Министерство 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»

Выполнили:

студенты группы 21ВВ1.2

Сагателов А.К  
Митрошин А.Д

Приняли:

Митрохин М. А.

Юрова О. В.

Пенза 2022

**Название**

Динамические списки.

**Лабораторное задание.**

1. Реализовать приоритетную очередь, путём добавления элемента в список в

соответствии с приоритетом объекта (т.е. объект с большим приоритетом

становится перед объектом с меньшим приоритетом).

2. \* На основе приведенного кода реализуйте структуру данных Очередь.

3. \* На основе приведенного кода реализуйте структуру данных Стек.

**Листинг**

Задание 1

#include "stdafx.h"

#include <string.h>

#include <memory.h>

#include <stdlib.h>

#include <locale.h>

struct node

{

char inf[256];

int P;

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

};

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

void push(const char\* str, int P) {

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

if (head == NULL) {

head = tmp;

last = tmp;

tmp->P = P;

strcpy\_s(tmp->inf, str);

}

else {

char flag = 0;

tmp->P = P;

strcpy\_s(tmp->inf, str);

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

while (flag == 0) {

while (1) {

if (sravn->P == tmp->P) { //если номера равны

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

last = tmp;

sravn->prev = tmp;

flag = 1;

break;

}

else {

tmp->prev = sravn->prev;

sravn->prev = tmp;

flag = 1;

break;

}

}

else if (sravn->P < tmp->P) { //если номер с меньшим приоритетом

if (sravn == last) {

last = tmp;

sravn->prev = tmp;

flag = 1;

break;

}

else {

sravnprev = sravn;

sravn = sravn->prev;

break;

}

}

else if (sravn->P > tmp->P) { //если номер с большим приоритетом

if (sravn == head) {

tmp->prev = head;

head = tmp;

flag = 1;

break;

}

else {

tmp->prev = sravn;

sravnprev->prev = tmp;

flag = 1;

break;

}

}

}

}

}

}

void pop() {

if (head == NULL) {

printf("Очередь пустая\n\n");

return;

}

else if (head == last) {

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

free(head);

head = NULL;

last = NULL;

}

else {

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

node\* tmp = head;

head = head->prev;

free(tmp);

}

}

int \_tmain(int argc, \_TCHAR\* argv[])

{

setlocale(LC\_ALL, "Russian");

char com[10], data[256];

int pr;

while (1) {

if (head != NULL) {

node\* tmp = head;

printf("Очередь: ");

while (1) {

printf("%s ", tmp->inf);

if (tmp == last) {

printf("\n\n");

break;

}

tmp = tmp->prev;

}

}

printf("Введите команду push или pop для работы с очередью. Чтобы выйти введите exit: ");

scanf("%s", &com);

printf("\n");

if (!strcmp(com, "push")) {

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

scanf("%s", &data);

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

scanf("%d", &pr);

printf("\n");

push(data, pr);

}

else if (!strcmp(com, "pop")) {

pop();

}

else if (!strcmp(com, "exit")) {

exit(0);

}

else {

printf("Вы ввели команду неверно\n\n");

}

}

return 0;

}

Задание 2

#include "stdafx.h"

#include <string.h>

#include <memory.h>

#include <stdlib.h>

#include <locale.h>

struct node

{

char inf[256];

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

};

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

void push(const char\* str) {

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

if (head == NULL) {

head = tmp;

last = tmp;

strcpy\_s(last->inf, str);

}

else {

last->prev = tmp;

strcpy\_s(tmp->inf, str);

last = tmp;

}

}

void pop() {

if (head == NULL) {

printf("Очередь пустая\n\n");

return;

}

else if (head == last) {

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

free(head);

head = NULL;

last = NULL;

}

else {

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

node\* tmp = head;

head = head->prev;

free(tmp);

}

}

int \_tmain(int argc, \_TCHAR\* argv[])

{

setlocale(LC\_ALL, "Russian");

char com[10], data[256];

while (1) {

if (head != NULL) {

node\* tmp = head;

printf("Очередь: ");

while (1) {

printf("%s ", tmp->inf);

if (tmp == last) {

printf("\n\n");

break;

}

tmp = tmp->prev;

}

}

printf("Введите команду push или pop для работы с очередью. Чтобы выйти введите exit: ");

scanf("%s", &com);

printf("\n");

if (!strcmp(com, "push")) {

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

scanf("%s", &data);

printf("\n");

push(data);

}

else if (!strcmp(com, "pop")) {

pop();

}

else if (!strcmp(com, "exit")) {

exit(0);

}

else {

printf("Вы ввели команду неверно\n\n");

}

}

return 0;

}

Задание 3

#include "stdafx.h"

#include <string.h>

#include <memory.h>

#include <stdlib.h>

#include <locale.h>

struct node

{

char inf[256];

struct node\* next; // ссылка на следующий элемент

};

struct node\* last = NULL; // указатели на последний элементы списка

void push(const char\* str) {

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

if (last == NULL) {

last = tmp;

strcpy\_s(last->inf, str);

last->next = NULL;

}

else {

tmp->next = last;

strcpy\_s(tmp->inf, str);

last = tmp;

}

}

void pop() {

if (last == NULL) {

printf("Стэк пустой\n\n");

return;

}

else if (last->next == NULL) {

printf("Извлечённый элемент: %s\n\n", last->inf);

free(last);

last = NULL;

}

else {

printf("Извлечённый элемент: %s\n", last->inf);

node\* tmp = last;

last = last->next;

free(tmp);

}

}

int \_tmain(int argc, \_TCHAR\* argv[])

{

setlocale(LC\_ALL, "Russian");

char com[10], data[256];

while (1) {

if (last != NULL) {

node\* tmp = last;

printf("Стэк: ");

while (1) {

printf("%s ", tmp->inf);

if (tmp->next == NULL) {

printf("\n\n");

break;

}

tmp = tmp->next;

}

}

printf("Введите команду push или pop для работы с очередью. Чтобы выйти введите exit: ");

scanf("%s", &com);

printf("\n");

if (!strcmp(com, "push")) {

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

scanf("%s", &data);

printf("\n");

push(data);

}

else if (!strcmp(com, "pop")) {

pop();

}

else if (!strcmp(com, "exit")) {

exit(0);

}

else {

printf("Вы ввели команду неверно\n\n");

}

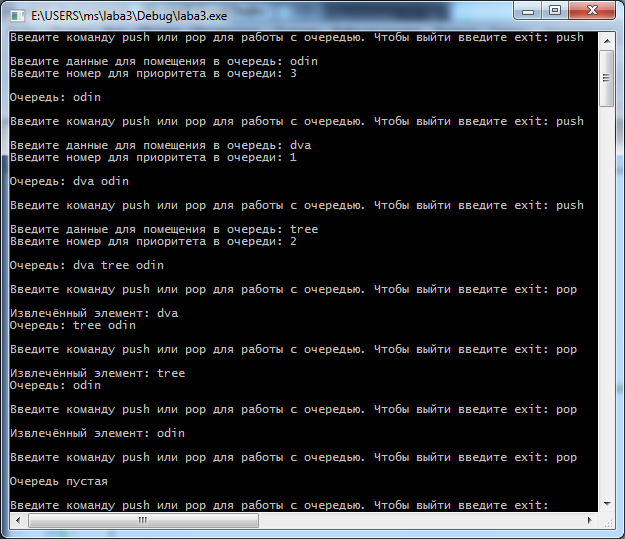
}

return 0;

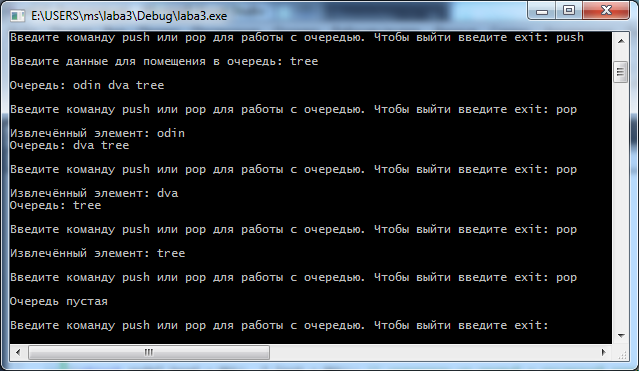
}

**Результат работы программы**

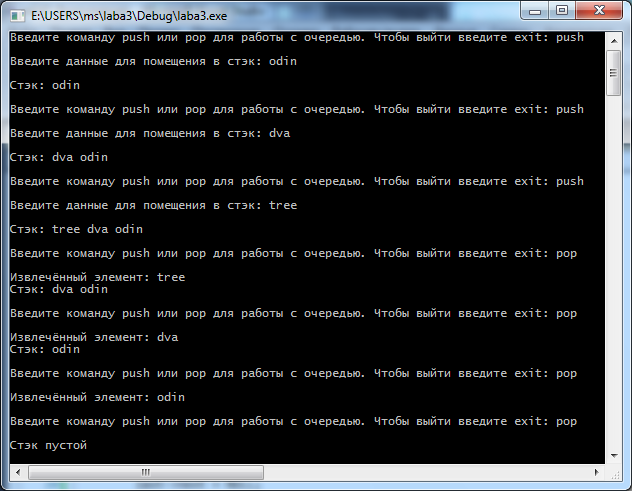
Задание 1

****

Задание 2

****

Задание 3

****

### Выводы

### В ходе данной работы мы научились разрабатывать программу, реализующую очередь с приоритетом, очередью и стеком.