АННОТАЦИЯ

В данной пояснительной записке содержится характеристика приложения, содержащего сведения о времени выполнения заданий на ЭВМ.  Поля базы данных содержат шифр задания, код отдела, Ф.И.О. программиста, общее время прохождения задания и время центрального процессора. В приложении осуществляется добавление и удаление, просмотр, поиск, сортировка и корректировка записей. База данных имеет применение в учебных учреждениях и различных предприятиях, выполняя функцию хранения и обработки данных.

СОДЕРЖАНИЕ

АННОТАЦИЯ 4

ВВЕДЕНИЕ 6

1. НАЗНАЧЕНИЕ И ОБЛАСТЬ ПРИМЕНЕНИЯ ПРОГРАММЫ 7
2. ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ ПРОГРАММЫ 8
   1. Постановка задачи на разработку программы 8
   2. Применяемые математические методы 9
   3. Описание и обоснование выбора метода организации входных,

выходных и промежуточных данных 9

* 1. Описание алгоритмов функционирования программы 10
  2. Обоснование состава технических и программных средств 22

1. ВЫПОЛНЕНИЕ ПРОГРАММЫ 23
   1. Условия выполнения программы 23
   2. Загрузка и запуск программы 23
   3. Проверка работоспособности программы 23

ВЫВОДЫ 31

ПЕРЕЧЕНЬ ССЫЛОК 32

ПРИЛОЖЕНИЕ А 33

ВВЕДЕНИЕ

Целью данного курсового проекта является систематизация, закрепление и углубление знаний в области основ программирования и совершенствование практических навыков разработки программ на языке Си и С++.

Для достижения цели на разных этапах курсового проектирования должны быть решены следующие задачи:

- выбор варианта задания и детализация поставки задачи;

- определение требований к функциям, выполняемых разрабатываемой программой;

- выбор типов и проектирование структур данных, определяющих способы представления, хранения и преобразования входных, выходных и промежуточных данных;

- разработка модульной структуры программы, определение функций модулей и способов их взаимодействия;

- написание текста (кодирование) программных модулей на алгоритмическом языке;

- разработка тестовых примеров;

- тестирование и отладка программы;

- разработка программных документов в соответствии с действующими стандартами.

В настоящее время, данная работа является актуальной только для малого количества учреждений, в связи с существованием большого количества приложений, имеющих более совершенствованный функционал.

Курсовой проект состоит из пояснительной записки, включающей в себя аннотацию, введение, основные разделы, выводы, список литературы, приложение и программы на электронном носителе.

1 НАЗНАЧЕНИЕ И ОБЛАСТЬ ПРИМЕНЕНИЯ ПРОГРАММЫ

Программа предназначена для работы с данными программистов, выполнивших задание на ЭВМ. В базу данных входят такие параметры, как шифр задания, код отдела, ФИО программиста, общее время прохождения задания и время центрального процессора. Программа позволяет выполнять создание списка (вручную или из файла), просмотр таблицы, редактирование информации, поиск и сортировку по различным полям, сохранение в файл и расчет процентного и среднего времени.

Область применения программы ограничена выбранным вариантом. Данные поля наиболее подходят для применения сотрудниками различных учреждений для хранения информации о программистах, выполнявших задания на ЭВМ.

2 ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ ПРОГРАММЫ

* 1. Постановка задачи на разработку программы

Задача: Даны сведения о времени выполнения заданий на ЭВМ (время измеряется в секундах). Структура записи: шифр задания (8 символов), код отдела (3символа), Ф.И.О. программиста (15 символов), общее время прохождения задания, время центрального процессора − 5 знаков (t\_задания > t\_цп). Получить процент процессорного времени по каждому заданию, а также суммы по видам времени по всем заданиям и средний процент времени центрального процессора по всем заданиям.

Разрабатываемая программа должна использовать меню-ориентированный интерфейс, обеспечивающий выполнение следующего минимального состава действий:

1. Начальное создание таблицы. При необходимости создания новой таблицы исходные данные считываются из текстового файла. Имя файла должен задавать пользователь.

2. Просмотр таблицы. При этом необходимо предусмотреть возможность скроллинга.

3. Добавление новой записи в таблицу.

4. Удаление записи. Удаляемый элемент выбирается по одному из полей таблицы (ключевому). Ключевое поле выбирает студент.

5. Корректировка записи в таблице. Корректируемую запись выбирают по одному из полей таблицы (ключевому).

6. Сортировка таблицы. Сортировка производится по одному из полей таблицы (ключевому). Метод сортировки выбирает студент.

7. Поиск записи в таблице по ключевому полю.

8. Сохранение таблицы в текстовом файле. Имя файла должен вводить пользователь. Сохранение таблицы в текстовом файле обеспечит при необходимости возможность её печати.

9. Обработка таблицы и просмотр результатов обработки. Результат обработки необходимо вывести на экран и в текстовый файл. Имя файла вводит пользователь.

10. Выход ― завершение работы программы

* 1. Применяемые математические методы

Для нахождения процента процессорного времени по каждому заданию необходимо данное время центрального процессора поделить на общее время и полученный результат умножить на 100. Суммируем все результаты нахождения процента процессорного времени и делим на число заданий, полученный результат является средним процентом времени центрального процессора.

* 1. Описание и обоснование выбора метода организации входных, выходных и промежуточных данных

Исходные данные хранятся в файле неопределенного размера, содержащем сведения о времени прохождения заданий на ЭВМ. Каждая запись содержит:

- шифр задания (8 символов);

- код отдела (3 символа);

- ФИО программиста (15 символов);

- общее время прохождения задания (5 знаков);

- время центрального процессора (5 знаков, t\_задания>t\_цп);

Для хранения сведений можно использовать динамический двунаправленный список, т.к. он позволяет работать сразу с тремя элементами: текущий, предыдущий, следующий.

Выходные данные могут выводиться в файл или на экран. На экран сведения выводятся в виде таблицы с возможностью перемещения между страницами.

Языком программирования в работе был выбран язык С++, а в качестве среды программирования – Microsoft Visual Studio 2017, т.к. он более удобный и бесплатный.

2.4 Описание алгоритмов функционирования программы

Перечень всех функций:

job\* dob\_beg(job\* end, job\* beg, const job& z) – функция добавления в начало. Принимает в качестве аргументов:

1) job \*beg – указатель на начало списка;

2) job \*end – указатель на конец списка;

3) const job &z – ссылка на структуру z;

job\* dob\_end(job\* end, job\* beg, const job& z) – функция добавления в конец. Принимает в качестве аргументов:

1) job \*beg – указатель на начало списка;

2) job \*end – указатель на конец списка;

3) const job &z – ссылка на структуру z;

job\* dob\_first(const job& z) – функция добавления первого элемента, принимает в качестве аргумента const job &z – ссылку на структуру z.

job\* dob\_after(job\* beg, int index, const job& z) – функция добавления после индекса. Принимает в качестве аргументов:

1) job \*beg – указатель на начало списка;

2) int index – номер после которого необходимо добавить;

3) const job &z – ссылка на структуру z;

job\* del\_index(job\* beg, int index) – функция удаления по индексу. Принимает в качестве аргументов:

1) job \*beg – указатель на начало списка;

2) int index – номер записи, которую необходимо удалить;

job\* del\_all(job\* beg) – функция удаления всех элементов. Принимает в качестве аргументов job \*beg – указатель на начало списка.

job vvod() – функция ввода данных с клавиатуры.

int menu\_main() – функция выводящая главное меню на экран.

int menu\_add() – функция выводящая меню добавления на экран.

int menu\_del() – функция выводящая меню удаления на экран.

int menu\_sort() – функция выводящая меню сортировки на экран.

int menu\_edit() – функция выводящая меню редактирования на экран.

int menu\_search() – функция выводящая меню поиска на экран.

int menu\_save() – функция выводящая меню сохранения на экран.

void sort\_sh(job\* beg), void sort\_kod(job\* beg), void sort\_fio(job\* beg), void sort\_tz(job\* beg), void sort\_tcp(job\* beg) – функции сортировки по шифру, коду, ФИО, общему времени и времени центрального процессора. Принимают в качестве аргументов job \*beg – указатель на начало списка.

void edit(job\* beg) – функция редактирования. Принимает в качестве аргументов job \*beg – указатель на начало списка.

void search(job\* beg) – функция поиска. Принимает в качестве аргументов job\*beg – указатель на начало списка.

void time(job\* beg) – функция по варианту, т.е. нахождения среднего процента времени и сумму по видам времени. Принимает в качестве аргументов job \*beg – указатель на начало списка.

void print(const job& z) – функция вывода одной записи.

void prosmotr(job\* beg) – функция просмотра таблицы. Принимает в качестве аргументов job \*beg – указатель на начало списка.

void Tab() – функция вывода заголовка таблицы.

int read\_file(char\* filefio, job\*\* beg, job\*\* end) – функция чтения файла. Принимает в качестве аргументов:

1) job \*\*beg – двойной указатель на начало списка;

2) job \*\*end – двойной указатель на конец списка;

3) char\* filefio – название файла для чтения;

int write\_file(char\* filename, job\* temp) – функция записи в файл. Принимает в качестве аргументов:

1) job \*temp – указатель на начало списка;

2) char\* filename – название файла для сохранения;

void TextColorO(char \*text), void TextColorP(char \*text), void TextColorB(char \*text), void TextColorR(char \*text), void TextColorG(char \*text) – функция вывода текста в цвете. Принимают в качестве аргумента char \*text – текст для изменения цвета.

Дополнительное описание некоторых функций:

Схема алгоритма основной программы на Рисунке 1.

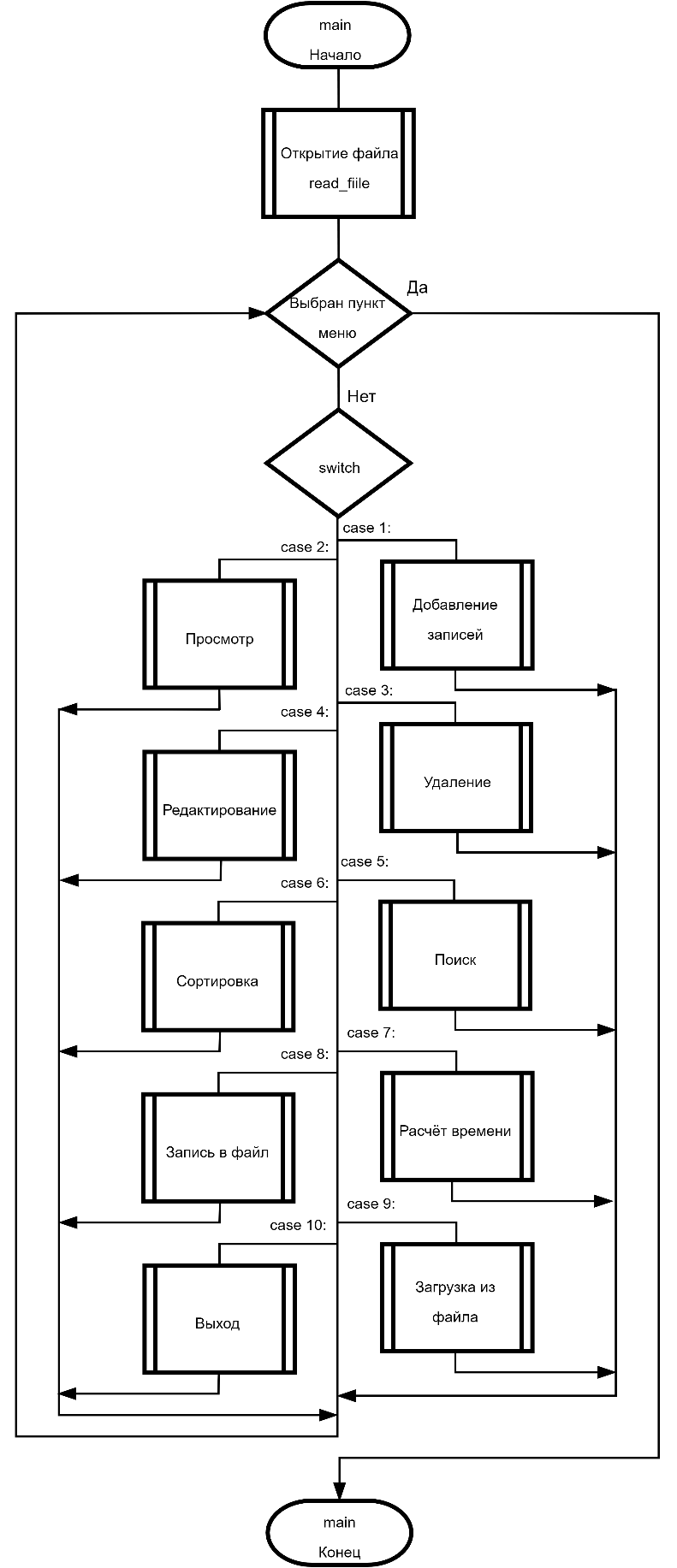


Рисунок 1 – Структурная схема основной программы

Блоки:

1 блок – функция открытия файла

2 блок – бесконечный цикл пока не выбран пункт меню

3 блок – оператор выбора

4 – выбор функции добавления записи

5 – выбор функции просмотра

6– выбор функции удаления

7 – выбор функции редактирования

8 – выбор функции поиска

9 – выбор функции сортировки

10 – выбор функции расчета времени

11 – выбор функции записи в файл

12 – выбор функции загрузка из файла

13 – выбор функции выхода

Основная программа начинается с автоматического открытия файла, затем запускается функция menu\_main() (рис. 2), которая отображает главное меню и считывает код введенной клавиши, после чего передает его в главную функцию, откуда запускается необходимая функция.

Блоки:

1 блок – инициализация начальных переменных

2 блок – функция отображения пунктов меню

3 блок – проверка нажатия клавиши Вверх

4 блок – проверка нажатия клавиши Вниз

5 блок – проверка значения переменной point после декрементации

6 блок – проверка значения переменной point после инкрементации

7 блок – выделение последнего пункта меню

8 блок – выделение первого пункта меню

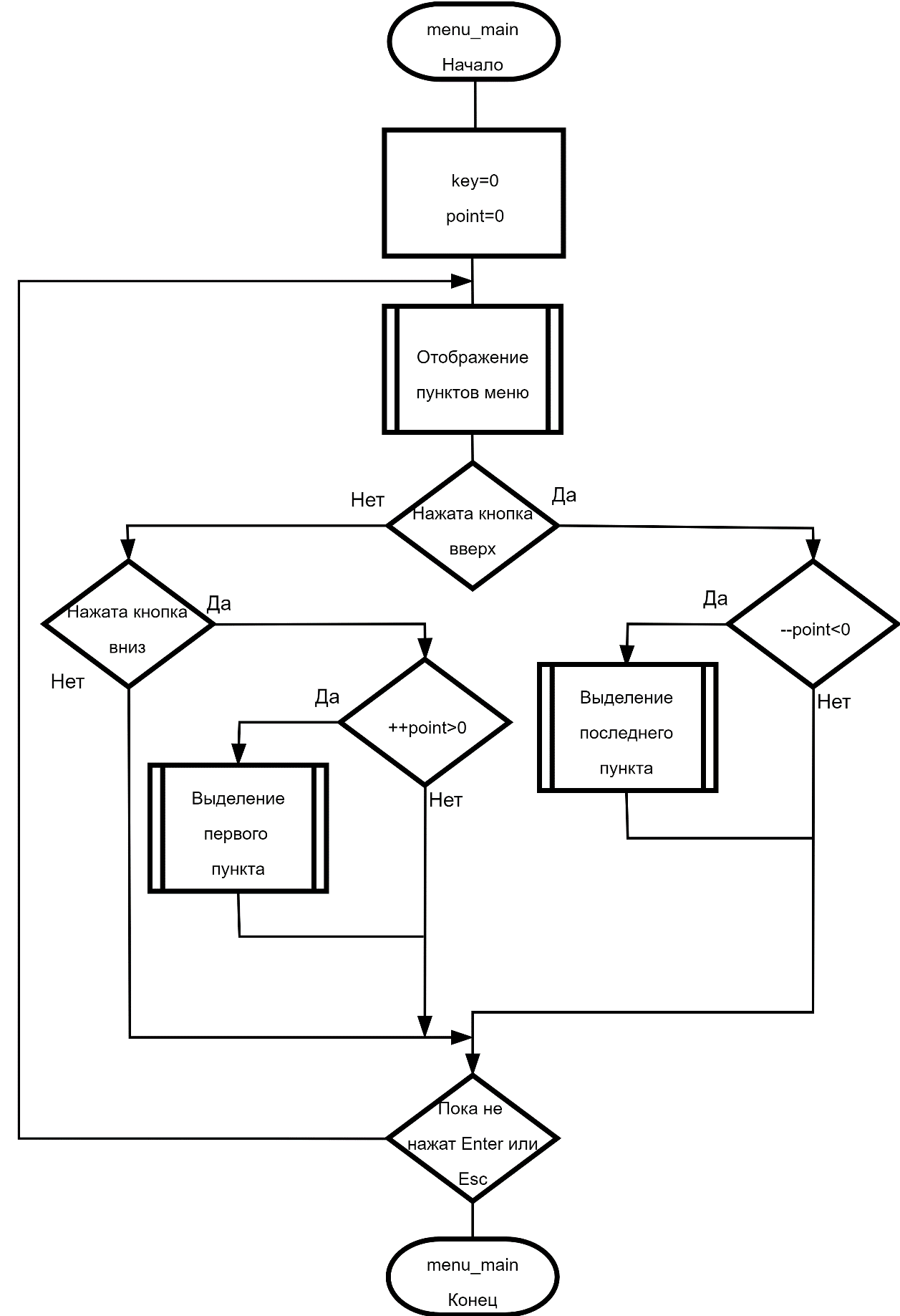


Рисунок 2 – Структурная схема функции menu\_main()

Следующая функция del\_index() удаляет элемент номер которого был введен. Ее схема представлена на Рисунке 3.

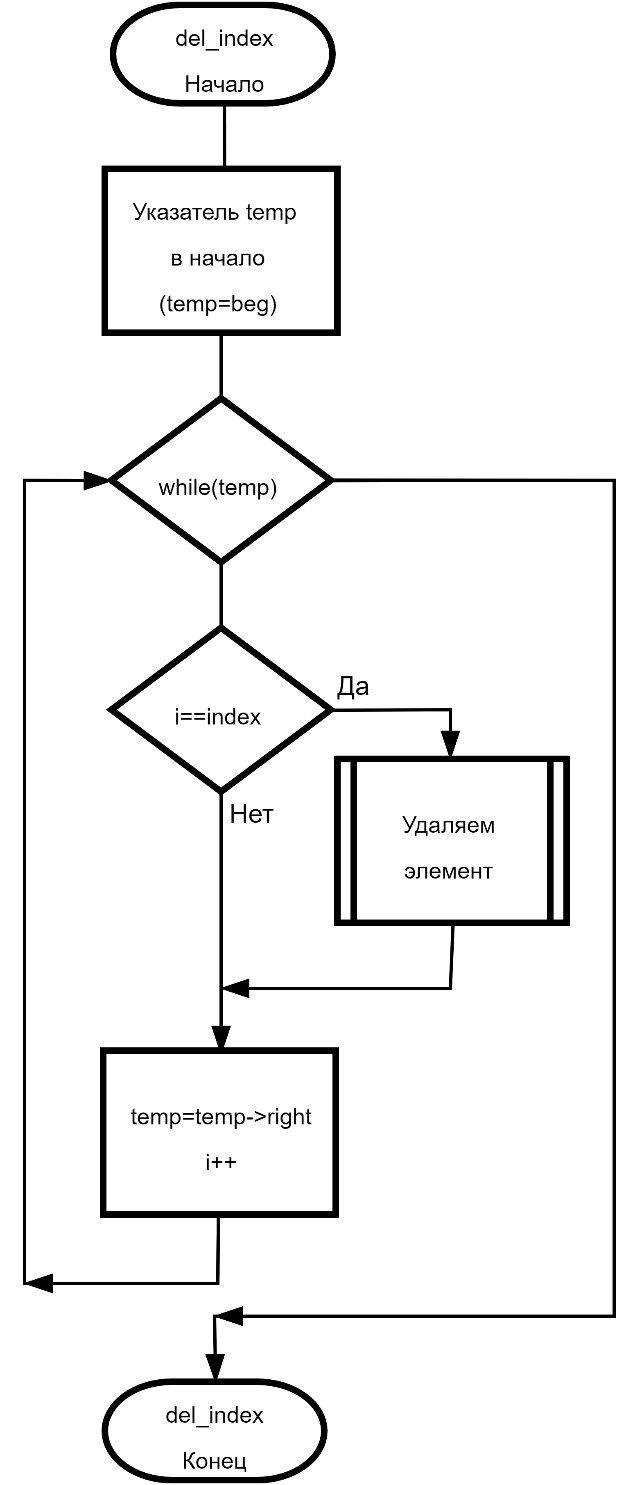


Рисунок 3 – Структурная схема функции del\_index()

Блоки:

1 блок – присваивание указателю temp начало списка

2 блок – цикл до конца списка

3 блок – проверка совпадения индексов

4 блок – удаление элемента

5 блок – сдвиг указателя temp на следующий элемент

Функция sort\_fio() и другие, аналогичные ей сортировки в программе, реализованы по схеме на Рисунке 4. Она поочерёдно сравнивает поля двух аргументов и меняет местами их данные, если это необходимо.

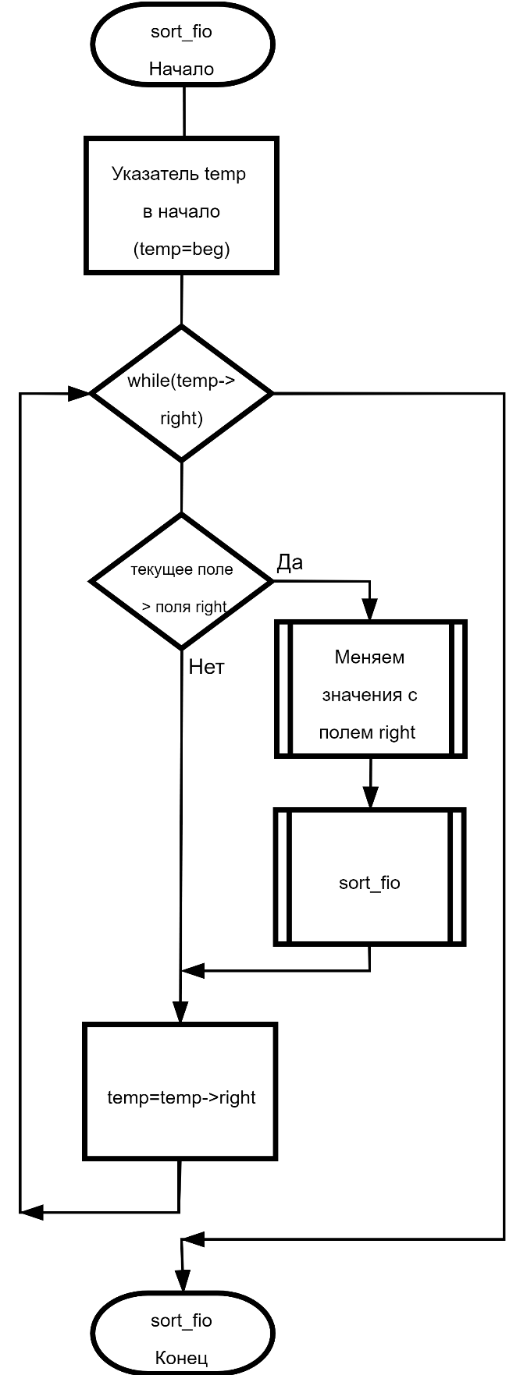


Рисунок 4 – Структурная схема функции sort\_fio()

Блоки:

1 блок – присваивание указателю temp начало списка

2 блок – цикл до конца списка

3 блок – проверка нахождения большего поля

4 блок – обмен значениями со следующим полем

5 блок – вызов этой же функции

6 блок – сдвиг указателя temp на следующий элемент

Функция search() получает от пользователя ключевое поле, по которому необходимо выполнить поиск. Если запись найдена будут выведены все данные для этой записи, иначе выведется текст “Данные не найдены”. Схема алгоритма функции search() представлена на Рисунке 5.

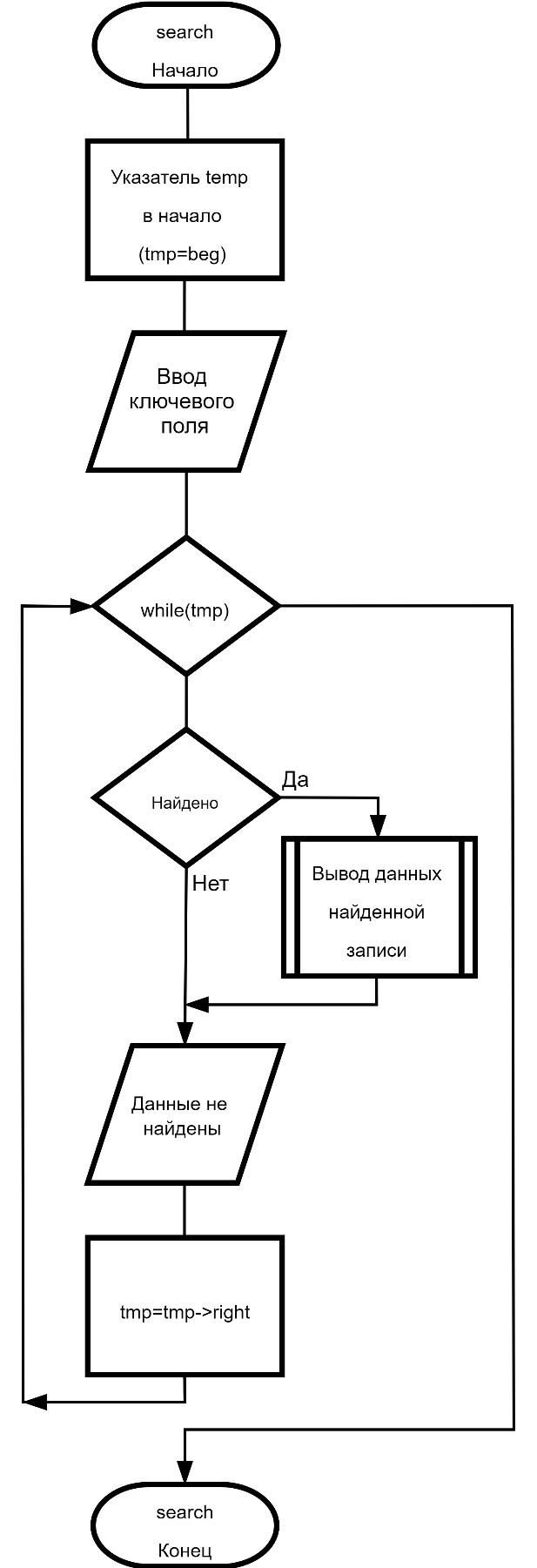


Рисунок 5 – Структурная схема функции search()

Блоки:

1 блок – присваивание указателю temp начало списка

2 блок – ввод ключевого поля для поиска

3 блок – цикл до конца списка

4 блок – проверка на нахождение элемента

5 блок – вывод данных найденной записи

6 блок – вывод сообщения о том, что данные не найдены

7 блок – сдвиг указателя temp на следующий элемент

Функция расчета времени time() выводит на экран время центрального процессора в процентах и сумму по видам времени. Схема функции time() приводится на Рисунке 6.



Рисунок 6 – Структурная схема функции time()

Блоки:

1 блок – присваивание указателю temp начало списка

2 блок – цикл до конца списка

3 блок – подсчет процентного времени и сумм по времени

4 блок – сдвиг указателя temp на следующий элемент

5 блок – вывод полученных данных

Функция prosmotr() реализует вывод таблицы базы данных на экран(рис. 7). Имеет возможность переключения между страницами, на которых находится по 10 записей.

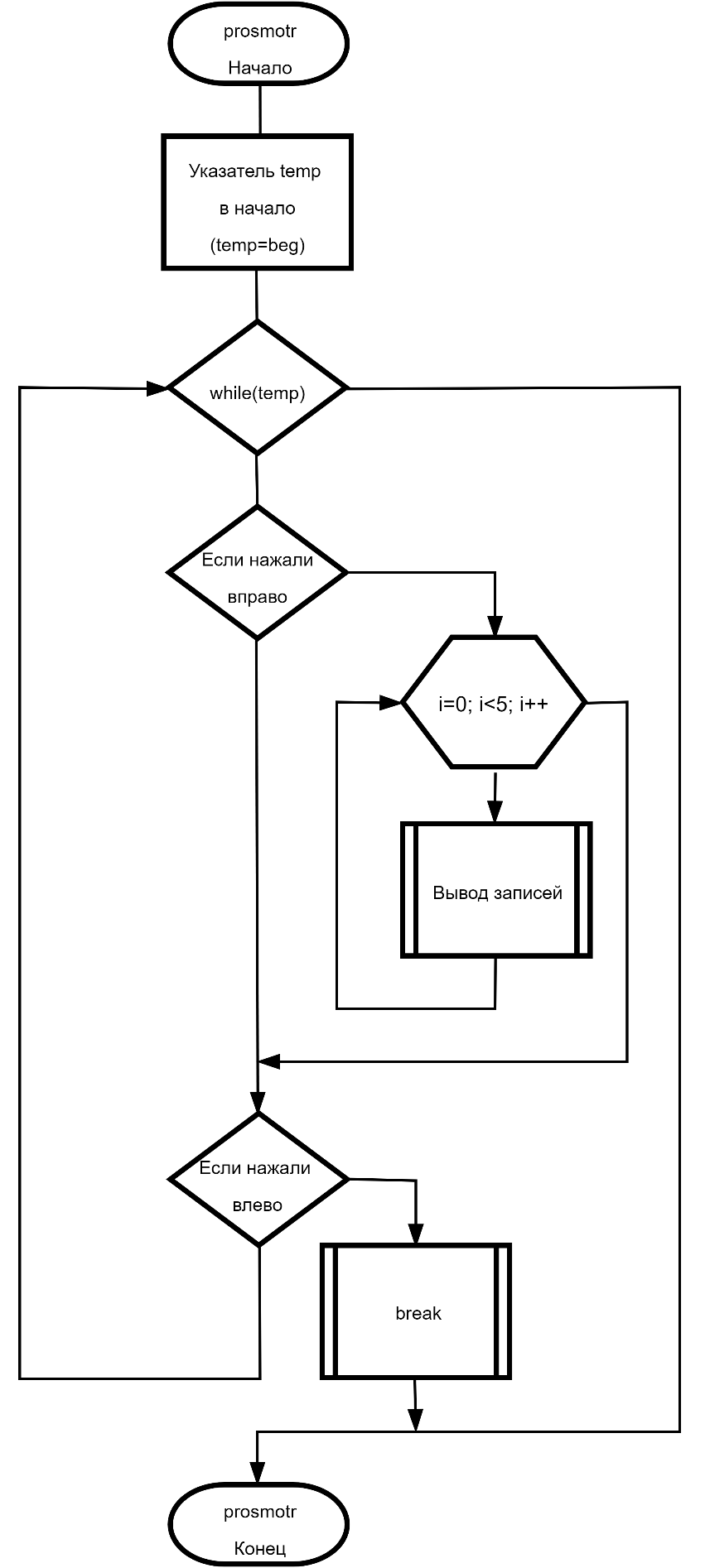


Рисунок 7 – Структурная схема функции prosmotr()

Блоки:

1 блок – присваивание указателю temp начало списка

2 блок – цикл до конца списка

3 блок – проверка на нажатие клавиши Вправо

4 блок – цикл для вывода записей

5 блок – функция вывода записей

6 блок – проверка на нажатие клавиши Влево

7 блок – функция break;

Схема алгоритма функции read\_file() представлена на Рисунке 8. Сохраняет таблицу в файл или выводит сообщение об ошибке.

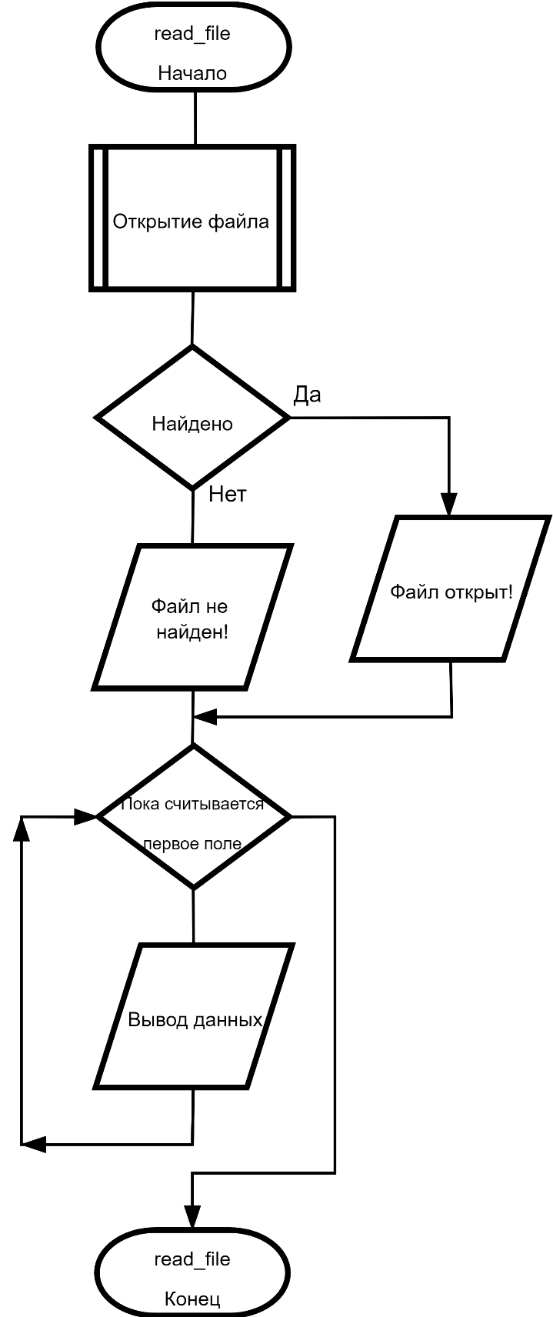


Рисунок 8 – Структурная схема функции read\_file()

Блоки:

1 блок – открытие файла

2 блок – проверка на открытие файла

3 блок – вывод сообщения о том, что файл не найден

4 блок – вывод сообщения о том, что файл найден

5 блок – цикл до конца файла

6 блок – вывод данных

Функция write\_file(), представленная на Рисунке 9 считывает базу данных из файла. Если считать файл не удалось, выведет сообщение об ошибке.

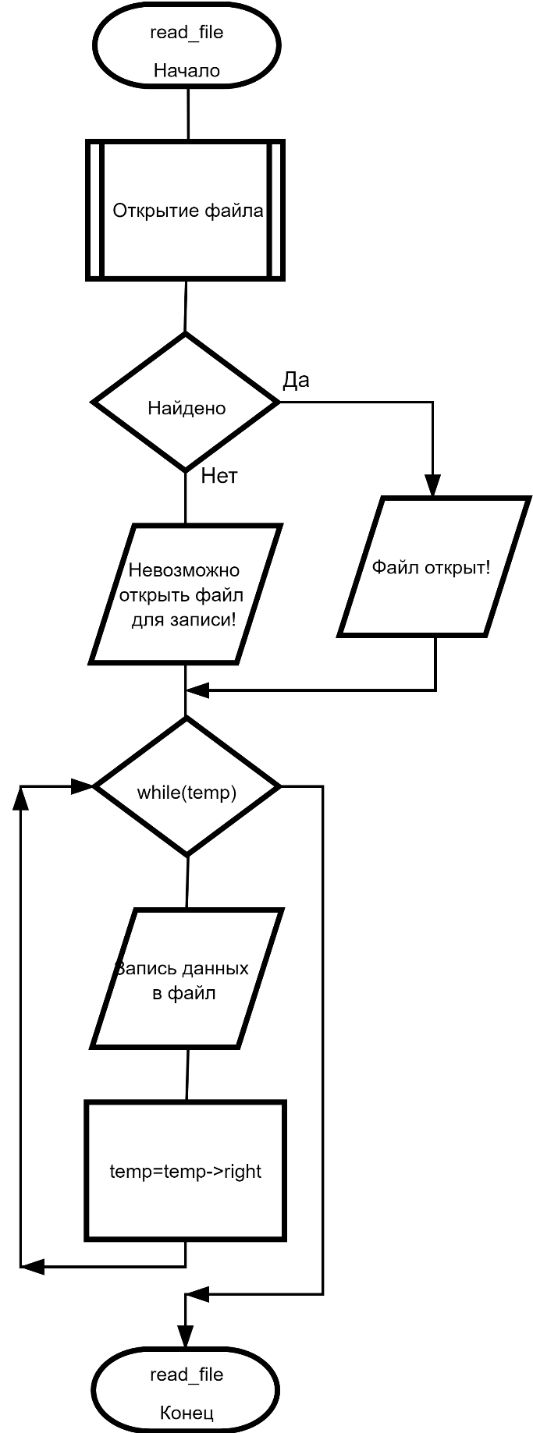


Рисунок 9 – Структурная схема функции write\_file()

Блоки:

1 блок – открытие файла

2 блок – проверка на открытие файла

3 блок – вывод сообщения о том, что файл не найден

4 блок – вывод сообщения о том, что файл найден

5 блок – цикл до конца списка

6 блок – запись данных в файл

7 блок – сдвиг указателя temp на следующий элемент

2. 5. Обоснование состава технических и программных средств

Размер файла, содержащий основной код программы составляет 43 Кб. В результате компиляции получается файл с расширением exe и размером 7,10 Мб (7449601байт). Программа потребляет около 6,3 Мб оперативной памяти. Поэтому она доступна для любого компьютера на базе операционной системой Windows.

Размер одной записи в файле составляет 16 байт.

Размер 1000 записей в файле будет составлять 16 Кб или 0,016 байт.

3 ВЫПОЛНЕНИЕ ПРОГРАММЫ

* 1. Условия выполнения программы

Минимальные системные требования:

- Операционная система: Windows XP / 7 / 8 / 10

- Оперативная память: 7 Мб

- Свободное место на жестком диске: 8 Мб

* 1. Загрузка и запуск программы

Запуск программы производится путем открытия файла EVM.exe. После запуска на экран будет выведен результат автоматического открытия стандартного файла (рис. 10).



Рисунок 10 – Открытие файла

Завершение программы выполняется после выбора пункта меню Выход. Результат выхода будет выведен на экран (рис. 11).



Рисунок 11 – Выход

* 1. Проверка работоспособности программы

После автоматического открытия файла пользователю станет доступно главное меню с названиями основных функций программы. Изображение данного меню приводится на рисунке 12.

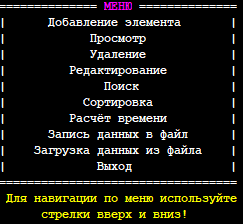
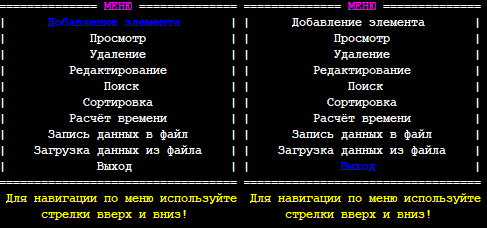


Рисунок 12 – Главное меню

Используя стрелки, пользователь перемещается между пунктами меню. Выбранный пункт подсвечивается синим цветом. Если пользователь, находясь на 1 пункте меню, нажимает стрелку Вверх, то выбирается последний пункт меню, а если пользователь нажмет стрелку Вниз то активным станет первый пункт меню. Данная возможность представлена на Рисунке 13.



а) б)

Рисунок 13 – Нажатие стрелки Вниз (а) и Вверх (б)

Добавление элемента в список происходит путем выбора соответствующего пункта в главном меню и, в меню добавления, пункта, указывающий на место, куда необходимо добавить элемент. Предусмотрена возможность добавления в начало списка, в середину и в конец. Демонстрация меню Добавления элемента представлена на Рисунке 14.

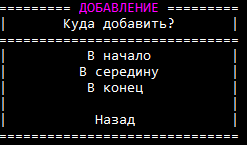


Рисунок 14 – Меню добавления

В функции добавления есть возможность добавления сразу нескольких элементов, а также реализована проверка на корректный ввод значений (рис. 15). Добавление в конец происходит аналогичным образом.

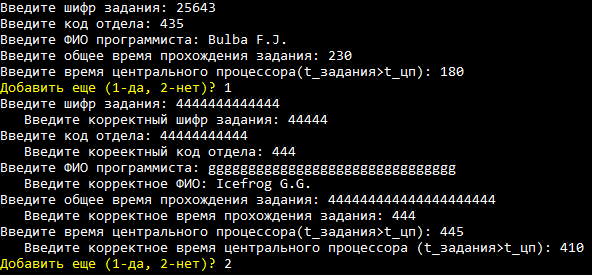


Рисунок 15 – Добавление в начало

При выборе пункта Добавления в середину, пользователю необходимо ввести номер элемента, после которого необходимо создать новую запись. Работа функции изображена на Рисунке 16.

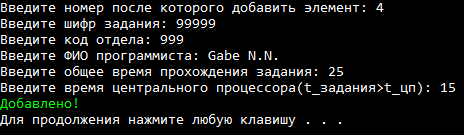


Рисунок 16 – Добавление в середину

Результаты добавления проиллюстрированы ниже, на Рисунке 17.

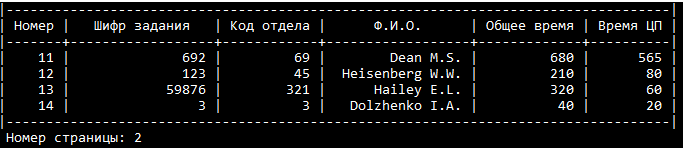


Рисунок 17 – Результат добавления

Просмотр списка активируется выбором соответствующего пункта главного меню (рис. 12). На каждой странице списка отображается только по 10 записей, поэтому пользователь имеет возможность переключения между страницами. Полный список базы показан на Рисунке 18.



а)



б)

Рисунок 18 – Полный список:

a) Первая страница;

б) Вторая страница

Для удаления реализовано отдельное меню с выбором того, чего необходимо удалить пользователю. Так при выборе удаления по номеру пользователю необходимо будет ввести номер удаляемой записи, после чего она будет удалена (рис. 19). Результат удаления продемонстрирован на Рисунке 20.



Рисунок 19 – Удаление по номеру



Рисунок 20 – Результат удаления по номеру

При выборе удаления всего списка, все записи будут очищены. Результат работы функции на Рисунке 21.



Рисунок 21 – Удаление всего списка

В программе предусмотрена сортировка по всем полям базы данных. Она имеет свое меню с выбором поля, по которому необходимо сортировать список (рис.22), и результат работы одной из них проиллюстрирован на Рисунке 23.

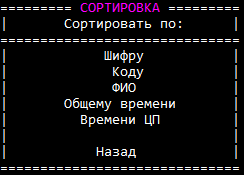
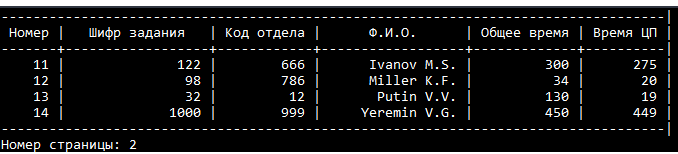


Рисунок 22 – Меню сортировки



а)



б)

Рисунок 23 – Результат сортировки по ФИО:

a) Первая страница;

б) Вторая страница

Поиск по базе данных имеет свое меню с выбором необходимых полей (рис. 24).

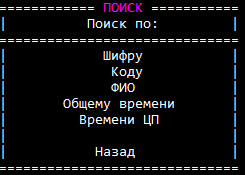


Рисунок 24 – Меню поиска

Результат поиска по ФИО представлен на Рисунке 25.

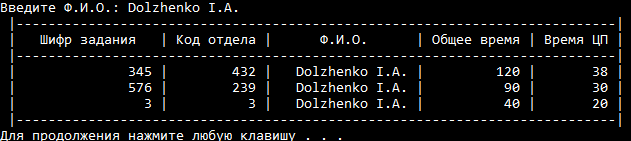


Рисунок 25 – Результат поиска по ФИО

Редактирование работает аналогично поиску, только с возможностью изменять нужные поля. Работа функции показана на рисунке 26.

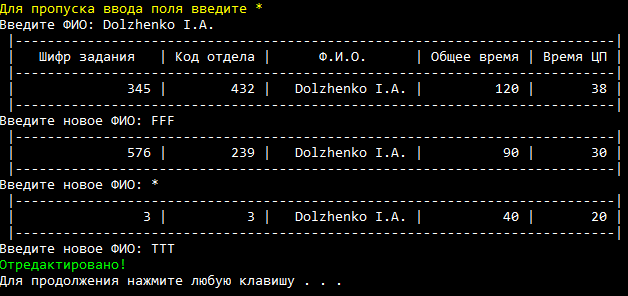


Рисунок 26 – Редактирование по ФИО

В программе реализована возможность подсчета процента времени центрального процессора по каждому заданию, а также суммы по видам времени по всем заданиям и средний процент времени по всем заданиям. Данная возможность продемонстрирована на Рисунке 27.

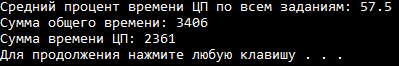


Рисунок 27 – Результат работы с временем

Если пользователю будет необходимо загрузить или сохранить файл, он сможет это сделать, выбрав необходимый пункт в главном меню (рис. 12). При выборе сохранения, будет открыто дополнительное меню сохранения (рис. 28) с возможностью сохранить в стандартный файл или новый, введя его название. Сохранения показано на Рисунке 29.

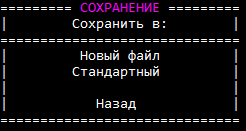


Рисунок 28 – Меню сохранения

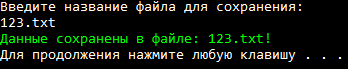


Рисунок 29 – Сохранение в свой файл

При выборе пункта загрузки из файла, пользователю будет предложено ввести название файла (рис. 30).

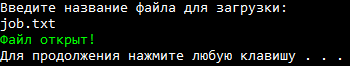


Рисунок 30 – Загрузка из файла

ВЫВОДЫ

Целью курсового проекта являлась систематизация, закрепление и углубление знаний в области программирования и совершенствование практических навыков разработки программ на языке Си и С++. Этим задачам было уделено огромное внимание во время написания программы.

В ходе выполнения курсового проекта была реализована программа, при помощи которой можно вести управление списком программистов. Программа способна создавать список, добавлять записи в начало, в конец и в середину, выполнять поиск, сортировку и редактирование по всем полям, удалять весь список и по номеру записи, а также сохранять и считывать данные в файле. В качестве алгоритма была выбрана структура данных в виде двунаправленного списка. Этот подход к организации данных помог упростить написание программы.

Таким образом, цель курсового проекта была достигнута. При небольших изменениях разработанная программа может применятся в различных предприятиях. Знания, полученные при разработке курсового проекта, оказались весьма обширными и полезными.

ПЕРЕЧЕНЬ ССЫЛОК

1. Павловская Т.А. C/C++. Программирование на языке высокого уровня. / Т.А. Павловская. – СПб.: Питер, 2003. – 461 с.
2. Боровский, А.Н. Qt4.7+. Практическое программирование на C++. / А.Н. Боровский. – СПб.: BHV, 2012. – 496 c.
3. Standard C++ Library reference [Электронный ресурс] cplusplus.com: информ.-справочный портал, 2000–2017. URL: http://www.cplusplus.com/reference/ (дата обращения: 25.11.2018).
4. Forever. Списки, стеки, очереди [Электронный ресурс] www.cyberforum.ru: форум программистов, компьютерный форум, программирование, 2000–2018. URL: http://www.cyberforum.ru/cpp-beginners/thread48705.html (дата обращения: 25.11.2018).
5. Операторы в C и C++ [Электронный ресурс]: Материал из Википедии — свободной энциклопедии. — Электрон. дан. — Сан-Франциско: Фонд Викимедиа, 2018. — URL: https://ru.wikipedia.org/?oldid=96144719
6. Связный список [Электронный ресурс]: Материал из Википедии — свободной энциклопедии. — Электрон. дан. — Сан-Франциско: Фонд Викимедиа, 2018. — URL: https://ru.wikipedia.org/?oldid=95249415

ПРИЛОЖЕНИЕ А

Код программы

#include <fstream>

#include <stdlib.h>

#include <stdio.h>

#include <string.h>

#include <string>

#include <time.h>

#include <iomanip>

#include <conio.h>

#include <iostream>

#include <Windows.h>

using namespace std;

//------------------------------- типы и глобальные переменные -------------------------------

struct job

{

char sh[9];

char kod[4];

char fio[16];

int t\_z;

int t\_cp;

job\* right, \*left;

};

//------------------------------- прототипы функций -------------------------------

job\* dob\_beg(job\* end, job\* beg, const job& z);

job\* dob\_end(job\* end, job\* beg, const job& z);

job\* dob\_first(const job& z);

job\* dob\_after(job\* beg, int index, const job& z);

job\* del\_index(job\* beg, int index);

job\* del\_all(job\* beg);

job vvod();

int menu\_main();

int menu\_add();

int menu\_del();

int menu\_sort();

int menu\_edit();

int menu\_search();

int menu\_save();

void sort\_sh(job\* beg);

void sort\_kod(job\* beg);

void sort\_fio(job\* beg);

void sort\_tz(job\* beg);

void sort\_tcp(job\* beg);

void edit(job\* beg);

void search(job\* beg);

void time(job\* beg);

void print(const job& z);

void prosmotr(job\* beg);

void Tab();

int read\_file(char\* filefio, job\*\* beg, job\*\* end);

int write\_file(char\* filename, job\* temp);

void TextColorO(char \*text);

void TextColorP(char \*text);

void TextColorB(char \*text);

void TextColorR(char \*text);

void TextColorG(char \*text);

//------------------------------- главная функция -------------------------------

int main()

{

setlocale(0,"RUS");

system ("mode con cols=100 lines=25");

system ("title Курсовой проект");

HANDLE hConsole=GetStdHandle(STD\_OUTPUT\_HANDLE);

int index=0;

int index1=0;

int number1=0;

int number=0;

job \*beg=0, \*end=0;

char \*filename="job.txt";

char filename\_s[15];

char filename\_z[15];

cout<<"Открытие стандартного файла job.txt: ";

read\_file(filename, &beg, &end);

while (1)

{

switch (menu\_main())

{

case 1:

{

switch (menu\_add())

{

case 1:

{

if (beg)

{

float iii=1;

while(iii==1)

{

beg=dob\_beg (end, beg, vvod());

TextColorO("Добавить еще (1-да, 2-нет)? ");

while (!(cin>>iii) || iii>2 || iii<1)

{

cin.clear();

while(cin.get()!='\n') continue;

cout<<"Введите корректное значение: ";

};

cin.get();

}

}

else

{

float iii=0;

beg=dob\_first(vvod());

end=beg;

TextColorO("Добавить еще (1-да, 2-нет)? ");

while (!(cin>>iii) || iii>2 || iii<1)

{

cin.clear();

while(cin.get()!='\n') continue;

cout<<" Введите корректное значение: ";

};

cin.get();

while(iii==1)

{

beg=dob\_beg (end, beg, vvod());

TextColorO("Добавить еще (1-да, 2-нет)? ");

while (!(cin>>iii) || iii>2 || iii<1)

{

cin.clear();

while(cin.get()!='\n') continue;

cout<<"Введите корректное значение: ";

};

cin.get();

}

}

TextColorG("Добавлено!");

system("pause");

break;

}

case 2:

{

job \*temp=beg;

if (beg)

{

cout<<"Введите номер после которого добавить элемент: ";

while (!(cin>>index1) )

{

cin.clear();

while(cin.get()!='\n') continue;

cout<<" Введите корректное значение: ";

};

do

{

++number1;

temp = temp->right;

}

while (temp!= NULL);

if(index1 > number1 || index1==0)

{

number1=0;

TextColorR("Такого номера нет!");

while(cin.get()!='\n');

system("pause");

break;

}

if(index1==number1)

{

while(cin.get()!='\n');

end=dob\_end (end, beg, vvod());

number1=0;

TextColorG("Добавлено!");

system("pause");

break;

}

while(cin.get()!='\n');

beg=dob\_after(beg, index1, vvod());

while(cin.get()!='\n');

number1=0;

TextColorG("Добавлено!");

}

else TextColorR("Список пустой!");

system("pause");

break;

}

case 3:

{

if (beg)

{

float iii=1;

while(iii==1)

{

end=dob\_end (end, beg, vvod());

TextColorO("Добавить еще (1-да, 2-нет)? ");

while (!(cin>>iii) || iii>2 || iii<1)

{

cin.clear();

while(cin.get()!='\n') continue;

cout<<"Введите корректное значение: ";

};

cin.get();

}

}

else

{

float iii=0;

beg=dob\_first(vvod());

end=beg;

TextColorO("Добавить еще (1-да, 2-нет)? ");

while (!(cin>>iii) || iii>2 || iii<1)

{

cin.clear();

while(cin.get()!='\n') continue;

cout<<" Введите корректное значение: ";

};

cin.get();

while(iii==1)

{

end=dob\_end (end, beg, vvod());

TextColorO("Добавить еще (1-да, 2-нет)? ");

while (!(cin>>iii) || iii>2 || iii<1)

{

cin.clear();

while(cin.get()!='\n') continue;

cout<<"Введите корректное значение: ";

};

cin.get();

}

}

TextColorG("Добавлено!");

system("pause");

break;

}

case 4: break;

}

break;

}

case 2:

{

system("cls");

if (!beg) {

TextColorR("Список пустой!");

system("pause");

}

else prosmotr(beg);

break;

}

case 3:

{

switch (menu\_del())

{

case 1:

{

if (!beg) TextColorR("Список пустой!");

else {

beg=del\_all(beg);

TextColorG("Удален весь список!");

}

system("pause");

break;

}

case 2:

{

if (beg)

{

job \*temp=beg;

cout<<"Введите номер удаляемого элемента: ";

while(!(cin>>index))

{

cin.clear();

while(cin.get()!='\n') continue;

cout<<" Введите корректный номер: ";

}

do

{

++number;

temp = temp->right;

}

while (temp!= NULL);

if(index > number)

{

TextColorR("Такого номера нет!");

system("pause");

cin.get();

}

else

{

beg=del\_index(beg, index);

SetConsoleTextAttribute(hConsole, FOREGROUND\_GREEN | FOREGROUND\_INTENSITY);

cout<<"Удалён элемент: "<<index<<endl;

SetConsoleTextAttribute(hConsole, FOREGROUND\_INTENSITY | FOREGROUND\_RED | FOREGROUND\_GREEN | FOREGROUND\_BLUE);

system("pause");

cin.get();

}

number=0;

}

else

{

TextColorR("Список пустой!");

cout<<"Для продолжения нажмите любую клавишу";

cin.get();

}

break;

}

case 3: break;

}

break;

}

case 4:

{

if (beg) edit(beg);

else

{

TextColorR("Список пустой!");

system("pause");

}

break;

}

case 5:

{

if (beg) search(beg);

else

{

TextColorR("Список пустой!");

system("pause");

}

break;

}

case 6:

{

if (beg)

{

switch (menu\_sort())

{

case 1:

{

sort\_sh(beg);

TextColorG("Отсортировано!");

system("pause");

break;

}

case 2:

{

sort\_kod(beg);

TextColorG("Отсортировано!");

system("pause");

break;

}

case 3:

{

sort\_fio(beg);

TextColorG("Отсортировано!");

system("pause");

break;

}

case 4:

{

sort\_tz(beg);

TextColorG("Отсортировано!");

system("pause");

break;

}

case 5:

{

sort\_tcp(beg);

TextColorG("Отсортировано!");

system("pause");

break;

}

case 6: break;

}

}

else

{

TextColorR("Список пустой!");

system("pause");

}

break;

}

case 7:

{

if (beg) time(beg);

else TextColorR("Список пустой!");

system("pause");

break;

}

case 8:

{

switch (menu\_save())

{

case 1:

{

cout<<"Введите название файла для сохранения:"<<endl; cin>>filename\_s;

write\_file(filename\_s, beg);

system("pause");

break;

}

case 2:

{

write\_file(filename, beg);

system("pause");

break;

}

case 3: break;

}

break;

}

case 9:

{

cout<<"Введите название файла для загрузки:"<<endl;

while(!(cin.getline(filename\_z, 16)))

{

cin.clear();

while(cin.get()!='\n') continue;

cout<<" Введите корректное название файла: ";

}

read\_file(filename\_z, &beg, &end);

break;

}

case 10:

TextColorG("Конец программы!");

return 0;

}

}

}

//------------------------------- добавление в начало -------------------------------

job\* dob\_beg(job \*end, job \*beg, const job& z)

{

job \*newE=new job;

\*newE=z;

newE->right=beg;

newE->left=0;

beg->left=newE;

beg=newE;

return beg;

}

//------------------------------- добавление в конец -------------------------------

job\* dob\_end(job \*end, job \*beg, const job& z)

{

job \*newE=new job;

\*newE=z;

newE->right=0;

newE->left=beg;

end->right=newE;

end=newE;

return end;

}

//------------------------------- добавление первым элементом -------------------------------

job\* dob\_first(const job& z)

{

job \*beg=new job;

\*beg=z;

beg->right=0;

beg->left=0;

return beg;

}

//------------------------------- добавление после -------------------------------

job\* dob\_after(job\* beg, int index1, const job& z)

{

int number=0;

job \*temp = beg;

job \*tmp = beg;

do

{

++number;

tmp = tmp->right;

}

while(tmp != NULL);

for(int i = 1; i < number; i++)

{

if(index1 == i)

{

job \*NewE = new job;

\*NewE = z;

NewE->right = temp->right;

Temp->right = NewE;

}

else temp = temp->right;

}

return beg;

}

//------------------------------- удалить все -------------------------------

job\* del\_all(job \*beg)

{

job \*p=beg;

while (p->right)

{

job \*temp;

temp = p;

p = p->right;

delete temp;

}

p=p->right;

delete p;

beg=NULL;

return beg;

}

//------------------------------- удаление по индексу -------------------------------

job\* del\_index(job\* beg, int index)

{

job\* p=0;

job\* temp=beg;

int i=1;

while(temp)

{

if(i==index)

{

if(p) p->right=temp->right;

else beg=temp->right;

delete temp;

return beg;

}

i++;

p=temp;

temp=temp->right;

}

return beg;

}

//------------------------------- сортировка по шифру -------------------------------

void sort\_sh(job\* beg)

{

int b1, b2;

job \*temp=beg;

while(temp->right)

{

b1=atoi(temp->sh);

b2=atoi(temp->right->sh);

if(b1>b2)

{

swap(temp->sh, temp->right->sh);

swap(temp->kod, temp->right->kod);

swap(temp->fio, temp->right->fio);

int tmp = temp->right->t\_z;

temp->right->t\_z = temp->t\_z;

temp->t\_z = tmp;

int tmp1 = temp->right->t\_cp;

temp->right->t\_cp = temp->t\_cp;

temp->t\_cp = tmp1;

temp=temp->right;

sort\_sh(beg);

}

else temp=temp->right;

}

}

//------------------------------- сортировка по коду -------------------------------

void sort\_kod(job\* beg)

{

int b1, b2;

job \*temp=beg;

while(temp->right)

{

b1=atoi(temp->kod);

b2=atoi(temp->right->kod);

if(b1>b2)

{

swap(temp->sh, temp->right->sh);

swap(temp->kod, temp->right->kod);

swap(temp->fio, temp->right->fio);

int tmp = temp->right->t\_z;

temp->right->t\_z = temp->t\_z;

temp->t\_z = tmp;

int tmp1 = temp->right->t\_cp;

temp->right->t\_cp = temp->t\_cp;

temp->t\_cp = tmp1;

temp=temp->right;

sort\_kod(beg);

}

else temp=temp->right;

}

}

//------------------------------- сортировка по фио -------------------------------

void sort\_fio(job\* beg)

{

job \*temp=beg;

while(temp->right)

{

if(strcmp(temp->fio, temp->right->fio)>0)

{

swap(temp->sh, temp->right->sh);

swap(temp->kod, temp->right->kod);

swap(temp->fio, temp->right->fio);

int tmp = temp->right->t\_z;

temp->right->t\_z = temp->t\_z;

temp->t\_z = tmp;

int tmp1 = temp->right->t\_cp;

temp->right->t\_cp = temp->t\_cp;

temp->t\_cp = tmp1;

temp=temp->right;

sort\_fio(beg);

}

else temp=temp->right;

}

}

//------------------------------- сортировка по общему времени -------------------------------

void sort\_tz(job\* beg)

{

job \*temp=beg;

while(temp->right)

{

if((temp->t\_z)>(temp->right->t\_z))

{

swap(temp->sh, temp->right->sh);

swap(temp->kod, temp->right->kod);

swap(temp->fio, temp->right->fio);

int tmp = temp->right->t\_z;

temp->right->t\_z = temp->t\_z;

temp->t\_z = tmp;

int tmp1 = temp->right->t\_cp;

temp->right->t\_cp = temp->t\_cp;

temp->t\_cp = tmp1;

temp=temp->right;

sort\_tz(beg);

}

else temp=temp->right;

}

}

//------------------------------- сортировка по времени цп -------------------------------

void sort\_tcp(job\* beg)

{

job \*temp=beg;

while(temp->right)

{

if((temp->t\_cp)>(temp->right->t\_cp))

{

swap(temp->sh, temp->right->sh);

swap(temp->kod, temp->right->kod);

swap(temp->fio, temp->right->fio);

int tmp = temp->right->t\_z;

temp->right->t\_z = temp->t\_z;

temp->t\_z = tmp;

int tmp1 = temp->right->t\_cp;

temp->right->t\_cp = temp->t\_cp;

temp->t\_cp = tmp1;

temp=temp->right;

sort\_tcp(beg);

}

else temp=temp->right;

}

}

//------------------------------- редактирование -------------------------------

void edit(job \*beg)

{

int index, cp, o\_t;

int number=0; int number1=0;

char fio1[16];

bool not\_found = true;

switch (menu\_edit())

{

case 1:

{

system("cls");

cout<<"Введите номер: ";

while(!(cin>>index))

{

cin.clear();

while(cin.get()!='\n') continue;

cout<<" Введите корректный номер: ";

}

job\* tmp=beg;

do

{

++number1;

tmp = tmp->right;

}

while (tmp!= NULL);

if(index > number1)

{

TextColorR("Такого номера нет!");

system("pause");

cin.get();

break;

}

Tab();

job\* temp = beg;

int i=1;

while(temp)

{

if(i == index)

{

print(\*temp);

cout << " |---------------------------------------------------------------------------|" <<endl;

cout<<"Введите новый шифр задания: ";

while(cin.get()!='\n');

while(!cin.getline(temp->sh, 9))

{

cin.clear();

while(cin.get()!='\n');

cout<<" Введите корректный новый шифр задания (8 символов): ";

}

cout<<"Введите новый код отдела: ";

while(!cin.getline(temp->kod, 4))

{

cin.clear();

while(cin.get()!='\n');

cout<<" Введите корректный новый код отдела (3 символа): ";

}

cout<<"Введите новое ФИО программиста: ";

while(!cin.getline(temp->fio, 16))

{

cin.clear();

while(cin.get()!='\n');

cout<<" Введите корректное новое ФИО (15 символов): ";

}

cout<<"Введите новое общее время прохождения задания: ";

while(!(cin>>temp->t\_z) || !(temp->t\_z>0 && temp->t\_z<99999))

{

cin.clear();

while(cin.get()!='\n') continue;

cout<<" Введите корректное новое время прохождения задания (5 знаков): ";

}

cout<<"Введите новое время центрального процессора:: ";

while(!(cin>>temp->t\_cp) || (temp->t\_cp>temp->t\_z))

{

cin.clear();

while(cin.get()!='\n') continue;

cout<<" Введите корректное новое время центрального процессора (t\_задания>t\_цп): ";

}

not\_found = false;

}

i++;

temp = temp->right;

}

if (!not\_found) TextColorG("Отредактировано!");

system("pause");

cin.get();

break;

}

case 2:

{

char bfio[16];

char \*b="\*";

int i=0;

system("cls");

TextColorO("Для пропуска ввода поля введите \*\n");

cout<<"Введите ФИО: ";

while(!cin.getline(fio1, 16))

{

cin.clear();

while(cin.get()!='\n');

cout<<" Введите корректное ФИО (15 символов): ";

}

Tab();

job\* temp = beg;

while(temp)

{

if(strncmp(temp->fio, fio1, 4)==0)

{

if(i>0) cout << " |---------------------------------------------------------------------------|" <<endl;

print(\*temp);

cout << " |---------------------------------------------------------------------------|" <<endl;

cout<<"Введите новое ФИО (15 символов): ";

cin.getline(bfio, 16);

if (!strcmp(bfio, b))

{

temp = temp->right;

not\_found = false; i++;

continue;

}

else

{

strcpy(temp->fio, bfio);

not\_found = false; i++;

temp = temp->right;

continue;

}

}

else temp = temp->right;

}

if (not\_found) TextColorR("Данные не найдены!");

else TextColorG("Отредактировано!");

system("pause");

break;

}

case 3:

{

int i=0;

system("cls");

cout<<"Введите общее время: ";

while(!(cin>>o\_t))

{

cin.clear();

while(cin.get()!='\n') continue;

cout<<" Введите корректное общее время (5 знаков): ";

}

Tab();

job\* temp = beg;

while(temp)

{

if(temp->t\_z == o\_t)

{

if(i>0) cout << " |---------------------------------------------------------------------------|" <<endl;

print(\*temp);

cout << " |---------------------------------------------------------------------------|" <<endl;

cout<<"Введите новое общее время: ";

while(!(cin>>temp->t\_z) || !(temp->t\_z>0 && temp->t\_z<99999))

{

cin.clear();

while(cin.get()!='\n') continue;

cout<<" Введите корректное новое общее время (5 знаков): ";

}

not\_found = false; i++;

temp = temp->right;

}

else temp = temp->right;

}

if (not\_found) TextColorR("Данные не найдены!");

else TextColorG("Отредактировано!");

system("pause");

cin.get();

break;

}

case 4:

{

int i=0;

system("cls");

cout<<"Введите время центрального процессора: ";

while(!(cin>>cp))

{

cin.clear();

while(cin.get()!='\n') continue;

cout<<" Введите корректное время центрального процессора (t\_задания>t\_цп): ";

}

Tab();

job\* temp = beg;

while(temp)

{

if(temp->t\_cp == cp)

{

if(i>0) cout << " |---------------------------------------------------------------------------|" <<endl;

print(\*temp);

cout << " |---------------------------------------------------------------------------|" <<endl;

cout<<"Введите новое время центрального процессора: ";

while(!(cin>>temp->t\_cp) || (temp->t\_cp>temp->t\_z))

{

cin.clear();

while(cin.get()!='\n') continue;

cout<<" Введите корректное новое время центрального процессора (t\_задания>t\_цп): ";

}

not\_found = false; i++;

}

else temp = temp->right;

}

if (not\_found) TextColorR("Данные не найдены!");

else TextColorG("Отредактировано!");

system("pause");

cin.get();

break;

}

case 5: break;

}

}

//------------------------------- поиск -------------------------------

void search(job \*beg)

{

int o\_t, t\_cp;

char fio[16], sh[9], kod[4];

bool not\_found = true;

job\* tmp = beg;

switch (menu\_search())

{

case 1:

{

cout<<"Введите шифр: "; cin>>sh; cin.get();

Tab();

while(tmp)

{

if(!strcmp(tmp->sh, sh))

{

print(\*tmp);

not\_found = false;

}

tmp = tmp->right;

}

if (not\_found) TextColorR("Данные не найдены!");

else cout << " |---------------------------------------------------------------------------|" <<endl;

system("pause");

break;

}

case 2:

{

cout<<"Введите код: "; cin>>kod; cin.get();

Tab();

while(tmp)

{

if(!strcmp(tmp->kod, kod))

{

print(\*tmp);

not\_found = false;

}

tmp = tmp->right;

}

if (not\_found) TextColorR("Данные не найдены!");

else cout << " |---------------------------------------------------------------------------|" <<endl;

system("pause");

break;

}

case 3:

{

cout<<"Введите Ф.И.О.: "; cin>>fio;

while(cin.get()!='\n');

Tab();

while(tmp)

{

if(strncmp(tmp->fio, fio, 4)==0)

{

print(\*tmp);

not\_found = false;

}

tmp = tmp->right;

}

if (not\_found) TextColorR("Данные не найдены!");

else cout << " |---------------------------------------------------------------------------|" <<endl;

system("pause");

break;

}

case 4:

{

cout<<"Введите общее время: "; cin>>o\_t; cin.get();

Tab();

while(tmp)

{

if(tmp->t\_z==o\_t)

{

print(\*tmp);

not\_found = false;

}

tmp = tmp->right;

}

if (not\_found) TextColorR("Данные не найдены!");

else cout << " |---------------------------------------------------------------------------|" <<endl;

system("pause");

break;

}

case 5:

{

cout<<"Введите время ЦП: "; cin>>t\_cp; cin.get();

Tab();

while(tmp)

{

if(tmp->t\_cp==t\_cp)

{

print(\*tmp);

not\_found = false;

}

tmp = tmp->right;

}

if (not\_found) TextColorR("Данные не найдены!");

else cout << " |---------------------------------------------------------------------------|" <<endl;

system("pause");

break;

}

case 6: break;

}

}

//------------------------------- функция ввода -------------------------------

job vvod()

{

job z;

cout<<"Введите шифр задания: ";

while(!cin.getline(z.sh, 9))

{

cin.clear();

while(cin.get()!='\n');

cout<<" Введите корректный шифр задания (8 символов): ";

}

cout<<"Введите код отдела: ";

while(!cin.getline(z.kod, 4))

{

cin.clear();

while(cin.get()!='\n');

cout<<" Введите кореектный код отдела (3 символа): ";

}

cout<<"Введите ФИО программиста: ";

while(!cin.getline(z.fio, 16))

{

cin.clear();

while(cin.get()!='\n');

cout<<" Введите корректное ФИО (15 символов): ";

}

cout<<"Введите общее время прохождения задания: ";

while(!(cin>>z.t\_z) || !(z.t\_z>0 && z.t\_z<99999))

{

cin.clear();

while(cin.get()!='\n') continue;

cout<<" Введите корректное время прохождения задания (5 знаков): ";

}

cout<<"Введите время центрального процессора (t\_задания>t\_цп): ";

while(!(cin>>z.t\_cp) || (z.t\_cp>z.t\_z))

{

cin.clear();

while(cin.get()!='\n') continue;

cout<<" Введите корректное время центрального процессора (t\_задания>t\_цп): ";

}

return z;

}

//------------------------------- главное меню -------------------------------

int menu\_main()

{

char key = 0;

int point = 0;

do

{

system("cls");

if (key==72)

{

if (--point<0) point=10;

}

else if (key==80)

{

if (++point>10) point=0;

}

cout<<"============== "; TextColorP("МЕНЮ"); cout<<" =============="<<endl;

if(point==1)

{

cout<<"|";

TextColorB(" Добавление элемента ");

cout<<"|"<<endl;

}

else cout<<"| Добавление элемента"<<setw(8)<<"|"<<endl;

if(point==2)

{

cout<<"|";

TextColorB(" Просмотр ");

cout<<"|"<<endl;

}

else cout<<"| Просмотр"<<setw(13)<<"|"<<endl;

if(point==3)

{

cout<<"|";

TextColorB(" Удаление ");

cout<<"|"<<endl;

}

else cout<<"| Удаление"<<setw(13)<<"|"<<endl;

if(point==4)

{

cout<<"|";

TextColorB(" Редактирование ");

cout<<"|"<<endl;

}

else cout<<"| Редактирование"<<setw(10)<<"|"<<endl;

if(point==5)

{

cout<<"|";

TextColorB(" Поиск ");

cout<<"|"<<endl;

}

else cout<<"| Поиск"<<setw(14)<<"|"<<endl;

if(point==6)

{

cout<<"|";

TextColorB(" Сортировка ");

cout<<"|"<<endl;

}

else cout<<"| Сортировка"<<setw(12)<<"|"<<endl;

if(point==7)

{

cout<<"|";

TextColorB(" Расчёт времени ");

cout<<"|"<<endl;

}

else cout<<"| Расчёт времени"<<setw(10)<<"|"<<endl;

if(point==8)

{

cout<<"|";

TextColorB(" Запись данных в файл ");

cout<<"|"<<endl;

}

else cout<<"| Запись данных в файл"<<setw(7)<<"|"<<endl;

if(point==9)

{

cout<<"|";

TextColorB(" Загрузка данных из файла ");

cout<<"|"<<endl;

}

else cout<<"| Загрузка данных из файла"<<setw(5)<<"|"<<endl;

if(point==10)

{

cout<<"|";

TextColorB(" Выход ");

cout<<"|"<<endl;

}

else cout<<"| Выход"<<setw(15)<<"|"<<endl;

cout<<"=================================="<<endl;

TextColorO(" Для навигации по меню используйте\n");

TextColorO(" стрелки вверх и вниз!\n");

key=getch();

}

while (key!=13 && key!=27);

system ("cls");

return key==13 ? point : 10;

}

//------------------------------- меню добавления -------------------------------

int menu\_add()

{

char key = 0;

int point = 0;

do

{

system("cls");

if (key==72)

{

if (--point<0) point=4;

}

else if (key==80)

{

if (++point>4) point=0;

}

cout<<"========= "; TextColorP("ДОБАВЛЕНИЕ"); cout<<" ========="<<endl;

cout<<"| Куда добавить? |" << endl;

cout<<"=============================="<<endl;

if(point==1)

{

cout<<"|";

TextColorB(" В начало");

cout<<setw(11)<<"|"<<endl;

}

else cout<<"| В начало"<<setw(11)<<"|"<<endl;

if(point==2)

{

cout<<"|";

TextColorB(" В середину");

cout<<setw(10)<<"|"<<endl;

}

else cout<<"| В середину"<<setw(10)<<"|"<<endl;

if(point==3)

{

cout<<"|";

TextColorB(" В конец");

cout<<setw(12)<<"|"<<endl;

}

else cout<<"| В конец"<<setw(12)<<"|"<<endl;

cout<<"|"<<setw(29)<<"|"<<endl;

if(point==4)

{

cout<<"|";

TextColorB(" Назад");

cout<<setw(13)<<"|"<<endl;

}

else cout<<"| Назад"<<setw(13)<<"|"<<endl;

cout<<"=============================="<<endl;

key=getch();

}

while (key!=13 && key!=27);

system ("cls");

return key==13 ? point : 4;

}

//------------------------------- меню удаления -------------------------------

int menu\_del()

{

char key = 0;

int point = 0;

do

{

system("cls");

if (key==72)

{

if (--point<0) point=3;

}

else if (key==80)

{

if (++point>3) point=0;

}

cout<<"========== "; TextColorP("УДАЛЕНИЕ"); cout<<" =========="<<endl;

cout<<"| Что удалить? |" << endl;

cout<<"=============================="<<endl;

if(point==1)

{

cout<<"|";

TextColorB(" Весь список");

cout<<setw(10)<<"|"<<endl;

}

else cout<<"| Весь список"<<setw(10)<<"|"<<endl;

if(point==2)

{

cout<<"|";

TextColorB(" По номеру");

cout<<setw(11)<<"|"<<endl;

}

else cout<<"| По номеру"<<setw(11)<<"|"<<endl;

cout<<"|"<<setw(29)<<"|"<<endl;

if(point==3)

{

cout<<"|";

TextColorB(" Назад");

cout<<setw(13)<<"|"<<endl;

}

else cout<<"| Назад"<<setw(13)<<"|"<<endl;

cout<<"=============================="<<endl;

key=getch();

}

while (key!=13 && key!=27);

system ("cls");

return key==13 ? point : 3;

}

//------------------------------- меню сортировки -------------------------------

int menu\_sort()

{

char key = 0;

int point = 0;

do

{

system("cls");

if (key==72)

{

if (--point<0) point=6;

}

else if (key==80)

{

if (++point>6) point=0;

}

cout<<"========= "; TextColorP("СОРТИРОВКА"); cout<<" ========="<<endl;

cout<<"| Сортировать по: |" << endl;

cout<<"=============================="<<endl;

if(point==1)

{

cout<<"|";

TextColorB(" Шифру");

cout<<setw(12)<<"|"<<endl;

}

else cout<<"| Шифру"<<setw(12)<<"|"<<endl;

if(point==2)

{

cout<<"|";

TextColorB(" Коду");

cout<<setw(12)<<"|"<<endl;

}

else cout<<"| Коду"<<setw(12)<<"|"<<endl;

if(point==3)

{

cout<<"|";

TextColorB(" ФИО");

cout<<setw(13)<<"|"<<endl;

}

else cout<<"| ФИО"<<setw(13)<<"|"<<endl;

if(point==4)

{

cout<<"|";

TextColorB(" Общему времени");

cout<<setw(8)<<"|"<<endl;

}

else cout<<"| Общему времени"<<setw(8)<<"|"<<endl;

if(point==5)

{

cout<<"|";

TextColorB(" Времени ЦП");

cout<<setw(10)<<"|"<<endl;

}

else cout<<"| Времени ЦП"<<setw(10)<<"|"<<endl;

cout<<"|"<<setw(29)<<"|"<<endl;

if(point==6)

{

cout<<"|";

TextColorB(" Назад");

cout<<setw(13)<<"|"<<endl;

}

else cout<<"| Назад"<<setw(13)<<"|"<<endl;

cout<<"=============================="<<endl;

key=getch();

}

while (key!=13 && key!=27);

system ("cls");

return key==13 ? point : 6;

}

//------------------------------- меню редактирования -------------------------------

int menu\_edit()

{

char key = 0;

int point = 0;

do

{

system("cls");

if (key==72)

{

if (--point<0) point=5;

}

else if (key==80)

{

if (++point>5) point=0;

}

cout<<"======= "; TextColorP("РЕДАКТИРОВАНИЕ"); cout<<" ======="<<endl;

cout<<"| Что отредактировать? |" << endl;

cout<<"=============================="<<endl;

if(point==1)

{

cout<<"|";

TextColorB(" Весь элемент");

cout<<setw(9)<<"|"<<endl;

}

else cout<<"| Весь элемент"<<setw(9)<<"|"<<endl;

if(point==2)

{

cout<<"|";

TextColorB(" ФИО");

cout<<setw(13)<<"|"<<endl;

}

else cout<<"| ФИО"<<setw(13)<<"|"<<endl;

if(point==3)

{

cout<<"|";

TextColorB(" Общее время");

cout<<setw(9)<<"|"<<endl;

}

else cout<<"| Общее время"<<setw(9)<<"|"<<endl;

if(point==4)

{

cout<<"|";

TextColorB(" Время ЦП");

cout<<setw(11)<<"|"<<endl;

}

else cout<<"| Время ЦП"<<setw(11)<<"|"<<endl;

cout<<"|"<<setw(29)<<"|"<<endl;

if(point==5)

{

cout<<"|";

TextColorB(" Назад");

cout<<setw(13)<<"|"<<endl;

}

else cout<<"| Назад"<<setw(13)<<"|"<<endl;

cout<<"=============================="<<endl;

key=getch();

}

while (key!=13 && key!=27);

system ("cls");

return key==13 ? point : 5;

}

//------------------------------- меню поиска -------------------------------

int menu\_search()

{

char key = 0;

int point = 0;

do

{

system("cls");

if (key==72)

{

if (--point<0) point=6;

}

else if (key==80)

{

if (++point>6) point=0;

}

cout<<"============ "; TextColorP("ПОИСК"); cout<<" ==========="<<endl;

cout<<"| Поиск по: |" << endl;

cout<<"=============================="<<endl;

if(point==1)

{

cout<<"|";

TextColorB(" Шифру");

cout<<setw(12)<<"|"<<endl;

}

else cout<<"| Шифру"<<setw(12)<<"|"<<endl;

if(point==2)

{

cout<<"|";

TextColorB(" Коду");

cout<<setw(12)<<"|"<<endl;

}

else cout<<"| Коду"<<setw(12)<<"|"<<endl;

if(point==3)

{

cout<<"|";

TextColorB(" ФИО");

cout<<setw(13)<<"|"<<endl;

}

else cout<<"| ФИО"<<setw(13)<<"|"<<endl;

if(point==4)

{

cout<<"|";

TextColorB(" Общему времени");

cout<<setw(8)<<"|"<<endl;

}

else cout<<"| Общему времени"<<setw(8)<<"|"<<endl;

if(point==5)

{

cout<<"|";

TextColorB(" Времени ЦП");

cout<<setw(10)<<"|"<<endl;

}

else cout<<"| Времени ЦП"<<setw(10)<<"|"<<endl;

cout<<"|"<<setw(29)<<"|"<<endl;

if(point==6)

{

cout<<"|";

TextColorB(" Назад");

cout<<setw(13)<<"|"<<endl;

}

else cout<<"| Назад"<<setw(13)<<"|"<<endl;

cout<<"=============================="<<endl;

key=getch();

}

while (key!=13 && key!=27);

system ("cls");

return key==13 ? point : 6;

}

//------------------------------- меню сохранения -------------------------------

int menu\_save()

{

char key = 0;

int point = 0;

do

{

system("cls");

if (key==72)

{

if (--point<0) point=3;

}

else if (key==80)

{

if (++point>3) point=0;

}

cout<<"========= "; TextColorP("СОХРАНЕНИЕ"); cout<<" ========="<<endl;

cout<<"| Сохранить в: |" << endl;

cout<<"=============================="<<endl;

if(point==1)

{

cout<<"|";

TextColorB(" Новый файл");

cout<<setw(10)<<"|"<<endl;

}

else cout<<"| Новый файл"<<setw(10)<<"|"<<endl;

if(point==2)

{

cout<<"|";

TextColorB(" Стандартный");

cout<<setw(10)<<"|"<<endl;

}

else cout<<"| Стандартный"<<setw(10)<<"|"<<endl;

cout<<"|"<<setw(29)<<"|"<<endl;

if(point==3)

{

cout<<"|";

TextColorB(" Назад");

cout<<setw(13)<<"|"<<endl;

}

else cout<<"| Назад"<<setw(13)<<"|"<<endl;

cout<<"=============================="<<endl;

key=getch();

}

while (key!=13 && key!=27);

system ("cls");

return key==13 ? point : 3;

}

//------------------------------- вывод значений -------------------------------

void print(const job& z)

{

cout << " |" <<setw(17) << z.sh << " |" << setw(11) << z.kod << " |" << setw(17) << z.fio << " |" << setw(12) << z.t\_z << " |" <<setw(9) << z.t\_cp << " |";

cout<<endl;

}

//------------------------------- просмотр -------------------------------

void prosmotr(job \*beg)

{

int i=1; char key = 77; int gg=0; int str=1;

job \*temp=beg;

TextColorO(" Для пролистывания таблицы используйте стрелку ->\n");

TextColorO(" Либо <- для выхода из просмотра\n");

while(temp)

{

if (key==77)

{

cout << "|-----------------------------------------------------------------------------------|" <<endl;

cout<< "| Номер | Шифр задания | Код отдела | Ф.И.О. | Общее время | Время ЦП |" <<endl;

cout << "|-------+------------------+------------+------------------+-------------+----------|" <<endl;

for(gg=0;gg<10;)

{

if(temp)

{

cout << "|" << setw(6) << i;

print(\*temp);

temp=temp->right;

i++; gg++;

}

else break;

}

cout << "|-----------------------------------------------------------------------------------|" <<endl;

cout << " Номер страницы: " << str++ << endl;

}

if (key==75) break;

key=getch();

system("cls");

}

}

//------------------------------- расчет времени -------------------------------

void time(job\* beg)

{

int sum1=0; int sum2=0; int sum3=0; int procent=0;

float sr=0;

int ii=0;

int i=1; char key = 77; int gg=0;

int str=1;

TextColorO(" Для пролистывания таблицы используйте стрелку ->\n");

TextColorO(" Либо <- для перехода к итогу таблицы\n");

job \*temp=beg;

while(temp)

{

procent=(temp->t\_cp\*100)/temp->t\_z;

sum1+=temp->t\_cp;

sum2+=temp->t\_z;

sum3+=procent;

ii++;

procent=0;

temp=temp->right;

}

temp=beg;

while(temp)

{

if (key==77)

{

cout << "|------------------------------------------------------------------------------------------------|" <<endl;

cout<< "| Номер | Шифр задания | Код отдела | Ф.И.О. | Общее время | Время ЦП | % ЦП |" <<endl;

cout << "|-------+------------------+------------+------------------+-------------+----------+------------|" <<endl;

for(gg=0;gg<10;)

{

if(temp)

{

procent=(temp->t\_cp\*100)/temp->t\_z;

cout << "|" << setw(6) << i;

cout << " |" <<setw(17) << temp->sh;

cout << " |" << setw(11) << temp->kod;

cout << " |" << setw(17) << temp->fio;

cout << " |" << setw(12) << temp->t\_z;

cout << " |" <<setw(9) << temp->t\_cp;

cout << " |"<<setw(11) << procent << " |"<<endl;

procent=0;

temp=temp->right;

i++; gg++;

}

else break;

}

cout << "|------------------------------------------------------------------------------------------------|" <<endl;

cout << " Номер страницы: " << str++ << endl;

}

if (key==75) break;

key=getch();

system("cls");

}

sr=(float)sum3/ii;

cout<<"Средний процент времени ЦП по всем заданиям: "<<sr<<endl;

cout<<"Сумма общего времени: "<<sum2<<endl;

cout<<"Сумма времени ЦП: "<<sum1<<endl;

}

//------------------------------- чтение файла -------------------------------

int read\_file(char\* filename, job\*\* beg, job\*\* end)

{

ifstream fin(filename,ios::in);

if (!fin)

{

TextColorR("Файл не найден!");

cout<<"Для продолжения нажмите любую клавишу . . .";

cin.get();

return 1;

}

else

{

TextColorG("Файл открыт!");

cout<<"Для продолжения нажмите любую клавишу . . .";

cin.get();

}

job z;

\*beg = 0;

while (fin.getline(z.sh, 9))

{

fin.getline(z.kod, 4);

fin.getline(z.fio, 16);

fin>>z.t\_z;

fin>>z.t\_cp;

fin.get();

if (\*beg)

\*end=dob\_end(\*end, \*beg, z);

else

{ \*beg=dob\_first(z); \*end=\*beg; }

}

return 0;

}

//------------------------------- запись файла -------------------------------

int write\_file(char\* filename, job\* temp)

{

ofstream fout(filename);

if (!fout)

{

TextColorR("Невозможно открыть файл для записи!");

return 1;

}

while (temp)

{

fout<<temp->sh<<endl;

fout<<temp->kod<<endl;

fout<<temp->fio<<endl;

fout<<temp->t\_z<<endl;

fout<<temp->t\_cp<<endl;

temp=temp->right;

}

HANDLE hConsole = GetStdHandle (STD\_OUTPUT\_HANDLE);

SetConsoleTextAttribute(hConsole, FOREGROUND\_GREEN | FOREGROUND\_INTENSITY);

cout<<"Данные сохранены в файле: "<<filename<<"!"<<endl;

SetConsoleTextAttribute(hConsole, FOREGROUND\_INTENSITY | FOREGROUND\_RED | FOREGROUND\_GREEN | FOREGROUND\_BLUE);

cin.ignore(1);

return 0;

}

//------------------------------- функция вывода заголовка таблицы -------------------------------

void Tab()

{

cout << " |---------------------------------------------------------------------------|" <<endl;

cout<< " | Шифр задания | Код отдела | Ф.И.О. | Общее время | Время ЦП |" <<endl;

cout << " |---------------------------------------------------------------------------|" <<endl;

}

//------------------------------- зеленый цвет -------------------------------

void TextColorG(char \*text)

{

HANDLE hConsole=GetStdHandle(STD\_OUTPUT\_HANDLE);

SetConsoleTextAttribute(hConsole, FOREGROUND\_GREEN | FOREGROUND\_INTENSITY);

cout<<text<<endl;

SetConsoleTextAttribute(hConsole, FOREGROUND\_INTENSITY | FOREGROUND\_RED | FOREGROUND\_GREEN | FOREGROUND\_BLUE);

}

//------------------------------- красный цвет -------------------------------

void TextColorR(char \*text)

{

HANDLE hConsole=GetStdHandle(STD\_OUTPUT\_HANDLE);

SetConsoleTextAttribute(hConsole, FOREGROUND\_INTENSITY | FOREGROUND\_RED);

cout<<text<<endl;

SetConsoleTextAttribute(hConsole, FOREGROUND\_INTENSITY | FOREGROUND\_RED | FOREGROUND\_GREEN | FOREGROUND\_BLUE);

}

//------------------------------- синий цвет -------------------------------

void TextColorB(char \*text)

{

HANDLE hConsole=GetStdHandle(STD\_OUTPUT\_HANDLE);

SetConsoleTextAttribute(hConsole, FOREGROUND\_INTENSITY | FOREGROUND\_BLUE);

cout<<text;

SetConsoleTextAttribute(hConsole, FOREGROUND\_INTENSITY | FOREGROUND\_RED | FOREGROUND\_GREEN | FOREGROUND\_BLUE);

}

//------------------------------- фиолетовый цвет -------------------------------

void TextColorP(char \*text)

{

HANDLE hConsole=GetStdHandle(STD\_OUTPUT\_HANDLE);

SetConsoleTextAttribute(hConsole, FOREGROUND\_INTENSITY | FOREGROUND\_RED | FOREGROUND\_BLUE);

cout<<text;

SetConsoleTextAttribute(hConsole, FOREGROUND\_INTENSITY | FOREGROUND\_RED | FOREGROUND\_GREEN | FOREGROUND\_BLUE);

}

//------------------------------- оранжевый цвет -------------------------------

void TextColorO(char \*text)

{

HANDLE hConsole=GetStdHandle(STD\_OUTPUT\_HANDLE);

SetConsoleTextAttribute(hConsole, FOREGROUND\_INTENSITY | FOREGROUND\_RED | FOREGROUND\_GREEN);

cout<<text;

SetConsoleTextAttribute(hConsole, FOREGROUND\_INTENSITY | FOREGROUND\_RED | FOREGROUND\_GREEN | FOREGROUND\_BLUE);

}