Министерство науки и высшего образования Российской Федерации

Пензенский государственный университет

Кафедра «Вычислительная техника»

**ОТЧЕТ**

по лабораторной работе №3

по курсу «Логика и основы алгоритмизации в инженерных задачах»

на тему «Динамические списки»

Выполнил студент гр. 22ВВС1:

Беспалов Н.А.

Приняли:

Юрова О. В.

Акифьев И.В.

Пенза 2024

**Лабораторное задание:**

1) Реализовать приоритетную очередь, путём добавления элемента в список в соответствии с приоритетом объекта

**Задание 1**

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <stdlib.h>

#include <locale>

struct node

{

char data[256];

int elem;

struct node\* pred; // ссылка на предыдущий элемент

};

struct node\* head = NULL, \* last = NULL; // указатели на первый и последний элементы списка

void in(const char\* str, int val) {

node\* list = (node\*)malloc(sizeof(node));

if (head == NULL) {

head = list;

last = list;

list->elem = val;

strcpy\_s(list->data, str);

}

else {

char flag = 0;

list->elem = val;

strcpy\_s(list->data, str);

node\* sravn = head, \* sravnprev = head;

while (flag == 0) {

while (1) {

if (sravn->elem == list->elem) { //если номера равны

if (sravn == last) { //если последний элемент равен вводимому

last = list;

sravn->pred = list;

flag = 1;

break;

}

else {

list->pred = sravn->pred;

sravn->pred = list;

flag = 1;

break;

}

}

else if (sravn->elem < list->elem) { //если номер с меньшим приоритетом

if (sravn == last) {

last = list;

sravn->pred = list;

flag = 1;

break;

}

else {

sravnprev = sravn;

sravn = sravn->pred;

break;

}

}

else if (sravn->elem > list->elem) { //если номер с большим приоритетом

if (sravn == head) {

list->pred = head;

head = list;

flag = 1;

break;

}

else {

list->pred = sravn;

sravnprev->pred = list;

flag = 1;

break;

}

}

}

}

}

}

void out() {

if (head == NULL) {

("Очередь пустая\n");

return;

}

else if (head == last) {

printf("Извлечённый элемент: %s\n", head->data);

free(head);

head = NULL;

last = NULL;

}

else {

printf("Извлечённый элемент: %s\n", head->data);

node\* tmp = head;

head = head->pred;

free(tmp);

}

}

int main()

{

setlocale(LC\_ALL, "Russian");

char data[256];

int pr;

int buf;

while (1) {

if (head != NULL) {

node\* list = head;

printf("Очередь: ");

while (1) {

printf("%s ", list->data);

if (list == last) {

printf("\n");

break;

}

list = list->pred;

}

}

printf("Введите: 1 для помещения в очередь, 2 для извлечения из очереди, 0 для выхода ");

scanf("%d", &buf);

printf("\n");

if (buf == 1) {

printf("Введите данные для помещения в очередь: ");

scanf("%s", data);

printf("Введите номер для приоритета в очереди: ");

scanf\_s("%d", &pr);

printf("\n");

in(data, pr);

}

else if (buf == 2) {

out();

}

else if (buf == 0) {

exit(0);

}

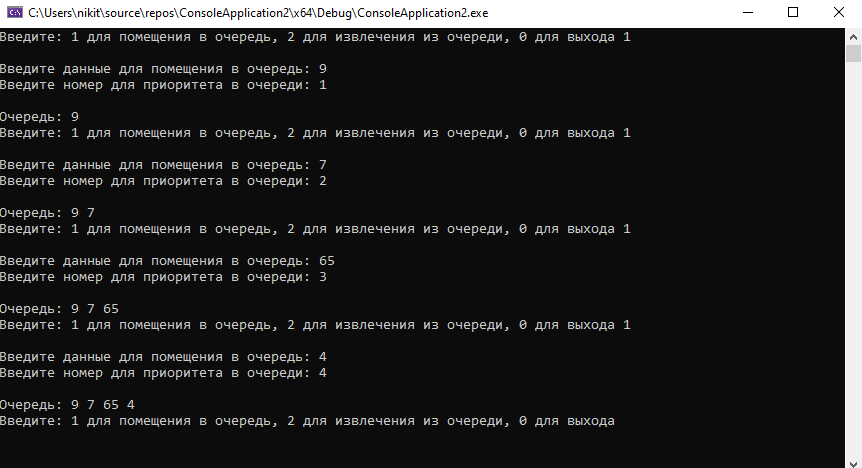
else {

printf("Вы ввели команду неверно\n");

}

}

}

****

**Вывод:** Мы научились реализовывать структуры данных в виде приоритетной очереди, очереди, стек в динамических списках.