# ФЕДЕРАЛЬНОЕ АГЕНТСТВО ВОЗДУШНОГО ТРАНСПОРТА

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ

«МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ГРАЖДАНСКОЙ АВИАЦИИ»

Кафедра вычислительных машин, комплексов, систем и сетей

Отчет защищен с оценкой

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись преподавателя, дата)

ОТЧЕТ

по результатам выполнения вычислительной практики

по дисциплине «Алгоритмические языки и программирование»

Выполнил студент группы ЭВМ 1-2

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(Ф.И.О., № зачетной книжки)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись, дата)

Москва 2011

**Предисловие**

В данной работе представлены результаты выполнения летней практики по дисциплине ПнЯВУ, включающие в себя отчёты по каждому практическому занятию (всего 5 занятий).

Каждый отчёт включает в себя название, цель работы, техническое задание, вариант задания, структуру программы, разработку алгоритмов программы, таблицы локальных и глобальных переменных программы и два приложения с листингом программы и с её тестированием соответственно.

**Практическая работа №1**

**1.1 Цель работы**

Освоение методов работы с бинарными и текстовыми файлами, освоение алгоритмов удаления данных из бинарного файла.

**1.2 Техническое задание**

Разработка схемы алгоритмов, структуру и текст программы для обработки структурных данных текстового и бинарного файлов заданного варианта с использованием функций создания файлов и удаления ряда структур из файлов.

Схемы алгоритмов должны быть разработаны для каждой функции.

Тестирование проводить с помощью файла данных с поисковыми признаками для удаления: должны быть удалены соответствующие записи.

**1.3 Вариант задания**

Структура: Фамилия и инициалы, табельный номер, стаж работы, оклад в тыс. руб., дата рождения.

**1.4 Структура программы**

Программа содержит следующие функции:

**bin** – Функция создания бинарного файла.

**del4 –** Функция удаления записей из бинарного файла.

**sozdat –** Функция создания текстового файла.

**cht –** Функция чтения бинарного файла.

**dopq –** Функция дополнения очереди.

**chtq –** Функция чтения очереди.

**filtr –** Функция для удаления пробелов.

**kor –** Функция для записи из бинарного в текстовый файл.

**p –** Функция форматного(в виде элемента таблицы) вывода структуры на экран.



**Рисунок 1.1 –** Структура программы.

**1.5 Разработка алгоритмов программы**

**1.5.1** Алгоритм функции **bin**.

Функция **bin** используется для создания нового бинарного файла из уже существующего текстового.

Текст функции приведён в **приложении А**. Алгоритм функции представлен на **рисунке 1.2**. В функции используются глобальные и локальные переменные, которые приведены в **таблицах 1.1 и 1.2.**



**Рисунок 1.2** – Схема алгоритма функции **bin**.

**1.5.2** Алгоритм функции **cht.**

Функция **cht** используется для чтения бинарного файла.

Текст функции приведён в **приложении А**. Алгоритм функции представлен на **рисунке 1.3**. В функции используются глобальные и локальные переменные, которые приведены в **таблицах 1.1 и 1.2.**



**Рисунок 1.3** – Схема алгоритма функции **cht.**

**1.5.3** Алгоритм функции **del4.**

Функция **del4** используется для удаления записей из файла.

Текст функции приведён в **приложении А**. Алгоритм функции представлен на **рисунке 1.4**. В функции используются глобальные и локальные переменные, которые приведены в **таблицах 1.1 и 1.2.**



**Рисунок 1.4** – Схема алгоритма функции **del4.**



**Рисунок 1.4** – Продолжение



**Рисунок 1.4** – Продолжение

**1.5.4** Алгоритм функции **dopq.**

Функция **dopq** используется для дополнения очереди.

Текст функции приведён в **приложении А**. Алгоритм функции представлен на **рисунке 1.5**. В функции используются глобальные и локальные переменные, которые приведены в **таблицах 1.1 и 1.2.**



**Рисунок 1.5** – Схема алгоритма функции **dopq.**

**1.5.5** Алгоритм функции **chtq.**

Функция **chtq** используется для чтения очереди.

Текст функции приведён в **приложении А**. Алгоритм функции представлен на **рисунке 1.6**. В функции используются глобальные и локальные переменные, которые приведены в **таблицах 1.1 и 1.2.**



**Рисунок 1.6** – Схема алгоритма функции **chtq.**

**1.5.6** Алгоритм функции **filtr.**

Функция **filtr** используется удаления пробелов.

Текст функции приведён в **приложении А**. Алгоритм функции представлен на **рисунке 1.7**. В функции используются глобальные и локальные переменные, которые приведены в **таблицах 1.1 и 1.2.**



**Рисунок 1.7** – Схема алгоритма функции **filtr.**

**1.5.7** Алгоритм функции **pri.**

Функция **pri** используется для вывода сообщения на экран.

Текст функции приведён в **приложении А**. Алгоритм функции представлен на **рисунке 1.8**. В функции используются глобальные и локальные переменные, которые приведены в **таблицах 1.1 и 1.2.**



**Рисунок 1.8** – Схема алгоритма функции **pri.**

**1.5.8** Алгоритм функции **p.**

Функция **p** используется для форматного(в виде элемента таблицы) вывода структуры на экран.

Текст функции приведён в **приложении А**. Алгоритм функции представлен на **рисунке 1.9**. В функции используются глобальные и локальные переменные, которые приведены в **таблицах 1.1 и 1.2.**



**Рисунок 1.9** – Схема алгоритма функции **p.**

**1.5.9** Алгоритм функции **kor.**

Функция **kor** используется для записи из бинарного в текстовый файл.

Текст функции приведён в **приложении А**. Алгоритм функции представлен на **рисунке 1.10**. В функции используются глобальные и локальные переменные, которые приведены в **таблицах 1.1 и 1.2.**



**Рисунок 1.10** – Схема алгоритма функции **kor.**

**1.5.10** Алгоритм функции **sozdat.**

Функция **sozdat** используется для создания текстового файла с данными с клавиатуры. Текст функции приведён в **приложении А**. Алгоритм функции представлен на **рисунке 1.11**. В функции используются глобальные и локальные переменные, которые приведены в **таблицах 1.1 и 1.2.**

**Рисунок 1.11** – Схема алгоритма функции **sozdat.**

**1.5.11** Алгоритм главной функции **main.**

Текст функции приведён в **приложении А**. Алгоритм функции представлен на **рисунке 1.12**.



**Рисунок 1.12 -** Основной алгоритм программы.

**1.6 Таблицы локальных и глобальных переменных программы.**

В **таблице 1.1** приведены глобальные переменные программы. В **таблице 1.2** приведены локальные переменные программы.

**Таблица 1.1 –** Глобальные переменные программы.

|  |  |  |
| --- | --- | --- |
| Имя | Тип | Назначение |
| man | men | Промежуточная переменная структурного типа, используемая для чтения и записи. |
| z | int | Размер структуры типа men в байтах. |
| sh[] | char\* | Массив содержащий «шапку» таблицы. |
| fin | ifstream | Стандартный входной файловый поток. |
| fout | ofstream | Стандартный выходной файловый поток. |
| io | fstream | Стандартный двунаправленный файловый поток. |
| q | queue\* | Указатель на вершину очереди. |

**Таблица 1.2** – Локальные переменные программы.

|  |  |  |
| --- | --- | --- |
| Имя | Тип | Назначение |
| i,x | int | Используется в операторе цикла for. |
| n | int | Счетчик структур в бинарном файле. |
| s | int | Суммарных оклад. |
| binname | char\* | Имя бинарного файла. |
| datname,  textname | char\* | Имя текстового файла. |
| data | men | Переменная струкрутного типа для заполнения очереди данными. |
| X1 | int | Позиция первого не пробельного символа слева. |
| X2 | int | Позиция первого не пробельного символа справа. |
| Source[255] | char | Вспомогательная строка. |
| p | queue\* | Вспомогательная переменная, для организации просмотра очереди. |
| q | queue\* | Вспомогательная переменная, для организации просмотра очереди. |
| nst | queue\* | Добавляемая структура. |

В **приложении А** приведён текст программы «PR1.CPP».

В **приложении Б** представлен текст файла с поисковыми признаками для удаления «DELDAT.DAT» и результаты тестирования программы.

**Приложение А**

#include <conio.h>

#include <iostream.h>

#include <stdlib.h>

#include <string.h>

#include <iomanip.h>

#include <fstream.h>

#include <stdio.h>

struct men

{

long numb;

char fio[20];

char DOB[50];

float pay;

float exp;

} man;

int z = sizeof(men);

char \*sh[] =

{

"\nСВЕДЕНИЯ О СОТРУДНИКАХ ОТДЕЛА",

"╔═══════════════════╦═════╦════════╦═══════════╦═══════════╗",

"║ Фамилия, инициалы ║Таб. ║ Стаж ║ Оклад ║ Дата ║",

"║ ║номер║ работы ║ в тыс.руб.║ рождения ║",

"╠═══════════════════╬═════╬════════╬═══════════╬═══════════╣",

"╚═══════════════════╩═════╩════════╩═══════════╩═══════════╝"

};

ifstream fin;

ofstream fout;

fstream io;

void filtr(char\*), sozdat(char\*),bin(char\*,char\*),

delsozd(char\*),del4(char\*,char\*), kor(char\*,char\*),cht(char\*),

p(men&);

istream& operator>>(istream& in, men& st)

{

cout << '\272';

in.getline(st.fio,21); filtr(st.fio);

in >> st.numb >>st.pay >> st.exp;

in.getline(st.DOB,50); filtr(st.DOB);

return in;

}

ostream& operator<<(ostream& out, men& st)

{

out<<setw(20)<<setiosflags(ios::left)<<st.fio<<setw(11)

<<setiosflags(ios::left)<<st.numb<<setw(10)<<setprecision(2)

<<st.pay<<setw(10)<<setprecision(2)<<st.exp<<setw(11)

<<setiosflags(ios::left)<<st.DOB<< endl; return out;

}

void main()

{

clrscr();

sozdat("sozdat.dat");

bin("sozdat.dat","bindat.dat");

clrscr();

cht("bindat.dat");

getch();

del4("bindat.dat", "deldat.dat");

cht("bindat.dat");

getch();

}

void pri(char\*s){cout << s << endl;}

void p(men& st)

{

cout<<setw(19)<<setiosflags(ios::left)<<st.fio<<'\272'<<setw(5)

<<setiosflags(ios::left)<<st.numb<<'\272'<<setw(8)<<setprecision(2)

<<st.pay<<'\272'<<setw(11)<<setprecision(2)<<st.exp<<'\272'<<setw(11)

<<setiosflags(ios::left)<<st.DOB<<'\272'<< endl;

}

void sozdat(char\* datname)

{

io.open(datname, ios::in|ios::out|ios::trunc);

//io.open(datname, ios::in|ios::out);

if (!io) {pri("Ошибка при открытии текстового файла!"); exit(1);}

do

{

clrscr();

for(int i=0; i<5; i++) cout << sh[i] << endl;

io.seekg(0,ios::beg);

io.seekp(0,ios::beg);

while(io.peek()!=EOF)

{

io >> man;

p(man);

}

io.seekg(0,ios::end);

io.seekp(0,ios::end);

cin >> man;

io << man;

cout <<"\nПродолжить? (y/n): \n";

} while(getch()=='y');

io.close();

}

void bin(char\* textname, char\* binname)

{

fin.open(textname);

if (!fin) {pri("Ошибка при открытии текстового файла!"); exit(1);}

fout.open(binname, ios::binary);

if (!fout) {pri("Ошибка при открытии бинарного файла!"); exit(1);}

while(fin.peek()!=EOF)

{

fin >> man;

fout.write((char\*)&man,z);

}

fout.close();

fin.close();

}

void filtr(char\*a)

{

char Source[255] = "";

int X1 = 0;

int X2 = strlen(a) - 1;

while(a[X1]==' ') X1++;

while(a[X2]==' ') X2--;

for(int x=X1; x<=X2; x++) Source[x-X1]=a[x];

Source[x-X1]='\0';

strcpy(a,Source);

}

struct queue

{

men data;

queue\* next;

} \*q=0;

void dopq(queue\*&S, men data)

{

queue\* p=S, \*q=0, \*nst = new queue;

nst->data = data;

nst->next = 0;

while(p){q=p;p=p->next;}

if(q) q->next = nst;else S = nst;

}

void chtq(queue\*S)

{

queue\* p=S;

while(p) {fout.write((char\*)&p->data,z); p=p->next;}

}

void del4(char\* binname, char\* delname)

{

fin.open(delname);

if(!fin) {pri("Ошибка при открытии файла данных с поисковыми признаками!(del4)");exit(1);}

io.open(binname,ios::binary|ios::in|ios::out);

if(!io) {pri("Ошибка открытия бинарного файла для удаления!(del4)"); exit(1);}

fout.open("temp.dat",ios::binary);

if(!fout){pri("Ошибка открытия бинарного файла!(del4)"); exit(1);}

cout << "\n\nУДАЛЕНИЕ СОТРУДНИКА ПО ФАМИЛИИ\n";

while(fin.peek()!=EOF)

{

char name[20];

fin.getline(name,20); filtr(name);

cout << "\nИщем данные сотрудника с фамилией - "<<"\"" << name << "\"" << endl;

if(!strcmp(name,"")){cout<<"Нет поискового признака для удаления!\n";continue;}

io.seekg(0,ios::beg);

while(io.peek()!=EOF)

{

io.read((char\*)&man, z);

if(strcmp(man.fio, name)==0)

{

strcpy(man.fio,"");

io.seekp(-z,ios::cur);

io.write((char\*)&man, z);

cout <<"Сотрудник с фамилией \""<< name << "\" успешно удален!" << endl;

goto m;

}

}

cout << "Сотрудник с фамилией \"" << name << "\" не найден!" << endl;

m:;

}

io.seekg(0,ios::beg);

io.seekp(0,ios::beg);

while(io.peek()!=EOF)

{

io.read((char\*)&man, z);

if(strcmp(man.fio,"")) dopq(q,man);

}

chtq(q);

fin.close(); io.close(); fout.close();

remove(binname);

rename("temp.dat",binname);

kor("sozdat2.dat",binname);

}

void kor(char\* datname, char\* binname)

{

io.open(binname,ios::in|ios::binary);

if(!io) {pri("Ошибка открытия бинарного файла для удаления!(kor)"); exit(1);}

fout.open(datname);

if(!fout){pri("Ошибка открытия текстового файла!(kor)"); exit(1);}

while(io.peek()!=EOF)

{

io.read((char\*)&man, z);

fout << man;

}

io.close(); fout.close();

}

void cht(char\* binname)

{

int n=0;

float s=0;

fin.open(binname, ios::binary);

if(!fin) {pri("Ошибка открытия бинарного файла для чтения!"); exit(1);}

for(int i=0; i<5; i++) cout << sh[i] << endl;

while(fin.peek()!=EOF)

{

fin.read((char\*)&man, z);

cout << '\272';p(man);

s+=man.pay;

n++;

}

cout << sh[5] << endl << "Суммарный оклад: " << s << endl

<< "Средний оклад: " << s/n << endl

<< "Количество структур в файле: " << n << endl;

fin.close();

}

**Приложение Б**

***Исдодные файлы.***

**deldat.dat**

Первый А.А.

Бывалов Т.Т.

Небывалов Н.Н.

***Тестирование программы.***

СВЕДЕНИЯ О СОТРУДНИКАХ ОТДЕЛА

╔═══════════════════╦═════╦════════╦═══════════╦═══════════╗

║ Фамилия, инициалы ║Таб. ║ Стаж ║ Оклад ║ Дата ║

║ ║номер║ работы ║ в тыс.руб.║ рождения ║

╠═══════════════════╬═════╬════════╬═══════════╬═══════════╣

║Первый А.А. ║1 ║2 ║10 ║10.04.1999 ║

║Бывалов Т.Т. ║2 ║3 ║11 ║11.12.1990 ║

║Кукарин Г.Г. ║3 ║12 ║23 ║11.23.1980 ║

║Кропкин H.H ║5 ║5 ║5 ║12.04.1980 ║

║Строкин E.E. ║1 ║2 ║2 ║12.09.2010 ║

╚═══════════════════╩═════╩════════╩═══════════╩═══════════╝

Суммарный оклад: 51

Средний оклад: 10.2

Количество структур в файле: 5

УДАЛЕНИЕ СОТРУДНИКА ПО ФАМИЛИИ

Ищем сотрудника с фамилией – «Первый А.А.»

Сотрудник с фамилией «Первый А.А.» успешно удалён!

Ищем сотрудника с фамилией – «Бывалов Т.Т.»

Сотрудник с фамилией «Бывалов Т.Т.» успешно удалён!

Ищем сотрудника с фамилией – «»

Нет поискового признака для удаления!

Ищем сотрудника с фамилией – «Небывалов Н.Н.»

Сотрудник с фамилией «Небывалов Н.Н.» не найден!

СВЕДЕНИЯ О СОТРУДНИКАХ ОТДЕЛА

╔═══════════════════╦═════╦════════╦═══════════╦═══════════╗

║ Фамилия, инициалы ║Таб. ║ Стаж ║ Оклад ║ Дата ║

║ ║номер║ работы ║ в тыс.руб.║ рождения ║

╠═══════════════════╬═════╬════════╬═══════════╬═══════════╣

║Кукарин Г.Г. ║3 ║12 ║23 ║11.23.1980 ║

║Кропкин H.H ║5 ║5 ║5 ║12.04.1980 ║

║Строкин E.E. ║1 ║2 ║2 ║12.09.2010 ║

╚═══════════════════╩═════╩════════╩═══════════╩═══════════╝

Суммарный оклад: 30

Средний оклад: 10

Количество структур в файле: 3

**Практическая работа №2**

**2.1 Цель работы**

1) Освоение методов внутренней сортировки.

2) Освоение приёмов отладки программ, выполняющих сортировку данных.

**2.2 Техническое задание**

Разработка схемы алгоритмов, структуру и текст программы для обработки данных бинарного файла с использованием функций сортировки структурированных данных заданным методом.

**2.3 Вариант задания**

Структура: Фамилия и инициалы, табельный номер, стаж работы, оклад в тыс. руб., дата рождения.

Метод: Работа с линейной связной структурой – списком, в которой структуры соединены в порядке возрастания или убывания ключевого признак – одного из элементов структур, составляющих список.

**2.4 Структура программы**

Программа содержит следующие функции:

**sozdat –** Функция создания текстового файла.

**p –** Функция форматного(в виде элемента таблицы) вывода структуры на экран.

**bin** – Функция создания бинарного файла.

**cht –** Функция чтения бинарного файла.

**sortbin** – Функция сортировки бинарного файла.

**dops** – Функция дополнения списка.

**sozbin** – Функция записи списка в бинарный файл.

**soztext** – Функция записи списка в текстовый файл.



**Рисунок 2.1 –** Структура программы.

**2.5 Разработка алгоритмов программы**

**2.5.1** Алгоритм функции **sortbin**.

Функция **sortbin** используется для сортировки бинарного файла по ключевому признаку.

Текст функции приведён в **приложении В**. Алгоритм функции представлен на **рисунке 2.2**. В функции используются глобальные и локальные переменные, которые приведены в **таблицах 2.1 и 2.2.**



**Рисунок 2.2** – Схема алгоритма функции **sortbin**.

**2.5.2** Алгоритм функции **dops**.

Функция **dops** используется для дополнения списка.

Текст функции приведён в **приложении В**. Алгоритм функции представлен на **рисунке 2.3**. В функции используются глобальные и локальные переменные, которые приведены в **таблицах 2.1 и 2.2.**



**Рисунок 2.3** – Схема алгоритма функции **dops**.

**2.5.3** Алгоритм функции **sozbin**.

Функция **sozbin** используется для создания бинарного файла посредством записи в него списка.

Текст функции приведён в **приложении В**. Алгоритм функции представлен на **рисунке 2.4**. В функции используются глобальные и локальные переменные, которые приведены в **таблицах 2.1 и 2.2.**



**Рисунок 2.4** – Схема алгоритма функции **sozbin**.

**2.5.4** Алгоритм функции **soztext**.

Функция **soztext** используется для создания текстового файла посредством записи в него списка.

Текст функции приведён в **приложении В**. Алгоритм функции представлен на **рисунке 2.5**. В функции используются глобальные и локальные переменные, которые приведены в **таблицах 2.1 и 2.2.**



**Рисунок 2.5** – Схема алгоритма функции **soztext.**

**2.5.5** Алгоритм главной функции **main.**

Текст функции приведён в **приложении В**. Алгоритм функции представлен на **рисунке 2.6**.



**Рисунок 2.6 -** Основной алгоритм программы.

**2.6 Таблицы локальных и глобальных переменных программы.**

В **таблице 2.1** приведены глобальные переменные программы. В **таблице 2.2** приведены локальные переменные программы.

**Таблица 2.1 –** Глобальные переменные программы.

|  |  |  |
| --- | --- | --- |
| Имя | Тип | Назначение |
| man | men | Промежуточная переменная структурного типа, используемая для чтения и записи. |
| z | int | Размер структуры типа men в байтах. |
| sh[] | char\* | Массив содержащий «шапку» таблицы. |
| fin | ifstream | Стандартный входной файловый поток. |
| fout | ofstream | Стандартный выходной файловый поток. |
| io | fstream | Стандартный двунаправленный файловый поток. |
| s | spisok\* | Указатель на вершину списка. |

**Таблица 2.2** – Локальные переменные программы.

|  |  |  |
| --- | --- | --- |
| Имя | Тип | Назначение |
| i,x | int | Используется в операторе цикла for. |
| n | int | Счетчик структур в бинарном файле. |
| s | int | Суммарных оклад. |
| binname | char\* | Имя бинарного файла. |
| datname,  textname | char\* | Имя текстового файла. |
| data | men | Переменная структурного типа для заполнения списка данными. |
| X1 | int | Позиция первого не пробельного символа слева. |
| X2 | int | Позиция первого не пробельного символа справа. |
| Source[255] | char | Вспомогательная строка. |
| p | queue\* | Вспомогательная переменная, для организации просмотра списка. |
| q | queue\* | Вспомогательная переменная, для организации просмотра списка. |
| nst | queue\* | Добавляемая структура. |

В **приложении В** приведён текст программы «PR2.CPP».

В **приложении Г** представлены результаты тестирования программы.

**Приложение В**

#include <conio.h>

#include <iostream.h>

#include <stdlib.h>

#include <string.h>

#include <iomanip.h>

#include <fstream.h>

#include <stdio.h>

struct men

{

long numb;

char fio[20];

char DOB[50];

float pay;

float exp;

} man;

int z = sizeof(men);

struct spisok

{

men data;

spisok\* next;

} \*s=0;

char \*sh[] =

{

"\nСВЕДЕНИЯ О СОТРУДНИКАХ ОТДЕЛА",

"╔═══════════════════╦═════╦════════╦═══════════╦═══════════╗",

"║ Фамилия, инициалы ║Таб. ║ Стаж ║ Оклад ║ Дата ║",

"║ ║номер║ работы ║ в тыс.руб.║ рождения ║",

"╠═══════════════════╬═════╬════════╬═══════════╬═══════════╣",

"╚═══════════════════╩═════╩════════╩═══════════╩═══════════╝"

};

ifstream fin;

ofstream fout;

fstream io;

void filtr(char\*), sozdat(char\*),bin(char\*,char\*),cht(char\*),p(men&);

istream& operator>>(istream& in, men& st)

{

cout << '\272';

in.getline(st.fio,21); filtr(st.fio);

in >> st.numb >>st.pay >> st.exp;

in.getline(st.DOB,50); filtr(st.DOB);

return in;

}

ostream& operator<<(ostream& out, men& st)

{

out<<setw(20)<<setiosflags(ios::left)<<st.fio<<setw(11)

<<setiosflags(ios::left)<<st.numb<<setw(10)<<setprecision(2)

<<st.pay<<setw(10)<<setprecision(2)<<st.exp<<setw(11)

<<setiosflags(ios::left)<<st.DOB<< endl; return out;

}

void dops(spisok\*&S, men data)

{

spisok\*p = S,\*q=0,\*nst=new spisok;

nst->data=data;

while(p && (p->data.numb<data.numb))

{q=p; p=p->next;};

nst->next=p;

if(q) q->next=nst;else S=nst;

}

void sozbin(spisok\*S,char\* binname)

{

fout.open(binname,ios::binary);

if(!fout){cout<<"Ошибка при откритии бинарного файла!";exit(1);}

spisok\*q=S;

while(q)

{

fout.write((char\*)&q->data,z);

q=q->next;

}

fout.close();

}

void soztext(spisok\*S,char\* textname)

{

fout.open(textname);

if(!fout){cout<<"Ошибка при откритии текстового файла!";exit(1);}

spisok\*q=S;

while(q)

{

fout << q->data;

q=q->next;

}

fout.close();

}

void sortbin(spisok\*S,char\* binname,char\* textname)

{

fin.open(binname,ios::binary);

if(!fin){cout<<"Ошибка при откритии бинарного файла!";exit(1);}

while(fin.peek()!=EOF)

{

fin.read((char\*)&man,z);

dops(S,man);

}

fin.close();

sozbin(S,binname);

soztext(S,textname);

}

void main()

{

clrscr();

sozdat("sozdat.dat");

bin("sozdat.dat","bindat.dat");

clrscr();

cht("bindat.dat");

sortbin(s, "bindat.dat", "sozdat.dat");

cht("bindat.dat");

getch();

}

void pri(char\*s){cout << s << endl;}

void p(men& st)

{

cout<<setw(19)<<setiosflags(ios::left)<<st.fio<<'\272'<<setw(5)

<<setiosflags(ios::left)<<st.numb<<'\272'<<setw(8)<<setprecision(2)

<<st.pay<<'\272'<<setw(11)<<setprecision(2)<<st.exp<<'\272'<<setw(11)

<<setiosflags(ios::left)<<st.DOB<<'\272'<< endl;

}

void bin(char\* textname, char\* binname)

{

fin.open(textname);

if (!fin) {pri("Ошибка при открытии текстового файла!"); exit(1);}

fout.open(binname, ios::binary);

if (!fout) {pri("Ошибка при открытии бинарного файла!"); exit(1);}

while(fin.peek()!=EOF)

{

fin >> man;

fout.write((char\*)&man,z);

}

fout.close();

fin.close();

}

void sozdat(char\* datname)

{

io.open(datname, ios::in|ios::out|ios::trunc);

//io.open(datname, ios::in|ios::out);

if (!fout) {pri("Ошибка при открытии текстового файла!"); exit(1);}

do

{

clrscr();

for(int i=0; i<5; i++) cout << sh[i] << endl;

io.seekg(0,ios::beg);

io.seekp(0,ios::beg);

while(io.peek()!=EOF)

{

io >> man;

p(man);

}

io.seekg(0,ios::end);

io.seekp(0,ios::end);

cin >> man;

io << man;

cout <<"\nПродолжить? (y/n): \n";

} while(getch()=='y');

io.close();

}

void filtr(char\*a)

{

char Source[255] = "";

int X1 = 0;

int X2 = strlen(a) - 1;

while(a[X1]==' ') X1++;

while(a[X2]==' ') X2--;

for(int x=X1; x<=X2; x++) Source[x-X1]=a[x];

Source[x-X1]='\0';

strcpy(a,Source);

}

void cht(char\* binname)

{

int n=0;

float s=0;

fin.open(binname, ios::binary);

if(!fin) {pri("Ошибка открытия бинарного файла для чтения!"); exit(1);}

for(int i=0; i<5; i++) cout << sh[i] << endl;

while(fin.peek()!=EOF)

{

fin.read((char\*)&man, z);

cout << '\272';p(man);

s+=man.pay;

n++;

}

cout << sh[5] << endl << "Суммарный оклад: " << s << endl

<< "Средний оклад: " << s/n << endl

<< "Количество структур в файле: " << n << endl;

fin.close();

}

**Приложение Г**

***Тестирование программы.***

СВЕДЕНИЯ О СОТРУДНИКАХ ОТДЕЛА

╔═══════════════════╦═════╦════════╦═══════════╦═══════════╗

║ Фамилия, инициалы ║Таб. ║ Стаж ║ Оклад ║ Дата ║

║ ║номер║ работы ║ в тыс.руб.║ рождения ║

╠═══════════════════╬═════╬════════╬═══════════╬═══════════╣

║Первый А.А. ║1 ║5 ║10 ║11.11.1981 ║

║Последний Ю.Ю. ║99 ║10 ║15 ║31.12.1950 ║

║Третьяков П.М. ║3 ║3 ║3 ║03.03.1983 ║

║Промежутков А.Я. ║10 ║6 ║9 ║01.01.1985 ║

╚═══════════════════╩═════╩════════╩═══════════╩═══════════╝

Суммарный оклад: 24

Средний оклад: 6

Количество структур в файле: 4

СВЕДЕНИЯ О СОТРУДНИКАХ ОТДЕЛА

╔═══════════════════╦═════╦════════╦═══════════╦═══════════╗

║ Фамилия, инициалы ║Таб. ║ Стаж ║ Оклад ║ Дата ║

║ ║номер║ работы ║ в тыс.руб.║ рождения ║

╠═══════════════════╬═════╬════════╬═══════════╬═══════════╣

║Первый А.А. ║1 ║5 ║10 ║11.11.1981 ║

║Третьяков П.М. ║3 ║3 ║3 ║03.03.1983 ║

║Промежутков А.Я. ║10 ║6 ║9 ║01.01.1985 ║

║Последний Ю.Ю. ║99 ║10 ║15 ║31.12.1950 ║

╚═══════════════════╩═════╩════════╩═══════════╩═══════════╝

Суммарный оклад: 24

Средний оклад: 6

Количество структур в файле: 4

**Практическая работа №3**

**3.1 Цель работы**

1) Приобретение навыков разработки многомодульной программ.

2) Освоение методов разработки программы, управление которой осуществляется с помощью меню.

3) Освоение препроцессорных средств управления ходом компиляции программы.

**3.2 Техническое задание**

Разработать многомодульную программу, включающую:

1. заголовочный файл, с объявлением внешних переменных и прототипов используемых пользовательских функций;
2. библиотеку функций пользователя – файл с определениями функций;
3. файл с главной функцией, содержащий систему меню для выбора варианта обработки данных.

Обработать структурные данные, записи которых содержатся в бинарном и текстовом файлах.

Разработать структуру программы и схем алгоритмов каждой функции, включая главную.

**3.3 Вариант задания**

Структура: Фамилия и инициалы, табельный номер, стаж работы, оклад в тыс. руб., дата рождения.

Меню реализовать с помощью массива указателей на функции.

**3.4 Структура программы**

Программа содержит следующие функции:

**bin** – Функция создания бинарного файла.

**del4 –** Функция удаления записей из бинарного файла.

**sozdat –** Функция создания текстового файла.

**cht –** Функция чтения бинарного файла.

**dopq –** Функция дополнения очереди.

**chtq –** Функция чтения очереди.

**filtr –** Функция для удаления пробелов.

**kor –** Функция для записи из бинарного в текстовый файл.

**p –** Функция форматного(в виде элемента таблицы) вывода структуры на экран.

**sortbin** – Функция сортировки бинарного файла.

**dops** – Функция дополнения списка.

**sozbin** – Функция записи списка в бинарный файл.

**soztext** – Функция записи списка в текстовый файл.

**p1** – Функция – пункт в меню для создания текстового и бинарного файлов.

**p1\_1** – Функция – пункт в меню, создающий новый текстовый и новый бинарный файл данных.

**p1\_2** – Функция – пункт в меню, создающий новый бинарный файл по уже существующему текстовому.

**p2** – Функция – пункт в меню, отвечающая за чтение.

**p3** – Функция – пункт в меню, отвечающая за сортировку.

**p4** – Функция – пункт в меню, отвечающая за удаление записей.



**Рисунок 3.1 –** Структура программы.

**3.5 Разработка алгоритмов программы**

**3.5.1** Алгоритм функции **p1**.

Функция **p1** используется для создания текстового и бинарного файлов путем вызова соответствующих функций.

Текст функции приведён в **приложении Д**. Алгоритм функции представлен на **рисунке 3.2**. В функции используются глобальные и локальные переменные, которые приведены в **таблицах 3.1 и 3.2.**



**Рисунок 3.2** – Схема алгоритма функции **p1**.

**3.5.2** Алгоритм функции **p1\_1**.

Функция **p1\_1** используется для создания новых текстового и бинарного файлов путем вызова соответствующих функций.

Текст функции приведён в **приложении Д**. Алгоритм функции представлен на **рисунке 3.3**. В функции используются глобальные и локальные переменные, которые приведены в **таблицах 3.1 и 3.2.**



**Рисунок 3.3** – Схема алгоритма функции **p1\_1**.

**3.5.3** Алгоритм функции **p1\_2**.

Функция **p1\_2** используется для создания бинарного файла по уже существующему текстовому посредствам вызова соответствующих функций.

Текст функции приведён в **приложении Д**. Алгоритм функции представлен на **рисунке 3.4**. В функции используются глобальные и локальные переменные, которые приведены в **таблицах 3.1 и 3.2.**



**Рисунок 3.4** – Схема алгоритма функции **p1\_2**.

**3.5.4** Алгоритм функции **p2**.

Функция **p2** используется для чтения бинарного файла посредствам вызова соответствующих функций..

Текст функции приведён в **приложении Д**. Алгоритм функции представлен на **рисунке 3.5**. В функции используются глобальные и локальные переменные, которые приведены в **таблицах 3.1 и 3.2.**



**Рисунок 3.5** – Схема алгоритма функции **p2**.

**3.5.5** Алгоритм функции **p3**.

Функция **p3** используется для сортировки текстового и бинарного файлов посредствам вызова соответствующих функций.

Текст функции приведён в **приложении Д**. Алгоритм функции представлен на **рисунке 3.6**. В функции используются глобальные и локальные переменные, которые приведены в **таблицах 3.1 и 3.2.**



**Рисунок 3.6** – Схема алгоритма функции **p3**.

**3.5.6** Алгоритм функции **p4**.

Функция **p4** используется для сортировки текстового и бинарного файлов посредствам вызова соответствующих функций.

Текст функции приведён в **приложении Д**. Алгоритм функции представлен на **рисунке 3.7**. В функции используются глобальные и локальные переменные, которые приведены в **таблицах 3.1 и 3.2.**



**Рисунок 3.7** – Схема алгоритма функции **p4**.

**3.5.7** Алгоритм главной функции **main.**

Текст функции приведён в **приложении Д**. Алгоритм функции представлен на **рисунке 3.8**.

В функции используются глобальные и локальные переменные, которые приведены в **таблицах 3.1 и 3.2.**



**Рисунок 3.8** – Основной алгоритм программы.

**3.6 Таблицы локальных и глобальных переменных программы.**

В **таблице 3.1** приведены глобальные переменные программы. В **таблице 3.2** приведены локальные переменные программы.

**Таблица 3.1 –** Глобальные переменные программы.

|  |  |  |
| --- | --- | --- |
| Имя | Тип | Назначение |
| man | men | Промежуточная переменная структурного типа, используемая для чтения и записи. |
| z | int | Размер структуры типа men в байтах. |
| sh[] | char\* | Массив содержащий «шапку» таблицы. |
| fin | ifstream | Стандартный входной файловый поток. |
| fout | ofstream | Стандартный выходной файловый поток. |
| io | fstream | Стандартный двунаправленный файловый поток. |
| q | queue\* | Указатель на вершину очереди. |
| s | spisok\* | Указатель на вершину списка. |

**Таблица 3.2** – Локальные переменные программы.

|  |  |  |
| --- | --- | --- |
| Имя | Тип | Назначение |
| i,x | int | Используется в операторе цикла for. |
| n | int | Счетчик структур в бинарном файле. |
| s | int | Суммарных оклад. |
| binname | char\* | Имя бинарного файла. |
| datname,  textname | char\* | Имя текстового файла. |
| data | men | Переменная структурного типа для заполнения списка/очереди данными. |
| X1 | int | Позиция первого не пробельного символа слева. |
| X2 | int | Позиция первого не пробельного символа справа. |
| Source[255] | char | Вспомогательная строка. |
| p | queue\* | Вспомогательная переменная, для организации просмотра списка/очереди. |
| q | queue\* | Вспомогательная переменная, для организации просмотра списка/очереди. |
| nst | queue\* | Добавляемая структура. |
| menu | M | Массив указателей на функции, реализующие меню. |
| c | char | Переменная для анализа нажатой клавиши. |

В **приложении Д** приведён текст программы «PR3.CPP», «PR3H.H», «PR3FUNC.CPP».

В **приложении Е** представлен текст файла с поисковыми признаками для удаления «DELDAT.DAT» и результаты тестирования программы.

**Приложение Д**

***«PR3.CPP»***

#include "PR3H.H"

#include "PR3FUNC.CPP"

typedef void (\*M)();

void p1\_1()

{

sozdat("sozdat.dat");

bin("sozdat.dat","bindat.dat");

clrscr();

cout<<"Меню -> Создание файла -> Создать новый файл данных:\n";

cht("bindat.dat");

cout<<"\nСоздан текстовый файл данных sozdat.dat\n";

cout<<"Создан бинарный файл данных bindat.dat\n";

}

void p1\_2()

{

bin("sozdat.dat","bindat.dat");

clrscr();

cout<<"Меню -> Создание файла -> Использовать уже имеющийся:\n";

cht("bindat.dat");

cout<<"\nСоздан бинарный файл данных bindat.dat\n";

}

void p1()

{

M menu[2] = {p1\_1, p1\_2};

clrscr();

cout << "Меню -> Создание файла:\n\n"

<< "Создать новый файл данных ............ 1\n"

<< "Использовать уже имеющийся ........... 2\n"

<< "Возврат в меню ....................... ESC\n";

char c;

do

{

c = getch();

if(c==27) return;

if(c-48==1||c-48==2) break;

}while(1);

menu[c-49]();

cout<<"\nДля выхода в меню нажмите любую клавишу . . .";

getch();

}

void p2()

{

clrscr();

cout<<"Меню -> Чтение файла:\n";

cht("bindat.dat");

cout<<"\nДля выхода в меню нажмите любую клавишу . . .";

getch();

}

void p3()

{

clrscr();

cout<<"Меню -> Сортировка:\n";

sortbin(s,"bindat.dat", "sozdat.dat");

cout<<"\nОтсортированы бинарный файл bindat.dat и текстовый файл sozdat.dat\n";

cout<<"\nДля выхода в меню нажмите любую клавишу . . .";

getch();

}

void p4()

{

clrscr();

cout<<"Меню -> Удаление данных из файла:\n";

del4("bindat.dat","deldat.dat");

cout<<"\nДля выхода в меню нажмите любую клавишу . . .";

getch();

}

void main()

{

M menu[4] = {p1, p2, p3, p4};

textbackground(CYAN);

textcolor(BLACK);

do

{

clrscr();

cout

<<"Меню:\n\n"

<<"Создание файла ....................... 1\n"

<<"Чтение файла ......................... 2\n"

<<"Сортировка файла ..................... 3\n"

<<"Удаление данных из файла ............. 4\n"

<<"Выход ................................ ESC\n";

char c=getch();

if(c!=27) {if((c-48>=1)&&(c-48<=4)) menu[int(c-49)]();}

else break;

} while(1);

}

***«PR3H.H»***

#ifndef PR3H

#define PR3H

#include <conio.h>

#include <iostream.h>

#include <stdlib.h>

#include <string.h>

#include <iomanip.h>

#include <fstream.h>

#include <stdio.h>

struct men

{

long numb;

char fio[20];

char DOB[50];

float pay;

float exp;

} man;

int z = sizeof(men);

struct spisok

{

men data;

spisok\* next;

} \*s=0;

struct queue

{

men data;

queue\* next;

} \*q=0;

char \*sh[] =

{

"\nСВЕДЕНИЯ О СОТРУДНИКАХ ОТДЕЛА",

"╔═══════════════════╦═════╦════════╦═══════════╦═══════════╗",

"║ Фамилия, инициалы ║Таб. ║ Стаж ║ Оклад ║ Дата ║",

"║ ║номер║ работы ║ в тыс.руб.║ рождения ║",

"╠═══════════════════╬═════╬════════╬═══════════╬═══════════╣",

"╚═══════════════════╩═════╩════════╩═══════════╩═══════════╝"

};

ifstream fin;

ofstream fout;

fstream io;

istream& operator>>(istream& in, men& st);

ostream& operator<<(ostream& out, men& st);

void dopq(queue\*&S, men data);

void chtq(queue\*S);

void dops(spisok\*&S, men data);

void sozbin(spisok\*S,char\* binname);

void soztext(spisok\*S,char\* textname);

void sortbin(spisok\*S,char\* binname,char\* textname);

void pri(char\*s);

void p(men& st);

void bin(char\* textname, char\* binname);

void sozdat(char\* datname);

void filtr(char\*a);

void del4(char\* binname, char\* delname);

void kor(char\* datname, char\* binname);

void cht(char\* binname);

#endif

***«PR3FUNC.CPP»***

istream& operator>>(istream& in, men& st)

{

cout << '\272';

in.getline(st.fio,21); filtr(st.fio);

in >> st.numb >>st.pay >> st.exp;

in.getline(st.DOB,50); filtr(st.DOB);

return in;

}

ostream& operator<<(ostream& out, men& st)

{

out<<setw(20)<<setiosflags(ios::left)<<st.fio<<setw(11)

<<setiosflags(ios::left)<<st.numb<<setw(10)<<setprecision(2)

<<st.pay<<setw(10)<<setprecision(2)<<st.exp<<setw(11)

<<setiosflags(ios::left)<<st.DOB<< endl; return out;

}

void filtr(char\*a)

{

char Source[255] = "";

int X1 = 0;

int X2 = strlen(a) - 1;

while(a[X1]==' ') X1++;

while(a[X2]==' ') X2--;

for(int x=X1; x<=X2; x++) Source[x-X1]=a[x];

Source[x-X1]='\0';

strcpy(a,Source);

}

void dopq(queue\*&S, men data)

{

queue\* p=S, \*q=0, \*nst = new queue;

nst->data = data;

nst->next = 0;

while(p){q=p;p=p->next;}

if(q) q->next = nst;else S = nst;

}

void chtq(queue\*S)

{

queue\* p=S;

while(p) {fout.write((char\*)&p->data,z); p=p->next;}

}

void dops(spisok\*&S, men data)

{

spisok\*p = S,\*q=0,\*nst=new spisok;

nst->data=data;

while(p && (p->data.numb<data.numb))

{q=p; p=p->next;};

nst->next=p;

if(q) q->next=nst;else S=nst;

}

void dels(spisok\*&S)

{

spisok\*p = S,\*q=0;

while(p) {q=p; p=p->next; delete q;}

S=0;

}

void sozbin(spisok\*S,char\* binname)

{

fout.open(binname,ios::binary);

if(!io){cout<<"Ошибка при откритии бинарного файла!";exit(1);}

spisok\*q=S;

if(!q) fout<<"\nНет записей!";

else

while(q)

{

fout.write((char\*)&q->data,z);

q=q->next;

}

fout.close();

}

void soztext(spisok\*S,char\* textname)

{

fout.open(textname);

if(!io){cout<<"Ошибка при откритии текстового файла!";exit(1);}

spisok\*q=S;

if(!q) fout<<"\nНет записей!";

else

while(q)

{

fout << q->data;

q=q->next;

}

fout.close();

}

void sortbin(spisok\*S,char\* binname,char\* textname)

{

fin.open(binname,ios::binary);

if(!fin){cout<<"Ошибка при откритии бинарного файла!";exit(1);}

while(fin.peek()!=EOF)

{

fin.read((char\*)&man,z);

dops(S,man);

}

fin.close();

sozbin(S,binname);

soztext(S,textname);

dels(S);

}

void pri(char\*s){cout << s << endl;}

void bin(char\* textname, char\* binname)

{

fin.open(textname);

if (!fin) {pri("Ошибка при открытии текстового файла!"); exit(1);}

fout.open(binname, ios::binary);

if (!fout) {pri("Ошибка при открытии бинарного файла!"); exit(1);}

while(fin.peek()!=EOF)

{

fin >> man;

fout.write((char\*)&man,z);

}

fout.close();

fin.close();

}

void kor(char\* datname, char\* binname)

{

io.open(binname,ios::in|ios::binary|ios::trunc);

if(!io) {pri("Ошибка открытия бинарного файла для удаления!(kor)"); exit(1);}

fout.open(datname);

if(!fout){pri("Ошибка открытия текстового файла!(kor)"); exit(1);}

while(io.peek()!=EOF)

{

io.read((char\*)&man, z);

fout << man;

}

io.close(); fout.close();

}

void del4(char\* binname, char\* delname)

{

fin.open(delname);

if(!fin) {pri("Ошибка при открытии файла данных с поисковыми признаками!(del4)"); exit(1);}

io.open(binname,ios::binary|ios::in|ios::out);

if(!io) {pri("Ошибка открытия бинарного файла для удаления!(del4)"); exit(1);}

fout.open("temp.dat",ios::binary);

if(!fout){pri("Ошибка открытия бинарного файла!(del4)"); exit(1);}

cout << "\n\nУДАЛЕНИЕ СОТРУДНИКА ПО ФАМИЛИИ\n";

while(fin.peek()!=EOF)

{

char name[20];

fin.getline(name,20); filtr(name);

cout << "\nИщем данные сотрудника с фамилией - " << "\"" << name << "\"" << endl;

if(!strcmp(name,"")) {cout << "Нет поискового признака для удаления!\n"; continue;}

io.seekg(0,ios::beg);

while(io.peek()!=EOF)

{

io.read((char\*)&man, z);

if(strcmp(man.fio, name)==0)

{

strcpy(man.fio,"");

io.seekp(-z,ios::cur);

io.write((char\*)&man, z);

cout << "Сотрудник с фамилией \"" << name << "\" успешно удален!" << endl;

goto m;

}

}

cout << "Сотрудник с фамилией \"" << name << "\" не найден!" << endl;

m:;

}

io.seekg(0,ios::beg);

io.seekp(0,ios::beg);

while(io.peek()!=EOF)

{

io.read((char\*)&man, z);

if(strcmp(man.fio,"")) dopq(q,man);

}

chtq(q);

fin.close(); io.close(); fout.close();

remove(binname);

rename("temp.dat",binname);

kor("sozdat.dat",binname);

}

void p(men& st)

{

cout<<setw(19)<<setiosflags(ios::left)<<st.fio<<'\272'<<setw(5)

<<setiosflags(ios::left)<<st.numb<<'\272'<<setw(8)<<setprecision(2)

<<st.pay<<'\272'<<setw(11)<<setprecision(2)<<st.exp<<'\272'<<setw(11)

<<setiosflags(ios::left)<<st.DOB<<'\272'<< endl;

}

void sozdat(char\* datname)

{

io.open(datname, ios::in|ios::out|ios::trunc);

//io.open(datname, ios::in|ios::out);

if (!io) {pri("Ошибка при открытии текстового файла!"); exit(1);}

do

{

clrscr();

for(int i=0; i<5; i++) cout << sh[i] << endl;

io.seekg(0,ios::beg);

io.seekp(0,ios::beg);

while(io.peek()!=EOF)

{

io >> man;

p(man);

}

io.seekg(0,ios::end);

io.seekp(0,ios::end);

cin >> man;

io << man;

cout <<"\nПродолжить? (y/n): \n";

} while(getch()=='y');

io.close();

}

void cht(char\* binname)

{

int n=0;

float s=0;

fin.open(binname, ios::binary);

if(!fin) {pri("Ошибка открытия бинарного файла для чтения!"); exit(1);}

for(int i=0; i<5; i++) cout << sh[i] << endl;

while(fin.peek()!=EOF)

{

fin.read((char\*)&man, z);

cout << '\272';p(man);

s+=man.pay;

n++;

}

cout << sh[5] << endl << "Суммарный оклад: " << s << endl

<< "Средний оклад: " << s/n << endl

<< "Количество структур в файле: " << n << endl;

fin.close();

**}**

**Приложение Е**

***Исдодные файлы.***

**deldat.dat**

Небывалов Н.Н.

Удалеев Д.Д.

***Тестирование программы.***

***Выбор пункта 1 в главном меню.***

Меню -> Создание файла -> Создать новый файл данных:

СВЕДЕНИЯ О СОТРУДНИКАХ ОТДЕЛА

╔═══════════════════╦═════╦════════╦═══════════╦═══════════╗

║ Фамилия, инициалы ║Таб. ║ Стаж ║ Оклад ║ Дата ║

║ ║номер║ работы ║ в тыс.руб.║ рождения ║

╠═══════════════════╬═════╬════════╬═══════════╬═══════════╣

║Первый А.А. ║1 ║5 ║10 ║11.11.1981 ║

║Последний Ю.Ю. ║99 ║10 ║15 ║31.12.1950 ║

║Удалеев Д.Д. ║70 ║30 ║50 ║01.05.1980 ║

║Третьяков П.М. ║3 ║3 ║3 ║03.03.1983 ║

║Промежутков А.Я. ║10 ║6 ║9 ║01.01.1985 ║

╚═══════════════════╩═════╩════════╩═══════════╩═══════════╝

Суммарный оклад: 87

Средний оклад: 17.4

Количество структур в файле: 5

Создан текстовый файл данных sozdat.dat

Создан бинарный файл данных bindat.dat

Для выхода в меню нажмите любую клавишу . . .

***Выбор пункта 2 в главном меню.***

Меню -> Чтение файла:

СВЕДЕНИЯ О СОТРУДНИКАХ ОТДЕЛА

╔═══════════════════╦═════╦════════╦═══════════╦═══════════╗

║ Фамилия, инициалы ║Таб. ║ Стаж ║ Оклад ║ Дата ║

║ ║номер║ работы ║ в тыс.руб.║ рождения ║

╠═══════════════════╬═════╬════════╬═══════════╬═══════════╣

║Первый А.А. ║1 ║5 ║10 ║11.11.1981 ║

║Последний Ю.Ю. ║99 ║10 ║15 ║31.12.1950 ║

║Удалеев Д.Д. ║70 ║30 ║50 ║01.05.1980 ║

║Третьяков П.М. ║3 ║3 ║3 ║03.03.1983 ║

║Промежутков А.Я. ║10 ║6 ║9 ║01.01.1985 ║

╚═══════════════════╩═════╩════════╩═══════════╩═══════════╝

Суммарный оклад: 87

Средний оклад: 17.4

Количество структур в файле: 5

Для выхода в меню нажмите любую клавишу . . .

***Выбор пункта 3 в главном меню.***

Меню -> Сортировка:

Отсортированны бинарный файл bindat.dat и текстовый файл sozdat.dat

Для выхода в меню нажмите любую клавишу . . .

***Выбор пункта 2 в главном меню.***

Меню -> Чтение файла:

СВЕДЕНИЯ О СОТРУДНИКАХ ОТДЕЛА

╔═══════════════════╦═════╦════════╦═══════════╦═══════════╗

║ Фамилия, инициалы ║Таб. ║ Стаж ║ Оклад ║ Дата ║

║ ║номер║ работы ║ в тыс.руб.║ рождения ║

╠═══════════════════╬═════╬════════╬═══════════╬═══════════╣

║Первый А.А. ║1 ║5 ║10 ║11.11.1981 ║

║Третьяков П.М. ║3 ║3 ║3 ║03.03.1983 ║

║Промежутков А.Я. ║10 ║6 ║9 ║01.01.1985 ║

║Удалеев Д.Д. ║70 ║30 ║50 ║01.05.1980 ║

║Последний Ю.Ю. ║99 ║10 ║15 ║31.12.1950 ║

╚═══════════════════╩═════╩════════╩═══════════╩═══════════╝

Суммарный оклад: 87

Средний оклад: 17.4

Количество структур в файле: 5

Для выхода в меню нажмите любую клавишу . . .

***Выбор пункта 4 в главном меню.***

Меню -> Удаление данных из файла:

УДАЛЕНИЕ СОТРУДНИКА ПО ФАМИЛИИ

Ищем сотрудника с фамилией – “Небывалов Н.Н.”

Сотрудник с фамилией «Небывалов Н.Н.» не найден!

Ищем сотрудника с фамилией – “”

Нет поискового признака для удаления!

Ищем сотрудника с фамилией – “Удалеев Д.Д.”

Сотрудник с фамилией “ Удалеев Д.Д.”успешно удалён!

Для выхода в меню нажмите любую клавишу . . .

***Выбор пункта 2 в главном меню.***

Меню -> Чтение файла:

СВЕДЕНИЯ О СОТРУДНИКАХ ОТДЕЛА

╔═══════════════════╦═════╦════════╦═══════════╦═══════════╗

║ Фамилия, инициалы ║Таб. ║ Стаж ║ Оклад ║ Дата ║

║ ║номер║ работы ║ в тыс.руб.║ рождения ║

╠═══════════════════╬═════╬════════╬═══════════╬═══════════╣

║Первый А.А. ║1 ║5 ║10 ║11.11.1981 ║

║Третьяков П.М. ║3 ║3 ║3 ║03.03.1983 ║

║Промежутков А.Я. ║10 ║6 ║9 ║01.01.1985 ║

║Последний Ю.Ю. ║99 ║10 ║15 ║31.12.1950 ║

╚═══════════════════╩═════╩════════╩═══════════╩═══════════╝

Суммарный оклад: 37

Средний оклад: 9.25

Количество структур в файле: 4

Для выхода в меню нажмите любую клавишу . . .

**Практическая работа №4**

**4.1 Цель работы**

Освоение методов обработки символьной информации, хранящейся в текстовых файлах:

1. ввода исходных данных из текстового файла;
2. обработка текста;
3. вывода результатов в текстовый файл.

**4.2 Техническое задание**

Разработать схему алгоритма и программу для обработки символьной информации в соответствии с вариантом задания.

В каждом варианте задания исходным является файл состоящий из нескольких предложений. Тексты формируются самостоятельно и предназначены для тестирования правильности работы программы.

Задание:

1. посимвольно считать исходные данные и вывести их посимвольно в текстовый

файл - протокол;

1. выполнить над текстом заданный вариант обработки и вывести результаты обработки в рабочий файл;
2. ввести данные построчно из рабочего файла и вывести их в протокол.

Файл – протокол должен включать:

* заголовки каждого этапа обработки;
* исходный текст, предназначенный для обработки;
* результаты обработки, выводимые в рабочий файл;
* исходный текст после обработки;

По завершению программы вывести на печать файлы: файл с исходными данными, рабочий файл и файл – протокол.

**4.3 Вариант задания**

Найти в тексте слова длинной не менее 5 символов, которые встречаются в тексте более 3-х рах, и заменить их кодом(специальным символом). У каждого слова должен быть свой код. За счёт замены текст «сжать». Расшифровать закодированный текст.

**4.4 Структура программы**

Программа содержит следующие функции:

**sozdq –** Функция считывания файла в очередь.

**dopq –** Функция дополнения очереди.

**zapq –** Функция записи очереди в файл.

**chtq –** Функция чтения очереди из файла.

**shifrq –** Функция шифровки текста.

**deshifrq –** Функция расшифровки текста.

**delq –** Функция удаления очереди.



**Рисунок 4.1 –** Структура программы.

**4.5 Разработка алгоритмов программы**

**4.5.1** Алгоритм функции **sozdq**.

Функция **sozdq** используется для считывания файла с текстом в очередь.

Текст функции приведён в **приложении Ж**. Алгоритм функции представлен на **рисунке 4.2**. В функции используются глобальные и локальные переменные, которые приведены в **таблицах 4.1 и 4.2.**



**Рисунок 4.2** – Схема алгоритма функции **sozdq**.

**4.5.2** Алгоритм функции **dopq.**

Функция **dopq** используется для дополнения очереди.

Текст функции приведён в **приложении Ж**. Алгоритм функции представлен на **рисунке 4.3**. В функции используются глобальные и локальные переменные, которые приведены в **таблицах 4.1 и 4.2.**



**Рисунок 4.3** – Схема алгоритма функции **dopq.**

**4.5.3** Алгоритм функции **zapq.**

Функция **zapq** используется записи очереди в файл.

Текст функции приведён в **приложении Ж**. Алгоритм функции представлен на **рисунке 4.4**. В функции используются глобальные и локальные переменные, которые приведены в **таблицах 4.1 и 4.2.**



**Рисунок 4.4** – Схема алгоритма функции **zapq.**

**4.5.4** Алгоритм функции **chtq.**

Функция **chtq** используется для чтения очереди из файла.

Текст функции приведён в **приложении Ж**. Алгоритм функции представлен на **рисунке 4.5**. В функции используются глобальные и локальные переменные, которые приведены в **таблицах 4.1 и 4.2.**



**Рисунок 4.5** – Схема алгоритма функции **chtq.**

**4.5.5** Алгоритм функции **shifrq.**

Функция **shifrq** используется для шифровки текста.

Текст функции приведён в **приложении Ж**. Алгоритм функции представлен на **рисунке 4.6**. В функции используются глобальные и локальные переменные, которые приведены в **таблицах 4.1 и 4.2.**

**

**Рисунок 4.6** – Схема алгоритма функции **shifrq.**

**4.5.6** Алгоритм функции **deshifrq.**

Функция **deshifrq** используется для расшифровки текста.

Текст функции приведён в **приложении Ж**. Алгоритм функции представлен на **рисунке 4.7**. В функции используются глобальные и локальные переменные, которые приведены в **таблицах 4.1 и 4.2.**



**Рисунок 4.7** – Схема алгоритма функции **deshifrq.**

**4.5.7** Алгоритм функции **delq.**

Функция **delq** используется для удаления очереди.

Текст функции приведён в **приложении Ж**. Алгоритм функции представлен на **рисунке 4.8**. В функции используются глобальные и локальные переменные, которые приведены в **таблицах 4.1 и 4.2.**



**Рисунок 4.8** – Схема алгоритма функции **delq.**

**4.5.8** Алгоритм главной функции **main.**

Текст функции приведён в **приложении Ж**. Алгоритм функции представлен на **рисунке 4.9**.



**Рисунок 4.9 -** Основной алгоритм программы.

**4.6 Таблицы локальных и глобальных переменных программы.**

В **таблице 4.1** приведены глобальные переменные программы. В **таблице 4.2** приведены локальные переменные программы.

**Таблица 4.1 –** Глобальные переменные программы.

|  |  |  |
| --- | --- | --- |
| Имя | Тип | Назначение |
| fin, in | ifstream | Стандартный входной файловый поток. |
| Fout, outP | ofstream | Стандартный выходной файловый поток. |

**Таблица 4.2** – Локальные переменные программы.

|  |  |  |
| --- | --- | --- |
| Имя | Тип | Назначение |
| i | int | Счётчик. |
| s | char\* | Слово. |
| qfile, from, to, txt | char\* | Имена текстовых файлов. |
| p | queue\* | Вспомогательная переменная, для организации просмотра очереди. |
| q | queue\* | Вспомогательная переменная, для организации просмотра очереди. |
| nst | queue\* | Добавляемая структура. |

В **приложении Ж** приведён текст программы «PR4.CPP».

В **приложении З** представлен текст файла с исходными данными, файл протокол и рабочий файл.

**Приложение Ж**

#include <conio.h>

#include <iostream.h>

#include <stdlib.h>

#include <string.h>

#include <iomanip.h>

#include <fstream.h>

#include <ctype.h>

#include <stdio.h>

ifstream fin, in;

ofstream outP, fout;

struct slovo

{

char s[50];

int kol;

};

struct queue

{

slovo data;

queue\* next;

};

void dopq(queue \*&Q,char\* s)

{

queue\* p=Q, \*q=0;

while(p)

{

if(!strcmp(p->data.s, s))

{

p->data.kol++;

return;

}

q=p;p=p->next;

}

queue\* nst = new queue;

nst->next = 0;

strcpy(nst->data.s, s);

nst->data.kol=1;

if(q) q->next = nst;

else Q = nst;

}

void chtq(char\* qfile,queue \*&Q)

{

fin.open(qfile);

while(fin.peek()!=EOF)

{

char s[50];

fin >> s;

queue\* p=Q, \*q=0;

queue\* nst = new queue;

strcpy(nst->data.s, s);

nst->data.kol=5;

nst->next = 0;

while(p)

{

if(!strcmp(p->data.s, s))

{

p->data.kol++;

return;

}

q=p;p=p->next;

}

if(q) q->next = nst;

else Q = nst;

}

fin.close();

}

void zapq(char\* qfile,queue \*&Q)

{

fout.open(qfile);

queue\* p=Q;

while(p)

{

if(p->data.kol>3)

fout<<p->data.s<<' ';

p=p->next;

}

fout.close();

}

void delq(queue \*&Q)

{

queue\*p=Q,\*q=0;

while(p) {q=p; p=p->next; delete q;}

Q=0;

}

void sozdq(queue \*&Q,char\* txt)

{

fin.open(txt);

while(fin.peek()!=EOF)

{

char s[50];

fin >> s;

if(strlen(s) >= 5 ) dopq(Q,s);

}

fin.close();

}

void shifrq(char\* from, char\* to, queue\* Q)

{

fin.open(from);

fout.open(to);

while(fin.peek()!=EOF)

{

char c;

while(isspace(fin.peek()))

{fin.get(c);fout.put(c);}

char s[50];

fin >> s;

queue\*p=Q,\*q=0;

int i=0, flag=0;

while(p)

{

if(!strcmp(s,p->data.s)&&p->data.kol>3)

{flag=1; break;}

q=p; p=p->next; i++;

}

if(flag==0) fout << s;

else fout << '$' << i << ' ';

}

fin.close();

fout.close();

}

void deshifrq(char\* from, char\* to, queue\* Q)

{

fin.open(from);

fout.open(to);

while(fin.peek()!=EOF)

{

char c;

fin.get(c);

if(c=='$')

{

int N;

fin >> N; fin.get(c);

queue\*p=Q,\*q=0;

int i=0;

while(p)

{

if(i==N) fout << p->data.s;

q=p; p=p->next; i++;

}

}

else fout.put(c);

}

fin.close();

fout.close();

}

void main()

{

clrscr();

in.open("TEXT.TXT");

outP.open("PROTOCOL.RES");

outP << "1. ИСХОДНЫЕ ДАННЫЕ:\n\n";

char c;

while(in.peek()!=EOF)

{in.get(c);outP.put(c);}

queue\*Q=0;

sozdq(Q,"TEXT.TXT");

zapq("TEXTQ.TXT",Q);

delq(Q);

chtq("TEXTQ.TXT",Q);

shifrq("TEXT.TXT", "TEXTS.TXT", Q);

deshifrq("TEXTS.TXT", "TEXTDS.TXT", Q);

delq(Q);

outP << "\n\n2. РЕЗУЛЬТИРУЮЩИЕ ДАННЫЕ:\n\n";

outP << "2.1 Зашифрованный файл.\n";

in.close(); in.open("TEXTS.TXT");

char s[255];

while(in.peek()!=EOF)

{in.getline(s,255); outP << s << endl;}

outP << "\n\n2.2 Дешифрованный файл.\n";

in.close(); in.open("TEXTDS.TXT");

while(in.peek()!=EOF)

{in.getline(s,255); outP << s<< endl;}

outP.close(); in.close();

}

**Приложение З**

***Файл с исходным текстом TEXT.TXT.***

Александр Блок

Ночь, улица, фонарь , аптека,

Бессмысленный и тусклый свет.

Живи еще хоть четверть века -

Все будет так. Исхода нет.

Умрешь - начнешь опять сначала,

И повторится все, как встарь,

Ночь, ледяная рябь канала,

аптека, улица, фонарь .

Александр Блок

Ночь, улица, фонарь , аптека,

Бессмысленный и тусклый свет.

Живи еще хоть четверть века -

Все будет так. Исхода нет.

Умрешь - начнешь опять сначала,

И повторится все, как встарь,

Ночь, ледяная рябь канала,

аптека, улица, фонарь .

***Файл протокол PROTOCOL.RES.***

1. ИСХОДНЫЕ ДАННЫЕ:

Александр Блок

Ночь, улица, фонарь , аптека,

Бессмысленный и тусклый свет.

Живи еще хоть четверть века -

Все будет так. Исхода нет.

Умрешь - начнешь опять сначала,

И повторится все, как встарь,

Ночь, ледяная рябь канала,

аптека, улица, фонарь .

Александр Блок

Ночь, улица, фонарь , аптека,

Бессмысленный и тусклый свет.

Живи еще хоть четверть века -

Все будет так. Исхода нет.

Умрешь - начнешь опять сначала,

И повторится все, как встарь,

Ночь, ледяная рябь канала,

аптека, улица, фонарь .

2. РЕЗУЛЬТИРУЮЩИЕ ДАННЫЕ:

2.1 Зашифрованный файл.

Александр Блок

$0 $1 $2 , $3

Бессмысленный и тусклый свет.

Живи еще хоть четверть века -

Все будет так. Исхода нет.

Умрешь - начнешь опять сначала,

И повторится все, как встарь,

$0 ледяная рябь канала,

$3 $1 $2 .

Александр Блок

$0 $1 $2 , $3

Бессмысленный и тусклый свет.

Живи еще хоть четверть века -

Все будет так. Исхода нет.

Умрешь - начнешь опять сначала,

И повторится все, как встарь,

$0 ледяная рябь канала,

$3 $1 $2 .

2.2 Дешифрованный файл.

Александр Блок

Ночь, улица, фонарь , аптека,

Бессмысленный и тусклый свет.

Живи еще хоть четверть века -

Все будет так. Исхода нет.

Умрешь - начнешь опять сначала,

И повторится все, как встарь,

Ночь, ледяная рябь канала,

аптека, улица, фонарь .

Александр Блок

Ночь, улица, фонарь , аптека,

Бессмысленный и тусклый свет.

Живи еще хоть четверть века -

Все будет так. Исхода нет.

Умрешь - начнешь опять сначала,

И повторится все, как встарь,

Ночь, ледяная рябь канала,

аптека, улица, фонарь .

***Рабочий файл TEXT1.TXT.***

Александр Блок

$0 $1 $2 , $3

Бессмысленный и тусклый свет.

Живи еще хоть четверть века -

Все будет так. Исхода нет.

Умрешь - начнешь опять сначала,

И повторится все, как встарь,

$0 ледяная рябь канала,

$3 $1 $2 .

Александр Блок

$0 $1 $2 , $3

Бессмысленный и тусклый свет.

Живи еще хоть четверть века -

Все будет так. Исхода нет.

Умрешь - начнешь опять сначала,

И повторится все, как встарь,

$0 ледяная рябь канала,

$3 $1 $2 .

**Практическая работа №5**

**5.1 Цель работы**

1) Приобретение навыков разработки алгоритмов и программ для представления и движенмия графических изображений в окне MS-DOS.

2) Освоение методов определения классов, создания и использования их экземпляров, приобретение первичных навыков объектно ориентированного программирования.

3) Приобретение навыков создания производных классов на базе определённых.

**5.2 Техническое задание**

1) Разработать алгоритм и программу вывода на экран графика функции представленной в Вашем варианте задания лабораторной работы № 5 второго семестра.

В графическом режиме вывести на экран:

а) линии осей координат со стрелками на концах, чёрточками оцифровки и с оцифровкой осей;

б) график функций;

в) надписи у графика кривой.

Вывод графика на экран производить с помощью одной из графических функций:

putpixel() – вывода точки;

drawpoly() – вывода ломанной;

lineto() – вывода отрезка;

lineral() – вывода приращения;

2) Отпечатать вид экрана с выведенным графиком функции.

**5.3 Вариант задания**

Изобразить некоторую составную фигуру, например, окружность, вписанную в квадрат с точкой в центре. Двигать эту фигуру по некоторой траектории с некоторой скоростью.

**5.4 Структура программы**

Программа содержит следующие функции:

**screen::osi** – Вывод осей на экран.

**screen::gr** – Вывод графига на экран.

**screen::title** – Вывод заголовка на экран.

**circsqrt::move** – Перемещение фигуры.

**circsqrt::show** – Вывод фигуры на экран.

**point::move** – Передвигает точку.

**point::hide** – Прячет точку.

**point::show** – Показывает точку.

**circ::move** – Передвигает окружность.

**circ::hide** – Прячет окружность.

**circ::show** – Показывает окружность.

**square::hide** – Прячет квадрат.

**square::show** – Показывает квадрат.

**square::move** – Передвигает квадрат.

**square::rissquare** – Рисует квадрат.



**Рисунок 5.1 –** Структура программы.

**5.5 Разработка алгоритмов программы**

**5.5.1** Алгоритм функции **screen::osi**.

**screen::osi** – Вывод осей на экран.

Текст функции приведён в **приложении И**. Алгоритм функции представлен на **рисунке 5.2**. В функции используются глобальные и локальные переменные, которые приведены в **таблицах 5.1 и 5.2.**



**Рисунок 5.2** – Схема алгоритма функции **screen::osi**.



**Рисунок 5.2** – Продолжение.

**5.5.2** Алгоритм функции **screen::gr**.

**screen::gr** – Вывод графика на экран.

Текст функции приведён в **приложении И**. Алгоритм функции представлен на **рисунке 5.3**. В функции используются глобальные и локальные переменные, которые приведены в **таблицах 5.1 и 5.2.**



**Рисунок 5.3** – Схема алгоритма функции **screen::osi**.

**5.5.3** Алгоритм функции **screen::title**.

**screen::title** – Вывод заголовка на экран.

Текст функции приведён в **приложении И**. Алгоритм функции представлен на **рисунке 5.4**. В функции используются глобальные и локальные переменные, которые приведены в **таблицах 5.1 и 5.2.**



**Рисунок 5.4** – Схема алгоритма функции **screen::title**.

**5.5.4** Алгоритм функции **circsqrt::move**.

**circsqrt::move** – Перемещение фигуры.

Текст функции приведён в **приложении И**. Алгоритм функции представлен на **рисунке 5.5**. В функции используются глобальные и локальные переменные, которые приведены в **таблицах 5.1 и 5.2.**



**Рисунок 5.5** – Схема алгоритма функции **circsqrt::move**.

**5.5.5** Алгоритм функции **circsqrt::show**.

**circsqrt::show** – Вывод фигуры на экран.

Текст функции приведён в **приложении И**. Алгоритм функции представлен на **рисунке 5.6**. В функции используются глобальные и локальные переменные, которые приведены в **таблицах 5.1 и 5.2.**



**Рисунок 5.6** – Схема алгоритма функции **circsqrt::show**.

**5.5.6** Алгоритм функции **circsqrt::circsqrt**.

**circsqrt::circsqrt** – Конструктор класса.

Текст функции приведён в **приложении И**. Алгоритм функции представлен на **рисунке 5.7**. В функции используются глобальные и локальные переменные, которые приведены в **таблицах 5.1 и 5.2.**



**Рисунок 5.7** – Схема алгоритма функции **screen::osi**.

**5.5.7** Алгоритм функции **point::move**.

**point::move** – Передвигает точку.

Текст функции приведён в **приложении И**. Алгоритм функции представлен на **рисунке 5.8**. В функции используются глобальные и локальные переменные, которые приведены в **таблицах 5.1 и 5.2.**



**Рисунок 5.8** – Схема алгоритма функции **point::move**.

**5.5.8** Алгоритм функции **point::hide**.

**point::hide** – Прячет точку.

Текст функции приведён в **приложении И**. Алгоритм функции представлен на **рисунке 5.9**. В функции используются глобальные и локальные переменные, которые приведены в **таблицах 5.1 и 5.2.**



**Рисунок 5.9** – Схема алгоритма функции **point::hide**.

**5.5.9** Алгоритм функции **point::show**.

**point::show** – Показывает точку.

Текст функции приведён в **приложении А**. Алгоритм функции представлен на **рисунке 5.10**. В функции используются глобальные и локальные переменные, которые приведены в **таблицах 5.1 и 5.2.**



**Рисунок 5.10** – Схема алгоритма функции **point::show**.

**5.5.10** Алгоритм функции **point:: point**.

**point:: point**– Конструктор класса.

Текст функции приведён в **приложении И**. Алгоритм функции представлен на **рисунке 5.11**. В функции используются глобальные и локальные переменные, которые приведены в **таблицах 5.1 и 5.2.**



**Рисунок 5.11** – Схема алгоритма функции **point::point**.

**5.5.11** Алгоритм функции **circ::move**.

**circ::move** – Передвигает окружность.

Текст функции приведён в **приложении И**. Алгоритм функции представлен на **рисунке 5.12**. В функции используются глобальные и локальные переменные, которые приведены в **таблицах 5.1 и 5.2.**



**Рисунок 5.12** – Схема алгоритма функции **circ::move**.

**5.5.12** Алгоритм функции **circ::hide**.

**circ::hide** – Прячет окружность.

Текст функции приведён в **приложении И**. Алгоритм функции представлен на **рисунке 5.13**. В функции используются глобальные и локальные переменные, которые приведены в **таблицах 5.1 и 5.2.**



**Рисунок 5.13** – Схема алгоритма функции **circ::hide**.

**5.5.13** Алгоритм функции **circ::show**.

**circ::show** – Показывает окружность.

Текст функции приведён в **приложении И**. Алгоритм функции представлен на **рисунке 5.14**. В функции используются глобальные и локальные переменные, которые приведены в **таблицах 5.1 и 5.2.**



**Рисунок 5.14** – Схема алгоритма функции **circ::show**.

**5.5.14** Алгоритм функции **circ:: circ**.

**circ:: circ** – Конструктор класса.

Текст функции приведён в **приложении И**. Алгоритм функции представлен на **рисунке 5.15**. В функции используются глобальные и локальные переменные, которые приведены в **таблицах 5.1 и 5.2.**



**Рисунок 5.15** – Схема алгоритма функции **circ:: circ**.

**5.5.15** Алгоритм функции **square::hide**.

**square::hide** – Прячет квадрат.

Текст функции приведён в **приложении И**. Алгоритм функции представлен на **рисунке 5.16**. В функции используются глобальные и локальные переменные, которые приведены в **таблицах 5.1 и 5.2.**



**Рисунок 5.16** – Схема алгоритма функции **square::hide**.

**5.5.16** Алгоритм функции **square::show**.

**square::show** – Показывает квадрат.

Текст функции приведён в **приложении И**. Алгоритм функции представлен на **рисунке 5.17**. В функции используются глобальные и локальные переменные, которые приведены в **таблицах 5.1 и 5.2.**



**Рисунок 5.17** – Схема алгоритма функции **square::show**.

**5.5.17** Алгоритм функции **square::move**.

**square::move** – Передвигает квадрат.

Текст функции приведён в **приложении И**. Алгоритм функции представлен на **рисунке 5.18**. В функции используются глобальные и локальные переменные, которые приведены в **таблицах 5.1 и 5.2.**



**Рисунок 5.18** – Схема алгоритма функции **square::move**.

**5.5.18** Алгоритм функции **square::rissquare**.

**square::rissquare** – Рисует квадрат.

Текст функции приведён в **приложении И**. Алгоритм функции представлен на **рисунке 5.19**. В функции используются глобальные и локальные переменные, которые приведены в **таблицах 5.1 и 5.2.**



**Рисунок 5.19** – Схема алгоритма функции **square::rissquare**.

**5.5.19** Алгоритм функции **square::square**.

**square::square** – Конструктор класса.

Текст функции приведён в **приложении И**. Алгоритм функции представлен на **рисунке 5.20**. В функции используются глобальные и локальные переменные, которые приведены в **таблицах 5.1 и 5.2.**



**Рисунок 5.20** – Схема алгоритма функции **square::square**.

**5.5.20** Алгоритм функции **screen::~screen**.

**screen::~screen** – Деструктор класса.

Текст функции приведён в **приложении И**. Алгоритм функции представлен на **рисунке 5.21**. В функции используются глобальные и локальные переменные, которые приведены в **таблицах 5.1 и 5.2.**



**Рисунок 5.21** – Схема алгоритма функции **screen::~screen**.

**5.5.21** Алгоритм функции **screen::screen**.

**screen::screen** – Конструктор класса.

Текст функции приведён в **приложении И**. Алгоритм функции представлен на **рисунке 5.22**. В функции используются глобальные и локальные переменные, которые приведены в **таблицах 5.1 и 5.2.**



**Рисунок 5.22** – Схема алгоритма функции **screen::screen**.

**5.5.22** Алгоритм главной функции **main.**

Текст функции приведён в **приложении И**. Алгоритм функции представлен на **рисунке 1.23**.



**Рисунок 5.23 -** Основной алгоритм программы.



**Рисунок 5.23 –** Продолжение.

**5.6 Таблицы локальных и глобальных переменных программы.**

В **таблице 5.1** приведены глобальные переменные программы. В **таблице 5.2** приведены локальные переменные программы.

**Таблица 5.1 –** Глобальные переменные программы.

|  |  |  |
| --- | --- | --- |
| Имя | Тип | Назначение |
| mx,my | int | Масштабы по х и у. |
| msx, msy | int | Максимальные размеры экрана в пикселях по х и по у. |

**Таблица 5.2** – Локальные переменные программы.

|  |  |  |
| --- | --- | --- |
| Имя | Тип | Назначение |
| x, y, xc, yc, xq, yq | int | Центры фигур. |
| rc, lq | int | Радиус окружности и длины сторон квадрата. |
| bk, cc | int | Запоминают цвет. |
| a | double | Значение в «нашем» масштабе. |
| st[4] | char | Строка для числа. |
| x, y | double | Промежуточные значения для вычислений. |
| c | char | Индикатор выхода из цикла. |
| storona | int | Флаг направления. |

В **приложении И** приведён текст программы «PR5.CPP».

В **приложении К** представлена распечатка работы графической программы.

**Приложение И**

#include <conio.h>

#include <graphics.h>

#include <stdio.h>

#include <math.h>

#include <stdlib.h>

#include <dos.h>

#include <bios.h>

double mx,my, msx, msy;

class point

{

protected:

double x,y;

public:

point(int xi=0, int yi=0)

{x=xi;y=yi;}

void show()

{putpixel(x,y,getcolor());}

void hide()

{putpixel(x,y,getbkcolor());}

void move(int xi=0, int yi=0)

{

hide();

x=xi,y=yi;

show();

}

};

class circ

{

double xc,yc,rc;

public:

circ(int xi, int yi, int ri)

{xc = xi; yc=yi; rc=ri;}

void show()

{circle(xc,yc,rc);}

void hide()

{

int bk, cc;

bk = getbkcolor();

cc = getcolor();

setcolor(bk);

circle(xc,yc,rc);

setcolor(cc);

}

void move(int xi=0, int yi=0)

{

hide();

xc=xi,yc=yi;

show();

}

};

class square

{

int xq,yq,lq;

void rissquare()

{

int d = lq/2;

line(xq-d,yq-d,xq+d,yq-d);

line(xq-d,yq+d,xq+d,yq+d);

line(xq-d,yq-d,xq-d,yq+d);

line(xq+d,yq-d,xq+d,yq+d);

}

public:

square(int xi, int yi, int li)

{xq=xi,yq=yi;lq=li;}

void show()

{rissquare();}

void hide()

{

int bk, cc;

bk = getbkcolor();

cc = getcolor();

setcolor(bk);

rissquare();

setcolor(cc);

}

void move(int xi=0, int yi=0)

{

hide();

xq=xi,yq=yi;

show();

}

};

class circsqrt : public circ, public square, public point

{

public:

circsqrt(int xi, int yi, int ri):

circ(xi, yi, ri),

square(xi, yi, 2\*ri),

point(xi,yi)

{}

void show()

{circ::show();square::show();point::show();}

void hide()

{circ::hide();square::hide();point::hide();}

void move(int xi=0, int yi=0)

{circ::move(xi,yi); square::move(xi,yi);point::move(xi,yi);}

};

class screen

{

public:

screen()

{

int dr =DETECT, mod;

initgraph(&dr, &mod, "C:\\BORLANDC\\BGI");

mx=getmaxx(); my=getmaxy();

msx=(0.6\*mx)/2.0; msy=(0.8\*my)/10.0;

}

~screen() {closegraph();}

void osi ();

void gr ();

void title ();

};

void screen:: osi ()

{

line(0.5\*mx,0.1\*my,0.5\*mx,0.9\*my);

line(0.1\*mx-40,0.5\*my,0.9\*mx+40,0.5\*my);

line(0.1\*mx-40,0.5\*my, 0.1\*mx-30,0.5\*my-3);

line(0.1\*mx-40,0.5\*my, 0.1\*mx-30,0.5\*my+3);

line(0.9\*mx+40,0.5\*my, 0.9\*mx+30,0.5\*my-3);

line(0.9\*mx+40,0.5\*my, 0.9\*mx+30,0.5\*my+3);

line(0.5\*mx,0.1\*my, 0.5\*mx+3, 0.1\*my+10);

line(0.5\*mx,0.1\*my, 0.5\*mx-3, 0.1\*my+10);

line(0.5\*mx,0.9\*my, 0.5\*mx+3, 0.9\*my-10);

line(0.5\*mx,0.9\*my, 0.5\*mx-3, 0.9\*my-10);

setcolor(11);

settextstyle(0,0,1);

outtextxy(0.9\*mx +30,0.5\*my+10,"X");

outtextxy(0.5\*mx - 15,0.1\*my,"Y");

double x = -1.4;

while (x<=1.4)

{

double a=0.5\*mx+msx\*x;

char st[4];

sprintf(st,"%4.1f",x);

outtextxy(a-20,0.5\*my +10,st);

line(a,0.5\*my+3,a,0.5\*my-3);

x+=0.2;

}

double y = 0.1;

while (y<=4.501)

{

double a= 0.5\*my - msy\*y;

char st[4];

sprintf(st,"%4.1f",y);

line(0.5\*mx-3,a,0.5\*mx+3, a);

outtextxy(0.5\*mx -40, a-5, st);

y+=0.5;

}

}

void screen::gr()

{

double x=-1.4;

while(x<=1.4)

{

double y=exp(x);

putpixel(0.5\*mx+msx\*x,0.5\*my-msy\*y,2);

x+=0.001;

}

}

void screen::title()

{

setcolor (11);

settextstyle (0,0,2);

outtextxy (0.2\*mx,0.1\*my,"GRAPHIC");

outtextxy (0.2\*mx,0.15\*my,"EXP(X)");

}

void main()

{

screen n;

n.osi();

getch();

n.gr();

getch();

n.title();

getch();

circsqrt CSP(100,100,30);

CSP.show();

getch();

int storona=1;

float x = -1.4;

while(1)

{

char c=0;

if(bioskey(1)) c = bioskey(0);

if(c==27) break;

double y=exp(x);

CSP.move(0.5\*mx+x\*msx,

0.5\*my-y\*msy);

n.osi();

n.gr();

n.title();

if(x>=1.3&&storona==1) storona=0; else if(x<=-1.3&&storona==0) storona=1;

if(storona) x+=0.015; else x-=0.015;

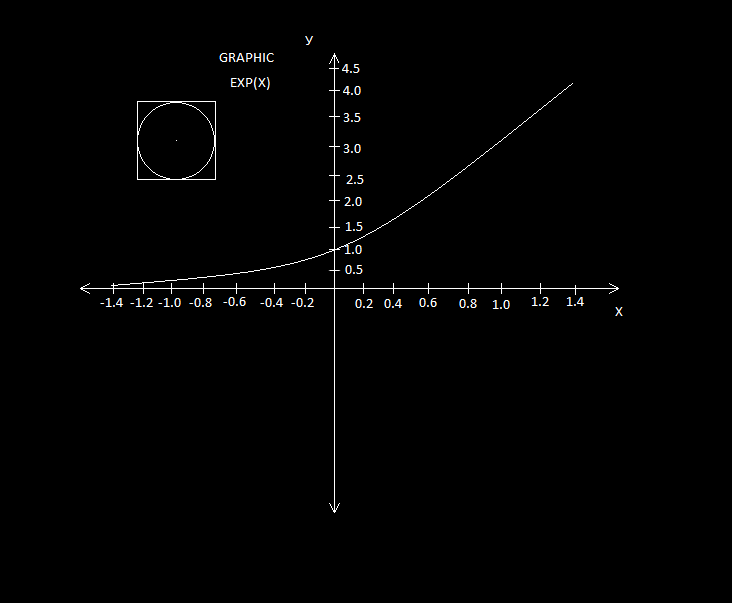
delay(50);

}

}

**Приложение К**

Снимок экрана работы программы.

****

**Содержание**

Предисловие . . . . . . . . . . . . . . . . . . . . . . . . . . . 2

Практическая работа №1 . . . . . . . . . . . . . . . . . . . . . 3

Цель работы . . . . . . . . . . . . . . . . . . . . . . . . . 3

Техническое задание . . . . . . . . . . . . . . . . . . . . . 3

Вариант задания . . . . . . . . . . . . . . . . . . . . . . . 3

Структура программы . . . . . . . . . . . . . . . . . . . . . 3

Разработка алгоритмов программы . . . . . . . . . . . . . . . 4

Таблицы локальных и глобальных переменных программы . . . . . 16

Приложение А . . . . . . . . . . . . . . . . . . . . . . . . 17

Приложение Б . . . . . . . . . . . . . . . . . . . . . . . . 22

Практическая работа №2 . . . . . . . . . . . . . . . . . . . . . 23

Цель работы . . . . . . . . . . . . . . . . . . . . . . . . . 23

Техническое задание . . . . . . . . . . . . . . . . . . . . . 23

Вариант задания . . . . . . . . . . . . . . . . . . . . . . . 23

Структура программы . . . . . . . . . . . . . . . . . . . . . 23

Разработка алгоритмов программы . . . . . . . . . . . . . . . 24

Таблицы локальных и глобальных переменных программы . . . . . 29

Приложение В . . . . . . . . . . . . . . . . . . . . . . . . 30

Приложение Г . . . . . . . . . . . . . . . . . . . . . . . . 34

Практическая работа №3 . . . . . . . . . . . . . . . . . . . . . 35

Цель работы . . . . . . . . . . . . . . . . . . . . . . . . . 35

Техническое задание . . . . . . . . . . . . . . . . . . . . . 35

Вариант задания . . . . . . . . . . . . . . . . . . . . . . . 35

Структура программы . . . . . . . . . . . . . . . . . . . . . 35

Разработка алгоритмов программы . . . . . . . . . . . . . . . 36

Таблицы локальных и глобальных переменных программы . . . . . 43

Приложение Д . . . . . . . . . . . . . . . . . . . . . . . . 44

Приложение Е . . . . . . . . . . . . . . . . . . . . . . . . 52

Практическая работа №4 . . . . . . . . . . . . . . . . . . . . . 54

Цель работы . . . . . . . . . . . . . . . . . . . . . . . . . 54

Техническое задание . . . . . . . . . . . . . . . . . . . . . 54

Вариант задания . . . . . . . . . . . . . . . . . . . . . . . 54

Структура программы . . . . . . . . . . . . . . . . . . . . . 54

Разработка алгоритмов программы . . . . . . . . . . . . . . . 55

Таблицы локальных и глобальных переменных программы . . . . . 63

Приложение Ж . . . . . . . . . . . . . . . . . . . . . . . . 64

Приложение З . . . . . . . . . . . . . . . . . . . . . . . . 68

Практическая работа №5 . . . . . . . . . . . . . . . . . . . . . 71

Цель работы . . . . . . . . . . . . . . . . . . . . . . . . . 71

Техническое задание . . . . . . . . . . . . . . . . . . . . . 71

Вариант задания . . . . . . . . . . . . . . . . . . . . . . . 71

Структура программы . . . . . . . . . . . . . . . . . . . . . 71

Разработка алгоритмов программы . . . . . . . . . . . . . . . 72

Таблицы локальных и глобальных переменных программы . . . . . 86

Приложение И . . . . . . . . . . . . . . . . . . . . . . . . 87

Приложение К . . . . . . . . . . . . . . . . . . . . . . . . 91