Министерство науки и высшего образования Российской Федерации Федеральное государственное автономное учреждение высшего образования «Пермский национальный исследовательский политехнический университет»

ПНИПУ

**Лабораторная работа №9**

**«Чтение и запись в файл»**

**19 вариант**

Выполнил:

Студент группы РИС-23-1б

Шароглазов Егор Алексеевич

Проверила:

Доцент кафедры ИТАС

О.А. Полякова

2024 г

**Постановка задачи:**

Записи в линейном списке содержат ключевое поле типа \*char(строка символов). Сформировать двунаправленный список. Удалить элемент с заданным номером. Добавить К элементов в начало списка.

**Анализ задачи:**

1. Создание структуры Node
2. Создание структуры Queue, конец и начало очереди указывают на nullptr.
3. Функция enqueue() добавляет элемент в очередь.
4. Функция dequeue() удаляет элемент из очереди.
5. Функция front() возвращает первый элемент очереди.
6. Функция print() выводит содержимое очереди.
7. Вводится количество элементов в очереди, затем сами элементы
8. Вводится ключ для удаления элемента и он удаляется с помощью функции dequeue().
9. Вводится количество новых элементов, затем номер, с какого их нужно вставить в очередь
10. В цикле for, с помощью функций enqueue() и dequeue(), доставая элементы из q2, происходит вставка.

**Код программы**

#include <iostream>

#include <string>

using namespace std;

template <typename T>

struct Node

{

    T data;

    Node\* next;

};

template <typename T>

struct Queue {

    Node<T>\* head = nullptr;

    Node<T>\* tail = nullptr;

    int size = 0;

};

template <typename T>

void enqueue(Queue<T>& q, T a) {

    Node<T>\* new\_node = new Node<T>;

    new\_node->data = a;

    new\_node->next = nullptr;

    if (q.tail != nullptr) {

        q.tail->next = new\_node;

    }

    else {

        q.head = new\_node;

    }

    q.tail = new\_node;

    q.size++;

}

template <typename T>

void dequeue(Queue<T>& q) {

    Node<T>\* delete\_node = q.head;

    q.head = delete\_node->next;

    if (q.head == nullptr) {

        q.tail = nullptr;

    }

    delete delete\_node;

    q.size--;

}

template <typename T>

T front(Queue<T>& q) {

    return q.head->data;

}

template <typename T>

void print(Queue<T>& q1, Queue<T>& q2) {

    int p = q1.size;

    if (p == 0) cout << "Очередь пуста";

    else {

        for (int i = 0; i < p; i++) {

            cout << front(q1) << ' ';

            enqueue(q2, front(q1));

            dequeue(q1);

        }

    }

    cout << '\n';

}

int main()

{

    setlocale(LC\_ALL, "ru");

    int n;

    Queue<string> q1, q2;

    cout << "Введите количество элементов в очереди:";

    cin >> n;

    cout << "Вводите элементы очереди:";

    cin.ignore();

    for (int i = 0; n > i; i++) {

        string a;

        getline(cin, a);

        enqueue(q1, a);

    }

    cout << "Очередь: ";

    print(q1, q2);

    cout << "Введите ключ для удаления: ";

    string key;

    getline(cin, key);

    for (int i = 0; i < n; i++) {

        if (front(q2) == key) {

            dequeue(q2);

        }

        else {

            enqueue(q1, front(q2));

            dequeue(q2);

        }

    }

    cout << "Очередь после удаления: ";

    print(q1, q2);

    cout << "Введите количество новых элементов: ";

    int k, s;

    cin >> k;

    cout << "C какого номера их вставить? ";

    cin >> s;

    s = n - s+1;

    cin.ignore();

    n = q2.size;

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

        enqueue(q1, front(q2));

        dequeue(q2);

    }

    cout << "Вводите новые элементы: ";

    for (int i = 0; i < k; i++) {

        string a;

        getline(cin, a);

        enqueue(q1, a);

    }

    while (q2.head != nullptr) {

        enqueue(q1, front(q2));

        dequeue(q2);

    }

    cout << "Очередь после добавления: ";

    print(q1, q2);

}

**Скрины работы программы**



