Министерство образования Республики Беларусь

Учреждение образования «Полоцкий государственный университет»

Факультет информационных технологий

Кафедра технологий программирования

**Лабораторная работа №8**

**По дисциплине «Основы алгоритмизации и программирования»**

**Тема: «методы организации списочных структур в динамической памяти»**

Выполнил: Макеёнок Д.И.

Группа 21-ИТ-1

Проверила: Пантелейко А.Ф.

Преподаватель-стажер кафедры ТП

Полоцк 2022

**Цель:** изучить методы организации списочных структур в динамической памяти. Реализовать алгоритмы помещения и изъятия элементов из стека, дека или очереди.

**Вариант № 13**

**Задание 1.** Функции для создания новой ДСД и ее удаления.

**Задание 2.** Функции добавления элементов (все, что необходимы для полноценной работы).

**Задание 3.** Функции добавления элементов (все, что необходимы для полноценной работы).

**Задание 4.** Функция печати всех элементов ДСД.

**Задание 5.** Функция очистки ДСД (то есть удаления всех вложенных в нее элементов).

очередь **FIFO**

**char[20]**

#include <iostream>

#include <stdio.h>

#include <Windows.h>

struct node {

std::string data; // указатель на адрес data в памяти

node\* next; // указатель на следующий

node(std::string c, node\* n) {

c.resize(20); data = c; next = n; // data = c -> указатель получает значение "c[20]", resize

}

};

class quene {

node\* root; // Указатель на корневой узел

public:

quene(std::string data) { // конструктор дэка

root = new node(data, nullptr); // Самый первый узел

}

node\* get\_last(node\* temp) {

if (temp->next) {

get\_last(temp->next);

}

else {

return temp;

}

}

void insert\_node(std::string data) { // Вставка до последнего корня

node\* last = get\_last(root);

node\* new\_node = new node(data, nullptr);

last->next = new\_node; // Перестановка new\_node на end.

}

void remove\_node(quene\* d) {

node\* new\_root = root->next;

delete root;

root = new\_root;

}

void print\_quene() {

node\* temp = root;

while (temp) {

std::cout << temp->data << " "; temp = temp->next;

}

std::cout << "\n";

}

void done\_quene(quene\* d) {

while (root) {

remove\_node(d);

}

delete root;

}

};

int main() {

SetConsoleOutputCP(1251);

SetConsoleCP(1251);

std::string root\_node;

std::string end\_node;

std::cout << "Начальный символ -> ";

std::cin >> root\_node;

quene\* d = new quene(root\_node);

std::string data;

int task = 0;

std::cout << "\n1-Добавить символ\n2-Удалить символ\n3-Вывести\n4-Очистить ДСД\n";

while (task != -1) {

std::cout << "Номер действия ->";

std::cin >> task;

switch (task) {

case 1:

std::cout << "Напишите символ ->";

std::cin >> data;

d->insert\_node(data);

break;

case 2:

d->remove\_node(d);

break;

case 3:

d->print\_quene();

break;

case 4:

d->done\_quene(d);

task = -1;

break;

default:

break;

}

}

main();}

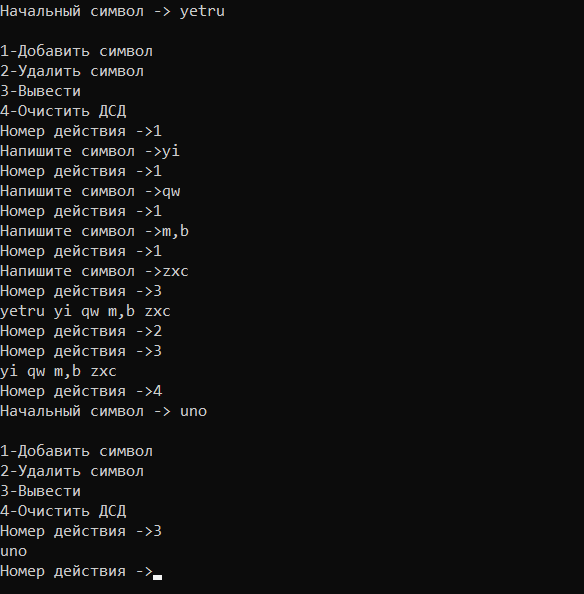


Рисунок 1 – Результат работы заданий

**Вывод:** в данной лабораторной работе были изучены методы организации списочных структур в динамической памяти. Реализовать алгоритмы помещения и изъятия элементов из стека, дека или очереди.