линии. Все шарики при этом сдвигаются друг к другу, и ситуация может повториться. Написать программу, которая по данной ситуации определяет, сколько шариков будет сейчас уничтожено. Непрерывных цепочек из трех и более одноцветных шаров в начальный момент может быть не более одной. Даны количество шариков в цепочке (не более 10^5) и цвета шариков (от 0 до 9, каждому цвету соответствует свое целое число). Требуется вывести количество шариков, которое будет уничтожено.

Алгоритм решения

Для решения задачи было решено использовать связные списки, поскольку они позволяют удалять последовательности элементов без необходимости сдвига оставшихся элементов, как, например, в массиве или векторе.

Сначала создаются два класса: Node и LinkedList. Класс Node представляет узел списка и содержит значение “цвета” шарика (цифру от 0 до 9) и указатель на следующий узел. Класс LinkedList реализует сам связный список, в нем содержатся методы для добавления элементов, удаления последовательностей, подсчета уничтоженных шариков, вывода списка.

В методе main программа запрашивает количество шариков n, проверяя ввод на корректность. Затем пользователь вводит сами шарики (цифры от 0 до 9), которые добавляются в связный список с помощью метода PushBack. Когда все шарики введены, вызывается функция DestroySeq.

Основная логика работы программы как раз таки и заключается в DestroySeq, она ищет последовательности из трех или более одинаковых цифр подряд пробегаясь по элементам в списке. Если такие последовательности найдены, они убираются из списка с помощью Erase, который обновляет связи между узлами и уменьшает размер списка. Удаление повторяется до тех пор, пока в списке не останется последовательностей. Количество уничтоженных шариков суммируется и выводится в конце, так же показывается оставшийся список.

Метод Erase удаляет узлы в диапазоне, начиная с узла start и заканчивая узлом перед end. Если удаление затрагивает начало или конец списка, обновляются указатели _head и _tail.

Тестирование

```
========================
Добро пожаловать в SHARIQI!
========================
Тут конечно нет никаких шариков, но есть цифры. Введи количество шариков, которые будут задействованы.
 >>
```

```
 >> text
НЕПРАВИЛЬНО! Введи положительное целое число меньшее 10^5!
 >> -1
НЕПРАВИЛЬНО! Введи положительное целое число меньшее 10^5!
 >> 1000000
НЕПРАВИЛЬНО! Введи положительное целое число меньшее 10^5!
 >>
```

В программе присутствует дружественный интерфейс и “защита от дурака”, которая не дает пользователю вводить ничего кроме цифр. Так же не принимаются значения, которые не соответствуют условиям задачи или вне диапазона.

Для начального тестирования возьмем значения, которые были предоставлены в задании в качестве примера:

1 3 3 3 2
Ожидаемые выходные данные: **3**
3 3 2 1 1 1 2 2 3 3
Ожидаемые выходные данные: **10**

Пробуем вписать их в программу:

```
 >> 5
Замечательно. Теперь введи сами 'шарики', это цифры в диапазоне от 0 до 9.
 >> 1 3 3 3 2
Уничтожено шариков: 3
Оставшиеся шарики: 1 2
```

```
 >> 10
Замечательно. Теперь введи сами 'шарики', это цифры в диапазоне от 0 до 9.
 >> 3 3 2 1 1 1 2 2 3 3
Уничтожено шариков: 10
Оставшиеся шарики:
```

Уничтожено шариков: 3 и 10 соответственно. Программа работает корректно.

В условии подмечатеся, что изначально цепочек из трех или более шариков может быть не более одной, но в случае, где их все-таки несколько, они будут обрабатываться последовательно:

```
 >> 3 7 8 3 3 3 3 3 8 9 1 6 6 6 0
Уничтожено шариков: 8
Оставшиеся шарики: 3 7 8 8 9 1 0
```

Здесь сначала уничтожаются цепочки из троек, затем из шестерок. Итого уничтожено 5 троек и 3 шестерки, всего 8, что и показывает программа.

Код программы

```
#include <iostream>
#include <limits>
using namespace std;

// узел связного списка
class Node {
public:
    int value;
    Node* next;

    // конструктор узла
    Node(int val) : value(val), next(nullptr) {}
};

// односвязный список
class LinkedList {
private:
    Node* _head;
    Node* _tail;
    int _size; // количество элементов в списке

public:
    LinkedList() : _head(nullptr), _tail(nullptr), _size(0) {}

    // добавление элемента в конец
    void PushBack(int val) {
        Node* newNode = new Node(val);
        if (!_head) {
            _head = _tail = newNode; // новый узел становится и
головой и хвостом
        }
        else {
            _tail->next = newNode; // добавляем узел в конец
            _tail = newNode; // обновляем указатель на хвост
        }
```

```
        _size++; // увеличиваем счетчик элементов
    }
// удаление цепочки узлов от start до end (без end)
    void Erase(Node* start, Node* end) {
        if (!start || !_head) return;

        // поиск узла перед start
        Node* prev = nullptr;
        Node* current = _head;
        while (current && current != start) {
            prev = current;
            current = current->next;
        }

        // удаление узлов
        Node* temp;
        int count = 0;
        while (current && current != end) {
            temp = current;
            current = current->next;
            delete temp;
            count++; // считаем удаленные узлы
        }

        // обновление связей
        if (prev) {
            prev->next = current;
        }
        else {
            _head = current;
        }

        // обновление хвоста
        if (!current) {
            _tail = prev;
        }

        _size -= count;
    }

    // поиск и удаление цепочки из 3+ шариков
    int DestroySeq() {
        if (!_head) return 0;

        int destroyed = 0; // счетчик уничтоженных шариков
        bool found = true; // флаг для продолжения поиска
```

```
       while (found) {
            found = false;
            Node* current = _head;
            Node* rangeStart = nullptr;  // начало
последовательности
            int count = 1;               // длина

            // поиск последовательностей
            while (current) {
                if (current->next && current->value == current-
>next->value) {
                    if (!rangeStart) rangeStart = current;  //
запоминаем начало
                    count++;
                }
                else {
                    if (count >= 3) { // нашли последовательность
                        Node* rangeEnd = current->next;
                        Erase(rangeStart, rangeEnd); // удаляем
                        destroyed += count;
                        found = true; // продолжаем поиск
                        break;
                    }
                    rangeStart = nullptr; // сброс для новой
последовательности
                    count = 1;
                }
                current = current->next;
            }
        }
        return destroyed;
    }

    // вывод списка
    void Print() {
        Node* current = _head;
        while (current) {
            cout << current->value << " ";
            current = current->next;
        }
        cout << endl;
    }

    ~LinkedList() {
        Node* current = _head;
        while (current) {
            Node* temp = current;
            current = current->next;
            delete temp;
        }
    }
```

```
};

int main() {
    cout << "=========================" << endl;
    cout << "Добро пожаловать в SHARIQI! " << endl;
    cout << "=========================" << endl;
    cout << "Тут конечно нет никаких шариков, но есть цифры.
Введи количество шариков, которые будут задействованы." << endl;

    // ввод количества шариков с проверкой
    int n;
    while (true) {
        cout << " >> ";
        cin >> n;

        if (cin.fail() || n <= 0 || n > 100000) {
            cin.clear();
            cin.ignore(numeric_limits<streamsize>::max(), '\n');
            cout << "НЕПРАВИЛЬНО! Введи положительное целое число
меньшее 10^5!" << endl;
        }
        else {
            break;
        }
    }

    LinkedList list;  // создаем список

    // ввод последовательности шариков
    cout << "Замечательно. Теперь введи сами 'шарики', это цифры
в диапазоне от 0 до 9." << endl;
    cout << " >> ";
    for (int i = 0; i < n; i++) {
        int num;
        while (true) {
            cin >> num;
            if (cin.fail() || num < 0 || num > 9) {
                cin.clear();
                cin.ignore(numeric_limits<streamsize>::max(),
'\n');
                cout << "НЕПРАВИЛЬНО! Введи цифру от 0 до 9!" <<
endl;
                cout << " >> ";
            }
            else {
                break;
            }
        }
        list.PushBack(num);  // добавляем шарик в список
    }
```

```
// вывод результатов
    int totalDestroyed = list.DestroySeq();
    cout << "Уничтожено шариков: " << totalDestroyed << endl;
    cout << "Оставшиеся шарики: ";
    list.Print();
    return 0;
}
```