МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение высшего образования «Казанский национальный исследовательский технический университет им. А.Н. Туполева-КАИ»

Чистопольский филиал «Восток»

Кафедра компьютерных и телекоммуникационных систем

МЕТОДИЧЕСКИЕ УКАЗАНИЯ

по выполнению

КУРСОВОЙ РАБОТЫ

по дисциплине (модулю)

Программирование и основы алгоритмизации

Методические указания (рекомендации) по выполнению курсовой работы/курсового проекта предназначены для обучающихся всех форм обучения по направлениям подготовки:

Код и наименование направления подготовки / специальности	Направленность (профиль, специализация, магистерская программа)	ФГОС ВО утвержден приказом Минобрнауки России
09.03.03 Прикладная информатика	Прикладная информатика в цифровой экономике	№ 922 от 19.09.2017

В методических указаниях приведены требования к выполнению курсовой работы, даны рекомендации по структуре, содержанию, оформлению, порядку выполнения и защите курсовой работы по дисциплине (модулю) Программирование и основы алгоритмизации.

Разработчик(и):

Ефимова Юлия Викторовна, кандидат педагогических наук

Методические указания рассмотрены на заседании кафедры Компьютерных и телекоммуникационных систем, протокол № 6 от 27.02.2023г.

Заведующий кафедрой _компьютерных и телекоммуникационных систем_____

Классен Виктор Иванович, д.т.н.

Содержание

1 Цели и задачи курсовой работы

Целью курсовой работы научиться разрабатывать на языке высокого уровня (C/C++) сложные программы, имеющие многофункциональную структуру, а также описания разработанного программного продукта как с точки зрения его пользователя, так и с точки зрения разработчика.

В результате выполнения курсовой работы студенты должны:

1. Изучить и применить принципы поэтапной разработки и отладки программ средней сложности.

2. Показать умение самостоятельно и грамотно формализовать поставленные перед ним задачи;

3. Пользоваться научной, учебной и справочной литературой;

4. Знать основные алгоритмические конструкции, знать алфавит, лексемы, типы данных и операторы языка программирования высокого уровня;

5. Уметь составлять алгоритмы решения прикладных задач и реализовывать алгоритмы на языке высокого уровня;

6. Иметь навыки разработки программного обеспечения решения прикладных задач;

7. Иметь навыки тестирования прикладных программ.

2 Требования к курсовой работе

Работа должна быть выполнена в течение одного семестра. По выданному преподавателем заданию необходимо разработать и отладить на компьютере программу на языке С (или С++) и оформить отчет по курсовой работе.

Оценка курсовой работы зависит от качества выполнения и оформления курсовой работы, от результатов и сроков ее защиты.

Программа должна иметь многофункциональную структуру: состоять не из одной главной функции main( ), но и еще из нескольких подпрограмм. Можно оформить отдельные функции в виде отдельных модулей, которые компилируются раздельно, а потом с помощью компоновщика объединяются в единый загрузочный модуль (exe-файл).

Всю необходимую функциям информацию следует передавать в виде параметров, а не через глобальные переменные, изменение которых трудно отследить.

Программа должна удовлетворять требованиям структурного программирования, должна содержать комментарии, поясняющие назначение каждой функции, ее параметров, локальных переменных и отдельных ее частей. Программа должна быть хорошо читаемой и понятной не только Вам, но и любому пользователю, знакомому с языком С (С++). Для наглядности программы соблюдайте ступенчатую форму записи, имена функций и переменных выбирайте осмысленные (это поможет при отладке программы).

Программа должна быть отлажена. Для этого нужно обдуманно подобрать набор тестов, на котором будет проверяться программа, причем делать это нужно не после написания программы, а до написания (тесты "черного ящика") и в процессе написания (тесты "белого ящика").

3 Тематика курсовых работ

Примерная тематика курсовых работ:

1. Дан файл, содержащий информацию о сотрудниках отдела учреждения. Структура записи файла:

фамилия и инициалы сотрудника;

год рождения;

пол;

стаж работы;

отдел;

оклад.

Написать программу, выдающую

– информацию о сотрудниках с заданным стажем;

– средний оклад сотрудников заданного отдела.

2. Дан файл, содержащий сведения о сдаче сессии студентами группы. Структура записи файла:

фамилия и инициалы студента;

оценки по четырем экзаменам

оценки по пяти зачетам («з» - зачет, «н» - незачет).

Написать программу, выдающую следующую информацию:

– фамилии студентов, не имеющих задолженностей;

– процент неуспевающих студентов.

3. Дан файл, содержащий информацию о пользователях сайта знакомств. Структура записи файла:

фамилия, имя;

знак Зодиака;

дата рождения.

Написать программу, выдающую:

– информацию о людях, родившихся под заданным знаком Зодиака;

– информацию о людях с заданной фамилией.

4. Дан файл, содержащий информацию о маршрутах городского транспорта. Структура записи файла:

номер маршрута;

вид транспорта (а - автобус, т – трамвай, м -маршрутное такси),

начальный пункт,

конечный пункт,

время в пути.

Написать программу, выдающую

– список всех маршрутов из пункта А или в пункт А;

– номер самого длинного маршрута из А или в А;

5. Дан файл, содержащий сведения о сдаче сессии студентами группы. Структура записи файла:

фамилия и инициалы студента;

оценки по четырем экзаменам

оценки по пяти зачетам («з» - зачет, «н» - незачет).

Написать программу, выдающую следующую информацию:

– фамилии студентов, не имеющих задолженностей по экзаменам;

– фамилии студентов, имеющих наибольшее число задолженностей.

6. Дан файл, содержащий сведения о сдаче сессии студентами группы. Структура записи файла:

фамилия и инициалы студента;

оценки по четырем экзаменам

оценки по пяти зачетам («з» - зачет, «н» - незачет).

Написать программу, выдающую следующую информацию:

– фамилии студентов, имеющих не более одной задолженности по экзаменам;

– фамилии студентов, имеющих наибольшее число задолженностей по экзаменам.

7. Дан файл, содержащий информацию о поступлении и продаже товаров, хранящихся на складе, в течение месяца. Структура записи файла:

шифр товара;

наименование товара;

число месяца;

количество единиц поступившего товара.

Написать программу, выдающую

– количество товара с заданным шифром;

– количество товара заданного наименования.

8. Дан файл, содержащий информацию о сотрудниках организации. Структура записи файла:

фамилия, имя и отчество сотрудника;

должность;

возраст сотрудника;

год поступления на работу в организацию.

Написать программу, выдающую

– список сотрудников, поступивших в заданный год;

– информацию о сотрудниках заданного возраста.

9. Дан файл, содержащий информацию о маршрутах городского транспорта. Структура записи файла:

номер маршрута,

вид транспорта (а - автобус, т - троллейбус, м - маршрутное такси),

начальный пункт,

конечный пункт,

время в пути.

Написать программу, выдающую

– список всех маршрутов указанного вида транспорта,

– номер маршрута, время в пути для которого из А в В максимальное.

10. Дан файл, содержащий информацию о маршрутах городского транспорта. Структура записи файла:

номер маршрута;

вид транспорта (а - автобус, т – трамвай, м -маршрутное такси),

начальный пункт,

конечный пункт,

время в пути.

Написать программу, выдающую

– номер маршрута с минимальным временем следования из А или в А;

– количество маршрутов каждого вида транспорта, следующих из А или в А.

11. Дан файл, содержащий информацию о сотрудниках организации. Структура записи файла:

фамилия, имя и отчество сотрудника;

должность;

год поступления на работу в организацию.

Написать программу, выдающую

– список сотрудников, чей стаж работы в организации превышает заданный;

– сведения о заданном сотруднике.

12. Дан файл, содержащий информацию о маршрутах городского транспорта. Структура записи файла:

номер маршрута,

вид транспорта (а - автобус, т - троллейбус, м - маршрутное такси),

начальный пункт,

конечный пункт,

время в пути.

Написать программу, выдающую

– количество маршрутов из пункта A в пункт B;

– номер маршрута, время в пути для которого из A в B минимальное.

13. Дан файл, содержащий информацию о сотрудниках организации. Структура записи файла:

фамилия, имя и отчество сотрудника;

должность;

год поступления на работу в организацию.

Написать программу, выдающую

– список сотрудников, занимающих указанную должность;

– список сотрудников, чей стаж работы в организации не превышает заданный.

14. Дан файл, содержащий сведения о мобильных телефонах сотрудников предприятия.

Структура записи файла:

фамилия и инициалы сотрудника;

год поступления на работу в организацию.

номер телефона.

Написать программу, выдающую:

– номер телефона заданного сотрудника;

– сведения о заданном сотруднике.

15. Дан файл, содержащий сведения о телефонах абонентов. Структура записи файла:

фамилия и инициалы абонента;

год установки телефона;

номер телефона.

Написать программу, выдающую:

– телефон заданного абонента;

– фамилию абонента с заданным номером телефона.

16. Дан файл, содержащий сведения об ассортименте обуви фирмы. Структура записи файла:

артикул. Артикул начинается с буквы Д для дамской обуви, М для мужской, П для детской;

наименование;

количество;

стоимость одной пары.

Написать программу, выдающую информацию:

– информацию об обуви артикула X;

– количество пар женской обуви с указанием ее стоимости.

17. Дан файл, содержащий информацию о поездах дальнего следования. Структура записи файла:

номер поезда;

станция назначения;

станция отправления;

время отправления;

время в пути.

Написать программу, выдающую

– информацию о поездах, следующих до города X;

– номер поезда с минимальным временем следования до города X (с указанием времени отправления и прибытия).

18. Дан файл, содержащий информацию о сотрудниках организации. Структура записи файла:

фамилия, имя и отчество сотрудника;

должность;

возраст сотрудника;

год поступления на работу в организацию.

Написать программу, выдающую

– список сотрудников пенсионного возраста;

– информацию о заданном сотруднике.

19. Дан файл, содержащий информацию о поступлении и продаже товаров, хранящихся на складе, в течение месяца. Структура записи файла:

шифр товара;

наименование товара;

число месяца;

количество единиц поступившего товара.

Написать программу, выдающую

– количество товара, поступившего в заданный день;

– количество товара заданного количества.

20. Дан файл, содержащий информацию о движении пригородных поездов. Структура записи файла:

номер поезда;

время отправления;

станция назначения;

время прибытия;

дни недели движения (е - ежедневно, р - рабочие дни, с - субботние и воскресные дни).

Написать программу, выдающую

– информацию о поездах, следующих до заданной станции ежедневно, с указанием времени в пути;

– количество поездов, прибывающих на заданную станцию.

20. Дан файл, содержащий информацию о сотрудниках отдела учреждения. Структура записи файла:

фамилия и инициалы сотрудника;

год рождения;

пол;

стаж работы;

отдел;

оклад.

Написать программу, выдающую

– информацию о сотрудниках женского пола;

– средний оклад сотрудников пенсионного возраста заданного отдела.

21. Дан файл, содержащий информацию о движении пригородных поездов. Структура записи файла:

номер поезда;

время отправления;

станция назначения;

время прибытия;

дни недели движения (е - ежедневно, р - рабочие дни, с - субботние и воскресные дни).

Написать программу, выдающую

– информацию о поездах, следующих от заданной станции, с указанием времени в пути;

– количество поездов, с заданным временем в пути.

22. Дан файл, содержащий информацию о сотрудниках отдела учреждения. Структура записи файла:

фамилия и инициалы сотрудника;

год рождения;

пол;

стаж работы;

оклад.

Написать программу, выдающую

– информацию о заданном сотруднике;

– информацию о сотрудниках заданного года рождения.

4 Этапы разработки программ

Разработка любой сложной программы осуществляется поэтапно.

I этап. Разобравшись в постановке задачи, нужно создать так называемую внешнюю спецификацию программы, которая включает в себя:

- описание входных и выходных данных;
- описание задачи, реализуемой программой;
- способ обращения к программе;
- описание возможных аварийных ситуаций и ошибок пользователя.

Входные и выходные данные должны быть представлены в удобной для пользователя форме.

На этом этапе полезно подобрать тесты “черного ящика” для проверки программы (т.е. рассматривая программу как черный ящик, подобрать такой набор тестов, чтобы учесть все возможные ситуации, в том числе и ошибочные). Это поможет лучше понять задачу.

II этап. Разработка внутренних структур для представления входных, выходных и промежуточных данных.

От выбора структур данных будет во многом зависеть алгоритм решения задачи. Например, если выходные данные должны представлять собой сортированную таблицу, можно создать в памяти таблицу в виде вектора, а затем выполнить ее сортировку, а можно создать динамическую сортированную таблицу в виде списка.

III этап. Проектирование структуры программы.

На этом этапе применяется технология нисходящего проектирования программ. Задача разбивается на подзадачи, которые можно рассматривать раздельно. Каждая подзадача оформляется как отдельная функция (подпрограмма). Для каждой подпрограммы составляется внешняя спецификация, аналогичная приведенной выше.

Затем при пошаговой детализации каждая подзадача может быть разбита на еще более мелкие подзадачи, которые также оформляются как отдельные функции. При разработке структуры функции вначале нужно выбрать наиболее эффективный метод решения задачи и записать в обобщенной форме алгоритм (например, в виде укрупненной блок-схемы), затем решить, какие части алгоритма (блоки схемы) целесообразно представить в виде отдельных функций.

Здесь важно продумать интерфейс – способ взаимодействия функций. Интерфейс функции определяется ее заголовком (где указываются имя функции, ее параметры и тип возвращаемого значения).

На этом этапе нужно также решить, из каких модулей будет состоять программа. Модуль может содержать одну или несколько взаимосвязанных функций и общие для функций данные. Каждый модуль помещается в отдельный файл (с расширением ***.c*** или ***.cpp***) и компилируется автономно. Получившиеся в результате компиляции объектные модули объединяются в единый исполняемый модуль (exe-файл) с помощью компоновщика (Linker).

Разбиение программы на модули уменьшает время перекомпиляции программы и облегчает процесс отладки, позволяя отлаживать программу по частям. Чем более независимы модули, тем легче отлаживать программу.

Одна из основных задач – хорошо продумать внешний интерфейс модулей. Интерфейсом модуля являются заголовки всех функций и описания доступных извне типов, переменных и констант.

Примечание. Описания глобальных типов данных, констант, переменных и прототипы всех функций лучше поместить в отдельный файл с расширением **.h** , а в остальные файлы (с расширением **.c** или **.cpp**) включать эти описания с помощью директивы **#include <имя.h>** . Глобальные переменные, правда, лучше не использовать (тогда интерфейс каждой функции будет полностью определяться ее заголовком).

Таким образом, на данном этапе строится модульная и функциональная структура программы и определяется интерфейс между модулями и функциями, а также разрабатываются обобщенные алгоритмы функций.

Процесс проектирования структуры программы является итерационным, поскольку в сложных программах невозможно сразу продумать все детали.

IV этап. Структурное программирование и тестирование программы.

Процесс программирования и тестирования также организуется по принципу «сверху-вниз». Вначале описывается главная функция программы, затем функции следующего уровня и т. д. Параллельно разрабатываются тесты для проверки каждой функции и каждого модуля. Набор тестов должен быть достаточно большим.

Тесты подбираются по методам «черного и белого ящиков». В методах «черного ящика» программа (подпрограмма) рассматривается, как черный ящик: известно только какие выходные данные должны получиться при определенных входных данных. Каждый тест должен содержать определенные входные данные и ожидаемый результат. Тесты должны охватывать как правильные, так и неправильные входные данные. Например, если входные данные читаются из заданного файла, то следует взять и «неправильные тесты»: например, когда файла с заданным именем нет в указанной папке, когда файл пустой. Тесты «черного ящика» следует подбирать еще на предыдущем этапе.

В тестах «белого ящика» учитывается алгоритм программы (подпрограммы). Каждая ветвь в структурах ветвления (операторы if, switch), каждый цикл должны проверяться при тестировании. Например, программа выводит меню из четырех пунктов, и в зависимости от номера выбранного пункта либо вызывает одну из трех подпрограмм, либо завершает свою работу (см. пример программы в приложении). Необходимо взять по крайней мере пять тестов: по одному для каждого пункта меню и с неверным номером пункта меню.

Прежде чем тестировать программу на компьютере полезно выполнить ручное тестирование. Это поможет выявить и исправить ошибки еще до ввода программы.

При написании программы и ее тестировании по принципу «сверху-вниз» функции нижнего уровня (еще не написанные) заменяются так называемыми «заглушками». В простейшем случае «заглушки» просто выдают сообщение о том, что подпрограмма вызвана. Например, для проверки работы главной функции программы, описанной в приложении, вместо каждой из трех вызываемых подпрограмм (Date, MiddleAge, Diagnos) можно написать «заглушку». Пример для функции Date:

```
void Date (FILE *f)
{   puts ("\nФункция Date вызвана ");
}
```

После отладки главной функции и написания определения, например, функции Date, «заглушка» этой функции заменяется на ее определение и выполняется тестирование этой функции (совместно с главной). Затем постепенно добавляются и проверяются остальные функции.

Если в процессе тестирования функции результаты окажутся неверными, то нужно будет отладить эту функцию. В процессе отладки может измениться не только алгоритм функции, но и появиться новые подпрограммы, т.е. может измениться функциональная структура программы.

5 Содержание отчета по курсовой работе

После отладки программы следует оформить отчет по курсовой работе. Отчет должен содержать полную документацию к разработанной программе. Пример отчета смотрите в приложении.

Текстовая часть курсовой работы оформляется в виде пояснительной записки (ПЗ) в соответствии с требованиями.

Тексты программ помещаются в приложении, в конце пояснительной записки. Тексты программ обязательно снабжаются блочными и построчными комментариями.

Пояснительная записка должна содержать:

Титульный лист;

Лист оглавления;

1 Задание;

2. Основную часть, содержащую описание применения программы;

3. Основную часть, содержащую описание разработанной программы;

4. Список используемой литературы.

5. Приложение. Текст программы.

В первом разделе приводится формулировка задания в том виде, в каком она дана преподавателем.

Второй раздел должен содержать описание применения программы. В этом разделе нужно дать всю необходимую информацию для пользователей: как запустить программу, как подготовить для нее входные данные (исходный файл и данные, вводимые с клавиатуры при выполнении программы), какие будут выходные данные, какие сообщения может выдавать программа и в каких случаях (в том числе и сообщения об ошибках).

В третьем разделе дается подробное описание программы: метод решения задачи, модульная (функциональная) структура программы, описание каждого модуля. При описании модуля указываете, из каких функций состоит модуль, и

даете подробное описание каждой функции.

Описание функции должно включать ее назначение, интерфейс (заголовок функции с пояснением ее параметров и возвращаемого значения), описание локальных переменных и описание алгоритма функции в виде блок-схемы (если структура линейная, то можно дать словесное описание алгоритма).

Приложения к отчету должны содержать листинг программы, примеры (один или несколько) входного файла и результаты тестирования программы.

6 Рекомендации по оформлению курсовой работы

В рамках рассмотренной выше структуры курсовой работы рекомендуется использовать следующие правила оформления.

Объем курсовой работы: 20-30 страниц стандартного текста формата А4 (210 х 297), набранных через полтора интервала на одной стороне листа белой бумаги в текстовом процессоре Word. Шрифт текста должен быть четким. Размер шрифта – 14 пунктов.

Поля: левое – 25-30 мм, правое - 10 мм, верхнее -20 мм, нижнее - 25 мм. Абзацный отступ должен быть одинаковым и равным 1,25-1,27 см (равен одному нажатию клавиши Tab).

Титульный лист оформляется по образцу, приведенному в приложении.

Каждый раздел (глава) начинаются с нового листа. Каждый параграф (подзаголовок) отделяются от текста двумя интервалами.

Все страницы курсовой работы, включая иллюстрации и приложения, нумеруются по порядку от титульного листа до последней страницы без пропусков и повторений. Первой страницей является титульный лист, оформленный в соответствующем порядке (см. приложение 1), номер страницы на нем не ставится. На последующих страницах порядковый номер печатается в середине нижнего края страницы или в правом нижним углу.

За титульным листом следует страница с указанием содержания (оглавления) работы в соответствии с ее планом и рубрикацией в тексте.

Иллюстрации (кроме таблиц) обозначаются словом «Рисунок» и нумеруются последовательно арабскими цифрами в пределах раздела (главы).

Номер рисунка и его наименование размещают ниже самого рисунка, подрисуночная подпись выравнивается по центру строки.

Если в работе приведена одна иллюстрация, то ее не номеруют и слово «Рисунок» не пишут.

Таблицы нумеруют последовательно арабскими цифрами в пределах раздела (главы). Каждой таблице предшествует заголовок таблицы, который помещается перед таблицей с выравниванием по правому краю текста. Заголовок начинается со слова «Таблица» с указанием номера этой таблицы, состоящего из номера раздела и порядкового номера таблицы, далее следует текстовая часть заголовка, например:

Таблица 2.3

Точка в конце заголовка таблицы и подрисуночной подписи не ставится. Таблицу размещают после первого упоминания о ней в тексте таким образом, чтобы читать ее можно было без поворота работы или с поворотом по часовой стрелке. Ссылка на таблицу по ходу текста выполняется так: в табл. 2.3 приводятся данные о…, при повторной ссылке – см. табл. 2.3.

Примечания к таблицам, иллюстрациям или пунктам и подпунктам текста размещают непосредственно после пункта, подпункта, таблицы, иллюстрации, к которым они относятся, и печатают с прописной буквы с абзацного отступа. Слово «Примечание» следует печатать с абзацного отступа жирным шрифтом.

Ссылки на разделы, подразделы, пункты, подпункты, иллюстрации, таблицы, формулы, уравнения, перечисления, приложения, на литературные источники следуют указывать порядковым номером, например: «… в разделе 4», «… по пункту 3.3.4», «… в подпункте 2.3.41, перечисление 3», «…по формуле (3)», «… в уравнении (2)»,-«… на рисунке 8», «… в приложении 6», «… в работе [2]».

Если в работе одна иллюстрация, таблица, формула, уравнение, или приложение следует при ссылках писать «на рисунке», «в таблице», «по формуле», «в уравнении», «в приложении».

Пояснение значений символов и числовых коэффициентов следует приводить непосредственно под формулой в той же последовательности, в которой даны в формуле. Значение каждого символа и числового коэффициента

следует давать с новой строки. Первую строку пояснения начинают со слова «где» без двоеточия.

Формулы в работе следует нумеровать порядковой нумерацией в пределах всей работы арабскими цифрами в круглых скобках в крайнем правом положении на строке. Если в работе только одна формула или уравнение, их не нумеруют.

7 Литература

1. Царев, Р. Ю. Программирование на языке Си [Электронный ресурс] : учеб. пособие / Р. Ю. Царев. – Красноярск : Сиб. федер. ун-т, 2014. – 108 с - Режим доступа: http://znanium.com/bookread2.php?book=510946

2. Программирование на языке Си/А.В.Кузин, Е.В.Чумакова - М.: Форум, НИЦ ИНФРА-М, 2015. - 144 с. - Режим доступа: http://znanium.com/bookread2.php?book=505194

3. Язык Си: кратко и ясно: Учебное пособие / Д.В. Парфенов. - М.: Альфа-М: НИЦ ИНФРА-М, 2014. - 320 с. - Режим доступа: https://znanium.com/read?id=356040

4. Программирование графики на C++. Теория и примеры : учеб. пособие / В.И. Корнеев, Л.Г. Гагарина, М.В. Корнеева. — М. : ИД «ФОРУМ» : ИНФРА-М, 2017. — 517 с. - Режим доступа: http://znanium.com/bookread2.php?book=562914

5. Программирование на языке C++: Учебное пособие / Т.И. Немцова, С.Ю. Голова, А.И. Терентьев; Под ред. Л.Г. Гагариной. - М.: ИД ФОРУМ: ИНФРА-М, 2012. - 512 с. - Режим доступа: http://znanium.com/bookread2.php?book=244875

6. Ефимова Ю.В. Программирование на языке высокого уровня: Практикум. - Казань: Изд-во Казан. гос. техн. ун-та, 2012. - 32 с.

8 Пример отчета по курсовой работе

Министерство науки и высшего образования Российской Федерации
федеральное государственное бюджетное образовательное учреждение высшего образования «Казанский национальный исследовательский технический университет им. А.Н. Туполева-КАИ»

Чистопольский филиал «Восток»
Кафедра компьютерные и телекоммуникационные системы

09.03.01 Информатика и вычислительная техника

КУРСОВАЯ РАБОТА

по дисциплине: Программирование и основы алгоритмизации

Обучающийся 21102 _____ Иванов И.И.
(номер группы) (подпись, дата) (Ф.И.О.)

Руководитель доц., к.п.н. Ефимова Ю.В.
(должность) (Ф.И.О.)

Курсовая работа зачтена с оценкой _____

_____
(подпись, дата)

Чистополь 2023 г

СОДЕРЖАНИЕ

1. Задание

Дан файл, содержащий сведения о пациентах клиники. Каждая строка файла содержит запись об одном пациенте.

Структура записи файла:

- фамилия и инициалы пациента;
- год рождения;
- дата поступления;
- диагноз.

Написать программу, которая

- выводит список пациентов, поступивших в заданный день;
- выводит средний возраст пациентов;
- записывает в новый файл список пациентов с заданным диагнозом.

2. Описание применения

2.1. Запуск программы

Запуск программы (файл patient.c) можно выполнить из среды Turbo C 2.0 (или Borland C++ 3.1), либо из командной строки MS DOS, введя patient.exe .

2.2. Входные данные

Входные данные программы находятся в файле “patient.txt”. Число строк в файле - произвольное. Каждая строка содержит фамилию с инициалами (25 символов), год рождения (5 символов), дату поступления (9 символов) и диагноз (до 40 символов).

Пример строки файла:

Анисимов Д.Г. 1961 12.05.04 инфаркт миокарда

| 25 | 5 | 9 | <= 40 |

Пример входного файла приведен в приложении 2.

По запросу программы с клавиатуры необходимо ввести номер пункта меню (см. раздел 2.3), дату поступления пациентов в виде “дд.мм.гг”, например, 05.09.04, имя выходного файла и диагноз.

2.3. Выходные данные

Программа выводит на экран меню:

==========

Выберите номер пункта меню:

1 - вывод списка пациентов, поступивших в заданный день

2 - определение среднего возраста пациентов

3 - запись в новый файл списка пациентов с заданным диагнозом

4 - выход

При выборе пункта 1 на экран выводится сообщение:

Введите дату поступления в виде: дд.мм.гг

После ввода даты, например 12.05.04, выводится результат в виде:

№ Фамилия И.О. Год рожд. Диагноз

---

1. Анисимов Д.Г. 1961 пневмония
2. Хайрутдинов И.С. 1945 инфаркт миокарда
3. Галиева А.И. 1960 инсульт
4. Могилевский С.П. 1943 стенокардия

Если ввести дату, которой нет в файле, выводится сообщение:

Нет пациентов, поступивших в заданный день

Примеры выходных данных для остальных пунктов меню см. в приложении 3.

2.4. Сообщения программы

Ниже приводится перечень возможных сообщений программы:

1. Файл patient.txt не найден;
2. Выберите номер пункта меню:;
3. Нужно вводить номер пункта от 1 до 4;
4. Введите дату поступления в виде: дд.мм.гг;
5. № Фамилия И.О. Год рожд. Диагноз;
6. Нет пациентов, поступивших в заданный день;
7. Средний возраст пациентов:;
8. Файл patient.txt пустой;
9. Введите имя выходного файла;
10. Укажите диагноз;
11. Нет пациентов с заданным диагнозом;
12. Запись в файл завершена;
13. Для продолжения нажмите любую клавишу.

3. Описание программы

3.1. Метод решения задачи

Задачу можно разбить на три отдельные подзадачи:

1. Поиск в файле записей, в которых дата поступления пациента в клинику совпадает с заданной датой, и вывод списка таких пациентов.

2. Последовательный просмотр записей файла, суммирование возрастов всех пациентов и определение количества всех пациентов, чтобы определить средний возраст.

3. Поиск во входном файле записей, в которых диагноз совпадает с заданным диагнозом, и вывод их в выходной файл.

Каждая подзадача решается методом линейного поиска (последовательного просмотра элементов таблицы).

Чтобы пользователь мог выбирать, какие подзадачи решать и в каком порядке, программа выводит на экран меню (см. раздел 2.3).

3.2. Структура программы

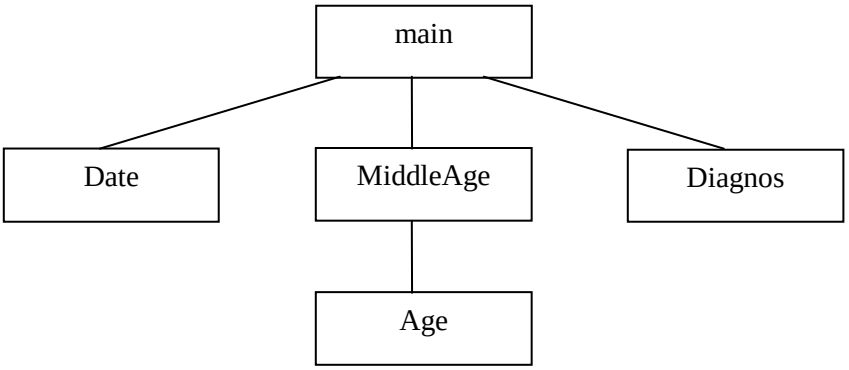

Рисунок 1 – Функциональная структура программы

Программа состоит из пяти функций: главной функции main и четырех подпрограмм. Date – вывод пациентов, поступивших в заданный день.

MiddleAge - определение среднего возраста пациентов.

Diagnos - запись в новый файл списка пациентов с заданным диагнозом.

Age - определение возраста пациента по году рождения.

3.3. Описание функций

3.3.1. main – главная функция

Заголовок функции:

int main ()

Значение функции:

0 – в случае успешного завершения;

1 – если входной файл не найден.

Рабочие данные:

f – указатель на структуру с информацией о входном файле;

n – номер пункта меню.

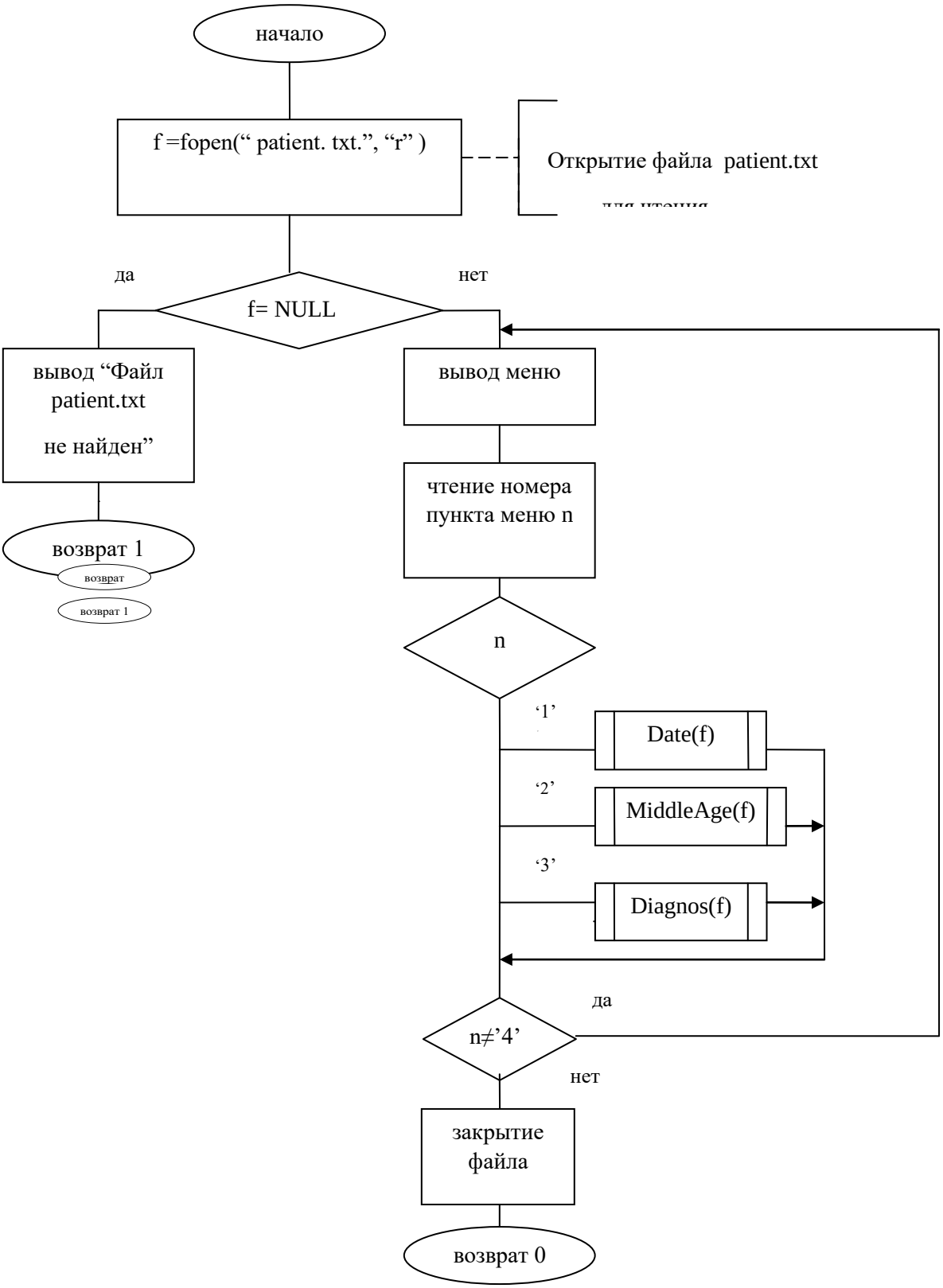

Рисунок 2 – Алгоритм работы функции main()

3.3.2. Date – вывод пациентов, поступивших в заданный день

Заголовок функции: void Date (FILE *f) Структура записи файла: fio - фамилия и инициалы пациента; gr - год рождения; diag – диагноз; data - дата поступления; Входные данные: f - ссылка на входной файл

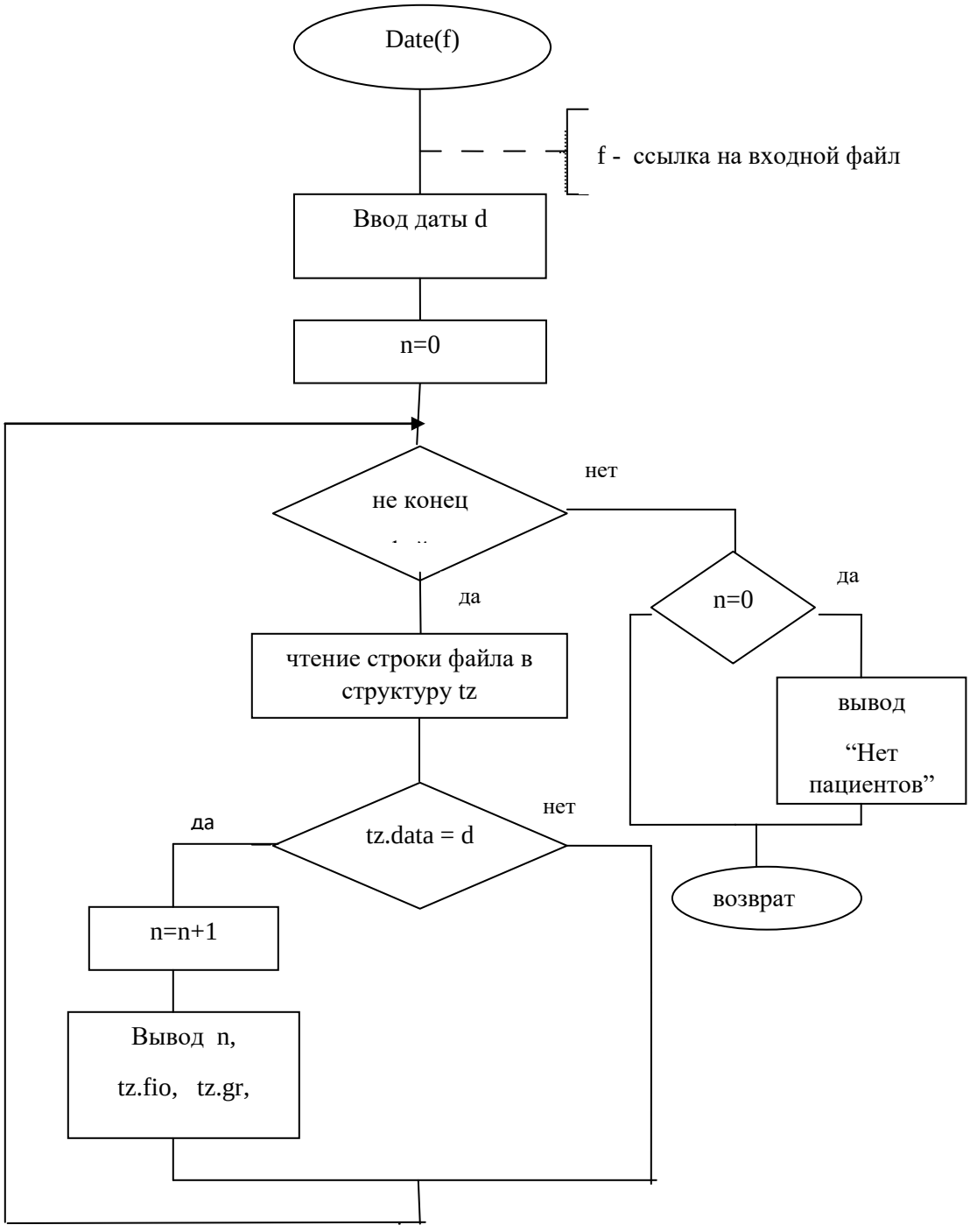

Рисунок 3 – Алгоритм работы функции вывода списка пациентов, поступивших в заданный день

3.3.3. MiddleAge - определение среднего возраста пациентов

Заголовок функции:

void MiddleAge(FILE *f)

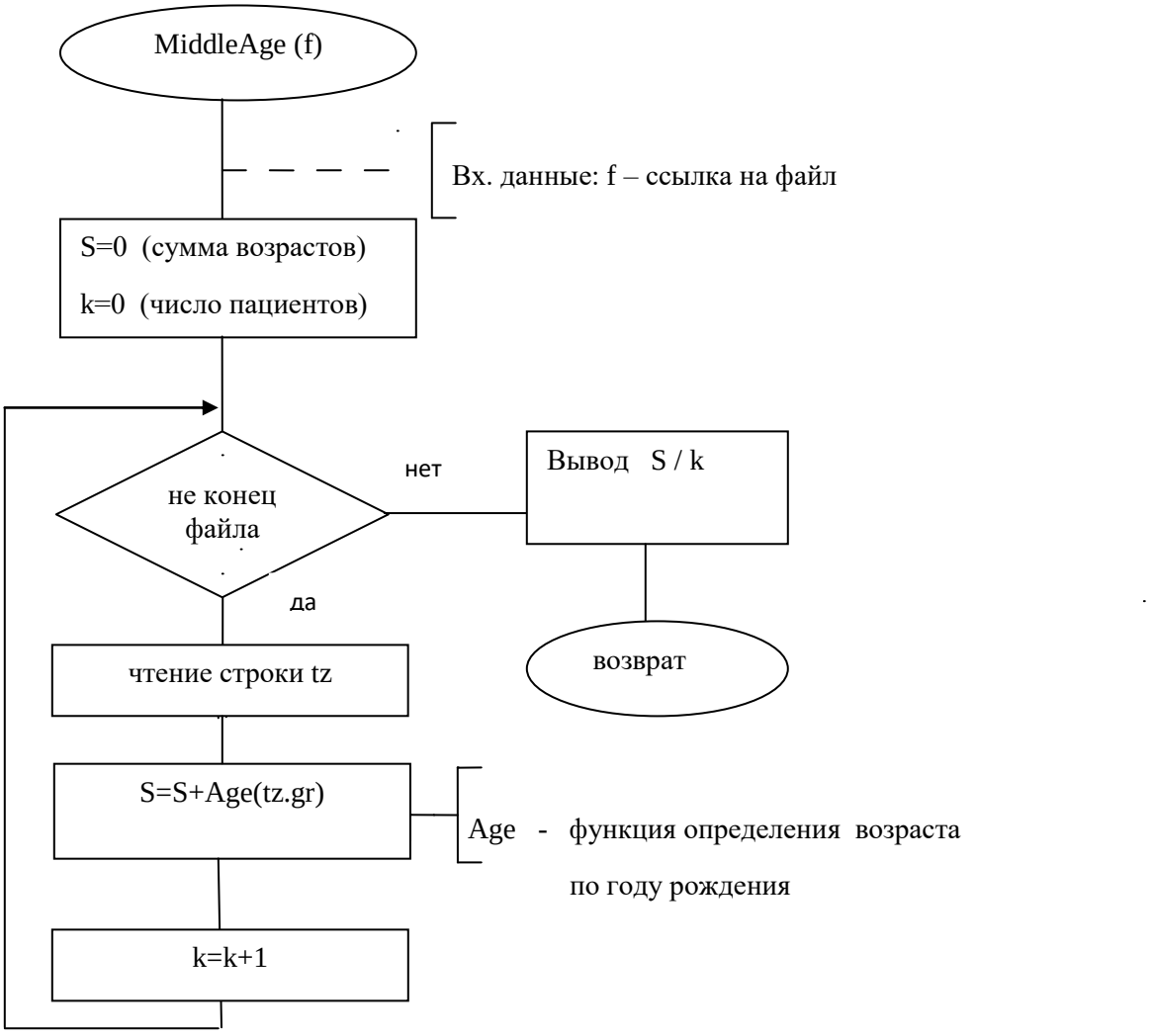

Рисунок 4 – Алгоритм работы функции определения среднего возраста пациентов

3.3.4. Diagnos - запись в новый файл списка пациентов с заданным диагнозом

Заголовок функции: void Diagnos (FILE *fin)

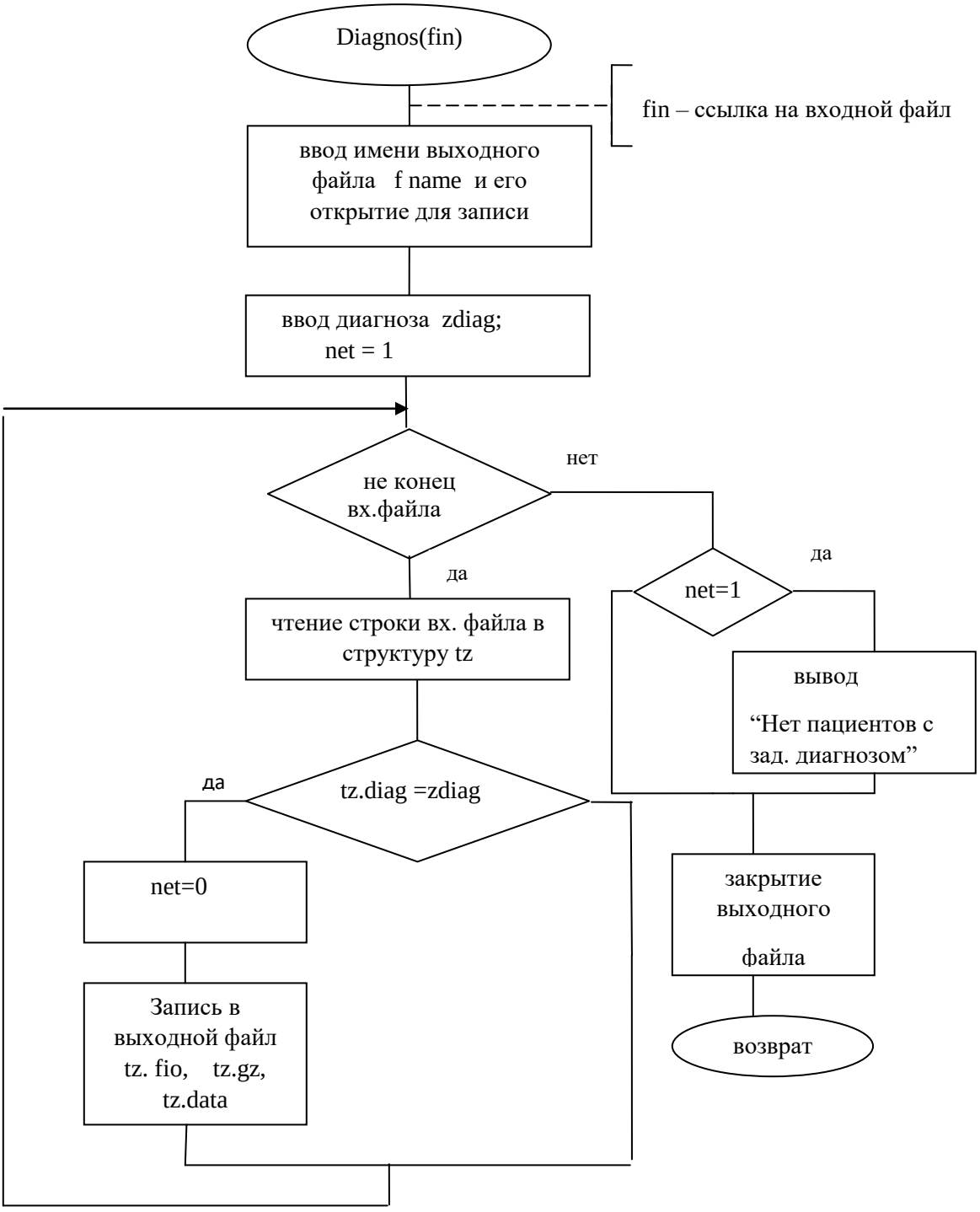

Рисунок 5 – Алгоритм работы функции записи в новый файл списка пациентов с заданным диагнозом

3.3.5. Age - определение возраста по году рождения

Заголовок функции:

int Age (char gr[DL_GR])

Входные данные:

gr - год рождения.

Значение функции: возраст.

Возраст пациента вычисляется как разность между текущим годом и годом рождения пациента. Для определения текущего года используются библиотечные функции, определенные в файле time.h . Функция time() позволяет получить текущую дату и время. Она возвращает ее в виде числа секунд, прошедших с полуночи 1 января 1970 года. Функция localtime() преобразует эту величину в стандартную структуру типа tm, который определен в том же файле time.h, и возвращает адрес этой структуры. Поле tm_year этой структуры содержит число лет, прошедших с 1900 года.

Текст функции:

```
int  Age (char gr[DL_GR])
{
int igr = atoi(gr);   /* преобразование года рождения в тип int */
long timer; /* число секунд, прошедших с начала 1970 года до текущего момента */
int year;       /* текущий год */
struct tm *t; /* указатель на структуру, содержащую текущую дату и время */
timer = time (NULL);/*получение числа секунд, прошедших с начала 1970 года*/
t = localtime(&timer);/* преобразование в структуру типа tm */
year = 1900+ t->tm_year; /* определение тек. года */
return year - igr;
}
```

Литература

1. Ефимова Ю.В. Программирование на языке высокого уровня: Практикум. - Казань: Изд-во Казан. гос. техн. ун-та, 2012. - 32 с.

2. Царев, Р. Ю. Программирование на языке Си [Электронный ресурс] : учеб. пособие / Р. Ю. Царев. – Красноярск : Сиб. федер. ун-т, 2014. – 108 с - Режим доступа: http://znanium.com/bookread2.php?book=510946

3. Программирование на языке Си/А.В.Кузин, Е.В.Чумакова - М.: Форум, НИЦ ИНФРА-М, 2015. - 144 с. - Режим доступа: http://znanium.com/bookread2.php?book=505194

Дополнительная литература:

4. Программирование на языке С++: Учебное пособие / Т.И. Немцова, С.Ю. Голова, А.И. Терентьев; Под ред. Л.Г. Гагариной. - М.: ИД ФОРУМ: ИНФРА-М, 2012. - 512 с. - Режим доступа: http://znanium.com/bookread2.php?book=244875

5. Язык Си: кратко и ясно: Учебное пособие / Д.В. Парфенов. - М.: Альфа-М: НИЦ ИНФРА-М, 2014. - 320 с. - Режим доступа: http://znanium.com/bookread2.php?book= 459254

6. Программирование графики на С++. Теория и примеры : учеб. пособие / В.И. Корнеев, Л.Г. Гагарина, М.В. Корнеева. — М. : ИД «ФОРУМ» : ИНФРА-М, 2017. — 517 с. - Режим доступа: http://znanium.com/bookread2.php?book=562914

Приложение 1. Текст программы

```
#include <stdio.h>
#include <string.h>
#include <conio.h>
#include <stdlib.h>
#include <time.h>

#define  DL_FIO 25        /* длина поля фамилии в файле */
#define  DL_GR  5         /* длина поля года рождения */
#define  DL_DATA          9          /* длина поля даты поступления */
#define  DL_DIAG          40         /* макс. длина диагноза */

struct pacient                /* структура записи входного файла */
{  char  fio[DL_FIO];         /* фамилия и инициалы пациента */
   char  gr [DL_GR];          /* год рождения */
   char  data[DL_DATA];  /* дата поступления */
   char  diag[DL_DIAG+2]; /* диагноз + '\n' + '\0'  */
};

  /* прототипы функций */
void Date(FILE *f);
void MiddleAge(FILE *f);
void Diagnos(FILE *fin);
int  Age (char gr[DL_GR]);

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

int main()
{
            FILE *f;  /* ссылка на входной файл */
            char n;   /* номер пункта меню */

            clrscr();
            f= fopen("patient.txt", "r");
            if (f==NULL)
            {  puts ("Файл  patient.txt не найден");
         getch();
              return 1;
            }
            do
            {  puts ("\n=========================================================");
              puts ("Выберите номер пункта меню:");
              puts ("1 - вывод списка пациентов, поступивших в заданный день");
              puts ("2 - определение среднего возраста пациентов");
              puts ("3 - запись в новый файл списка пациентов с заданным диагнозом");
              puts ("4 - выход");
              puts(("---------------------------------------------------");
              n = getche();
              switch(n)
              {
                        case '1': Date(f); break;
                        case '2': MiddleAge(f); break;
                        case '3': Diagnos(f); break;
                        case '4': break;
```

```
                default: puts("\nНужно вводить номер пункта от 1 до 4");
            }
            if (n!='4')
            {  puts("\nДля продолжения нажмите любую клавишу");
               getch();
            }
          }
          while (n != '4');
          fclose(f);
          return 0;
}
 /*---------------------------------------*/
 /*   Функция вывода списка пациентов,   */
 /*     поступивших в заданный день      */
 /*---------------------------------------*/

void Date (FILE *f)
{
          char zdata[DL_DATA]; /* заданная дата поступления */
          int n=0;               /* порядковый номер пациента в выводимом списке */
          struct pacient tz;   /* текущая запись файла */

          puts ("\nВведите дату поступления в виде: дд.мм.гг");
          gets  (zdata);
          rewind(f);
          while (fgets(&tz,sizeof(struct pacient),f) != NULL)
                    if (strncmp(tz.data,zdata,DL_DATA-1) == 0 )
                    {  if (n==0)
                       { puts("№   Фамилия И.О.  Год рожд. Диагноз");
                         puts("-------------------------------------------------");
                       }
                       tz.gr[DL_GR - 1] = '\0';
                       printf ("%d. %s %s", ++n, tz.fio, tz.diag);
                    }
          if (n==0)  puts ("\nНет пациентов, поступивших в заданный день");
}

 /*-------------------------------------------------*/
 /* функция определения среднего возраста пациентов */
 /*-------------------------------------------------*/

void MiddleAge(FILE *f)
{
          struct pacient tz;   /* текущая запись файла */
          float  s = 0;                  /* сумма возрастов всех пациентов */
          int k = 0;                              /* количество пациентов в файле */

          rewind (f);
          while (fgets(&tz,sizeof(struct pacient),f) != NULL)
          {  s += Age(tz.gr);
             k++;
          }
          if (k) printf ("\nСредний возраст пациентов:%.1f\n",s/k);
          else puts ("\nФайл patient.txt пустой");
}

 /*-----------------------------------------------*/
 /* функция определения возраста по году рождения */
 /*-----------------------------------------------*/
```

```
int  Age (char gr[DL_GR])
{
        int igr = atoi(gr); /* год рождения типа int */
        long timer;                     /* число секунд, прошедших с начала 1970 года
                                        до текущего момента */
        int year;               /* текущий год */
        struct tm *t;           /* указатель на структуру, содержащую текущую
                        дату и время */

        timer = time (NULL);  /* получение числа секунд, прошедших с начала 1970
                        года */
        t = localtime(&timer);/* преобразование в структуру типа tm */
        year = 1900+ t->tm_year; /* определение тек. года */
        return year - igr;
}

 /*-------------------------------------------------*/
 /*  функция записи в новый файл списка пациентов  */
 /*          с заданным диагнозом          */
 /*-------------------------------------------------*/

void Diagnos(FILE *fin)
{
        char  fname[13];        /* имя выходного файла */
        FILE *fout;                     /* ссылка на выходной файл */
        char zdiag[DL_DIAG+1]; /* заданный диагноз */
        int net=1;                      /* признак того, что нет пациентов с заданным
                                        диагнозом */
        struct pacient tz;  /* текущая запись файла */

        puts ("\nВведите имя выходного файла");
        gets  (fname);
        fout = fopen(fname,"w");
        puts ("Укажите диагноз");
        gets (zdiag);
        rewind(fin);
        while (fgets(&tz,sizeof(struct pacient),fin))
                if (strstr(tz.diag,zdiag))
                {  net=0;
                  tz.data[DL_DATA-1] = '\n';
                  fwrite (&tz,sizeof(struct pacient) - DL_DIAG - 2,1,fout);
                }
    fclose (fout);
        if (net) { puts ("Нет пациентов с заданным диагнозом");
          unlink(fname);  /* удаление созданного файла */
         }
        else puts("Запись в файл завершена");

}
```

Приложение 2. Пример входного файла patient.txt

Анисимов Д.Г. 1961 12.05.04 пневмония
Хайрутдинов И.С. 1945 12.05.04 инфаркт миокарда
Петрова Т.В. 1933 13.05.04 стенокардия
Аксенов-Зварчук И.А. 1955 13.05.04 пневмония
Галиева А.И. 1960 12.05.04 инсульт
Сидоров. П.Е. 1956 14.05.04 язва желудка
Хасанов И.Р. 1974 14.05.04 левосторонняя пневмония
Харламов Ф.А. 1957 13.05.04 цирроз печени
Иванов И.С. 1949 14.05.04 инфаркт миокарда
Могилевский С.П. 1943 12.05.04 стенокардия

Приложение 3. Результаты тестирования программы

Тест 1. Входного файла нет в текущем каталоге.
Результат:
Файл patient.txt не найден

В следующих тестах используется файл из приложения 2.

Тест 2 .

```
============================================================
Выберите номер пункта меню:
1 - вывод списка пациентов, поступивших в заданный день
2 - определение среднего возраста пациентов
3 - запись в новый файл списка пациентов с заданным диагнозом
4 - выход
------------------------------------------------------------------------------
1
Введите дату поступления в виде: дд.мм.гг
12.05.04

№  Фамилия И.О.       Год рожд.  Диагноз
------------------------------------------------------------------------
1.  Анисимов Д.Г.              1961  пневмония
2.  Хайрутдинов И.С.  1945  инфаркт миокарда
3.  Галиева А.И.               1960  инсульт
4.  Могилевский С.П.  1943  стенокардия

Для продолжения нажмите любую клавишу
```

Тест 3 .

```
============================================================
Выберите номер пункта меню:
1 - вывод списка пациентов, поступивших в заданный день
2 - определение среднего возраста пациентов
3 - запись в новый файл списка пациентов с заданным диагнозом
4 - выход
------------------------------------------------------------------------------
1
Введите дату поступления в виде: дд.мм.гг
01.05.04
```

Нет пациентов, поступивших в заданный день

Для продолжения нажмите любую клавишу
Тест 4 .

==========
Выберите номер пункта меню:
1 - вывод списка пациентов, поступивших в заданный день
2 - определение среднего возраста пациентов
3 - запись в новый файл списка пациентов с заданным диагнозом
4 - выход

2
Средний возраст пациентов: 50.7

Для продолжения нажмите любую клавишу

Тест 5 .

==========
Выберите номер пункта меню:
1 - вывод списка пациентов, поступивших в заданный день
2 - определение среднего возраста пациентов
3 - запись в новый файл списка пациентов с заданным диагнозом
4 - выход

3
Введите имя выходного файла
pnevmon.txt
Укажите диагноз
пневмония
Запись в файл завершена

Для продолжения нажмите любую клавишу

Содержимое файла pnevmon.txt:

Анисимов Д.Г. 1961 12.05.04
Аксенов-Зварчук И.А. 1955 13.05.04
Хасанов И.Р. 1974 14.05.04

Тест 6 .

==========
Выберите номер пункта меню:
1 - вывод списка пациентов, поступивших в заданный день
2 - определение среднего возраста пациентов
3 - запись в новый файл списка пациентов с заданным диагнозом
4 - выход

3
Введите имя выходного файла
orvi.txt
Укажите диагноз
ОРВИ
Нет пациентов с заданным диагнозом

Для продолжения нажмите любую клавишу
Тест 7 . Входной файл пустой

==========
Выберите номер пункта меню:
1 - вывод списка пациентов, поступивших в заданный день
2 - определение среднего возраста пациентов
3 - запись в новый файл списка пациентов с заданным диагнозом
4 - выход

2
Файл patient.txt пустой

Для продолжения нажмите любую клавишу

Тест 8 . Неверно выбран номер пункта меню

==========
Выберите номер пункта меню:
1 - вывод списка пациентов, поступивших в заданный день
2 - определение среднего возраста пациентов
3 - запись в новый файл списка пациентов с заданным диагнозом
4 - выход

6
Нужно вводить номер пункта от 1 до 4

Для продолжения нажмите любую клавишу

==========
Выберите номер пункта меню:
1 - вывод списка пациентов, поступивших в заданный день
2 - определение среднего возраста пациентов
3 - запись в новый файл списка пациентов с заданным диагнозом
4 - выход

4