Федеральное государственное бюджетное образовательное учреждение

высшего образования

"Пермский национальный исследовательский политехнический университет"

Лабораторная работа №12

по дисциплине:  
Основы алгоритмизации и программирования

Семестр 1, Вариант №1

Выполнил: Ильиных А. А.

Группа: АСУ-19-1бзу

Проверила: доцент кафедры ИТАС Полякова О.А.

Пермь

2020 г.

# 

Лабораторная работа № 12

"Информационные динамические структуры"

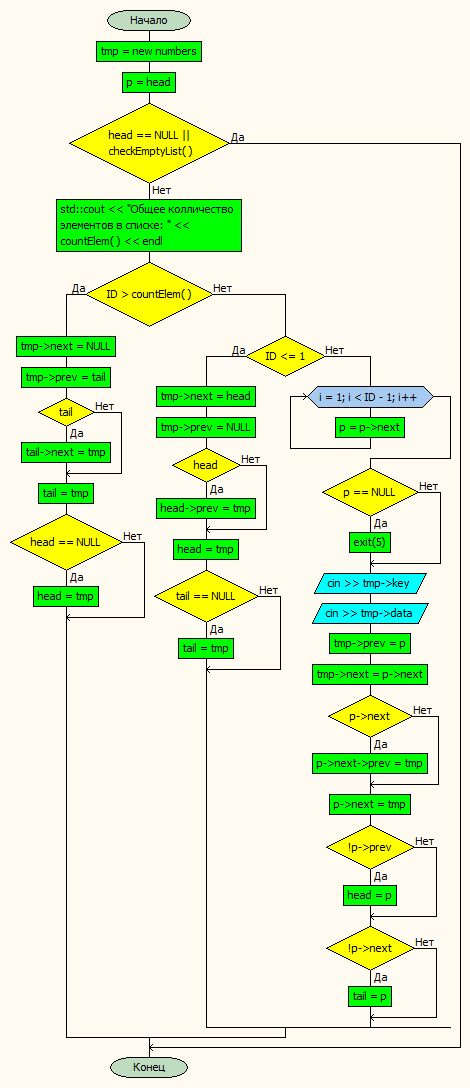
Постановка задачи:

1. Записи в линейном списке содержат ключевое поле типа int. Сформировать однонаправленный список. Удалить из него элемент с заданным номером, добавить элемент с заданным номером;

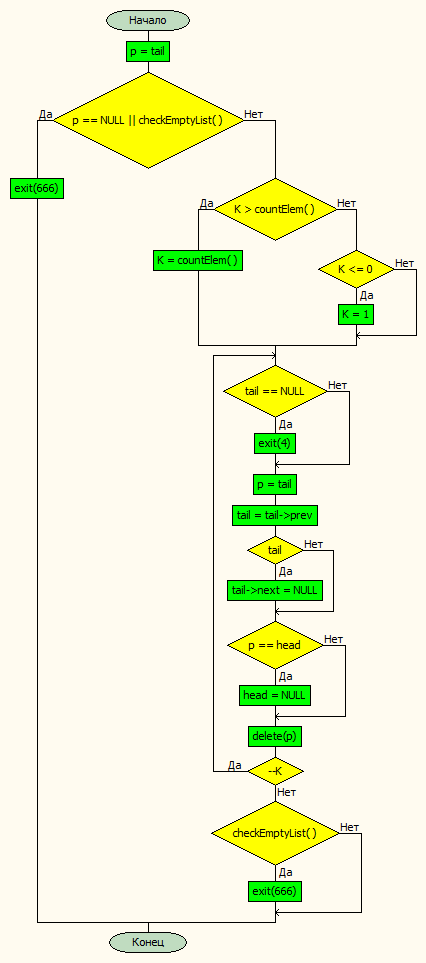
Решение задачи:

Блок схема алгоритма решения задачи:

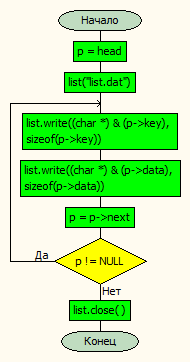
Вставка элемента с заданным номером:



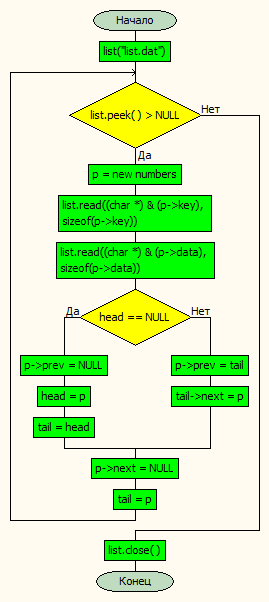
Удаление элемента с заданным номером:



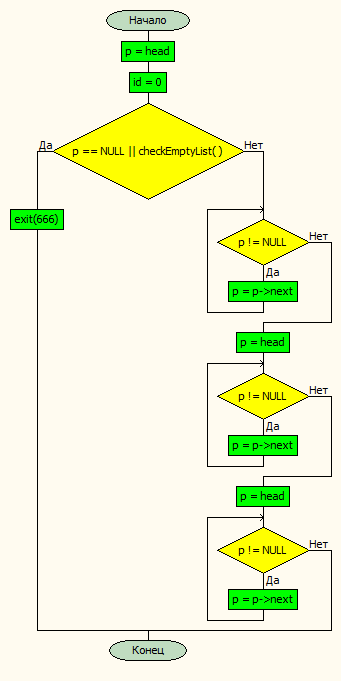
Вывод списка в файл:



Ввод списка из файла:



Вывод списка:



Программа решения задачи на С++:

#include <stdio.h>

#include <conio.h>

#include <fstream>

#include <iostream>

#include <locale>

#define NULL 0

#pragma warning(disable : 4996)

using namespace std;

struct numbers {

int key;

int data;

numbers\* next;

numbers\* prev;

} \*head, \* tail;

void createClearList() {

numbers\* p = head;

p = NULL;

head = tail = NULL;

std::cout << "Пустой список создан" << endl << endl;

}

bool checkEmptyList() {

bool CL = false;

numbers\* p = head;

if ((head && tail) == NULL) CL = true;

return CL;

}

void insertElement() {

numbers\* p = new numbers;

std::cout << "Введите новый ключевой элемент: ";

std::cin >> p->key;

std::cout << "Введите новые данные: ";

std::cin >> p->data;

if (head == NULL) {

p->prev = NULL;

head = p;

tail = head;

}

else {

p->prev = tail;

tail->next = p;

}

p->next = NULL;

tail = p;

std::cout << endl;

}

void insertElements() {

bool canContinue = true;

do {

insertElement();

std::cout << "Продолжить (y/n): ";

char ans;

std::cin >> ans;

std::cout << endl;

while (ans != 'y' && ans != 'n') {

std::cout << "Ошибка ввода" << endl;

std::cout << "Продолжить (y/n): ";

std::cin >> ans;

std::cout << endl;

}

if (ans == 'y') canContinue = true;

else if (ans == 'n') canContinue = false;

} while (canContinue);

}

int countElem() {

numbers\* p = head;

int count = 0;

while (p != NULL) {

count++;

p = p->next;

}

return count;

}

void insertElementByID() {

numbers\* tmp = new numbers;

numbers\* p = head;

if (head == NULL || checkEmptyList()) {

std::cout << "Список пуст. Вставка невозможна!" << endl;

}

else {

int ID;

std::cout << "Общее колличество элементов в списке: " << countElem() << endl;

std::cout << "Введите номер добавляемого элемента: ";

std::cin >> ID;

std::cout << endl;

if (ID > countElem()) {

std::cout << "Введите новый ключевой элемент: ";

std::cin >> tmp->key;

std::cout << "Введите новые данные: ";

std::cin >> tmp->data;

tmp->next = NULL;

tmp->prev = tail;

if (tail) tail->next = tmp;

tail = tmp;

if (head == NULL) head = tmp;

std::cout << endl;

}

else if (ID <= 1) {

std::cout << "Введите новый ключевой элемент: ";

std::cin >> tmp->key;

std::cout << "Введите новые данные: ";

std::cin >> tmp->data;

tmp->next = head;

tmp->prev = NULL;

if (head) head->prev = tmp;

head = tmp;

if (tail == NULL) tail = tmp;

std::cout << endl;

}

else {

for (int i = 1; i < ID - 1; i++) p = p->next;

if (p == NULL) exit(5);

std::cout << "Введите новый ключевой элемент: ";

cin >> tmp->key;

std::cout << "Введите новые данные: ";

cin >> tmp->data;

tmp->prev = p;

tmp->next = p->next;

if (p->next) p->next->prev = tmp;

p->next = tmp;

if (!p->prev) head = p;

if (!p->next) tail = p;

std::cout << endl;

}

}

}

void deleteKEndElem() {

numbers\* p = tail;

int K;

if (p == NULL || checkEmptyList()) {

std::cout << "Список пуст. Удаление невозможно!" << endl;

exit(666);

}

else {

std::cout << "Введите количество удаляемых с конца элементов: ";

std::cin >> K;

if (K > countElem()) K = countElem();

else if (K <= 0) K = 1;

do {

if (tail == NULL) exit(4);

p = tail;

tail = tail->prev;

if (tail) tail->next = NULL;

if (p == head) head = NULL;

delete(p);

} while (--K);

if (checkEmptyList()) {

std::cout << endl << "Удален последний элемент!!!" << endl << endl;

std::cout << "Дальнейшая работа невозможна!!!" << endl;

exit(666);

}

}

}

void deleteAllElem() {

numbers\* p = head;

if (p == NULL || checkEmptyList()) {

std::cout << "Список пуст. Удаление невозможно!" << endl;

exit(666);

}

else {

while (p != NULL) {

head = p->next;

delete p;

p = head;

}

}

if (checkEmptyList()) std::cout << "Список пуст" << endl << endl;

}

void printList() {

numbers\* p = head;

int id = 0;

if (p == NULL || checkEmptyList()) {

std::cout << "Список Пуст. Печать невозможна!" << endl;

exit(666);

}

else {

std::cout << "Список имеет вид: " << endl << endl;

while (p != NULL) {

++id;

std::cout << " | " << id << " | ";

p = p->next;

}

p = head;

std::cout << endl;

while (p != NULL) {

std::cout << " | " << "k : d" << " | ";

p = p->next;

}

p = head;

std::cout << endl;

while (p != NULL) {

std::cout << " | " << p->key << " : " << p->data << " | ";

p = p->next;

}

std::cout << endl << endl;

}

}

void fileInput(void) {

numbers\* p = head;

ofstream list("list.dat");

do {

list.write((char\*)&(p->key), sizeof(p->key));

list.write((char\*)&(p->data), sizeof(p->data));

p = p->next;

} while (p != NULL);

list.close();

}

void fileOutput(void) {

ifstream list("list.dat");

while (list.peek() > NULL) {

numbers\* p = new numbers;

list.read((char\*)&(p->key), sizeof(p->key));

list.read((char\*)&(p->data), sizeof(p->data));

if (head == NULL) {

p->prev = NULL;

head = p;

tail = head;

}

else {

p->prev = tail;

tail->next = p;

}

p->next = NULL;

tail = p;

}

list.close();

}

int main() {

setlocale(LC\_ALL, "rus");

createClearList();

insertElements();

printList();

insertElementByID();

printList();

deleteKEndElem();

printList();

fileInput();

deleteAllElem();

fileOutput();

printList();

return 0;

}

Результат выполнения программы:

