ФЕДЕРАЛЬНОЕ АГЕНТСТВО СВЯЗИ

БУРЯТСКИЙ ФИЛИАЛ ГОУ ВПО «СибГУТИ»

**КУРСОВАЯ РАБОТА**

**по дисциплине: Программирование на языке высокого уровня**

выполнил: ст. г. И-101 Дондуков А.С.

№ вариата: 14

проверила: Эрдынеева Л.И

Улан-Удэ

2011 г.

ФЕДЕРАЛЬНОЕ АГЕНТСТВО СВЯЗИ

БУРЯТСКИЙ ФИЛИАЛ ГОУ ВПО «СибГУТИ»

Кафедра «Информатики и вычислительной техники»

Рассмотрено

На кафедре «ИВТ»

Зав. кафедры

Чимитова Е.Г. /\_\_\_\_\_\_\_\_/

«\_\_\_\_\_»\_\_\_\_\_\_\_200\_г.

**ЗАДАНИЕ НА КУРСОВУЮ РАБОТУ**

по дисциплине: Программирование на языке высокого уровня.

Студенту \_гр. И-101\_.\_Дондукову Арсалану Сергеевичу

**Тема задания и исходные данные**

Разработать программу модульной структуры для создания и обработки внешнего файла в соответствии с вариантом задания. Структура данных по варианту № \_14\_

Программа должна выполнять в общем три задачи: 1. Создание файла. 2. Чтение данных из файла. 3. Вывод данных на экран дисплея. 4. Обработка данных по заданию Б.

Все полученные результаты отобразить на экране.

**Пояснительная записка**

1. Введение
2. Теоретическая часть
3. Проектная часть\_
4. Тестирование и отладка программы
5. Заключение
6. Список, использованных источников

**Проектная часть**

1. \_\_Структура основной программы. Описание основной программы.\_\_\_\_\_\_\_
2. \_\_Структура модуля работы с файлами, процедуры и функции содержащиеся в модуле (описание блок-схемы и текста подпрограммы)\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Приложение**

1. \_Блок- схема основного модуля, и подпрограмм\_
2. \_Листинг программы
3. Тестовые результаты\_

**Дата выдачи** «5» мая 2011 г.

**Срок окончания** «\_21\_» июня 2010 г.

Руководитель курсовой работы \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/Эрдынеева Л.И./\_

Задание принял к исполнению\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Содержание**

Введение…………………………………………………………………………..…..5

Теоретическая часть…………………………………………………………………...5

1. Структурированные типы данных……………………………………………..….6

2. Способы описание функций…………………………………………………….....9

3. Создание и обработка модулей…………………………………………………....12

4. Проектирование программ………………………………………………………...13

5. Создание графического интерфейса……………………………………………....14

6. Постановка задачи……………………………………………………………….…15

7.1 Исходные данные……………………………………………………………....16

7.2 Выходные данные……………………………………………………………...17

8.Таблица значения переменных………………………………………………….....18

9. Основная функция void main()……………………………………………………..19

9.1 Блок схема………………………………………………………………………20

9.2 Текст программы……………………………………………………………….21

10. Тестирование программы………………………………………………………....30

Заключение…………………………………………………………………….…..35

Список использованной литературы……………………………………….……36

Введение

Язык программирования С++ царит в компьютерном мире. Есть масса причин для узучения этого языка – это продвижение по служебной лестнице, предназначен для разработки высокопроизводительного программного обеспечения и получение высокооплачиваемой работы. С другой стороны, изучение С++ может превратиться в настоящую каторгу, так как на самом деле это язык с множеством мудреных правил. Обилие инструментов для разработки программ на языке С++ с первого взгляда могут перепутать начинающегося разработчика.

Однако в данном курсовом проекте сделан упор не на сложность, а на разработке программно-модульной структуры, которая должна в соответствии с вариантом задании выполнять общие действия, а именно создавать файл, читать данный файл, выводить данные на дисплей и обрабатывать данные в соответствии с вариантом задания.

Программа должна не только правильно решать задачу, но и иметь современный интерфейс, быть высоконадежной, дружественной по отношению к пользователю. Только такие программы могут выдерживать конкуренцию на мировом рынке программных продуктов.

Я хочу подчеркнуть, что С++ лишь стартовая площадка. С++ - это большой по объему средств и не самый простой язык программировая. Необходимым условием успешного программировая на С++ является не только ключевых слов, операторов и синтаксиса, определяющего возможности языка, но и библиотек классов и функций, которые существенно помогают в разработке программ. И хотя некоторые элементы библиотек рассматриваются в книгах, все же большинство из них не нашло здесь своего отражения. Чтобы стать первоклассным программистом на C++, необходимо в совершенстве изучить и С++-библиотеки. Знания, полученные при изучении, позволят вам освоить не только библиотеки, но и все остальные аспекты C++.

Для составления работающих программ достаточно иметь интуитивные представления об алгоритмах и устройстве компьютера. Часто бывает достаточно ознакомиться с несколькими работающими программами или даже с фрагментами таких программ, чтобы, в буквальном смысле используя образцы, успешно описывать собственные алгоритмы. Однако грамотная речь невозможна без знания языка, а профессиональное программирование требует глубоких знаний в области языков программирования. В ней на основе базовых языковых конструкций и элементарных понятий описываются всё более сложные элементы языка и связанные с ними концепции:

- алфавит и идентификаторы

Изм.

Лист

№ докум.

Подпись

Дата

Лист

5

- элементарные типы, объявления и пределения

- операции, выражения и операторы

- функции и их характеристики

- производные типы и средства их построения

- функции-члены, конструкторы и операторные функции

- механизмы наследования, инкапсуляции и полиморфизма

- шаблоны

- потоки ввода-вывода

- средства обработки исключительных ситуаций.

Информация о контроле типов, типизации и прочих важных характеристиках языков программирования; сведения об алгоритмах преобразования, дополнительном коде и о преобразовании дробной части вещественного числа. Эти приложения включены в книгу с единственной целью: чтобы при обсуждении того, что делается, не возникало вопросов по поводу того, как это делается; элементы теории комплексных чисел. Комплексные числа - наш полигон. Многие конструкции языка C++ мы в дальнейшем будем отрабатывать применительно к множеству комплексных чисел; элементарные сведения об устройстве ЭВМ.

# 2. Структурированный типы данных

Структура – это объединение одного или нескольких объектов (перемменых, массивов, указателец, других структур и т.д.) Как и массив, она представляет собой совокупность данных. Отличнем является те, что к ее элементам необходимо обращаться по имени и что различные элементы структуры не обязательно должны принадлежать одному типу. Переменные, из которых состоит структура, называется членами. Члены структуры еще называются элементами или полями.

Когда объявляется структура, то определяется агрегатный тип, а не переменная. Пока не объявите переменную этого типа, то существовать она не будет. Когда объявляется переменная – структура, компилятор автоматически выделяет количество памяти, достаточное, чтобы разместить все ее члены.

Вектор есть совокупность элементов одного типа; struct является совокупностью элементов (практически) произвольных типов. Например:

struct address { // почтовый адрес

char\* name; // имя "Jim Dandy"

long number; // номер дома 61

char\* street; // улица "South Street"

char\* town; // город "New Providence"

char\* state[2]; // штат 'N' 'J'

int zip; // индекс 7974

}

определяет новый тип, названный address (почтовый адрес), состоящий из пунктов, требующихся для того, чтобы послать кому-нибудь корреспонденцию (вообще говоря, address не является достаточным для работы с полным почтовым адресом, но в качестве примера достаточен). Обратите внимание на точку с запятой в конце; это одно из очень немногих мест в C++, где необходимо ставить точку с запятой после фигурной скобки, поэтому люди склонны забывать об этом.

Переменные типа address могут описываться точно также, как другие переменные, а доступ к отдельным членам получается с помощью операции. (точка). Например:

address jd;

jd.name = "Jim Dandy";

jd.number = 61;

Запись, которая использовалась для инициализации векторов, можно применять и к переменным структурных типов. Например:

address jd = {

"Jim Dandy",

61, "South Street",

"New Providence", {'N','J'}, 7974

};

Однако обычно лучше использовать конструктор. Заметьте, что нельзя было бы инициализировать jd.state строкой "NJ". Строки оканчиваются символом '\0', поэтому в "NJ" три символа, то есть на один больше, чем влезет в jd.state.

К структурным объектам часто обращаются посредством указателей используя операцию ->. Например:

void print\_addr(address\* p)

{

Изм.

Лист

№ докум.

Подпись

Дата

Лист

6

cout << p->name << "\n" << p->number << " " << p->street << "\n" << p->town << "\n" << chr(p->state[0]) << chr(p->state[1])

<< " " << p->zip << "\n"; }

Объекты типа структур можно присваивать, передавать как параметры функции и возвращать из функции в качестве результата. Например:

address current;

address set\_current(address next)

{

address prev = current;

current = next;

return prev;

}

Остальные осмысленные операции, такие как сравнение (== и !=) не определены. Однако пользователь может определить эти операции.

Размер объекта структурного типа нельзя вычислить просто как сумму его членов. Причина этого состоит в том, что многие машины требуют, чтобы объекты определенных типов выравнивались в памяти только по некоторым зависящим от архитектуры границам (типичный пример: целое должно быть выравнено по границе слова) или просто гораздо более эффективно обрабатывают такие объекты, если они выравнены в машине. Это приводит к "дырам" в структуре. Например, (на моей машине) sizeof (address) равен 24, а не 22, как можно было ожидать.

Заметьте, что имя типа становится доступным сразу после того, как оно встретилось, а не только после того, как полностью просмотрено все описание. Например:

struct link{

link\* previous;

link\* successor;

}

Новые объекты структурного типа не могут быть описываться, пока все описание не просмотрено, поэтому

struct no\_good {

no\_good member;

};

является ошибочным (компилятор не может установить размер no\_good). Чтобы дать возможность двум (или более) структурным типам ссылаться друг на друга, можно просто описать имя как имя структурного типа. Например:

struct list; // должна быть определена позднее

struct link {

link\* pre;

link\* suc;

link\* member\_of;

};

struct list {

link\* head;

}

Без первого описания list описание link вызвало бы к синтаксическую

ошибку.

# 3. Способы описания функций

Изм.

Лист

№ докум.

Подпись

Дата

Лист

7

Изм.

Лист

№ докум.

Подпись

Дата

Лист

8

Для начало надо разобраться, что такое функция. Программа на языке Си становится большого размера, и появляется необходимость разбивать код на части. Вот такие части на языке Си и называется функциями. Функция может возвращать любые типы значений, а может ничего не возвращать, тогда в качестве возвращаемого значения функция будет тип void. Имя функция может быть любым, но чаще всего имя функция дается от предназначения. Язык С можно назвать структурированным, единицей которого является функция.

Функция - это поименованная часть программы, которая может вызываться из других частей программы столько раз, сколько необходимо. Приведем программу, выдающую степени числа два:

extern float pow ( float, int ); // pow () определена в другом месте

int main ()

{

for ( int i=0; i<10; i++ ) cout << pow ( 2, i ) << '\n';

}

Первая строка является описанием функции. Она задает pow как функцию с параметрами типа float и int, возвращающую значение типа float. Описание функции необходимо для ее вызова, ее определение находится в другом месте.

При вызове функции тип каждого фактического параметра сверяется с типом, указанным в описании функции, точно так же, как если бы инициализировалась переменная описанного типа. Это гарантирует надлежащую проверку и преобразования типов. Например, вызов функции pow(12.3,"abcd") транслятор сочтет ошибочным, поскольку "abcd" является строкой, а не параметром типа int. В вызове pow(2,i) транслятор преобразует целую константу (целое 2) в число с плавающей точкой (float), как того требует функция. Функция pow может быть определена следующим образом:

float pow ( float x, int n )

{

if(n<0)

error ( "ошибка: для pow () задан отрицательный показатель");

switch ( n )

{

case 0: return 1;

case 1: return x;

default: return x \* pow ( x, n-1 );

}

}

Первая часть определения функции задает ее имя, тип возвращаемого значения (если оно есть), а также типы и имена формальных параметров (если они существуют).Значение возвращается из функции с помощью оператора return.

Разные функции обычно имеют разные имена, но функциям, выполняющим сходные операции над объектами разных типов, лучше дать одно имя. Если типы параметров таких функций различны, то транслятор всегда может

разобраться, какую функцию нужно вызывать. Например, можно иметь две функции возведения в степень: одну - для целых чисел, а другую - для чисел с плавающей точкой:

int pow ( int, int );

double pow ( double, double ); //

x = pow ( 2,10 );// вызов pow(int,int)

y = pow ( 2.0, 10.0 );// вызов pow(double,double)

Такое многократное использование имени называется перегрузкой имени функции или просто перегрузкой;

Параметры функции могут передаваться либо "по значению", либо "по ссылке". Рассмотрим определение функции, которая осуществляет взаимообмен

значений двух целых переменных. Если используется стандартный способ передачи параметров по значению, то придется передавать указатели:

void swap ( int \* p, int \* q )

{

int t = \* p;

\* p = \* q;

\* q = t;

}

Унарная операция \* называется косвенностью (или операцией разыменования), она выбирает значение объекта, на который настроен указатель. Функцию можно вызывать следующим образом:

void f ( int i, int j )

{ swap ( & i, & j ); }

Если использовать передачу параметра по ссылке, можно обойтись без явных операций с указателем:

void swap (int&r1,int&r2)

{ int t = r1;

r1 = r2;

r2 = t; }

void g (inti,intj)

Изм.

Лист

№ докум.

Подпись

Дата

Лист

9

{ swap (i,j);

}

Функции реализуют структурное программирование и исполняет все необходимые задания для решение общей задачи. Имеются стандартные функции, занесенные в программные библиотеке Си, как правило эти функции используются часто. Также для решение других часто используемых задач можно написать собственные функции. Для опредения функция в языке Си необходимо указать тип и имя, а также список параметров в круглых скобках.

В общем виде функцию можно представить так:

Тип имя\_функции (список\_параметров)

{

Инструкции

returne (возвращаемое значение функции)

}

# 3.Создание и обработка модулей

Модуль это набор ресурсов (функций, процедур, констант, перемменых, типов )разрабатываемых и хранимых независимо от используемых программ.

Программа на языке Си++ состоит из одного или более модулей. Каждый модуль компилирует отдельно от других. Каждый модуль содержит данные, имеющие общие назначение, и его разработку выполняет обычно один разработчик.

Модуль состоит из двух файлов:

1.Файл с расширением .h содержит функций, глобальные переменные и типы данных. Он обычно называется заголовочный файл.

2.Файл реализации, имеющий расширение ,с и содержащий определения функций и глобальных переменных.

# 4.Проектирование программ

Изм.

Лист

№ докум.

Подпись

Дата

Лист

*10*

Главное условие для создания крупных программ заключается а применение методов проектирования. Большое распространение при написание программ получили следующие три метода: нисходящий, восходящий и специальный.

Спецификация задачи требует особого внимания, поскольку даже небольшие изменения в ней могут повлечь за собой значительные затраты времени на их воплощение. Основным условием для создания крупных программ является применение методов проектирования. Большое распространение в программировании получили следующие три метода: нисходящий, восходящий и специальный.

Нисходящий метод, начинаемый созидательным процессом с программы высокого уровня и спускающейся до подпрограмм низкого уровня. Вначале пишется основная программа, используя средство вызова подпрограмм, а затем, уверены в правильности логического построения основной программы, расписываем подпрограммы, вызывая по необходимости подпрограммы более низкого уровня. Этот процесс не завершается пока программа не будет до конца написана и проверена.

Восходящий метод работает в обратном направление: писать начинают сначала с отдельных специальных, постепенно строится на их основе более сложные конструкции и заканчивается самым верхнем уровнем программы. То есть сначала пишутся подпрограммы нижнего уровня, тщательно тестируются и отлаживаются. Потом вызываются подпрограммы более высокого уровня, которые вызывают подпрограммы нижнего уровня. И так до сих пор пока не будет достигнута программы. Этот метод пригоден при наличии больших библиотек стандартных программ.

Специальный подход не имеет заранее установленного метода. Иногда лучше совмещать два метода. Так как программа должна быть небольшого размера, так чтобы можно было одним взглядом охватить всю её логику.

Язык Си является структурированным языком программирования, он лучше всего сочетается с нисходящим методом проектирования. Так как этот подход позволяет прояснить и сделать программу более понятной т он уменьшит потери времени, обусловленным неудачными или ошибочными начинаниями.

# 5. Создание графического интерфейса

Для оформления графического интерфейса программы используется библиотека «CONIO.H». Она позволяет установить текст в нужном месте, поменять цвет, сделать рамки, и т.д.

Conio.h (от англ. Console input-output – консольный ввод-вывод) - заголовочный файл, используемый в старых компиляторах, работающих в операционных системах MS-DOS, для создания текстового интерфейса пользователя. Тем не менее, он не является частью языка программирования Си, стандартной библиотеки языка Си.

Функции для работы библиотеке Conio.h

int kbhit(void); Определяет, было ли нажатие клавиш клавиатуры/

int getch(void); Считывает символ напрямую из консоли без использования буфера и echo-вывода

Изм.

Лист

№ докум.

Подпись

Дата

Лист

*11*

void window(int left, int top, int right, int bottom); Устанавливает текущее окно консоли по указанным координатам

void clreol(void); очищает текущую строку окна от позиции курсора до конца

void gotoxy(int x, int y); помещает курсор в указанные столбец и строку окна

void textbackground(int newcolor); устанавливает фоновый цвет окна.

void textcolor(int newcolor); Устанавливает указанный цвет вывода текста в окне.

# 6.Постановка задачи

Задача: разработать программу модульной структуры, для создание и обработки внешнего файла и соответствие с вариантом задания.

Программа должна выполнять в общем три задачи:

1.Создание файла.

2.Чтение данных из файла.

3.Вывод данных на экран дисплея.

4.Обработка данных по заданию Б.

Все полученные результаты отобразить на экране.

**Задание**

А. Создать файл содержащий сведения о количестве изделий, собранных сборщиками цеха за неделю. Каждая запись содержит поля – фамилия сборщика, количество изделий, собранных им ежедневно в течение шестидневной недели ( понедельник, вторник, среда, четверг, пятница, суббота) Количество записей –более 25

сборщику просуммировать количество деталей, собранное им за неделю. Определить сборщика, собравшего наибольшее число изделий и день когда он достиг наивысшей производительности труда.

# 

Изм.

Лист

№ докум.

Подпись

Дата

Лист

12

# 7.1 Исходные данные

Даны сведения о количестве изделий, собранных сборщиками цеха за неделю.

Составлена таблица исходных данных.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ФИО | ИЗДЕЛИЯ | | | | | |
| Понедельник | Вторник | Среда | Четверг | Пятница | Суббота |
| 1.Авраченков | 40 | 40 | 40 | 50 | 50 | 50 |
| 2.Агонян | 40 | 40 | 50 | 50 | 50 | 30 |
| 3.Андреев | 40 | 40 | 40 | 40 | 40 | 40 |
| 4.Анюков | 50 | 50 | 50 | 50 | 40 | 40 |
| 5.Большаков | 40 | 30 | 30 | 40 | 40 | 30 |
| 6.Брянцев | 30 | 30 | 40 | 40 | 30 | 40 |
| 7.Быстров | 40 | 40 | 40 | 40 | 40 | 40 |
| 8.Валерьев | 50 | 50 | 50 | 50 | 30 | 30 |
| 9.Васнецов | 20 | 40 | 40 | 50 | 50 | 40 |
| 10.Гагарин | 50 | 50 | 30 | 30 | 40 | 20 |
| 11.Гарын | 40 | 40 | 40 | 40 | 40 | 40 |
| 12.Грибов | 50 | 40 | 50 | 50 | 50 | 50 |
| 13.Григорьев | 40 | 40 | 40 | 50 | 50 | 50 |
| 14.Герасимов | 30 | 20 | 30 | 40 | 40 | 50 |
| 15.Дакур | 40 | 30 | 30 | 50 | 50 | 40 |
| 16.Денисов | 20 | 30 | 40 | 50 | 60 | 40 |
| 17.Дынин | 30 | 30 | 30 | 30 | 30 | 30 |
| 18.Евсеев | 50 | 50 | 50 | 40 | 40 | 30 |
| 19.Рыбин | 40 | 30 | 40 | 40 | 40 | 40 |
| 20.Тимощук | 40 | 50 | 60 | 50 | 30 | 40 |
| 21.Титов | 70 | 50 | 60 | 60 | 50 | 40 |
| 22.Никитов | 20 | 20 | 30 | 40 | 40 | 50 |
| 23.Мандрыкин | 40 | 50 | 50 | 50 | 40 | 30 |
| 24.Чентофальски | 30 | 30 | 30 | 40 | 40 | 50 |
| 25.Перов | 50 | 40 | 30 | 40 | 30 | 30 |

# 7.2 Выходные данные

Изм.

Лист

№ докум.

Подпись

Дата

Лист

13

По каждому сборщику просуммировать количество деталей, собранное им за неделю. Определить сборщика, собравшего наибольшее число изделий и день когда он достиг наивысшей производительности труда.

|  |  |  |
| --- | --- | --- |
| ФИО  Изм.  Лист  № докум.  Подпись  Дата  Лист  14 | Общее количество деталей за неделю | Определение сборщика и наибольших значений |
| 1.Авраченков | 270 |  |
| 2.Агонян | 260 |  |
| 3.Андреев | 240 |  |
| 4.Анюков | 280 |  |
| 5.Большаков | 210 |  |
| 6.Брянцев | 210 |  |
| 7.Быстров | 240 |  |
| 8.Валерьев | 260 |  |
| 9.Васнецов | 240 |  |
| 10.Гагарин | 220 |  |
| 11.Гарын | 240 |  |
| 12.Грибов | 290 |  |
| 13.Григорьев | 270 |  |
| 14.Герасимов | 210 |  |
| 15.Дакур | 240 |  |
| 16.Денисов | 240 |  |
| 17.Дынин | 180 |  |
| 18.Евсеев | 260 |  |
| 19.Рыбин | 230 |  |
| 20.Тимощук | 270 |  |
| 21.Титов | 330 | Среда |
| 22.Никитов | 200 |  |
| 23.Мандрыкин | 260 |  |
| 24.Чентофальски | 220 |  |
| 25.Перов | 220 |  |

# 8. Таблица значения переменных.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | имя поля | название | тип данных | размер (байт) |
| 1 | Фамилия | Фамилия | F | 1 |
| 2 | Понедельник | изделия | izd[0] |  |
| 3 | вторник | изделия | izd[1] |  |
| 4 | среда | изделия | izd[2] |  |
| 5 | четверг | изделия | izd[3] |  |
| 6 | пятница | изделия | izd[4] |  |
| 7 | суббота | изделия | izd[5] |  |

# 9. Основная функция void main()

Изм.

Лист

№ докум.

Подпись

Дата

Лист

*15*

**Использование void**

По мете усложнения нашей программы мы должны разделить ее на небольшие легко управляемые части – функция. Функция – это простой набор операторов, выполняющих определенную задачу. Если наша программа использует функцию, функция выполняет свою задачу и затем возвращает результат программе.

Каждая функция в программе имеет уникальное имя, а каждая программа имеет по крайней мере одну функцию с именем **main**.

Программа использует слово void для указания того, что функция не возвращает значения или не имеет значений, передаваемых в нее. Большинство простых программ на С++, которые будут созданы вами в процессе изучения, не возвращают выходное значение статуса операционной системе. Поэтому мы должны размещать слово void перед main.

Каждая программа на С++ имеет один вход, с которого начинается выполнение программы, - главную программу. В программах С++ оператор **void main(void)** указывает стартовую точку вашей программы.

В Си функция **void main()** – это самая главная функция, в ней программы всегда выполняется первой, а потом при необходимости из нее можно вызвать другие функции.

Моя программа включает в себя 4 подфункций:

Функция ввода информации – int vvod()

Функция чтение данных из файла – void ofile()

Функция вывода информации по номеру телефона – int vd1()

Функция вывода информации по году установки – int vd2()

# 9.1 Блок схема

Изм.

Лист

№ докум.

Подпись

Дата

Лист

16

Изм.

Лист

№ докум.

Подпись

Дата

Лист

*17*

Начало

Вывод меню

Пока не нажата клавиша ESC

Чтение данных из файла

Анализ нажатой клавиши

‘1’ Добавление записи

‘2’ Отображение всех данных

‘3’ Общее количество изделий за неделю

‘4’ наибольшее количество изделий

Конец

Запись данных в файл

Конец

-

+

9.2 Текст программы

#include <iostream.h>

#include <conio.h>

#include <stdio.h>

Изм.

Лист

№ докум.

Подпись

Дата

Лист

*18*

#include <string.h>

const N = 30; //максимальное количество возможных записей

//запись

struct TIzdelie{

char f[30]; //фамилия

unsigned int izd[6];//количество изделий собраных ежидневно

} Iz,Izdelie[N];

char ch; //анализируемый символ

int Recs; //последняя введенная запись

//чтение данных из файл fname, возврщает количество считанных данных

int rData(char \*fname){

FILE \*f;

int i=-1;

if ((f=fopen(fname,"r"))!=NULL){

i=0;

while(!feof(f)){

fscanf(f,"%15s",Izdelie[i].f);

fscanf(f,"%6d",&Izdelie[i].izd[0]);

fscanf(f,"%6d",&Izdelie[i].izd[1]);

fscanf(f,"%6d",&Izdelie[i].izd[2]);

fscanf(f,"%6d",&Izdelie[i].izd[3]);

fscanf(f,"%6d",&Izdelie[i].izd[4]);

fscanf(f,"%6d",&Izdelie[i].izd[5]);

fscanf(f,"\n");

++i;

}

--i;

fclose(f);

}

return i;

}

//запись данных в файл fname

void wData(char \*fname){

FILE \*f;

if (Recs>=0){

f = fopen(fname,"w");

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

fprintf(f,"%15s",Izdelie[i].f);

fprintf(f,"%6d",Izdelie[i].izd[0]);

fprintf(f,"%6d",Izdelie[i].izd[1]);

fprintf(f,"%6d",Izdelie[i].izd[2]);

fprintf(f,"%6d",Izdelie[i].izd[3]);

fprintf(f,"%6d",Izdelie[i].izd[4]);

fprintf(f,"%6d",Izdelie[i].izd[5]);

fprintf(f,"\n");

}

fclose(f);

}

}

//добавление записи

void AddRec(TIzdelie &r){

window(1,1,80,50);

textbackground(1);

clrscr();

textbackground(3);

window(10,20,70,30);

clrscr();

textcolor(0);

if(Recs == N-1){

gotoxy(2,2);

cout<<"Вы ввели максимальное количество записей.";

gotoxy(2,3);

cout<<"Для возврата в основное меню нажмите любую клавишу.";

getch();

} else {

gotoxy(2,2);

cout<<"Фамилия: "; cin>>r.f;

gotoxy(2,3);

cout<<"Количество изделий за понедельник: "; cin>>r.izd[0];

gotoxy(2,4);

cout<<"Количество изделий за вторник: "; cin>>r.izd[1];

gotoxy(2,5);

cout<<"Количество изделий за среду: "; cin>>r.izd[2];

gotoxy(2,6);

cout<<"Количество изделий за четверг: "; cin>>r.izd[3];

gotoxy(2,7);

cout<<"Количество изделий за пятницу: "; cin>>r.izd[4];

gotoxy(2,8);

cout<<"Количество изделий за субботу: "; cin>>r.izd[5];

gotoxy(2,10);

cout<<"Для возврата в основное меню нажмите любую клавишу.";

getch();

++Recs;

}

}

Изм.

Лист

№ докум.

Подпись

Дата

Лист

*19*

//вывод меню

void OutMenu(){

window(1,1,80,50);

textbackground(1);

clrscr();

textbackground(3);

Изм.

Лист

№ докум.

Подпись

Дата

Лист

*20*

window(20,20,60,30);

clrscr();

textcolor(0);

gotoxy(2,2);

cout<<"1 - ввод данных";

gotoxy(2,3);

cout<<"2 - отображение всех данных";

gotoxy(2,4);

cout<<"3 - общее колличество изделий на неделю";

gotoxy(2,5);

cout<<"4 - наибольшее количество изделий";

gotoxy(2,6);

cout<<"ESC - выход из программы";

}

//отображение всех записей

void OutAllData(){

window(1,1,80,50);

textbackground(1);

clrscr();

textbackground(3);

window(10,5,70,45);

clrscr();

textcolor(0);

gotoxy(2,2); cout<<"№";

gotoxy(5,2); cout<<"Фамилия";

gotoxy(25,2); cout<<"Пн.";

gotoxy(30,2); cout<<"Вт.";

gotoxy(35,2); cout<<"Ср.";

gotoxy(40,2); cout<<"Чт.";

gotoxy(45,2); cout<<"Пт.";

gotoxy(50,2); cout<<"Сб.";

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

gotoxy(2,2+2+i); cout<<i+1;

gotoxy(5,2+2+i); cout<<Izdelie[i].f;

gotoxy(25,2+2+i); cout<<Izdelie[i].izd[0];

gotoxy(30,2+2+i); cout<<Izdelie[i].izd[1];

gotoxy(35,2+2+i); cout<<Izdelie[i].izd[2];

gotoxy(40,2+2+i); cout<<Izdelie[i].izd[3];

gotoxy(45,2+2+i); cout<<Izdelie[i].izd[4];

gotoxy(50,2+2+i); cout<<Izdelie[i].izd[5];

}

gotoxy(2,2+2+1+i);

cout<<"Для возврата в основное меню нажмите любую клавишу.";

getch();

}

//Общее количество деталей за неделю

void sumIzdelies(){

window(1,1,80,50);

textbackground(1);

clrscr();

textbackground(3);

Изм.

Лист

№ докум.

Подпись

Дата

Лист

*21*

window(10,5,70,45);

clrscr();

textcolor(0);

gotoxy(2,2); cout<<"№";

gotoxy(5,2); cout<<"Фамилия";

gotoxy(25,2); cout<<"Общее количество деталей за неделю";

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

gotoxy(2,2+2+i); cout<<i+1;

gotoxy(5,2+2+i); cout<<Izdelie[i].f;

int sum=0;

for (int j=0;j<6;j++){

sum+=Izdelie[i].izd[j];

}

gotoxy(25,2+2+i); cout<<sum;

}

gotoxy(2,2+2+1+i);

cout<<"Для возврата в основное меню нажмите любую клавишу.";

getch();

}

//Определение сбощика и наибольших значений

void Sborshik(){

window(1,1,80,50);

textbackground(1);

clrscr();

textbackground(3);

window(10,10,70,40);

clrscr();

textcolor(0);

//наибольшее количество изделий

unsigned int max1=0; unsigned int max2=0;

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

int sum=0;

for (int j=0;j<6;j++){

sum+=Izdelie[i].izd[j];

}

if (sum>max1) max1 = sum;

}

//в какой день наибольшее количество изделий

for(i=0;i<=Recs;i++){

int sum=0;

for (int j=0;j<6;j++){

sum+=Izdelie[i].izd[j];

}

if (sum == max1){

max2 = Izdelie[i].izd[0];

for (int j=1;j<6;j++)

if (Izdelie[i].izd[j]>max2) max2=Izdelie[i].izd[j];

}

Изм.

Лист

№ докум.

Подпись

Дата

Лист

*22*

}

//отображение найденых значений

int c=0;

for(i=0;i<=Recs;i++){

int sum=0;

for (int j=0;j<6;j++){

sum+=Izdelie[i].izd[j];

}

if (sum == max1){

gotoxy(2,2+c);

cout<<"Наиболшее число изделий у: "<<Izdelie[i].f;

gotoxy(32,2+c);

cout<<"День: ";

int d=0;

for (int j=0;j<6;j++)

if (Izdelie[i].izd[j] == max2){

gotoxy(38+d\*4,2+c);

if (j==0){

cout<<"Пн.";

++d;

}else if (j==1){

cout<<"Вт.";

++d;

}else if (j==2){

cout<<"Ср.";

++d;

}else if (j==3){

cout<<"Чт.";

++d;

}else if (j==4){

cout<<"Пт.";

++d;

}else if (j==5){

cout<<"Сб.";

++d;

}

}

++c;

}

}

gotoxy(2,2+1+c);

cout<<"Для возврата в основное меню нажмите любую клавишу.";

getch();

}

//код основной программы

void main(){

clrscr();//очистка экрана

textmode(64);// задание тектового режима 80x50 (ширина х высота)

OutMenu();//вывод основного меню

Recs = rData("izdeliya.dat");//чтение данных из файла если таковой имеется

while((ch=getch())!=0x1B){

switch(ch){

case '1': AddRec(Iz);

Изм.

Лист

№ докум.

Подпись

Дата

Лист

*23*

Izdelie[Recs] = Iz;

break;

case '2': OutAllData();

break;

case '3': sumIzdelies();

break;

case '4': Sborshik();

break;

}

clrscr();

OutMenu();

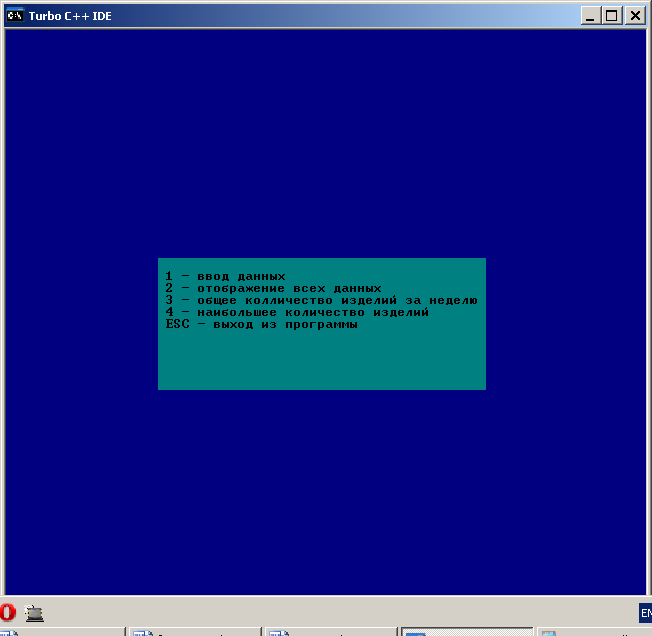
}

wData("izdeliya.dat"); //запись данных в файл

}

# 10. Тестирование программы

**1.Вывод меню.**



Изм.

Лист

№ докум.

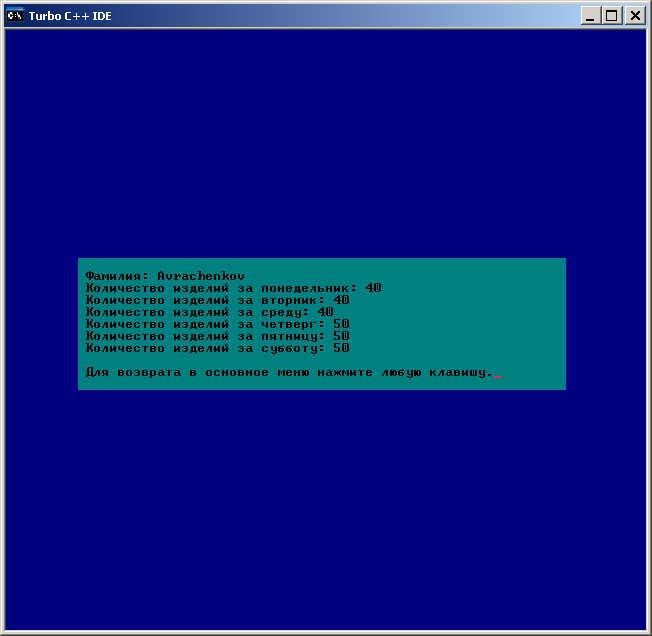
Подпись

Дата

Лист

*24*

**2.Ввод записей.**



**3.Отображение всех записей.**

Изм.

Лист

№ докум.

Подпись

Дата

Лист

*25*

Изм.

Лист

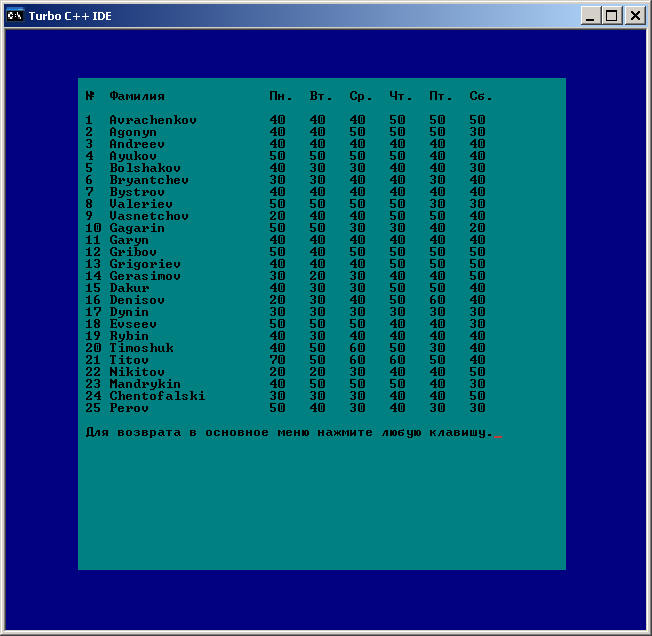
№ докум.

Подпись

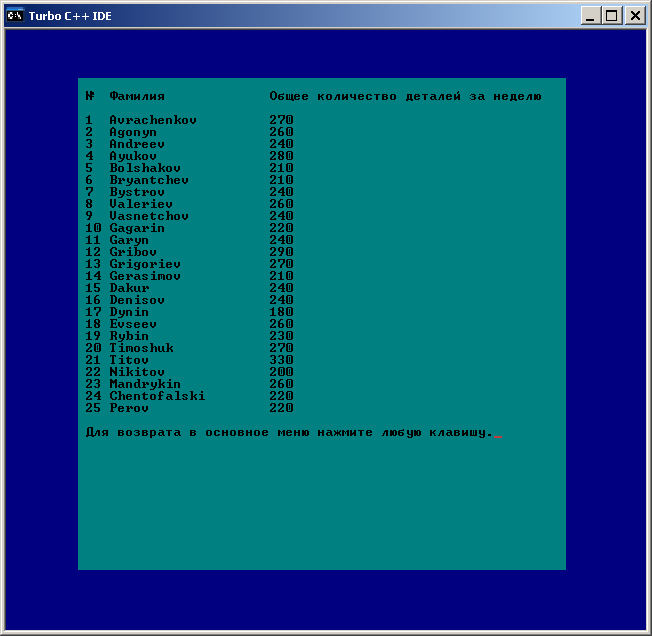
Дата

Лист

*26*



**4.Общее количество деталей за неделю**



**5.Определение сборщика и наибольших значений**

Изм.

Лист

№ докум.

Подпись

Дата

Лист

*27*

Изм.

Лист

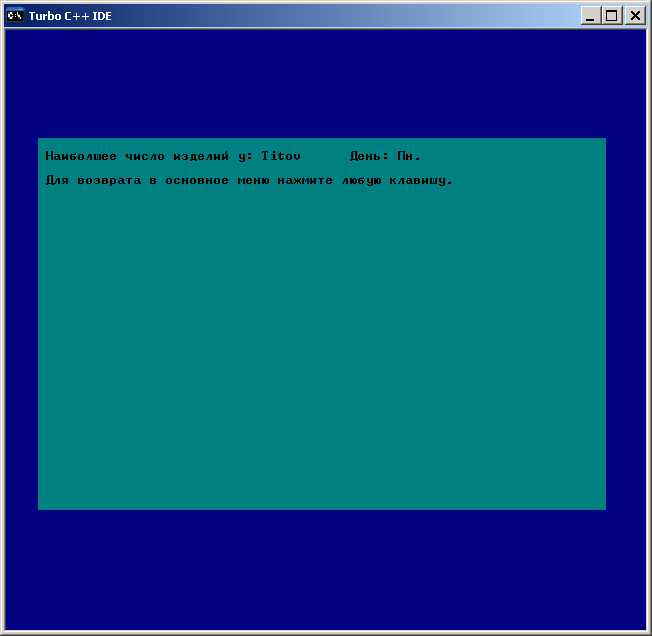
№ докум.

Подпись

Дата

Лист

*28*



Заключение.

Изм.

Лист

№ докум.

Подпись

Дата

Лист

*29*

В результате проделанной работы была разработана программа задач о сборщиках и сборщика собравшего наибольшее число изделий за неделю. Программа состоит из двух модулей, которые обеспечивают создание файла, его считывание, обработку файла по условию, вывод данных на экран. Программное средство разработано в среде программирования С++. В процессе написания программы я более подробно изучил раздел программирования на С++ и получил дополнительные навыки.

# Список использованной литературы.

Изм.

Лист

№ докум.

Подпись

Дата

Лист

*30*

1.Франка П. С++: учебный курс, Питер,2001-528 с.

2. Герберт Шилдт. Самоучитель С++, 3-е издание. Санкт-Петербург 1998-688 с.

3. И.Г. Семакин: Основы программирования: Учебник-Мастерство, 2002 г-432 с.

4. Бондарев В.М. Программирование на С++. 2-е изд. -Харьков: “Компания СМИТ”,2005.-284 с.

5.С/С++.Программирование на языке высокого уровня /Т.А.Павловская .- СПб.:Питер,2003.-461 с.: ил.

**Листинг программы:**

#include <iostream.h>

#include <conio.h>

#include <stdio.h>

#include <string.h>

const N = 30; //максимальное количество возможных записей

//запись

struct TIzdelie{

char f[30]; //фамилия

unsigned int izd[6];//количество изделий собраных ежидневно

} Iz,Izdelie[N];

char ch; //анализируемый символ

int Recs; //последняя введенная запись

//чтение данных из файл fname, возврщает количество считанных данных

int rData(char \*fname){

FILE \*f;

int i=-1;

if ((f=fopen(fname,"r"))!=NULL){

i=0;

while(!feof(f)){

fscanf(f,"%15s",Izdelie[i].f);

fscanf(f,"%6d",&Izdelie[i].izd[0]);

fscanf(f,"%6d",&Izdelie[i].izd[1]);

fscanf(f,"%6d",&Izdelie[i].izd[2]);

fscanf(f,"%6d",&Izdelie[i].izd[3]);

fscanf(f,"%6d",&Izdelie[i].izd[4]);

fscanf(f,"%6d",&Izdelie[i].izd[5]);

Изм.

Лист

№ докум.

Подпись

Дата

Лист

*31*

fscanf(f,"\n");

++i;

}

--i;

fclose(f);

}

return i;

}

//запись данных в файл fname

void wData(char \*fname){

FILE \*f;

if (Recs>=0){

f = fopen(fname,"w");

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

fprintf(f,"%15s",Izdelie[i].f);

fprintf(f,"%6d",Izdelie[i].izd[0]);

fprintf(f,"%6d",Izdelie[i].izd[1]);

fprintf(f,"%6d",Izdelie[i].izd[2]);

fprintf(f,"%6d",Izdelie[i].izd[3]);

fprintf(f,"%6d",Izdelie[i].izd[4]);

fprintf(f,"%6d",Izdelie[i].izd[5]);

Изм.

Лист

№ докум.

Подпись

Дата

Лист

*32*

fprintf(f,"\n");

}

fclose(f);

}

}

//добавление записи

void AddRec(TIzdelie &r){

window(1,1,80,50);

textbackground(1);

clrscr();

textbackground(3);

window(10,20,70,30);

clrscr();

textcolor(0);

if(Recs == N-1){

gotoxy(2,2);

cout<<"Вы ввели максимальное количество записей.";

gotoxy(2,3);

cout<<"Для возврата в основное меню нажмите любую клавишу.";

getch();

} else {

gotoxy(2,2);

cout<<"Фамилия: "; cin>>r.f;

gotoxy(2,3);

cout<<"Количество изделий за понедельник: "; cin>>r.izd[0];

gotoxy(2,4);

cout<<"Количество изделий за вторник: "; cin>>r.izd[1];

gotoxy(2,5);

cout<<"Количество изделий за среду: "; cin>>r.izd[2];

gotoxy(2,6);

cout<<"Количество изделий за четверг: "; cin>>r.izd[3];

gotoxy(2,7);

cout<<"Количество изделий за пятницу: "; cin>>r.izd[4];

gotoxy(2,8);

cout<<"Количество изделий за субботу: "; cin>>r.izd[5];

gotoxy(2,10);

cout<<"Для возврата в основное меню нажмите любую клавишу.";

getch();

++Recs;

}

}

//вывод меню

void OutMenu(){

window(1,1,80,50);

textbackground(1);

clrscr();

textbackground(3);

window(20,20,60,30);

clrscr();

textcolor(0);

gotoxy(2,2);

cout<<"1 - ввод данных";

gotoxy(2,3);

Изм.

Лист

№ докум.

Подпись

Дата

Лист

*33*

cout<<"2 - отображение всех данных";

gotoxy(2,4);

cout<<"3 - общее колличество изделий на неделю";

gotoxy(2,5);

cout<<"4 - наибольшее количество изделий";

gotoxy(2,6);

cout<<"ESC - выход из программы";

}

//отображение всех записей

void OutAllData(){

window(1,1,80,50);

textbackground(1);

clrscr();

textbackground(3);

window(10,5,70,45);

clrscr();

textcolor(0);

gotoxy(2,2); cout<<"№";

gotoxy(5,2); cout<<"Фамилия";

gotoxy(25,2); cout<<"Пн.";

gotoxy(30,2); cout<<"Вт.";

gotoxy(35,2); cout<<"Ср.";

gotoxy(40,2); cout<<"Чт.";

gotoxy(45,2); cout<<"Пт.";

gotoxy(50,2); cout<<"Сб.";

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

gotoxy(2,2+2+i); cout<<i+1;

gotoxy(5,2+2+i); cout<<Izdelie[i].f;

gotoxy(25,2+2+i); cout<<Izdelie[i].izd[0];

gotoxy(30,2+2+i); cout<<Izdelie[i].izd[1];

gotoxy(35,2+2+i); cout<<Izdelie[i].izd[2];

gotoxy(40,2+2+i); cout<<Izdelie[i].izd[3];

gotoxy(45,2+2+i); cout<<Izdelie[i].izd[4];

gotoxy(50,2+2+i); cout<<Izdelie[i].izd[5];

}

gotoxy(2,2+2+1+i);

cout<<"Для возврата в основное меню нажмите любую клавишу.";

getch();

}

//Общее количество деталей за неделю

void sumIzdelies(){

window(1,1,80,50);

textbackground(1);

clrscr();

textbackground(3);

window(10,5,70,45);

clrscr();

textcolor(0);

gotoxy(2,2); cout<<"№";

gotoxy(5,2); cout<<"Фамилия";

gotoxy(25,2); cout<<"Общее количество деталей за неделю";

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

gotoxy(2,2+2+i); cout<<i+1;

gotoxy(5,2+2+i); cout<<Izdelie[i].f;

int sum=0;

for (int j=0;j<6;j++){

sum+=Izdelie[i].izd[j];

}

gotoxy(25,2+2+i); cout<<sum;

}

gotoxy(2,2+2+1+i);

cout<<"Для возврата в основное меню нажмите любую клавишу.";

getch();

}

//Определение сбощика и наибольших значений

void Sborshik(){

window(1,1,80,50);

textbackground(1);

clrscr();

Изм.

Лист

№ докум.

Подпись

Дата

Лист

*34*

textbackground(3);

window(10,10,70,40);

clrscr();

textcolor(0);

//наибольшее количество изделий

unsigned int max1=0; unsigned int max2=0;

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

int sum=0;

for (int j=0;j<6;j++){

sum+=Izdelie[i].izd[j];

}

if (sum>max1) max1 = sum;

}

//в какой день наибольшее количество изделий

for(i=0;i<=Recs;i++){

int sum=0;

for (int j=0;j<6;j++){

sum+=Izdelie[i].izd[j];

}

if (sum == max1){

max2 = Izdelie[i].izd[0];

for (int j=1;j<6;j++)

if (Izdelie[i].izd[j]>max2) max2=Izdelie[i].izd[j];

}

}

//отображение найденых значений

int c=0;

for(i=0;i<=Recs;i++){

int sum=0;

for (int j=0;j<6;j++){

sum+=Izdelie[i].izd[j];

}

if (sum == max1){

gotoxy(2,2+c);

cout<<"Наиболшее число изделий у: "<<Izdelie[i].f;

gotoxy(32,2+c);

cout<<"День: ";

int d=0;

for (int j=0;j<6;j++)

if (Izdelie[i].izd[j] == max2){

gotoxy(38+d\*4,2+c);

if (j==0){

cout<<"Пн.";

++d;

}else if (j==1){

cout<<"Вт.";

++d;

}else if (j==2){

cout<<"Ср.";

++d;

}else if (j==3){

cout<<"Чт.";

++d;

}else if (j==4){

cout<<"Пт.";

++d;

}else if (j==5){

cout<<"Сб.";

++d;

Изм.

Лист

№ докум.

Подпись

Дата

Лист

*35*

}

}

++c;

}

}

gotoxy(2,2+1+c);

cout<<"Для возврата в основное меню нажмите любую клавишу.";

getch();

}

//код основной программы

void main(){

clrscr();//очистка экрана

textmode(64);// задание тектового режима 80x50 (ширина х высота)

OutMenu();//вывод основного меню

Recs = rData("izdeliya.dat");//чтение данных из файла если таковой имеется

while((ch=getch())!=0x1B){

switch(ch){

case '1': AddRec(Iz);

Izdelie[Recs] = Iz;

break;

case '2': OutAllData();

break;

case '3': sumIzdelies();

break;

case '4': Sborshik();

break;

}

clrscr();

OutMenu();

}

wData("izdeliya.dat"); //запись данных в файл

}

Изм.

Лист

№ докум.

Подпись

Дата

Лист

*36*