Выполнил

студент КТбо1-4 \_\_\_\_\_\_\_\_\_\_\_\_\_ А. А. Воронов

Принял

ст. преподаватель каф. САПР \_\_\_\_\_\_\_\_\_\_\_\_\_ В. И. Данильченко

Таганрог 2023

ЛАБОРАТОРНАЯ РАБОТА №5

по дисциплине «Алгоритмизация и программирование»

**«Работа со списком структур»**

Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования

«ЮЖНЫЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

(ФГАОУ ВО «ЮФУ»)

1. **Цель лабораторной работы:** Научиться с работать со списками структур в языке С++, научиться работать с односвязным списком.
2. **Задача:** Написать программу, которая записывает с клавиатуры в файл структуру: «Пациент»: фамилия, имя, отчество, год рождения, телефон, адрес, номер медицинской карты, группа крови, резус-фактор, диагноз. В программе требуется реализовать: а) дополнение существующего списка структур новыми структурами; б) поиск структуры с заданным значением выбранного элемента; в) вывод на экран содержимого списка структур; г) упорядочение массива структур по заданному полю (элементу).
3. **Ход работы:** я создал новый проект в программе «Visual Studio 2022» и написал программу, которая создает односвязный список структур *lst* и принимает на вход данные, вводимые с клавиатуры в консоль, записывая их в структуру *person*. В программе реализованы такие основные функции для работы с односвязным списком как: *push\_front* – добавляет узел в начало списка, *push\_back*- добавляет узел в конец списка, *pop\_front* – удаляет первый в списке узел, *pop\_back* – удаляет последний в списке узел, а также *print* – выводит содержимое структур, записанных в узлы, *write* – записывает содержимое структур в файл *patients.txt* через символ табуляции, *scan* - выполняет поиск в списке по фамилии пациента и выводит на экран информацию о нем, *sort* – сортирует узлы в списке по убыванию номера полиса пациента и *creator* – функция, позволяющая добавить нужное количество пациентов в список (количество вводится пользователем).

**Программный код:**

#include <iostream>

#include <string>

#include <fstream>

using namespace std;

struct person {

string lname, name, ptrnmc, birth\_y, ph\_num, adress,polis, bld\_gp, rh\_factor, diag;

void p\_write(ofstream& file) {

file

<< lname << "\t"

<< name << "\t"

<< ptrnmc << "\t"

<< birth\_y << "\t"

<< ph\_num << "\t"

<< adress << "\t"

<< polis << "\t"

<< bld\_gp << "\t"

<< rh\_factor << "\t"

<< diag << "\n";

}

void p\_print() {

cout

<<"\n"

<< "Data: " << "\n"

<< "Surname " << lname << "\n"

<< "Name: " << name << "\n"

<< "Patronymic: " << ptrnmc << "\n"

<< "Year: " << birth\_y << "\n"

<< "Phone number: " << ph\_num << "\n"

<< "Adress: " << adress << "\n"

<< "Polis No: " << polis << "\n"

<< "Blood group: " << bld\_gp << "\n"

<< "RH-factor: " << rh\_factor << "\n"

<< "Diagnosis: " << diag << "\n\n";

}

};

struct Node {

person val;

Node\* next;

Node(person val) {

this->val = val;

this->next = NULL;

}

};

person assign(person x);

struct list { //односвязный список

Node\* first, \* last;

list() : first(nullptr), last(nullptr) {}

void push\_back(person data) { // добавляет в конец списка

Node\* node = new Node(data);

if (first == NULL) first = node;

if (last != NULL) last->next = node;

last = node;

}

void push\_front(person data) { // добавляет в начало списка

Node\* node = new Node(data);

node->next = first;

first = node;

if (last = NULL) last = node;

}

void pop\_back() { // удаляет последний объект

if (last == NULL) return;

if (first == last) {

delete last;

first = last = NULL;

return;

}

Node\* node = first;

for (; node->next != last; node = node->next);

node->next = NULL;

delete last;

last = node;

}

void pop\_front() { // удаляет первый объект

if (first == NULL) return;

if (first == last) {

delete last;

first = last = NULL;

return;

}

Node\* node = first;

first = node->next;

delete node;

}

void print() {

Node\* node = first;

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

node->val.p\_print();

}

}

void write(ofstream& file) {

Node\* node = first;

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

node->val.p\_write(file);

}

void scan(string sname) { //поиск по фамилии

Node\* node = first;

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

if (node->val.lname == sname) node->val.p\_print();

}

}

void sort() {

bool swapped = true;

Node\* node, \*last = nullptr;

if (first == nullptr)

return;

while (swapped) {

swapped = false;

node = first;

while (node->next != last) {

if (node->val.polis > node->next->val.polis) {

swap(node->val, node->next->val);

swapped = true;

}

node = node->next;

}

last = node;

}

}

void creator(int amount) {

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

person p;

p = assign(p);

push\_back(p);

}

}

};

person assign(person x) {

getline(cin, x.lname, ' ');

getline(cin, x.name, ' ');

getline(cin, x.ptrnmc, ' ');

getline(cin, x.birth\_y, ' ');

getline(cin, x.ph\_num, ' ');

getline(cin, x.adress, ' ');

getline(cin, x.polis, ' ');

getline(cin, x.bld\_gp, ' ');

getline(cin, x.rh\_factor, ' ');

getline(cin, x.diag);

return x;

}

//----------------------------------------------------

int main() {

list lst;

int n1;

string n,s;

cout << "Enter ";

getline(cin, n);

n1 = stoi(n);

lst.creator(n1);

ofstream f;

f.open("patients.txt");

lst.write(f);

cout << "\n";

cin >> s;

lst.scan(s);

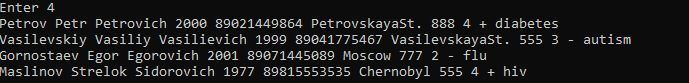
lst.sort();

lst.print();

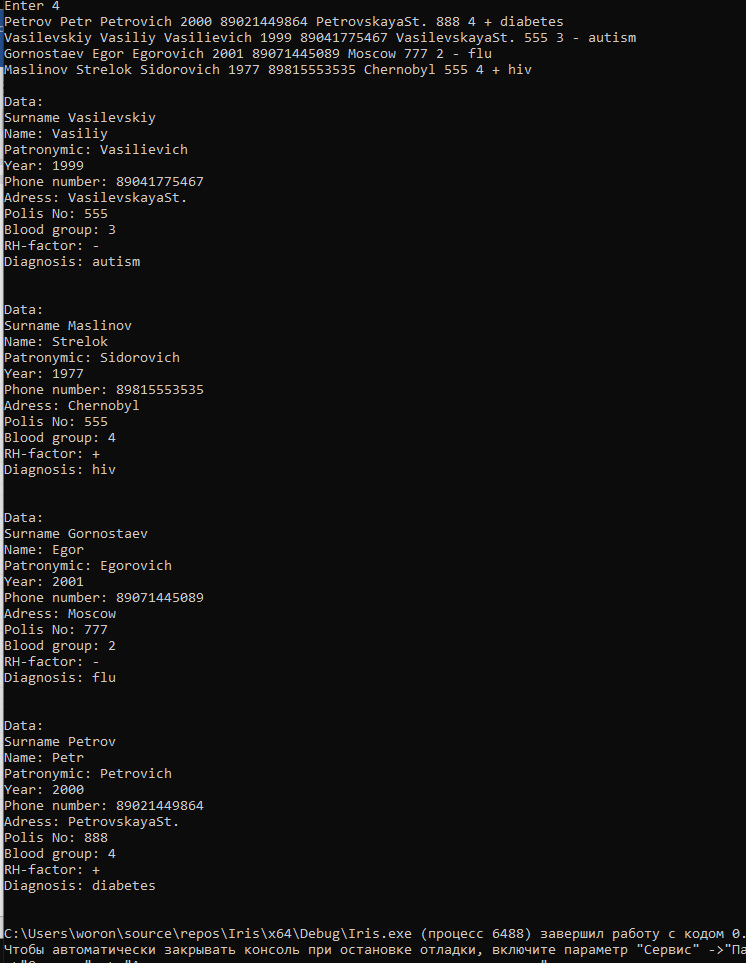
f.close();

}

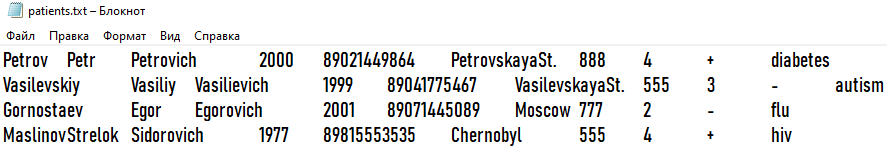
**Входные данные программы:**

****

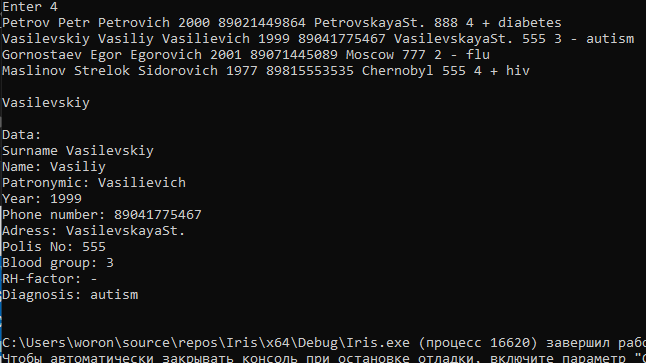
**Вывод отсортированного списка и запись его в файл:**

****

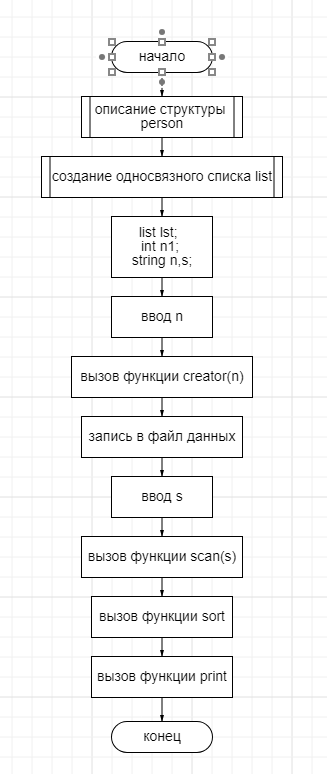
**Пример записи в файл (без сортировки):**

****

**Вывод данных по запросу (поиск по фамилии):**

****

1. **Блок-схема:**

****

**5. Вывод:** в ходе работы были выполнены все цели и задачи, было выяснено, как работать со списком структур и создавать односвязный список в языке С++, было реализовано: а) дополнение существующего списка структур новыми структурами; б) поиск структуры с заданным значением выбранного элемента; в) вывод на экран содержимого списка структур; г) упорядочение массива структур по заданному полю (элементу).

**6. Источники:**

1. [Работа со списком структур.pptx](https://sfedu.sharepoint.com/:p:/r/sites/1-423/Shared%20Documents/General/C1/%D0%9B5/%D0%A0%D0%B0%D0%B1%D0%BE%D1%82%D0%B0%20%D1%81%D0%BE%20%D1%81%D0%BF%D0%B8%D1%81%D0%BA%D0%BE%D0%BC%20%D1%81%D1%82%D1%80%D1%83%D0%BA%D1%82%D1%83%D1%80.pptx?d=wc597dd81cd5b48eb897db93462f6007e&csf=1&web=1&e=q3bZep)

2. <https://prog-cpp.ru/data-ols/>