МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Лабораторная работа №7

по дисциплине «Программирование»

Указатель на функцию

Группа: АВТ-041

Студент: Антонов А.  
Преподаватель: Балакин В.

НОВОСИБИРСК 2021

**Задача**

Преобразовать функцию сортировки с использованием массивов (b), списков (6.3), деревьев (8.4, 8.5) в итератор. Проверить его работу на двух структурах данных содержащих указатели на различные типы (например, целые и строки). Массив преобразовать в массив указателей.

**Проектирование программы**

Идея: Написать отдельно функции сортировки для двух списков и реализовать их в виде итератора.

Составные части программы:

1. Структура для целочисленных элементов:

struct list1 {

int value;

list1\* next, \* prev;

};

1. Структура для символьных элементов:

struct list2 {

string value;

list2\* next, \* prev;

};

1. Не пустые 2 списка:

list1\* list11 = NULL;

list2\* list12 = NULL;

1. Добавление нового элемента (для int), заполнение списка:

cin >> q->value;

if (ph == NULL) { //добавление нового элемента

ph = q;

ph->next = NULL;

}

else {

for (; p->next != NULL; p = p->next);

q->prev = p;

p->next = q;

}

1. Добавление нового элемента (для string), заполнение списка:

cin >> q->value;

if (ph == NULL) {

ph = q;

ph->next = NULL;

}

else {

for (; p->next != NULL; p = p->next);

q->prev = p;

p->next = q;

}

1. Вывод элементов в консоль (show1 - для int, show2 - для string):

void show1(list1\* p) { //вывод в консоль

for (; p != NULL; p = p->next) {

cout << p->value << " ";

}

}

void show2(list2\* p) {

for (; p != NULL; p = p->next) {

cout << p->value << " ";

}

}

1. Цикл для поиска минимального элемента:

for (; p != NULL; p = p->next) {

if (q->value > p->value) q = p;

}

1. Условие в сортировке для перестановки указателей элементов:

if (q->next != NULL) q->next->prev = q->prev;

q->prev->next = q->next;

q->next = p;

q->prev = p->prev;

if (p->prev != NULL) p->prev->next = q;

p->prev = q;

1. Итератор:

void sort(void (\*f)(), int v) { // итератор

if (v == 0) {

(\*f)(); // указатель на контейнерный класс

}

else if (v == 1) {

(\*f)(); // указатель на контейнерный класс

}

}

1. Функция main:

int main()

{

setlocale(LC\_ALL, "Russian");

int n, vv;

cout << "введите количество элементов: ";

cin >> n;

cout << "выберите: int - 0, string - 1 => ";

cin >> vv;

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

if (vv == 0) {

add1(list11);

}

else if (vv == 1) {

add2(list12);

}

}

if (vv == 0) {

sort(sort1, vv);

show1(list11);

}

else {

sort(sort2, vv);

show2(list12);

}

return 0;

}

Переменные:

int n - кол-во элементов

int vv - выбор типа данных

**Текст программы с комментариями:**

#include <iostream>

#include <string>

using namespace std;

struct list1 {

int value;

list1\* next, \* prev;

};

struct list2 {

string value;

list2\* next, \* prev;

};

list1\* list11 = NULL;

list2\* list12 = NULL;

//добавление очередного элемента в целочисленный список

void add1(list1\*& ph) { // добавление целочисленных переменных

list1\* p = ph;

list1\* q = new list1;

q->next = NULL;

q->prev = NULL;

//инт

cin >> q->value;

if (ph == NULL) { //добавление нового элемента

ph = q;

ph->next = NULL;

}

else {

for (; p->next != NULL; p = p->next);

q->prev = p;

p->next = q;

}

}

void add2(list2\*& ph) { //добавление символов

list2\* p = ph;

list2\* q = new list2;

q->next = NULL;

q->prev = NULL;

cin >> q->value;

if (ph == NULL) {

ph = q;

ph->next = NULL;

}

else {

for (; p->next != NULL; p = p->next);

q->prev = p;

p->next = q;

}

}

void show1(list1\* p) { //вывод в консоль

for (; p != NULL; p = p->next) {

cout << p->value << " ";

}

}

void show2(list2\* p) {

for (; p != NULL; p = p->next) {

cout << p->value << " ";

}

}

list1\* min1(list1\* p) { //поиск минимального целочисленного значения

list1\* q = p;

for (; p != NULL; p = p->next) {

if (q->value > p->value) q = p;

}

return q;

}

list2\* min2(list2\* p) { // поиск минимального символьного значения

list2\* q = p;

for (; p != NULL; p = p->next) {

if (q->value > p->value) q = p;

}

return q;

}

void sort1() { // сортировка целочисленных элементов

list1\* p, \* q;

p = list11;

while (p != NULL) {

q = min1(p);

if (p == list11) list11 = q;

if (q == p) {

p = p->next;

continue;

}

if (q->next != NULL) q->next->prev = q->prev;

q->prev->next = q->next;

q->next = p;

q->prev = p->prev;

if (p->prev != NULL) p->prev->next = q;

p->prev = q;

}

}

void sort2() { // сортировка символьных элементов

list2\* p, \* q;

p = list12;

while (p != NULL) {

q = min2(p);

if (p == list12) list12 = q;

if (q == p) {

p = p->next;

continue;

}

if (q->next != NULL) q->next->prev = q->prev;

q->prev->next = q->next;

q->next = p;

q->prev = p->prev;

if (p->prev != NULL) p->prev->next = q;

p->prev = q;

}

}

void sort(void (\*f)(), int v) { // итератор

if (v == 0) {

(\*f)(); // указатель на контейнерный класс

}

else if (v == 1) {

(\*f)(); // указатель на контейнерный класс

}

}

int main()

{

setlocale(LC\_ALL, "Russian");

int n, vv;

cout << "введите количество элементов: ";

cin >> n;

cout << "выберите: int - 0, string - 1 => ";

cin >> vv;

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

if (vv == 0) {

add1(list11);

}

else if (vv == 1) {

add2(list12);

}

}

if (vv == 0) {

sort(sort1, vv);

show1(list11);

}

else {

sort(sort2, vv);

show2(list12);

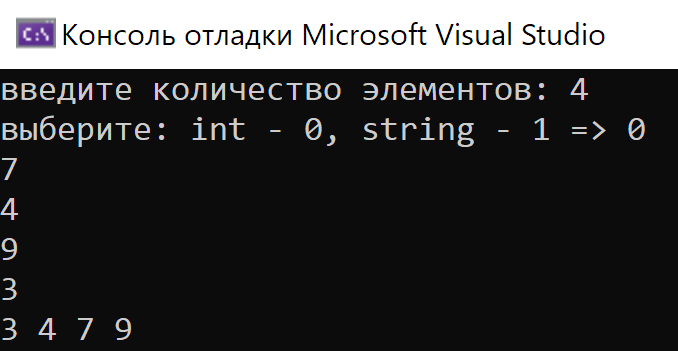
}

return 0;

}

**Пример работы программы:**

**Входные данные:** 4 0 7 4 9 3



**Вывод:**

Функция была преобразованна, программа работает корректно.