МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ
РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО
ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ «МОСКОВСКИЙ
ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
РАДИОТЕХНИКИ, ЭЛЕКТРОНИКИ И АВТОМАТИКИ»

ИНФОРМАТИКА

Методические указания
по выполнению курсовой работы
студентами очного отделения по направлению
«Управление в технических системах»

МОСКВА 2012

Составители: В.Т. Лузинский,
А.М. Набатчиков,
Е.А. Бурлак

Редактор Кочемасов А.В.

Методические указания содержат описание курсовой работы по курсу "Информатика" для студентов 1курса очного отделения.

Печатаются по решению редакционно-издательского совета университета.

Рецензенты: В.Л. Лотоцкий
К.Я. Вильданов

Методические указания напечатаны в авторской редакции

Подписано в печать XX.XX.2012. Формат 60x84 1/16.
Усл. печ. л. XX. Усл. кр.-отт. XX. Уч.-изд. л. XXX.
Тираж 100 экз. Заказ XXX. Бесплатно

Федеральное государственное бюджетное образовательное
учреждение высшего профессионального образования
“Московский государственный технический университет
радиотехники, электроники и автоматики”
119454, Москва, пр. Вернадского, 78

ВВЕДЕНИЕ

Согласно действующему учебному плану дисциплина «Информатика» на факультете Кибернетика изучается студентами очного отделения по направлению подготовки 220400 «Управление в технических системах» в течение 2-х семестров. Целью курсовой работы по данной дисциплине является закрепление тех навыков по составлению алгоритмов и программированию, которые получены на лекциях и практических занятиях. При этом студенты знакомятся с такими технологиями программирования, как процедурное программирование и модульное программирование, а также объектно-ориентированное программирование.

Специальность «Управление в технических системах» не предполагает подготовки профессиональных разработчиков программного обеспечения, и при этом в настоящее время уже имеется множество специализированных прикладных программ, позволяющих решать инженерные задачи в конкретной предметной области. Однако для освоения современных компьютерных технологий на профессиональном уровне будущему инженеру по прежнему необходимо быть знакомым с основами программирования. Поэтому студентам необходимо научиться разрабатывать алгоритмы умеренной сложности, писать и отлаживать программы размером, по меньшей мере, в несколько десятков строк исходного кода, а также использовать в своих программах стандартные библиотеки. Достижению этой цели и служит курсовая работа.

Знания, приобретаемые студентами на этом этапе обучения могут пригодится им например при освоении основ численного анализа, а также могут составить основу культуры программирования, а это необходимо при изучении ряда будущих изучаемых специальных дисциплин.

Кроме того, при выполнении курсовой работы решается также задача закрепления полученных знаний подготовке документов с использованием возможностей текстового процессора Microsoft Word. В связи с этим обязательное требование состоит в том, чтобы курсовая работа оформлялась как документ Word, в

соответствии с установленными на кафедре правилами. Эти навыки в дальнейшем могут быть использованы при подготовке отчетов по лабораторным работам, при написании пояснительных записок к курсовым работам и проектам по другим дисциплинам, а также при оформлении дипломной работы.

Тема курсовой работы задается и определяется преподавателем – руководителем курсовой работы, который также может уточнить некоторые из требований к содержанию и оформлению курсовой работы.

К защите работа должна быть представлена в виде распечатанного текста, содержащего все необходимые разделы, к которому должна быть приложена носитель информации (CD\DVD) с программами.

1. СОДЕРЖАНИЕ КУРСОВОЙ РАБОТЫ

Курсовая работа должна содержать следующие разделы:

1. Формулировка задания.
2. Краткие математические сведения, лежащие в основе разрабатываемых алгоритмов (при необходимости).
3. Спецификация. Пояснения для используемых переменных и функций.
4. Разработка и описание алгоритмов для решения поставленной задачи.
5. Тексты, реализующих алгоритмы программы, на языке C++
6. Текст тестирующей программы на языке C++ с результатами тестирования.
7. Список используемой литературы.

В разделе 1 должна быть дана формулировка задания, с подробным указанием назначения функций, которые требуется разрабатывать, и характера исходных и выдаваемых в качестве результата данных. Необходимо также указать требуемый способ представления данных (ввод с клавиатуры, вывод на экран, чтение из файла на диске или запись в файл и т.п.). Во всех случаях

формулировка задания должна начинаться со слов: «Требуется разработать программный модуль на языке C++, включающий функции, предназначенные для …». В формулировке задания также должны быть отражены требования по отладке и тестированию разработанного модуля. Прежде чем приступать к выполнению работы каждый студент должен оформить задание на отдельном листе и на нем получить подпись у своего руководителя (это необходимо сделать в самом начале семестра; в противном случае, а также в случае, если студент не предъявляет промежуточные результаты своей самостоятельной работы на консультациях в течение семестра задание может быть изменено по усмотрению преподавателя).

В разделе 2 должны содержаться формулировки важнейших математических фактов, которые лежат в основе разрабатываемых алгоритмов. Используемые определения, математические утверждения (теоремы, свойства и т.п.) и формулировки должны быть пронумерованы. Например, если дано задание написать функцию для вычисления определителя произвольной квадратной матрицы с помощью приведения ее к треугольному виду (в данной работе задания более простые), то в этом разделе должны быть перечислены все свойства определителей, которые для этого вычисления будут использоваться. Если задание не предполагает использование специальных математических фактов, то этот раздел может быть опущен.

В разделе 3 составляется таблица. В левой колонке список используемых переменных, функций, а в правой комментарии и пояснения к ним.

В разделе 4 должны содержаться детальные описания алгоритмов. Для этого каждая решаемая задача должна быть разбита на отдельные подзадачи, последовательность которые приводят к ее решению. При этом детализация каждого такого шага может быть представлена в виде отдельного алгоритма.

Для описания алгоритмов может использоваться как обычный язык, включающий упрощенную запись (без следования строгим синтаксическим правилам языка C++) условных операторов, операторов цикла и т.д., так и блок-схемы. Главное тре-

бование: показать, из каких именно элементарных шагов, не требующих пояснения, будет складываться решение задачи.

Следует стремиться составлять алгоритмы так, чтобы в них не было повторных вычислений одних и тех же величин, т.е. к разработке алгоритмов надо подходить с точки зрения экономии вычислительных ресурсов.

В разделе 5 (это основной раздел) должны содержаться тексты написанных на языке C++ и отлаженных функций, реализующих разработанные в предыдущем разделе алгоритмы. Как отмечалось во введении, одной из целей курсовой работы является освоение технологии процедурного и модульного программирования. Поэтому каждое выданное задание, независимо от его конкретного содержания, предполагает разработку модуля, в который должны быть включены некоторые функции (задание формулируется в стандартном виде: разработать такие-то функции, выполняющие то-то и то-то).

2. СТРУКТУРА ПРОГРАММЫ

Грамотно структурированный проект подразумевает разбиение исходного кода программы не только на функции, но и другие структурные элементы (классы, структуры, перечисления, пространства имён и т.п.), соответствующие разбиению решаемой задачи на отдельные этапы. Такой подход позволяет выделить обособленные шаги решения и некие примитивы, многократно используемые в ходе выполнения программы. Организация решения задачи в виде отдельной библиотеки позволяет абстрагироваться при дальнейшем использовании данного исходного кода от деталей реализации.

Отметим, что термином «*библиотека*», в зависимости от контекста, в программировании могут называться довольно разноплановые механизмы разделения программы на модули. Например, использование технологии *раздельной компиляции* (separate compilation) – раздельной трансляции частей программы в модули, которые затем собирает компоновщик в единое закон-

ченное (загружаемое операционной системой) целое. «В C++» (это вопрос скорее не языка, а компилятора, конкретно – компоновщика) использование оттранслированных (*объектных*) файлов, так называемых «*объектных библиотек*», хоть и возможно, на практике встречается нечасто (хотя компилятор имеет в поставке оттранслированные реализации библиотек, что позволяет ускорить сборку приложения, сам процесс их вовлечения скрыт от пользователя). Более часто встречающийся вариант: не статическая, а динамическая компоновка, при которой в роли подключаемого модуля выступает DLL-файл (динамически подключаемая библиотека). При динамической компоновке связывание библиотечного модуля с исполнимым происходит на этапе выполнения (а не компиляции), при помощи объектного файла (так называемая «*библиотеки импорта*»), содержащего необходимые для использования модуля данные, но, в отличие от *объектной библиотеки*, не содержащие самого кода. Возможна и *динамическая* загрузка *динамически* подключаемой библиотеки (выше рассмотрена *статическая* загрузка, называемая ещё «*неявным подключением*», *динамически* подключаемой библиотеки): в этом случае, работа с библиотекой осуществляется при помощи API ОС без использования *библиотеки импорта*.

Рассмотренный вариант весьма популярен, но далеко не всегда оправдан. Ему можно отдать предпочтение по разным причинам: исходный код библиотеки остаётся «закрытым»; модуль позволяет расширить возможности приложения; модуль сводит код, написанный на разных ЯП к единому интерфейсу; модуль позволяет оперировать с функционалом программы по принципу конструктора (использование плагинов). Тем не менее, наиболее распространенный вариант – поставка библиотеки в виде исходного кода. Кроме того, разработка DLL – отдельная большая тема, имеющая свою специфику. Экспортирование функций, отмена декорирования имён, учёт так называемого «*соглашения вызова*» (calling convention), проблема совместимости форматов Object Module Format (OMF) и Common Object File Format (COFF) и прочее. В частности, разные компиляторы используют тот или иной формат для библиотек импорта.

Более подробные сведения об особенностях взаимодействия модулей можно найти в [1, 10], а основные принципы построения компиляторов – в [2].

Исходя из вышесказанного, уточним, что в рамках данной работы под *библиотекой* подразумевается совокупность двух файлов: заголовочный файл (*.h) и файл реализации (*.cpp).

Такая библиотека может затем многократно использоваться в других проектах, в том числе и другими программистами. Напомним, что заголовочный файл, используемый при написании программ на языке C++, как правило, имеет следующую структуру.

Заголовочный файл (test.h):

```
#ifndef TEST_H_INCLUDED
#define TEST_H_INCLUDED

void foo(int &baz);
float bar(int baz);

#endif
```

Данный пример содержит прототипы функций (`foo` и `bar`), но здесь можно также описать используемые структуры, классы и т.п. Для того, чтобы избежать множественного подключения файла, основное его содержимое обрамлено конструкциями, реализующими макрозащиту: проверка `#ifndef` заставляет пропустить препроцессор весь код идущий до `#endif`, если уже было вызвано макроопределение `TEST_H_INCLUDED` (конкретное имя выбирается на усмотрение программиста).

Заголовочный файл должен быть включён в проект, а его использование должно быть означено при помощи подключения его директивой `#include` в необходимых файлах.

Файл реализации (test.cpp):

```
void foo(int &baz){
    baz++;
}
float bar(int baz){
    return baz*baz;}
```

Здесь содержатся реализации (тела) функций и методов.

Файл реализации должен быть включён в проект для трансляции его в объектный модуль.

Файлы реализации и заголовочный могут подключать другие заголовочные файлы при помощи директивы `#include`.

В курсовой работе требуется, чтобы была написана определенная в задании библиотека, а входящие в неё функции были отлажены (т.е. в них должны быть выявлены и исправлены возможные допущенные при разработке ошибки).

В разделе 5 должно содержаться описание тестовых примеров (в достаточном количестве), подтверждающих правильность написанных функций. При этом должна быть написана программа (основная, к которой должна быть подключена разработанная библиотека), из которой с подробными исходными данными вызываются разработанные функции. Должны быть приведены текст этой программы и результаты ее работы в виде распечатки таблиц, графиков и т.д.

Если выданное задание требует визуализации каких-либо функций, то эта задача (построение графиков) может быть решена с помощью специальных графических функций пакета MATLAB, Mathcad, SMath Studio или любого другого программного обеспечения. Как правило, для этого табличные значения функции должны быть предварительно записаны в текстовый файл в требуемом формате. Более подробные сведения по работе с системой MATLAB можно найти, например, в [3].

Основная программа, будет иметь следующую структуру:

```
#include “test.h”
using namespace std;

int main(){
    return 0;
}
```

Здесь, при помощи директивы include происходит подключение файла, содержащего прототипы функций, которые

будут использоваться основной программой. Подключать подобным образом файл реализации не нужно, но необходимо добавить его в проект при помощи соответствующей опции, используемой вами ИСР[1] (если вы разрабатываете этот файл «с нуля» в ИСР, то он будет добавлен автоматически или после утвердительного ответа на запрос о добавлении). При помощи объявления using вводится локальный синоним, позволяющий избежать повторяющейся утомительной квалификации имён из пространства имён std.

В разделе 6 должен быть приведен полный список используемой литературы, включающий справочные пособия по языку программирования, книги по математике и т.д. При необходимости в тексте курсовой следует делать ссылки на список.

Примечание. Строки содержат символы, слова, цифры и др. Т.е. группа символов без пробелов внутри и с пробелами вне группы, слева и справа.

3. ЧИСЛЕННЫЕ МЕТОДЫ. ПРОЦЕДУРЫ И ФУНКЦИИ.

Цель работы: приобрести навыки разработки программ с функциями.

Постановка задачи. Сформировать матрицу C(5,5), элементы которой являются значениями определённого интеграла

$$C(I,J) = \int_{I/20}^{(I+J)/20} f(x)dx$$

Алгоритмы выполнения функциональных задач: вычисление подынтегральной функции, вычисление определённого интеграла с заданной точностью двумя методами, формирование матрицы и вывода матрицы оформить в виде функций. Функция вычисления интеграла с заданной точностью должна

[1] Интегрированная среда разработки (IDE).

иметь следующие входные параметры: верхний и нижний пределы интегрирования, точность вычисления значения интеграла, начальное количество элементарных отрезков и подынтегральную функцию. Варианты заданий представлены в табл. 6. Использовать в работе работу с файловым типом данных.

Методические указания

1. При разработке программы используйте метод нисходящего программирования. Начните с программирования и отладки основной программы, заменив функции "заглушками". После отладки основной программы последовательно заменяйте "заглушки" телами функций. Пример "заглушки" для функции вычисления интеграла с заданной точностью:

```
float Integ(float a, float b, float e,
        int n, float (*f)(float))
{
    return 1;
}
```

Последний аргумент функции `Integ` – указатель на функцию с прототипом:

`float func(float)`. Можно осуществить, например, такой вызов: `Integ(1,2,0.03,12,G);` где `G` имеет прототип: `float G(float)` . В этом случае, все вызовы `f` в теле функции `Integ` будут трактовать как вызовы `G`.

2.Формулы численного интегрирования

- Формулы левых прямоугольников

$J = h\sum_{i=2}^{n+1} f(X_i)$ где n - количество элементарных отрезков, на которые разбивается отрезок интегрирования $[a,b]$, $h = (b-a)/n$, $x_1 = a$.

- Формула средних

$J = h\sum_{i=1}^{n} f(X_i + 0.5h)$, где $x_1 = a$.

- Формула правых прямоугольников

$$J = h\sum_{i=2}^{n+1} f(X_i), \text{ где } x_1 = a + h$$

- Формула трапеций

$$J = h(\frac{f(a) + f(b)}{2} + \sum_{i=2}^{n} f(X_i))$$

- Формула Симеона (п - чётное)

J = h/3(f(a) + 4f(a + h) + 2f(a + 2h)+4f(a+3h) + ...+4f(b-h)+f(b)).

- Метод Веддля

Метод базируется на применении к каждому из п элементарных отрезков [x ,x] длиной h = (b-a)/n формулы

$$\int_{x_1}^{x_7} f(x)dx = 3z/10(f(x_1) + 5f(x_2) + f(x_3) + 6f(x_4) + f(x_5) + 5f(x_6) + f(x_7))$$

где $x_i = x_1 + z(i-1),\ i = 2,3,4,5,6,7,\ z = h/6$

- Метод Ньютона

Метод базируется на применении к каждому из n элементарных отрезков $[x_1, x_4]$ длиной h = (b-a)/n формулы

$$\int_{x_1}^{x_4} f(x)dx = 3z/8(f(x_1) + 3f(x_2) + 3f(x_3) + f(x_4)),$$

где $x_i = x_1 + z(i-1),\ \ i = 2,3,4,\ \ z = h/3$

3. Для достижения заданной точности *E* используйте алгоритм последовательного удвоения количества элементарных отрезков интегрирования:

- Первоначально отрезок [a,b]разбивается на $n=n_0$ частей размером h=(b-a)/n;
- Вычисляется значение интеграла J_1 выбранным методом интегрирования;
- Число элементарных отрезков удваивается n=2n и вычисляется значение интеграла J_2 для h=h/2;
- Если $|J_1 - J_2| < E$, то конец алгоритма, иначе необходимо удвоить число элементарных отрезков и вычислить новое приближение интеграла и т. д.

Таблица 1 **Варианты заданий**

№	Метод интегрирования 1	Метод интегрирования 2	Подынтегральная функция
1	Метод трапеций	Метод средних	$\sin^2(2x)/\cos(2x)$
2	Метод Симпсона	Метод Ньютона	$\sin^3(3x)/\cos(2x)$
3	Метод средних	Метод трапеций	$1/\ln(x)$
4	Метод Веддля	Метод Симпсона	$\sin(\ln(x))$
5	Метод Симпсона	Метод левых прямоугольников	$1/\sin(2x)$
6	Метод трапеций	Метод Ньютона	$x/\ln(3x)$
7	Метод правых прямоугольников	Метод Симпсона	$1/(1+\sin(2x))$
8	Метод Веддля	Метод трапеций	$1/\cos(3x)$
9	Метод средних	Метод правых прямоугольников	$1/(1+\cos(3x))$
10	Метод Ньютона	Метод Веддля	$1/(\sin(2x)\cos(2x))$
11	Метод правых прямоугольников	Метод средних	$x\sin(x)$
12	Метод трапеций	Метод Веддля	$\ln(x)\cos(x)$
13	Метод левых прямоугольников	Метод правых прямоугольников	$\cos(2x)\sqrt{x}$
14	Метод Симпсона	Метод трапеций	$1/(\cos(2x)+2)$
15	Метод Ньютона	Метод левых прямоугольников	$1/(x^3\ln(x)))$
16	Метод Симпсона	Метод Веддля	$1/\cos(2x)$
17	Метод трапеций	Метод Веддля	$2x/\ln(3x)$
18	Метод правых прямоугольников	Метод Ньютона	$1/(1+\cos(2x))$
19	Метод Веддля	Метод трапеций	$1/\sin(3x)$
20	Метод средних	Метод Симпсона	$1/(1+\sin(3x))$
21	Метод Ньютона	Метод средних	$1/(\sin(3x)\cos(3x))$
22	Метод трапеций	Метод Веддля	$(\sin^2(4x)\cos^2(4x))$
23	Метод Симпсона	Метод правых прямоугольников	$1/(\sin(3x)\cos(3x))$
24	Метод средних	Метод трапеций	$1/\ln(x)$
25	Метод Веддля	Метод левых прямоугольников	$\cos(\ln(x))$
26	Метод правых прямоугольников	Метод Симпсона	$2x*3\cos(x)$
27	Метод трапеций	Метод Ньютона	$\ln(x)\sin(x)$
28	Метод правых прямоугольников	Метод Симпсона	$\cos(2x)\sqrt{x}$
29	Метод Симпсона	Метод трапеций	$1/(\sin(2x)+2)$
30	Метод Веддля	Метод правых прямоугольников	$1/(x^3\ln(x)))$
31	Метод Симпсона	Метод левых прямоугольников	$\sin(\ln(x))$

Контрольные вопросы 1

1. Что такое аргумент функции (в программировании)?
2. Что означают пустые скобки после имени функции?
3. Сколько значений может возвращать функция?
4. Какой тип возвращаемого значения имеет функция, если она не возвращает значения?
5. Какова причина использования ссылочного механизма передачи аргументов в функцию?
6. Что такое *перегруженные* функции?
7. Что такое *встроенные* функции?
8. Как задать значение аргумента функции по умолчанию?
9. Какие функции могут иметь доступ к *глобальной* переменной, расположенной в одном файле с ними?
10. Для чего используется *статическая локальная* переменная?
11. В каком необычном месте программы можно использовать вызов функции, если эта функция возвращает значение по ссылке?

В случае если приведённые вопросы вызывают затруднения, рекомендуем ознакомиться с [4, 5, 7].

4. МАССИВЫ, СТРОКОВЫЕ И СИМВОЛЬНЫЕ ПЕРЕМЕННЫЕ

Вариант 1

1. Написать программу согласно следующему условию: Дана целочисленная матрица A размером MxN, где M,N - заданные натуральные числа, а все элементы матрицы различные. Найти сумму S=X(1)*X(2)+X(2)*X(3)+... +X(M-1)*X(M), где X(i) - максимальный элемент в i-ой строке матрицы.

2. Написать программу согласно следующему условию: Дана строка символов. Группу символов, разделенную с одной или с обеих сторон одним или несколькими пробелами и не содержащую внутри себя пробелов, назовем словом. Распечатать самое короткое слово, начинающееся на букву "К". Пред-

полагается, что если такое слово есть, то оно единственное. Если таких слов нет, то выдать соответствующее текстовое сообщение.

Вариант 2

1. Написать программу согласно следующему условию: Дана квадратная целочисленная матрица A порядка N (N-заданное натуральное число), все элементы которой различны. Поменять местами строку, в которой находится наименьший элемент матрицы, со строкой, где находится наибольший элемент матрицы.

2. Написать программу согласно следующему условию: Дана строка символов. Группу символов, разделенную с одной или с обеих сторон одним или несколькими пробелами и не содержащую внутри себя пробелов, назовем словом. Вывести на экран все слова, которые отличны от слова INFORMATION. Если таких слов нет, то выдать соответствующее текстовое сообщение.

3. Написать программу согласно следующему условию: Дана действительная квадратная матрица A порядка N, где N - заданное натуральное число. Сколько элементов матрицы равны (MAX+MIN)/2, где MAX, MIN - соответственно, максимальное и минимальное значения среди элементов матрицы.

4. Написать программу согласно следующему условию: Дана строка символов. Группу символов, разделенную с одной или с обеих сторон одним или несколькими пробелами и не содержащую внутри себя пробелов, назовем словом. Распечатать все слова с количеством символов больше 4 и меньше 10. Если такого слова нет, то выдать соответствующее текстовое сообщение.

Вариант 3

1. Написать программу согласно следующему условию: Дана целочисленная матрица A размером MxN, где M, N - заданные натуральные числа. Сформировать одномерный массив B, где B (i) равно сумме элементов, кратных пяти и располо-

женных в i строке матрицы, i = l, 2, ... M. Если таких элементов в i строке нет, то элементу B(i) присвоить номер строки.

2. Написать программу согласно следующему условию: Дана строка символов. Группу символов, разделенную с одной или с обеих сторон одним или несколькими пробелами и не содержащую внутри себя пробелов, назовем словом. Распечатать самое длинное слово, начинающееся на букву "К". Если таких слов нет, то выдать соответствующее текстовое сообщение, а если такое слово есть, то предполагается, что оно единственное.

Вариант 4

1. Написать программу согласно следующему условию: Дана целочисленная матрица A размером MxN, где M, N - заданные натуральные числа. Найти количество столбцов матрицы, содержащих одни нулевые элементы.

2. Написать программу согласно следующему условию: Дана строка символов. Группу символов, разделенную с одной или с обеих сторон одним или несколькими пробелами и не содержащую внутри себя пробелов, назовем словом. Распечатать самое длинное слово, первые две буквы которого "КО" (предполагается, что если такое слово есть, то оно единственное). Если таких слов нет, то выдать соответствующее текстовое сообщение.

3. Написать программу согласно следующему условию: Дана действительная квадратная матрица A порядка N, где N – заданное натуральное число. Сколько элементов матрицы равны (MAX+MIN)/2, где MAX, MIN – соответственно, максимальное и минимальное значения среди элементов матрицы.

4. Написать программу согласно следующему условию: Дана строка символов. Группу символов, разделенную с одной или с обеих сторон одним или несколькими пробелами и не содержащую внутри себя пробелов, назовем словом. Распечатать все слова с количеством символов больше 4 и меньше 10. Если такого слова нет, то выдать соответствующее текстовое сообщение.

Вариант 5

1. Написать программу согласно следующему условию: Дана целочисленная матрица A размером MxN, где M, N - заданные натуральные числа. Сформировать одномерный массив B, где B (i) равно сумме элементов, кратных пяти и расположенных в i строке матрицы, i=l 2,... M. Если таких элементов в i строке нет, то элементу B(i) присвоить номер строки.

2. Написать программу согласно следующему условию: Дана строка символов. Группу символов, разделенную с одной или с обеих сторон одним или несколькими пробелами и не содержащую внутри себя пробелов, назовем словом. Распечатать самое длинное слово, начинающееся на букву "К". Если таких слов нет, то выдать соответствующее текстовое сообщение, а если такое слово есть, то предполагается, что оно единственное.

Вариант 6

1. Написать программу согласно следующему условию: Дана целочисленная матрица A размером MxN, где M,N - заданные натуральные числа. Найти количество столбцов матрицы, содержащих одни нулевые элементы.

2. Написать программу согласно следующему условию: Дана строка символов. Группу символов, разделенную с одной или с обеих сторон одним или несколькими пробелами и не содержащую внутри себя пробелов, назовем словом. Распечатать самое длинное слово, первые две буквы которого "КО" (предполагается, что если такое слово есть, то оно единственное). Если таких слов нет, то выдать соответствующее текстовое сообщение.

Вариант 7

1. Написать программу согласно следующему условию: Дана целочисленная матрица A размером MxN, где M, X - заданные натуральные числа, причем M>5. Найти количество столбцов матрицы, в каждом из которых содержится не менее пяти нулевых элементов, причем три из них обязательно подряд.

2. Написать программу согласно следующему условию: Дана строка символов. Группу символов, разделенную с одной или с обеих сторон одним или несколькими пробелами и не содержащую внутри себя пробелов, назовем словом. Распечатать самое длинное симметричное слово, первые две буквы которого "КО" (предполагается, что если такое слово есть, то оно единственное). Если таких слов нет, то выдать соответствующее текстовое сообщение.

Вариант 8

1. Написать программу согласно следующему условию: Дана квадратная целочисленная матрица A порядка N, где N – заданное натуральное число. Является ли заданная матрица магическим квадратом, то есть такой, в которой суммы элементов во всех строках и столбцах одинаковы.

2. Написать программу согласно следующему условию: Дана строка символов. Группу символов, разделенную с одной или с обеих сторон одним или несколькими пробелами и не содержащую внутри себя пробелов, назовем словом. Выяснить, какое слово встречается раньше в строке с наименьшим или наибольшим количеством символов. Предполагается, что в строке более двух слов и все слова различной длины.

Вариант 9

1. Написать программу согласно следующему условию: Дана целочисленная матрица A размером MxN, где M, N - заданные натуральные числа. Сформировать одномерный целочисленный массив B, где B(i) равно сумме элементов, кратных трем в i-ой строке матрицы I = 1, 2, ..., M. Если таких элементов в i-ой строке нет, то B(i) равно среднему арифметическому значению элементов, расположенных в i-ой строке матрицы.

2. Написать программу согласно следующему условию: Дана строка символов. Группу символов, разделенную с одной или с обеих сторон одним или несколькими пробелами и не содержащую внутри себя пробелов, назовем словом. Определить среднее количество символов в словах четной длины. Ес-

ли слов с четной длиной нет, то выдать соответствующее текстовое сообщение.

Вариант 10

1. Написать программу согласно следующему условию: Дана действительная матрица A размером MxN, где M, N – заданные натуральные числа, а все элементы матрицы различные. Сформировать одномерный целочисленный массив B, где B(j) равно среднему арифметическому значению всех индексов наибольшего и наименьшего элементов в j-ой столбце j=l,2, ...,N.

2. Написать программу согласно следующему условию: Дана строка символов. Группу символов, разделенную с одной или с обеих сторон одним или несколькими пробелами и не содержащую внутри себя пробелов, назовем словом. Распечатать все слова нечетной длины, начинающиеся и оканчивающиеся на букву T. Если таких слов нет, то выдать соответствующее текстовое сообщение.

Вариант 11

1. Написать программу согласно следующему условию: Дана действительная матрица A размером MxN, где M, N - заданные натуральные числа. Сформировать одномерный действительный массив B, в котором элемент B(j) равен 1, если элементы j-ro столбца матрицы составляют строго возрастающую последовательность и ноль в противном случае *(]* =\,*2*,..., N).

2. Написать программу согласно следующему условию: Дана строка символов. Группу символов, разделенную с одной или с обеих сторон одним или несколькими пробелами и не содержащую внутри себя пробелов, назовем словом. Вывести на экран все слова четной длины, расположенные после слова с наибольшей длиной. Предполагается, что слово с наибольшей длиной единственное. Если таких слов нет, то выдать соответствующее текстовое сообщение.

Вариант 12

1. Написать программу согласно следующему условию:

Дана действительная матрица A размером MxN, где M, N – заданные натуральные числа. Назовем элемент A (I, J) особым (i=1, 2, ...,M; j = l, 2, ..., N), если сумма элементов в i строке больше нуля, а сумма элементов в j столбце кратна 5. Найти сумму особых элементов в матрице A. Если особых элементов нет, то вывести соответствующее текстовое сообщение.

2. Написать программу согласно следующему условию: Дана строка символов. Группу символов, разделенную с одной или с обеих сторон одним или несколькими пробелами и не содержащую внутри себя пробелов, назовем словом. Вывести на экран слова в обратном порядке и инвертировав порядок букв («слово» → «оволс»), то есть первым будет записано последнее слово, вторым предпоследнее и т.д.

Вариант 13

1. Написать программу согласно следующему условию: Дана матрица A размером MxN, где M, N - заданные натуральные числа, состоящая из натуральных чисел, больших 1. Составить одномерный массив B, в котором B(J) (j= l, 2, ..., N) равно J, если j-ый столбец содержит только простые числа, в противном случае B(J)=0.

2. Написать программу согласно следующему условию: Дана строка символов. Группу символов, разделенную с одной или с обеих сторон одним или несколькими пробелами и не содержащую внутри себя пробелов, назовем словом. Вывести на экран строку из последовательности слов в обратном порядке (то есть первым - последнее слово, затем предпоследнее и т.д.). Между словами сделать один пробел.

Вариант 14

1. Написать программу согласно следующему условию: Дана действительная квадратная матрица A порядка N, (N - заданное натуральное число), а все элементы матрицы различные.

Вычислить

S=X(l)+X(l)*X(2)+X(l)*X(2)*X(3)+...+X(l)*X(2)*...*X(N),

Где X(1) максимальное из значений элементов матрицы, расположенных в i строке, 1=1,2,...., N.

2. Написать программу согласно следующему условию: Дана строка символов. Группу символов, разделенную с одной или с обеих сторон одним или несколькими пробелами и не содержащую внутри себя пробелов, назовем словом. Выяснить, имеется ли в самом коротком слове пара соседствующих букв "КА" или нет. Предполагается, что слово с наименьшей длиной единственное.

Вариант 15

1. Написать программу согласно следующему условию: Дана действительная матрица A размером MxN, где MN – заданные натуральные числа, а все элементы матрицы разлитые. Назовем элемент A(I,J) особым (i = l, 2, ..., M; j = l, 2, ..., N), если максимальное из значений элементов в i строке больше минимального из значений элементов в j столбце. Найти произведение особых элементов в матрице A. Если особых элементов нет, то вывести соответствующее текстовое сообщение.

2. Написать программу согласно следующему условию: Дана строка символов, содержащая четное количество символов. Группу символов, разделенную с одной или с обеих сторон одним или несколькими пробелами и не содержащую внутри себя пробелов, назовем словом. Присвоить переменной F=l, если слово с наибольшим количеством символов находится в первой половине строки, F=2 – во второй половине, F=0 – часть слова в первой половине, а часть во второй. Предполагается, что слово с наибольшей длиной единственное.

Вариант 16

1. Написать программу согласно следующему условию: Дана матрица A размером MxN, где M, N - заданные натуральные числа, состоящая из натуральных чисел, больших 1. Вывести на экран столбцы, которые содержат только простые числа. Если таких столбцов нет, то вывести соответствующее текстовое сообщение.

2. Написать программу согласно следующему условию: Дана строка символов. Группу символов, разделенную с одной или с обеих сторон одним или несколькими пробелами и не содержащую внутри себя пробелов, назовем словом. Напечатать строку из последовательности слов, в которых нет удвоенной буквы "н". Между словами сделать один пробел. Если таких слов нет, то выдать соответствующее текстовое сообщение.

Вариант 17

1. Написать программу согласно следующему условию: Дана действительная матрица A размером MxN, где M,N - заданные натуральные числа. Найти максимальное из значений всех элементов тех строк матрицы, среднее арифметическое значение элементов в каждой из которых больше 10 и меньше 100. Если таких строк нет, то вывести соответствующее текстовое сообщение.

2. Написать программу согласно следующему условию: Дана строка символов, содержащая четное количество символов. Группу символов, разделенную с одной или с обеих сторон одним или несколькими пробелами и не содержащую внутри себя пробелов, назовем словом. Вывести на экран слова нечетной длины, полностью расположенные в первой половине строки. Если таких слов нет, то вывести соответствующее текстовое сообщение.

Вариант 18

1. Написать программу согласно следующему условию: Дана матрица A размером MxN, где M, N – заданные натуральные числа, состоящая из натуральных чисел больше. 1. Сформировать одномерный массив B, где B(i) равно среднему арифметическому значению всех элементов, являющихся простыми числами в i-ой строке матрицы, (i = l, 2, ..., M). Если таких элементов в i-ой строке нет, то элементу B(1) присвоить номер строки.

2. Написать программу согласно следующему условию:

Дана строка символов. Группу символов, разделенную с одной или с обеих сторон одним или несколькими пробелами и не содержащую внутри себя пробелов, назовем словом. Распечатать все слова нечетной длины, отличные от второго слова. Предполагается, что в строке больше двух слов. Если таких слов нет, то вывести соответствующее текстовое сообщение.

Вариант 19

1. Написать программу согласно следующему условию: Дана целочисленная квадратная матрица A порядка N, где N - заданное натуральное число. Столбец с индексом J (J=l, 2, ..., N) назовем отмеченным, если все элементы в этом столбце, расположенные на главной диагонали и ниже нее неотрицательны и оканчиваются на цифру 7. Найти количество отмеченных столбцов в матрице A.

2. Написать программу согласно следующему условию: Дана строка символов. Группу символов, разделенную с одной или с обеих сторон одним или несколькими пробелами и не содержащую внутри себя пробелов, назовем словом. Распечатать все слова строки, отличные от первого слова и оканчивающиеся на букву "К". Предполагается, что в строке больше двух слов. Если таких слов нет, то выдать соответствующее текстовое сообщение.

Вариант 20

1. Написать программу согласно следующему условию: Дана квадратная матрица A порядка N, где N – заданное натуральное число, состоящая из натуральных чисел, больших 1. Строку с индексом i (i=l, 2, ..., N) назовем отмеченной, если все элементы в этой строке, расположенные на побочной диагонали и ниже нее являются простыми числами. Найти количество отмеченных строк в матрице A.

2. Написать программу согласно следующему условию: Дана строка символов, содержащая четное количество символов. Группу символов, разделенную с одной или с обеих сторон одним или несколькими пробелами и не содержащую

внутри себя пробелов, назовем словом. Вывести на экран слова, полностью расположенные во второй половине строки. Если таких слов нет, то выдать соответствующее текстовое сообщение.

Вариант 21

1. Написать программу согласно следующему условию: Дана квадратная матрица A порядка N, где N - заданное натуральное число, состоящая из натуральных чисел, больших 1. Строку с номером i (i=l,2, ..., N) назовем отмеченной, если элемент, расположенный в i-ой строке на побочной диагонали, является простым числом. Найти, максимальное значение среди всех элементов, расположенных в отмеченных строках матрицы A. Если таких строк нет, то выдать на печать соответствующее текстовое сообщение.

2. Написать программу согласно следующему условию: Дана строка символов. Группу символов, разделенную с одной или с обеих сторон одним или несколькими пробелами и не содержащую внутри себя пробелов, назовем словом. Распечатать самое длинное слово, предварительно перенеся первый символ в конец слова. Предполагается, что слово с наибольшим количеством символов единственное.

Вариант 22

1. 1. Написать программу согласно следующему условию: Дана квадратная матрица A порядка N, состоящая из натуральных чисел, больших 1, где N - заданное натуральное число. Строку с номером i (i$^{=}$l,2, ..., N) назовем отмеченной, если элемент, расположенный в i-ой строке на побочной диагонали является кратным трем и оканчивается на цифру 7. Найти максимальное значение среди всех элементов, расположенных в отмеченных строках. Если таких строк нет, то выдать соответствующее текстовое сообщение.

2. Написать программу согласно следующему условию: Дана строка символов. Группу символов, разделенную с одной или с обеих сторон одним или несколькими пробелами и не содержащую внутри себя пробелов, назовем словом. Выяснить

какое слово встречается раньше в строке - на букву "К" или на букву "Л". Предполагается, что в строке есть слова, начинающиеся как на букву "К", так и на букву "Л".

Вариант 23

1. Написать программу согласно следующему условию: Дана целочисленная квадратная матрица A порядка N, где N - заданное натуральное число. Строку с номером i (i=1,2, ..., N) назовем отмеченной, если элемент, расположенный в i-ой строке на побочной диагонали больше нуля и оканчивается на цифру 7. Найти максимальное значение среди всех элементов, расположенных в отмеченных строках матрицы A. Если таких строк нет, то выдать соответствующее текстовое сообщение.

2. Написать программу согласно следующему условию: Дана строка символов. Группу символов, разделенную с одной или с обеих сторон одним или несколькими пробелами и не содержащую внутри себя пробелов, назовем словом. Выяснить какое слово встречается раньше - начинающееся или оканчивающееся на букву "К". Предполагается, что в строке есть слова как начинающиеся на букву "К", так и оканчивающееся на букву "К" и нег слов начинающихся и оканчивающихся на букву "К".

Вариант 24

1. Написать программу согласно следующему условию: Дана действительная матрица A размером MxN, где M,N - заданные натуральные числа. Назовем элемент A(I,J) (1=1,2,..., M, J=1,2,...,N) особым, если A(I,J) больше суммы остальных элементов, расположенных в i-ой строке и A(I,J) больше суммы остальных элементов, расположенных в j столбце (1=1,2,..., M, J=1,2,...,N). Найти среднее арифметическое значение особых элементов. Если их нет, то выдать соответствующее текстовое сообщение.

2. Написать программу согласно следующему условию: Дана строка символов. Группу символов, разделенную с одной или с обеих сторон одним или несколькими пробелами и не

содержащую внутри себя пробелов, назовем словом. Распечатать все слова, оканчивающиеся группой букв "КОЙ". Если таких слов нет, то выдать соответствующее текстовое сообщение.

Вариант 25

1. Написать программу согласно следующему условию: Дана действительная квадратная матрица A порядка N, где N - заданное натуральное число. Назовем элемент A(I,J) особым, (i=l,2,... N; j=l,2,...,N), если сумма элементов в j строке равна 0, а сумма элементов в i столбце больше нуля. Найти количество особых элементов в матрице A.

2. Написать программу согласно следующему условию: Дана строка символов. Группу символов, разделенную с одной или с обеих сторон одним или несколькими пробелами и не содержащую внутри себя пробелов, назовем словом. Вывести на экран слова, начинающиеся со слога "ПО". Если таких слов нет, то выдать соответствующее текстовое сообщение.

Вариант 26

1. Написать программу согласно следующему условию: Дана действительная матрица A размером MxN, где M,N - заданные натуральные числа, а все элементы матрицы различные. Сформировать одномерный массив B, где B(j) равно среднему арифметическому значению наибольшего и наименьшего элементов в j-ом столбце (j=l,2, ..., N).

2. Написать программу согласно следующему условию: Дана строка символов. Группу символов, разделенную с одной или с обеих сторон одним или несколькими пробелами и не содержащую внутри себя пробелов, назовем словом. Распечатать все слова четной длины. Если таких слов нет, то выдать соответствующее текстовое сообщение.

Вариант 27

1. Написать программу согласно следующему условию: Дана матрица A размером MxN, где M,N - заданные натураль-

ные числа, состоящая из натуральных чисел, больших 1. Составить одномерный массив B, в котором элемент B(J) (j=1,2, ..., N) равен j, если j-ый столбец содержит только числа, кратные 5, в противном случае B(J)=0.

2. Написать программу согласно следующему условию: Дана строка символов. Группу символов, разделенную с одной или с обеих сторон одним или несколькими пробелами и не содержащую внутри себя пробелов, назовем словом. Распечатать все слова нечетной длины, в которых есть буква T. Если таких слов нет, то выдать соответствующее текстовое сообщение.

Вариант 28

1. Написать программу согласно следующему условию: Дана целочисленная матрица A размером MxN, где M,N - заданные натуральные числа. Сформировать одномерный массив B, где B(i) равно сумме элементов, кратных трем в i-ой строке матрицы i=l,2, ..., N. Если таких элементов в i-ой строке нет, то B(i) равно среднему арифметическому значению элементов, расположенных в i-ой строке матрицы.

2. Написать программу согласно следующему условию: Дана строка символов. Группу символов, разделенную с одной или с обеих сторон одним или несколькими пробелами и не содержащую внутри себя пробелов, назовем словом. Распечатать самое короткое симметричное слово. Если такого нет, то выдать соответствующее текстовое сообщение.

Контрольные вопросы

1. Элементы 5-элементного массива нумеруются начиная с _____ и до _____.
2. Имя массива, используемое в файлах исходного кода, представляет собой ____ массива.
3. Что определяет это выражение: `employee list [100];` ?
4. Строка в C++ – это ____ типа _____.
5. Напишите фрагмент программы, определяющий сум-

му элементов целочисленного массива, кратных пяти.

6. Напишите фрагмент программы, осуществляющий вывод элементов целочисленного массива с чётными индексами.

7. Напишите фрагмент программы заполнения массива из пяти элементов цифрами заданного пятиразрядного числа.

8. Напишите фрагмент программы, определяющий сумму тех элементов целочисленного массива, индексы которых совпадают со значениями соответствующих элементов массива.

9. Напишите фрагмент программы, осуществляющий вывод массива из 100 элементов по 10 элементов в строке.

10. Напишите фрагмент программы, осуществляющий циклический сдвиг элементов массива на одну позицию влево.

11. Напишите фрагмент программы, осуществляющий вывод элементов целочисленного массива, индексы которых являются полными квадратами: (1, 4, 9, 16, 25,...).

В случае если приведённые вопросы вызывают затруднения, рекомендуется ознакомиться с [4, 5, 7].

5. ОБЩИЕ РЕКОМЕНДАЦИИ ПО ВЫПОЛНЕНИЮ КУРСОВОЙ РАБОТЫ

По существу для написания курсовой работы необходимо выполнить *три функции*. Первая – составить алгоритмы разрабатываемых функций в виде блок-схем. Вторая - перевести составленные алгоритмы на язык программирования С++. И, наконец, третья и самая сложная – произвести отладку и тестирование разработанных функций.

При отладке функций считается, что написать достаточно сложную программу без ошибок попросту невозможно, поэтому предлагаемые в качестве курсовых работ задания, не являются очень сложными. Однако для начинающих программировать студентов они представляют определенную сложность. Так что ошибки неизбежны.

Для снижения вероятности допущения ошибок и облегчения их последующего устранения (если они все же были допущены) следует с самого начала придерживаться следующих принципов:

1) Необходимо тщательно продумывать алгоритм. Надо стремиться к тому, чтобы алгоритм был как можно более понятным. Во-вторых, данную программу следует писать с использованием необходимых по размеру функций и по отдельности отлаживать каждую маленькую функцию (сложность отладки при увеличении размера функции резко возрастает).

2) Для проверки алгоритма следует его «проигрывать» на бумаге, задавшись подходящими (для начала как можно более простыми) исходными данными.

3) Для повышения надежности, данные в функции следует стараться передавать без использования глобальных переменных.

4) Переменным и функциям желательно давать имена, их которых понятно их предназначение. Для не знающих английского языка совет такой: пишите по-русски, но латинскими буквами. Это позволит лучше ориентироваться в тексте программы. Для этой же цели систематически используйте поясняющие алгоритм комментарии и используйте отступы, для подчеркивания ее структуры операторов. Следует иметь в виду: написанный исходный код программы забывается очень быстро, и без соблюдения указанных мер по прошествии некоторого времени даже самому автору бывает трудно в нем разобраться.

5) При разработке программы (функций), из которой должны будут вызываться другие функции, на промежуточном этапе проектирования можно использовать «заглушки», т.е. функции, которые выдают некоторый постоянный результат, пригодный для тестирования вызывающей программы (функции). Обычно «заглушки» состоят всего из одного или нескольких операторов присваивания, и поэтому можно быть вполне уверенными в отсутствии в них ошибок. При этом использование таких функций позволяет реализовывать техноло-

гию программирования «сверху вниз».

Все ошибки, возникающие при программировании, можно разделить на следующие три группы. 1 – синтаксические ошибки, т. е. ошибки вызванные нарушением формальных правил, предусмотренных в языке программирования. Такие ошибки не страшны, поскольку тут же обнаруживаются компилятором и легко могут быть исправлены. 2 – ошибки, связанные, с неправильным переводом алгоритма на язык программирования, при отсутствии нарушений формальных синтаксических правил языка программирования. Обычно причина таких ошибок в невнимательности программиста, их причиной может также стать нетвердое знание базовых конструкций языка программирования. 3 – алгоритмические ошибки, т.е. ошибки, допущенные при разработке алгоритмов.

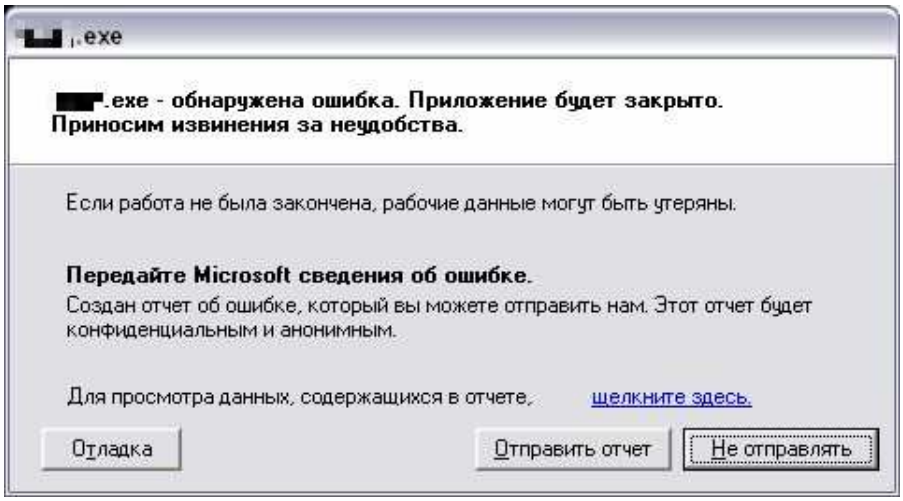
Рассмотрим пример исходного кода с ошибкой:

```
#include <iostream>
#include <conio.h>

using namespace std;

int main()
{
    int d=100;
    int r=d/(d-d);
    getch ();
    return 0;
}
```

Исправить ошибки второго вида обычно уже значительно сложнее, поскольку их обнаружение сопряжено с задачей статического анализа кода, тем не менее, большинство современных компиляторов предупреждает о потенциальных ошибках, позволяя при этом проигнорировать предупреждения. Существуют также специальные утилиты (например: Cppcheck, PVS-Studio), выполняющие статический анализ кода. Так, для приведённого выше кода, компилятор GNU GCC выдаст два предупреждения с указанием строки, вызвающей подозрения:

«warning: division by zero» (деление на ноль) и «unused variable 'r'» (неиспользуемая переменная 'r'). Аналогичные комментарии даёт и утилита Cppcheck, указывая, кроме того, на потенциальную логическую ошибку в вычитании (d - d). Обнаружение и исправление ошибок третьего вида полностью лежит на программисте. Стоит отметить, что программы могут быть собраны в двух режимах: *debug* и *release*. В режиме debug программа будет собрана с дополнительным отладочным кодом (узнать текущий режим сборки в коде программы, можно проверив макроопределение _DEBUG[2]), что сделает её поведение более документированным, но её бинарный файл избыточно (для отлаженной версии) больше. Разные режим сборки могут иметь разные конфигурации в рамках одного проекта. В зависимости от ИСР (или её настроек), программа, запущенная из среды разработки может запускаться с включённым отладчиком (*debugger*) (что более целесообразно для debug-сборок), или отладчик будет запускаться отдельной командой. Если в программе допущены ошибки второго и третьего вида, то последствия могут быть такими:

1. Программа досрочно прекратила свою работу с выдачей сообщения о произошедшей ошибке времени выполнения (рассмотрим на примере приведённого выше кода).



[2] Имя зависит от компилятора. Для некоторых компиляторов, вероятно, придется искать аналоги приведённых здесь директив и макрокоманд, но, в большинстве случаев это не понадобится.

Можно вызвать установленный в системе отладчик по умолчанию и попытаться отладить программу в нём.

Данная попытка не всегда (в частности, если для сборки приложения использовался другой компилятор) укажет на место сбоя в достаточно интуитивном формате (тем не менее, видно, что дизассемблер указывает на сбой при выполнении деления чисел с учётом знака [9]):

```
0040133A   idiv        eax,ecx
0040133C   mov         dword ptr [esp+0Ch],eax
00401340   call        00414338
00401345   mov         eax,0
```

2. Программа работает бесконечно долго (иногда это трудно отличить от просто долгой работы), в этом случае говорят, что программа «зависла» («не отвечает», «впала в бесконечный цикл»). Это всегда связано с тем, что по какой-либо причине где-то в программе условие завершения некоторого цикла никогда не выполняется. Причины этого могут быть самые разнообразные, однако выявить их обычно бывает несложно.

3. Программа корректно (без каких-либо сообщений об ошибках) завершила работу, однако выданный результат оказывается не тем, который ожидается. Это может быть связано с ошибками в самом алгоритме или с неверным переводом этого алгоритма.

4. При разных исходных данных (или, например, при использовании генератора псевдослучайных чисел) программа

ведет себя то одним, то другим образом.

Во всех перечисленных случаях необходим тщательный анализ программы. Часто, когда в коде программы никаких видимых недочетов не обнаруживается, причина ошибки кроется в неправильно составленном алгоритме. Также часто бывает, что допущенных ошибок сразу несколько, и они имеют разный характер.

Отладка программы – это трудоемкий и сложный этап ее разработки. Существуют некоторые стандартные приемы отладки программ. Обычно, для того чтобы понять причину неправильный работы программы требуется не только знать результат ее работы, но и проследить значения, которые принимают переменные, используемые в ней, и вывод значений которых не предусматривается. Бывает полезно также прослеживать динамику изменения таких переменных, находящихся внутри циклов.

Для этого интегрированная среда программирования предусматривает специальные средства (отладчик). К таким средствам относятся[3]: возможности пошагового исполнения программы без входа внутрь функций или с входом в них, возможность задания точек останова (breakpoints) и выполнения части программы до очередной такой точки, а также окна просмотра значений переменных, состояний регистров процессора, стека вызовов; окно дизассемблера, окно точек останова, окно просмотра памяти и потоков. Отладчик может указать на строку в исходном коде, выполнение которой привело к краху приложения:

```
 8   using namespace std;
 9
10   int main()
11   {
12       int d=100;
13       int r=d/(d-d);
14       getch ();
15       return 0;
16   }
```

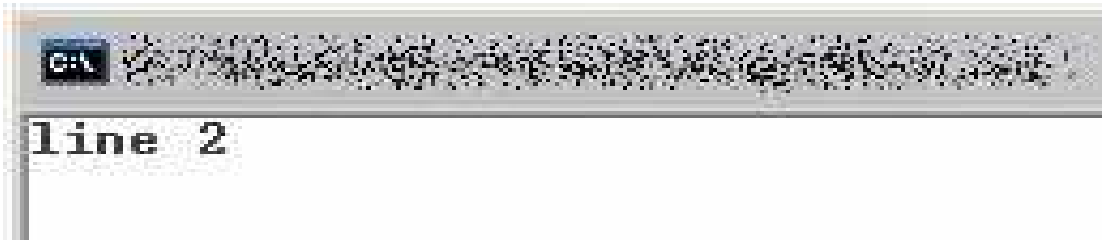
Кроме этого, иногда для этих же целей бывает удобно на

[3] Возможности могут отличаться в разных отладчиках.

время отладки вставлять в программу в нужных местах операторы вывода на экран. значений переменных, представляющих интерес. Этот вывод удобно сопровождать специальным сообщением о том, значения каких переменных выводятся в данный момент. При этом иногда удобно эти операторы вывода помещать внутрь некоторых условных операторов, что позволяет осуществлять вывод только при выполнении требуемых условий. Кроме того, повысить информативность отладочной информации может использование макрокоманды `__LINE__`[4], позволяющей узнать номер строки в исходном коде и директивы `#line`, позволяющей установить текущее значение счётчика строк:

```
    int main()
    {
        #line 1
//счётчик строк = 1
        int d=100;                          //1
        cout<<"line "<<__LINE__<<endl;  //2
        int r=d/(d-d);
        getch ();
        return 0;
    }
```

В ходе выполнения, программа выведет:



После отладки программы все эти конструкции можно не удалять из текста программы, а закомментировать или управлять включением отладочного кода при помощи директив пре-

[4] Помимо этой, могут быть полезны макрокоманды `__FILE__` (имя компилируемого файла с исходным кодом), `__DATE__` (дата компиляции), `__TIME__` (время компиляции) и другие.

процессора.

Важное предупреждение. Бывает, что начинающие программировать, отчаявшись найти ошибку в своей программе, поддаются искушению думать, что это не ими допущена ошибка, а ошибается компьютер или компилятор. В действительности вероятность такого события ничтожна мала. Тем не менее, в дальнейшем, при разработке более сложного программного обеспечения, такая ситуация может возникнуть. Например, при использовании beta-версий библиотек (такие ошибки, как артефакты при работе графического движка (graphics engine) на некоторых моделях видеокарт), эксплуатации недокументированных возможностей программных средств, более тесном взаимодействии с API операционной системы и аппаратной частью (что накладывает некоторый отпечаток на переносимость кода). Если вы окажитесь в такой ситуации, то, во-первых, постарайтесь локализовать проблему, составив минимальный по объёму код, доступно демонстрирующий ошибку. Во-вторых, произведите поиск информации в литературе и Интернете. В-третьих, проведите тестирование на разных ПЭВМ, составьте подробное описание проблемы и свяжитесь с разработчиками используемых вами программных средств (к этому моменту вы должны достаточно детально разобраться в проблеме, чтобы конкретизировать источник сбоя).

В качестве задач для тренировки умения отлаживать программы, рекомендуем [11].

В рамках дальнейшего знакомства с описанием языка C++ «из первых рук» рекомендуем [6]. В этой же книге вы найдёте советы по программированию. Для ознакомления с примерами решения типовых задач, шаблонами проектирования – *паттернами* (*design pattern*), обратитесь к [8].

Требования к оформлению курсовой работы

Курсовая работа должна быть подготовлена как документ Microsoft Word. Этот документ должен быть набран шрифтом Times New Roman размером 14пп через один интервал, с полями страницы: верхнее - 2,5пп, нижнее - 2пп, левое - 2пп, правое

-1,5пп. Каждый новый абзац должен начинаться с красной строки с отступом 1,25пп. Образец титульного листа приведен на последней странице данного пособия. В остальном документ должен быть отформатирован в соответствии с обычными требованиями к оформлению текстов научно-технического содержания.

Оглавление, заголовки разделов, нумерация страниц текста должны создаваться имеющимися стандартными средствами Microsoft Word. Курсовая работа должна быть представлена как в электронном виде, так и в печатном. В электронном виде должны быть представлены все работающие программы исходных текстов и откомпилированных исполняемых файлов.

При составлении блок-схем стоит опираться на ГОСТ 19.701-90 «Схемы алгоритмов, программ, данных и систем. Условные обозначения и правила выполнения». Сами схемы рекомендуется строить в векторных графических редакторах, таких как: MS Visio, InkScape, CorelDraw и т.п.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Бей И. Взаимодействие разноязыковых программ. Руководство программиста – М.: Издательский дом "Вильямс", 2005. – 880 с.: ил. эл. опт. диск (CD-ROM).

2. Ахо А. и др. Компиляторы: принципы, технологии и инструментарий, 2-е изд.: Пер. с англ. - М.: ООО "И.Д. Вильямс", 2008. - 1184 с. : ил. - Парал. тит. англ.

3. Мартынов Н.Н., Иванов А.П. MATLAB 5.x Вычисление, визуализация, программирование – М.: КУДИЦ-ОБРАЗ, 2000. – 336 с.

4. Лафоре Р. Объектно-ориентированное программирование в C++. 4-е издание — СПб.: Питер, 2005. — 924 с.

5. Фридман А.Л. Основы объектно-ориентированного программирования на языке Си++. Изд. 2-е перераб. и доп. - М.: Горячая линия - Телеком, 2001. - 232 с.. ил.

6. Бьерн Страуструп Язык программирования C++. Спе-

циальное издание. Пер. с англ. — М.: Издательство Бином, 2011 г. — 1136 с: ил.

7. Франка П. С++: Учебный курс: Учебное пособие — СПб.: Питер, 2003. — 521 с.

8. Гамма Э. и др. Приемы объектно-ориентированного проектирования. Паттерны проектирования Изд.: Питер, 2010 г.

9. Крупник А. Изучаем ассемблер. — СПб.: Питер, 2005. — 249 с.

10. Рихтер Дж. Windows для профессионалов: создание эффективных Win32 приложений с учетом специфики 64-разрядной версии Windows/Пер. с англ – 4-е изд. – Питер, Русская Редакция. 2001. – 752 с.

11. Уэллин С, Как не надо программировать на С++, — СПб.: Питер, 2004 г. — 240 с.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ 3

1. СОДЕРЖАНИЕ КУРСОВОЙ РАБОТЫ 4

2. СТРУКТУРА ПРОГРАММЫ.....6

3. ЧИСЛЕННЫЕ МЕТОДЫ. ПРОЦЕДУРЫ И ФУНКЦИИ. 10

4. МАССИВЫ, СТРОКОВЫЕ И СИМВОЛЬНЫЕ

ПЕРЕМЕННЫЕ 14
5. ОБЩИЕ РЕКОМЕНДАЦИИ ПО ВЫПОЛНЕНИЮ КУРСОВОЙ РАБОТЫ.....28
БИБЛИОГРАФИЧЕСКИЙ СПИСОК 36
ПРИЛОЖЕНИЕ 1 39
ПРИЛОЖЕНИЕ 2 40

ПРИЛОЖЕНИЕ 1

Пример оформления титульного листа

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ
РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО
ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ «МОСКОВСКИЙ
ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
РАДИОТЕХНИКИ, ЭЛЕКТРОНИКИ И АВТОМАТИКИ»
МГТУ МИРЭА

ФАКУЛЬТЕТ КИБЕРНЕТИКИ

КАФЕДРА АВТОМАТИЧЕСКИХ СИСТЕМ

КУРСОВАЯ РАБОТА

ПО ДИСЦИПЛИНЕ «ИНФОРМАТИКА»

Тема: <Название темы>

Студент: <Ф.и.о. студента> Группа: <Шифр группы> Руководитель: <Ф.и.о. преподавателя>

Москва <год>

ПРИЛОЖЕНИЕ 2

Пример оформления конверта диска