Министерство образования и науки
Российской Федерации

Московский государственный университет геодезии и картографии

Заблоцкий В.Р.

Базовые понятия языка C/C++

Методическое пособие по курсу
«Информатика и программно-алгоритмические языки»

для студентов первого курса все специальностей

Москва, 2004 г.

УДК 519.6; 519.68; 681.324

Заблоцкий В.Р.

Базовые понятия языка C/C++
Методическое пособие по курсу
«Информатика и программно-алгоритмические языки»
М.: МИИГАиК, 2004, 56с.

Методическое пособие является руководством по автоматизированному зачету для курса «Информатика и программно-алгоритмические языки (ПАЯ)», который читается автором студентам МИИГАиК по специальности «Исследование природных ресурсов авиа космическими средствами». С целью автоматизации процесса проверки усвоения студентами знаний основ языка C/C++ была разработана программа ТЕСТ. Программа предлагает 30 вопросов по базовым понятиям языка C/C++, причем каждый вопрос имеет 5 вариантов ответа, среди которых надо выбрать правильный ответ. Тестируемый вводит ответ и переходит к следующему вопросу. В результате программа выводит количество правильных ответов, данных студентом и полученную оценку. Программа ТЕСТ реализована в двух версиях: в форме оконного интерфейса *Borland C++ ver3.1* и в форме оконного интерфейса *C++Builder6.0.*. Программу тестирования в среде *Borland C++Builder6.0.* разработал студент С.А.Степанов.

Методическое пособие рассмотрено и рекомендовано к изданию кафедрой вычислительной техники и автоматизированной обработки аэрокосмической информации и утверждено к изданию редакционно-издательской комиссией факультета прикладной космонавтики (протокол №46 от 01.04.2004).

Библиография: 15 назв., табл.9, рис.5.

Рецензенты:
Проф., к.т.н. Л.Н. Капранов, МИИГАиК, каф. ВТиАОИ
к.т.н. Е.Е. Дмитреева, ГУЗ, каф. информатики

1.Введение

Дисциплина «Информатика и программно-алгоритмические языки (ПАЯ)» является одной из базовых дисциплин инженерного цикла в МИИГАиК, посвященных программированию на языках высокого уровня. Дипломированные специалисты, занятые в области аэрофотогеодезии, исследования природных ресурсов аэрокосмическими средствами, должны обладать знаниями и практическими навыками по программированию вычислительных средств.

Цель дисциплины – обеспечить подготовку студентов современным методам программирования на языке высокого уровня C/C++ для решения различных инженерных и организационных задач.

Задачи дисциплины – подготовить студентов к использованию вычислительных средств, дать знания о принципах и методах программирования на языке C/C++, ориентированных на решение широкого круга вопросов: от инженерно-расчетных до организационно-управленческих. Дисциплина «Информатика и ПАЯ» базируется на таких разделах математики как системы счисления, теория алгоритмов, языки и грамматики. Данная дисциплина обеспечивает наряду с общеинженерными дисциплинами, преемственность знаний при переходе от общенаучных знаний к специальным учебным дисциплинам.

В результате изучения курса «Информатика и ПАЯ» студенты должны:

-знать синтаксис и грамматику языка программирования C/C++, их особенности, отличие от других языков программирования высокого уровня, а также области наиболее предпочтительного их использования;

-знать правила написания программ на языке C/C++, их редактирования, отладки и тестирования;

-уметь строить алгоритмы решения общеинженерных и научно-исследовательских задач;

-иметь представление об особенностях написания программ для решения прикладных задач фотограмметрии и дистанционного зондирования.

Данная программа курса реализуется в форме лекций, лабораторных и расчетно-графических работ. Контроль освоения студентами знаний осуществляется в форме контрольных работ, а также при проверке домашних заданий. Заключительной формой контроля являются зачеты и экзамены.

Дисциплина «Информатика и ПАЯ» изучается в течение первых трех семестров. Первый и второй семестры заканчиваются зачетами, третий – экзаменом.

С целью автоматизации проверки знаний студентами первого семестра основ языка C была разработана программа тестирования. В ходе тестирования испытуемый получает вопросы, на которые он должен ответить, выбрать правильный ответ из пяти предложенных. Программа подсчитывает количество правильных ответов и оценивает знания испытуемого.

2. Кратко о языках программирования C и C++

Язык программирования C был разработан Деннисом Ритчи (Dennis M. Ritchie) из Bell Laboratories для компьютера DEC PDP-11 в 1972г. Широкую известность C получил, когда на его основе была реализована операционная система UNIX. Публикация в 1978 г. книги Кернигана Б. и Ритчи Д. «Язык программирования Си» превратила язык фактически в стандарт или в то, что называется «традиционный» C. Эта книга считается одной из самых удачных публикаций по языкам программирования. Язык C определен стандартом

ANSI (Американский национальный институт стандартов) в 1989 г., в настоящее время определен новый стандарт на язык С ANSI 1999.

На языке С, базируется другой язык программирования С++, созданный в начале 1980-х годов Бьярном Страуструпом (Bjarne Stroustrup), работавшим в AT&T Bell Laboratories. С++ унаследовал все возможности языка С и дополнил их средствами манипулирования объектами. В 1997г. ANSI опубликовал окончательный вариант стандарта языка С++. Объективно-ориентированное программирование стало ключевой методологией программирования. В настоящее время практически все основные ОС и многие коммерческие программные продукты написаны на С и/или С++. Многие считают, и автор разделяет эту точку зрения, что сегодня наилучшей стратегией обучения является освоение языка С, а затем, на его базе, языка С++.

3. Как работать с данным пособием

Традиционно, язык программирования С/С++, считается сложным языком программирования. В этой связи автор пособия избрал игровой метод подачи материала (с помощью компьютерного тестирования), который должен способствовать скорейшему освоению основ языка С/С++. Используя программу тестируемый получает вопрос на который дается, обычно пять вариантов ответа. Требуется выбрать правильный вариант. Методическое пособие позволяет выполнить программу тестирования, поскольку предлагаемые вопросы рассматриваются в разделе коллоквиум.

Структура коллоквиума следующая. В первой части коллоквиума, именуемой вопросы и ответы, для каждого вопроса указан правильный вариант ответа с помощью значка радиокнопка включена  . Каждый вопрос сопровождается комментарием. Он служит для оперативного напоминания синтаксиса языка, его основных конструкций.

Во второй части коллоквиума, именуемой вопросы для самоконтроля, для каждого вопроса также дается правильный вариант ответа, но ответ представлен отдельно от вопроса в конце пособия. Однако вместе с вопросом тестируемому предлагается и вариант ответа (отмечен значком ☒), который может быть правильным или неправильным.

Такой способ проверки знаний позволяет сконцентрировать внимание на типичных ошибках программирования. Студенты, начавшие изучать язык обычно, делают некоторые ошибки наиболее часто. Если был выбран в качестве ответа предложенный вариант с ошибкой, необходимо внимательно проанализировать типичную ошибку программирования, чтобы избегать ее в дальнейшем. Автор настоятельно советует попытаться самостоятельно ответить на вопросы второй части коллоквиума, не заглядывая в ответы и лишь потом сверить результаты.

Для третьей части коллоквиума, именуемой вопросы для самостоятельного решения, ответы в пособии не представлены. Однако, усвоив базовые понятия и основные конструкции языка С/С++ по первой и второй части коллоквиума, у читателя не должно возникнуть существенных проблем в выборе правильных ответов на вопросы третьей части.

4. Программа тестирования основ языка С/С++

4.1 Общее описание

С целью автоматизации тестирования усвоения студентами знаний основ языка С/С++ была разработана программа ТЕСТ. Программа написана на языке С и

предназначена для выполнения в среде Windows. В основе программы лежит алгоритм случайного выбора вопросов, реализованных в виде отдельных функций. С помощью генератора случайных чисел формируется случайное число в диапазоне от 1 до 30. Далее в цикле одномерный массив заполняется случайными числами. Вызов функции, содержащий вопрос, выполняется с помощью конструкции переключателя switch.

Программа ТЕСТ предлагает 30 вопросов по базовым понятиям языка С/С++, включая такие темы как: имена переменных, типы и размеры данных, константы, объявления переменных, арифметические операторы, операторы отношения и логические операторы, операторы инкремента и декремента, побитовые операторы, операторы и выражения присваивания, приоритет и очередность вычислений, инструкции и блоки, а также основные управляющие конструкции типа if, if- else, переключатель switch, циклы while, for и do- while, инструкции break u continue.

Программа ТЕСТ реализована в двух версиях: в форме оконного интерфейса Borland C++ ver3.1/5.0 и в форме оконного интерфейса Borland C++ Builder 6. На рис.1. представлен оконный интерфейс первого вида, на рис.2. – оконный интерфейс второго вида.

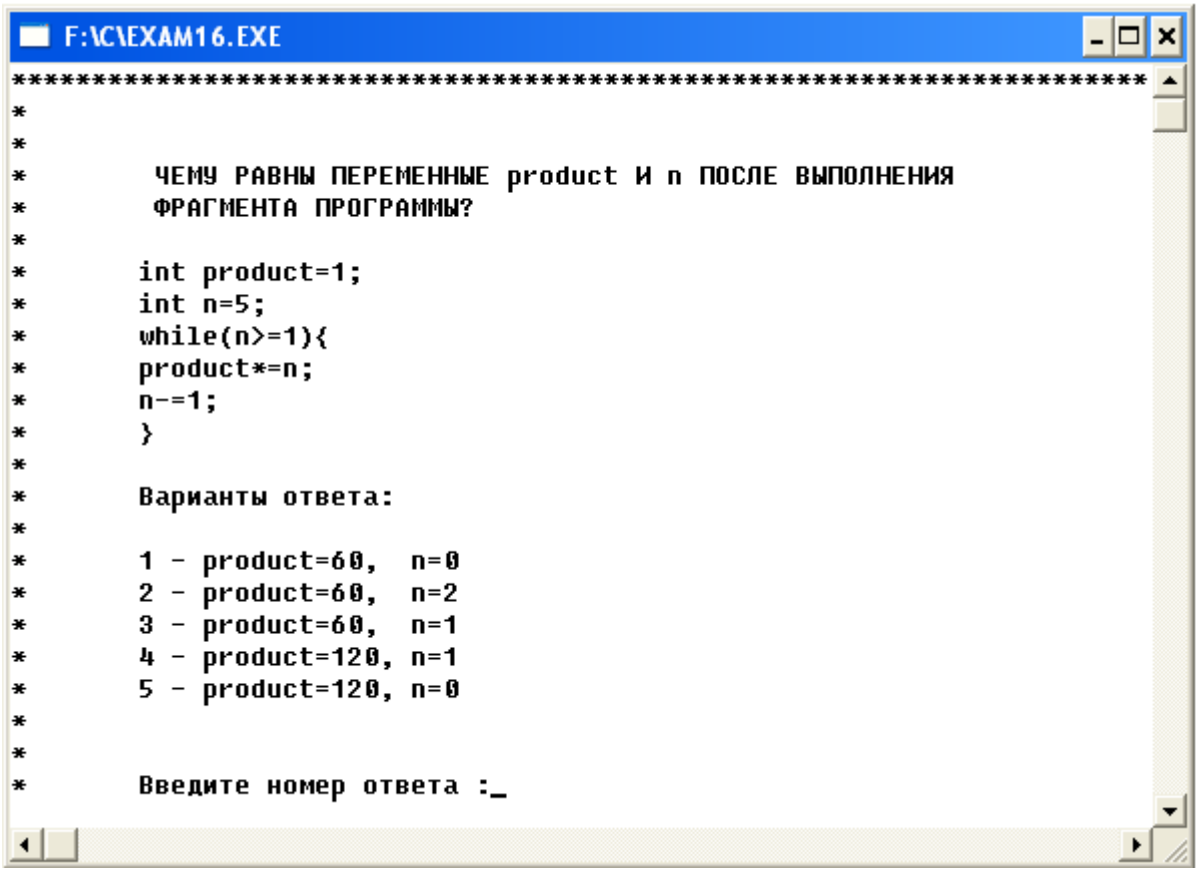


Рис.1. Оконный интерфейс программы тестирования в стиле Borland C++ver3.1.

Практически каждый вопрос содержит 5 вариантов ответа, среди которых находится правильный ответ. Тестируемый выбирает ответ и вводит его с клавиатуры, или, делая активной радиокнопку, и переходит к следующему вопросу. Правильный ответ фиксируется, и количество правильных ответов увеличивается на единицу.

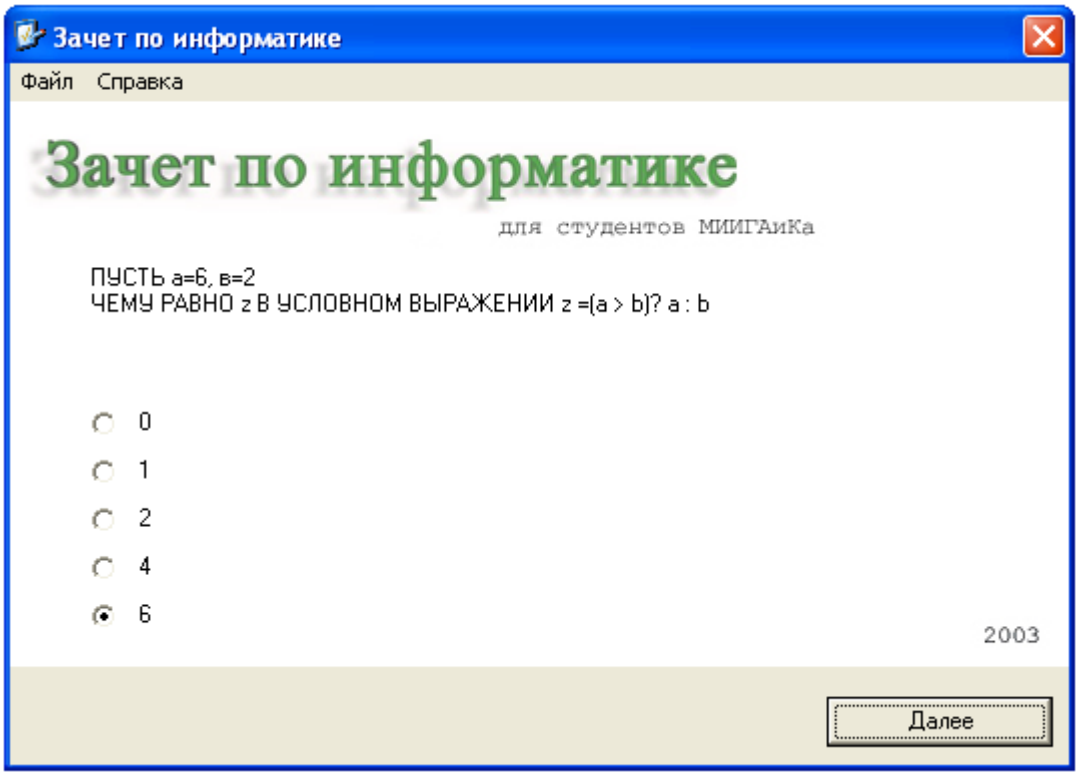


Рис.2. Оконный интерфейс программы тестирования в стиле Borland C++Builder6.0.

В результате тестирования программа выводит сообщение о количестве правильных ответов, введенным студентом и показывает количество полученных баллов. Это количество баллов оценивается по формуле floor((grade/6.0)+0.5) , т.е. имеет место следующая таблица 1 соответствия количества правильных ответов и полученных баллов.

Таблица 1

Количество правильных ответов	Полученный балл
от 27 до 30	Отлично
от 21 до 26	Хорошо
от 15 до 20	Удовлетворительно
Меньше 15	Неудовлетворительно

Несмотря на внешние отличия, сам алгоритм программы существенных изменений не претерпел.

4.2. Алгоритм программы

Перед написанием программы тестирования был спланирован подход к ее решению. В качестве основного метода решения были выбраны методы нисходящего и структурного программирования, традиционные для языка С. В начале задача была написана в общем виде на псевдокоде. Псевдокод – это искусственный неформальный язык, близкий к естественному, используемый для разработки алгоритма программы. Такой подход помогает продумать программу перед ее написанием. В качестве примера псевдокода для конструкции if –else напишем оператор псевдокода, используемый в дальнейшем в нашей программе тестирования:

Если оценка студента больше или равна 15
Вывести на экран “ Зачет! ”
иначе
Вывести на экран “ Незачет. Читайте книгу
Б. Кернигана и Д. Ритчи Язык программирования Си ”

При написании программы тестирования использовался метод нисходящего программирования, который позволяет осуществлять пошаговые уточнения программы и в итоге позволяет разрабатывать хорошо структурированные программы. Псевдокод для верхнего уровня нисходящего процесса разработки программы тестирования следующий:

Определить оценку студента за зачет.

Верхний уровень представляет собой одно предложение на псевдокоде, выражающее общее назначение программы. Дальнейшая разработка состоит из ряда уточнений. При этом верхний уровень подразделяется на набор более мелких задач, перечисляемых в том порядке, в котором они должны выполняться. В результате получается следующее первое уточнение.

Инициализировать переменные и массив
Вывести на экран 30 вопросов, сопровождая каждый возможными ответами.
Вычислить и вывести на экран оценку студента за зачет

При переходе ко второму уровню детализации оператор псевдокода

Инициализировать переменные и массив

может быть уточнен следующим образом.

Инициализировать оценку нулем.
Для каждого из 30 элементов массива
Сгенерировать случайное число, используемое в качестве индекса массива.
Каждому элементу массива присвоить очередной порядковый номер.

Такой алгоритм заполнения массива имеет одну важную особенность. Он лишен недостатка называемого неопределенной отсрочки. Простой алгоритм заполнения массива случайными числами может выполняться неопределенно долго, если случайные числа, которые уже выпали, продолжают выпадать в случайном выборе снова. Использованный нами алгоритм лишен этого недостатка.

Для оператора псевдокода

Вывести на экран 30 вопросов, сопровождая каждый возможными ответами.

потребуется цикл, в котором последовательно выводится вопрос и пять возможных ответов.

Уточнение предыдущего оператора псевдокода тогда будет

Для каждого из 30 элементов массива со случайным индексом
Переключателем вызвать функцию, которая выведет на экран вопрос и возможные ответы

Оператор псевдокода

Вычислить и вывести на экран оценку студента за зачет

может быть детализирован так:

Если выбран правильный ответ
функция возвращает единицу
иначе
возвращает нуль.
Если оценка студента больше или равна 15
Вывести на экран " Зачет! "
иначе
Вывести на экран " Незачет"

Таким образом на третьем уровне детализации алгоритм программы примет вид:

Инициализировать оценку нулем.
Для каждого из 30 элементов массива
Сгенерировать случайное число, используемое в качестве индекса массива.
Каждому элементу массива присвоить очередной порядковый номер.

Для каждого из 30 элементов массива со случайным индексом
Переключателем вызвать функцию, которая выведет на экран вопрос и возможные ответы
Если выбран правильный ответ
функция возвращает единицу
иначе
возвращает нуль.

Если оценка студента больше или равна 15
Вывести на экран " Зачет! "
иначе
Вывести на экран " Незачет"

При записи алгоритма на псевдокоде тело цикла отделяется отступами слева, это показывает, какие выполняемые операторы принадлежат циклу. Желательно, также включать в псевдокод пустые строки для повышения его читаемости. Пустые строки, фактически, разделяют алгоритм на разные этапы его работы.

Завершение процесса нисходящего пошагового уточнения (метод нисходящего программирования) завершается тогда, когда алгоритм на псевдокоде специфицирован (описан) достаточно подробно, чтобы можно было преобразовать псевдокод в программу на С. В нашем случае процесс детализации алгоритма можно завершить на третьем шаге

и приступить к структурному программированию. При данном подходе используются специальные блок-схемы.

Блок – схема является графическим представлением алгоритма программы или части программы. При рисовании блок-схем используются специальные геометрические фигуры, такие как прямоугольники, ромбы, овалы и т.д. Символ прямоугольника используется для обозначения любого действия, например вычисления или операции ввода/вывода. Символ ромба указывает на необходимость выбора (или принятия решения). Символы соединяются стрелками – линиями перехода, которые указывают на последовательность выполнения действий. При рисовании блок – схемы, представляющий полный алгоритм программы, первой графической фигурой является овал, содержащий слово “Начало”, последней фигурой также – овал, содержащий слово “Конец”. На рис.3 представлена простейшая блок – схема программы.

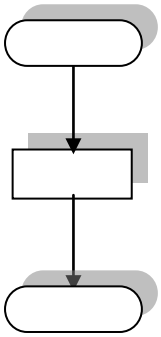
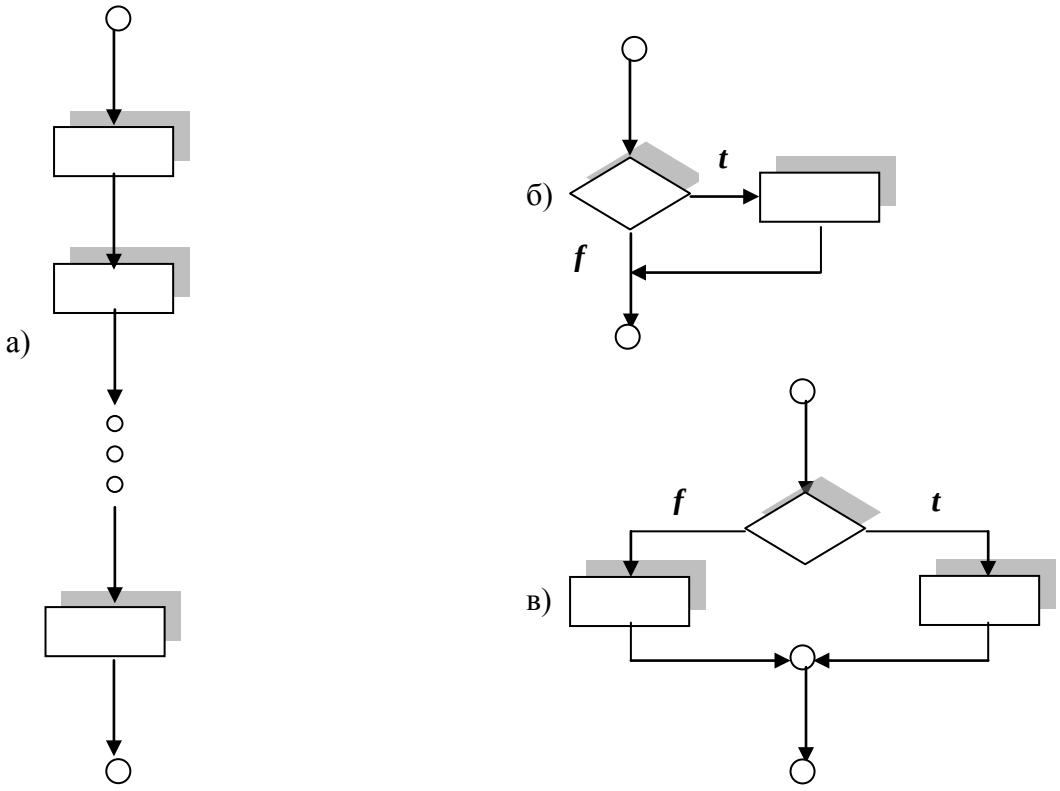


Рис.3. Простейшая блок – схема программы.

Структурированные (хорошо читаемые и понимаемые) программы создаются на основе ограниченного набора управляющих структур: линейной, выбора и повторения. На рис.4 приведены блок схемы управляющих структур языка С.



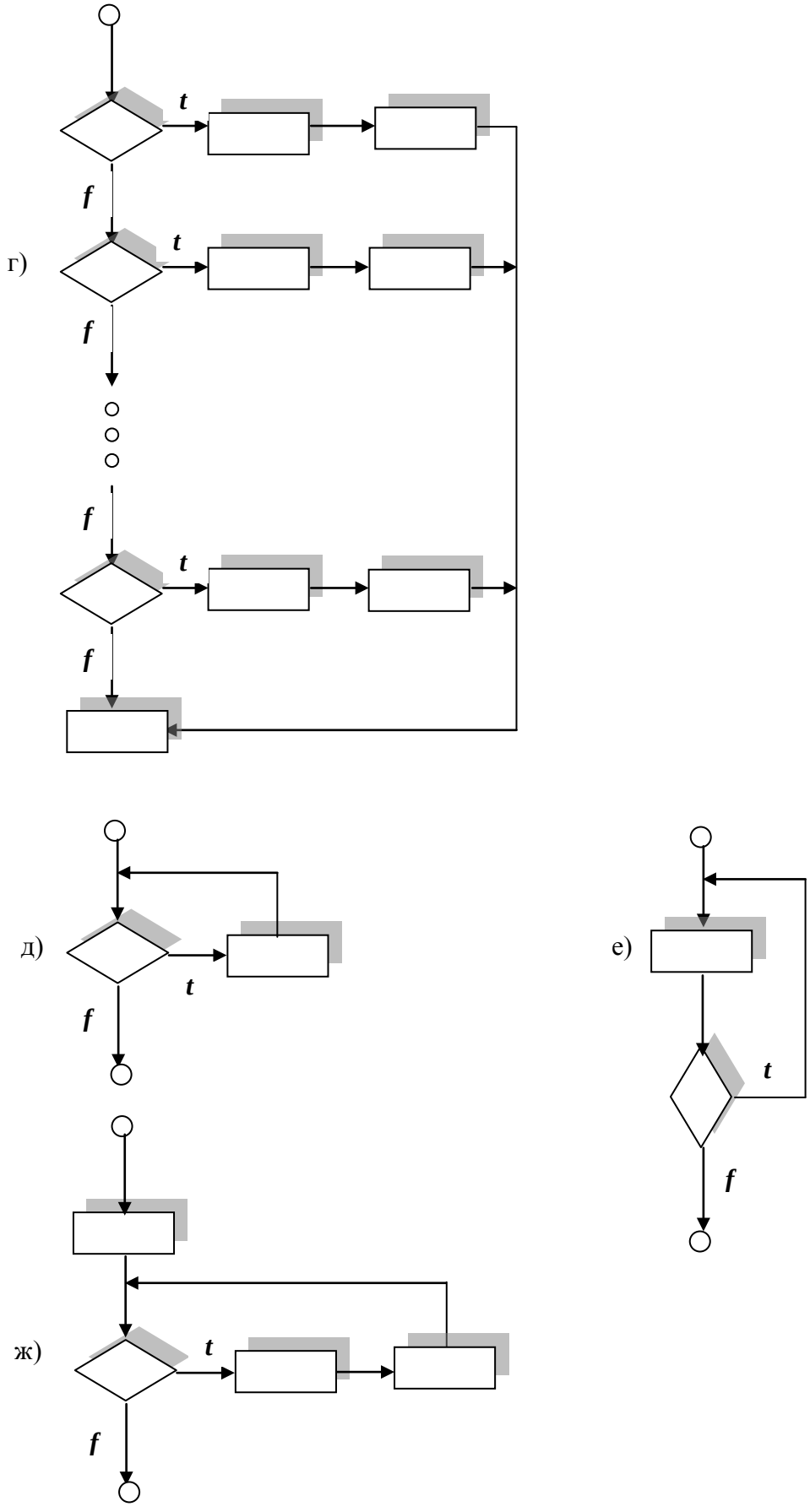


*Рис.4. Управляющие структуры языка С: линейная (а), выбора (б – **if**, в – **if –else**, г – **switch**) и повторения (д – **while**, е – **do-while**, ж - **for**).*

Соединять управляющие структуры можно только двумя простыми способами. Используются структуры с одним входом и одним выходом. При последовательном соединении управляющих структур точка выхода одной управляющей структуры соединяется с точкой входа следующей управляющей структуры. Таким образом, управляющие структуры просто помещаются одна за другой в программе. Второй способ соединения управляющих структур заключается во вложении управляющих структур друг в друга. Чтобы построить хорошо структурированную программу пользуются ограниченным набором управляющих структур.

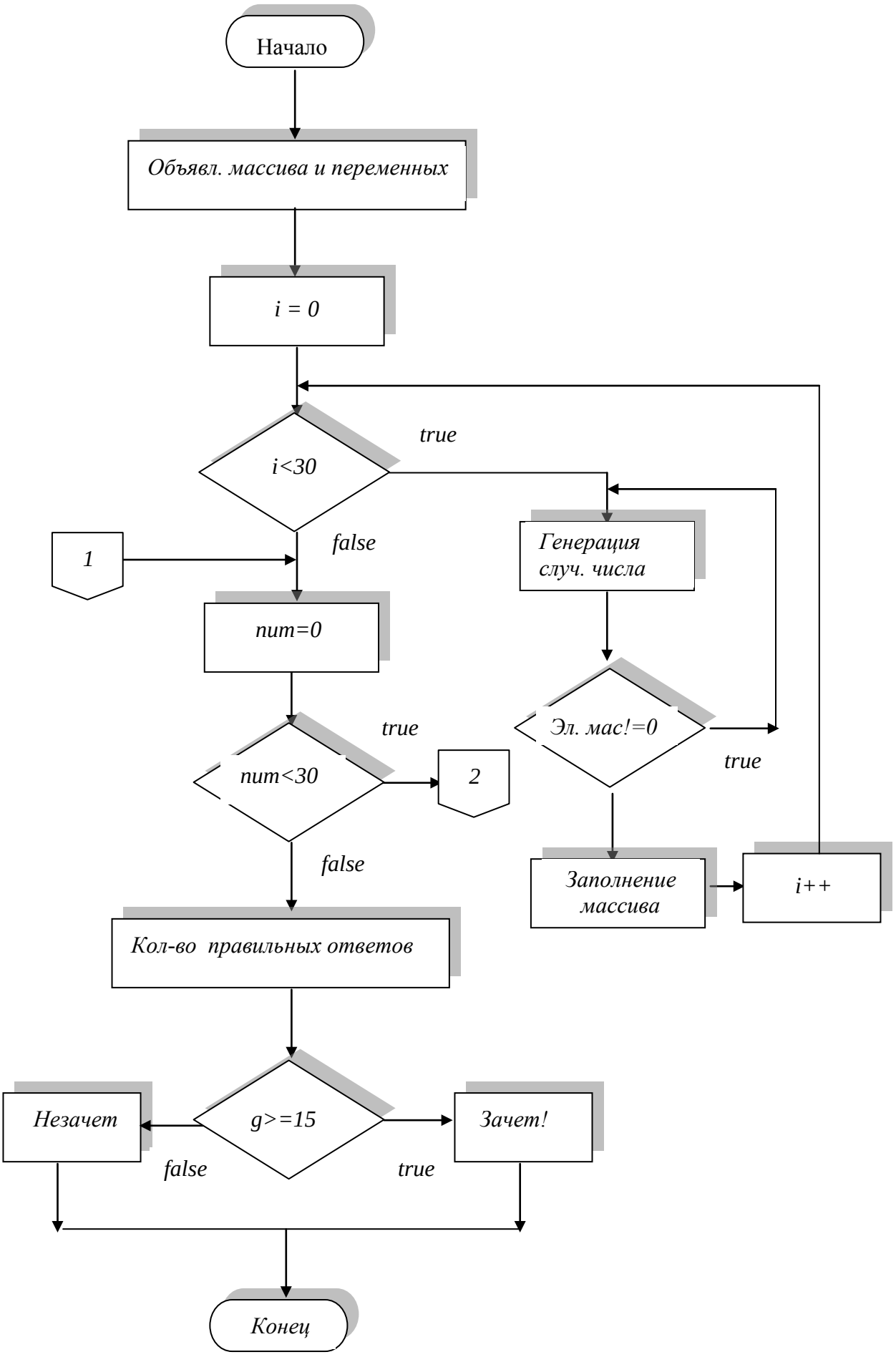
Алгоритм построения структурированных программ следующий:

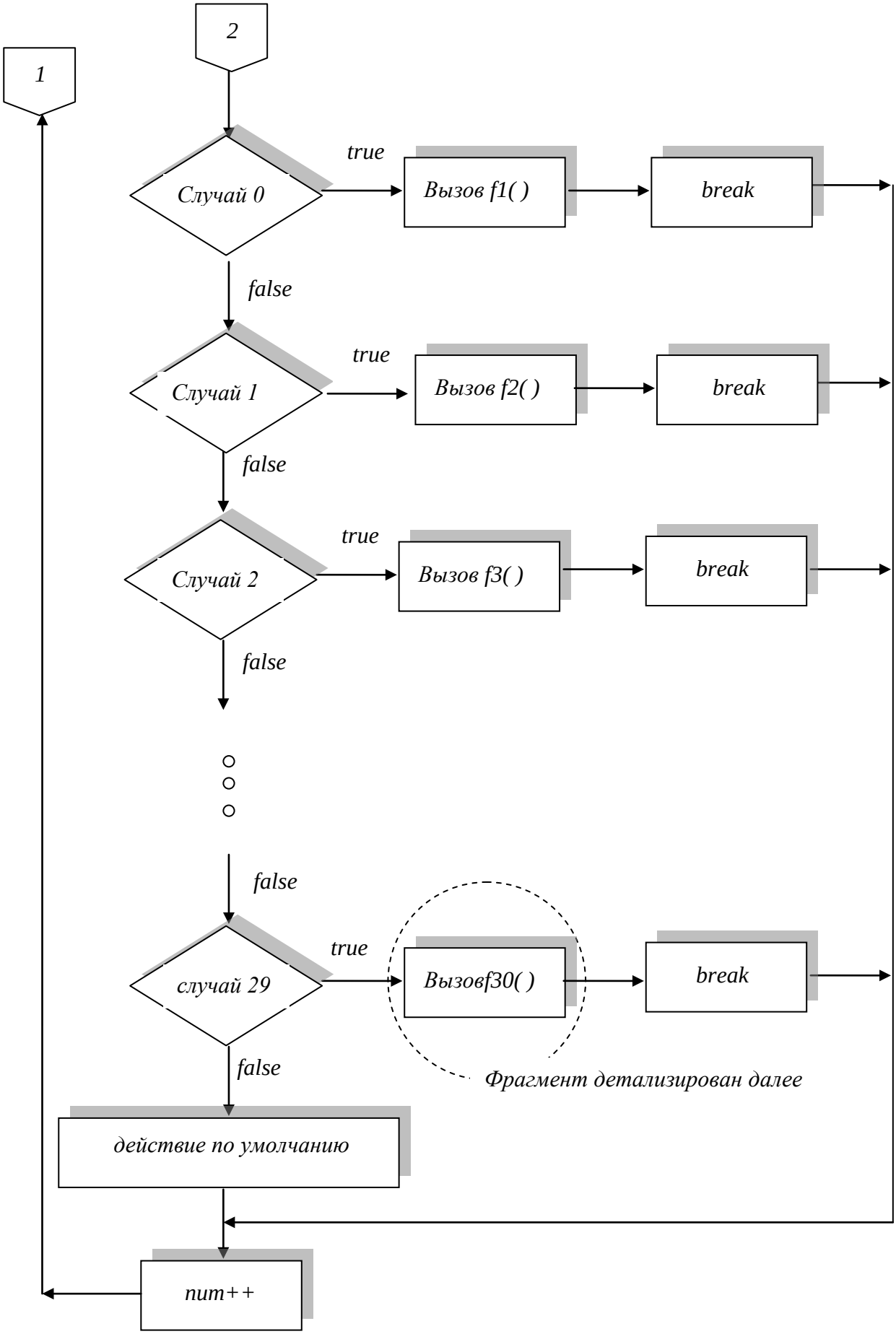
1. Начать создавать программу с простейшей блок – схемы (рис. 3).
2. Любой прямоугольник в блок – схеме можно заменить на два последовательных прямоугольника.
3. Любой прямоугольник в блок – схеме можно заменить на любую управляющую структуру (линейную, выбора, повторения).
4. Правила 2 и 3 можно применять в каком угодно порядке.

Таком образом любая программа на С может быть построена всего лишь из семи различных типов управляющих структур (линейной, трех структур выбора и трех структур повторения), объединенных одним из двух возможных способов. Структурное программирование явилось важным шагом в развитии технологии программирования, поскольку позволяло создавать относительно большие программные продукты в более короткий срок, с меньшим количеством ошибок и более простым и понятным кодом.

В настоящее время господствующей технологией программирования является объектно–ориентированное программирование, которое поддерживают многие современные языки программирования, в частности С++.

Автор считает, что студентам при изучении языка С/С++ полезно познакомится с принципами структурного программирования. Этим целям служит и разработанная в качестве учебных целей блок – схема программы ТЕСТ, которая приводится далее.


Начало
Объявл. массива и переменных
i = 0
i<30
true
false
Генерация случ. числа
1
num=0
Эл. мас!=0
true
num<30
true
2
false
Заполнение массива
i++
Кол-во правильных ответов
g>=15
Незачет
Зачет!
false
true
Конец


2
1
Случай 0
true
Вызов f1( )
break
false
Случай 1
true
Вызов f2( )
break
false
Случай 2
true
Вызов f3( )
break
false
false
случай 29
true
Вызовf30( )
break
false
Фрагмент детализирован далее
действие по умолчанию
num++

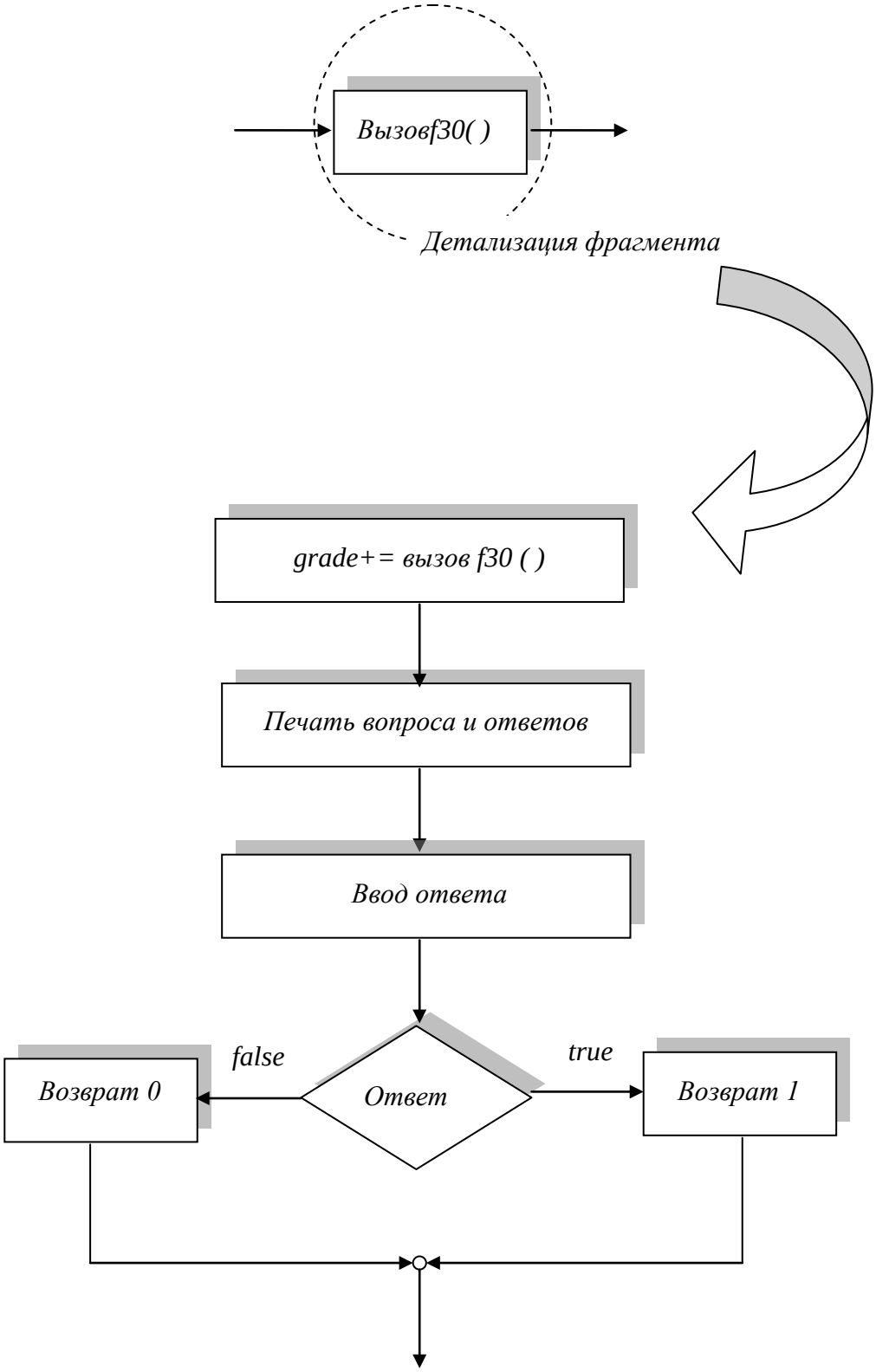


Рис.5. Укрупненная блок – схема программы ТЕСТ.

5.Коллоквиум

5.1. Вопросы с ответами

Вопрос №1
КАКОЙ ТИП ПЕРЕМЕННОЙ ПРЕДСТАВЛЕН КЛЮЧЕВЫМ СЛОВОМ "**int**"?
Варианты ответа:

- ○ вещественный с двойной точностью
- ○ символьный
- ○ вещественный с одинарной точностью
- ◉ целый

Комментарий. В языке С имеется несколько типов числовых переменных. Различные типы переменных используют разный объем памяти. Числовые переменные делятся на две основные категории: целочисленные переменные (**char**, **int**), в которых хранятся целые числа и переменные с плавающей точкой (**float**, **double**), в которых хранятся числа с дробной частью (т.е. вещественные или действительные числа).

На основе базовых типов данных **char** и **int** образуются другие (производные) типы данных с помощью модификаторов типа: **signed** (знаковые) и **unsigned** (беззнаковые). Целые со знаком, например **signed char** или **signed int**, могут хранить как положительные, так и отрицательные значения. Целые без знака, например **unsigned char** или **unsigned int** могут хранить только положительные значения и нуль. Другой модификатор **short** (короткое целое) применим только к типу **int**, модификатор **long** применим и к типу **int** (например, **long int**) и к типу **double** (например, **long double**). Тип **long double** предназначен для арифметики с плавающей точкой повышенной точности.

Вопрос №2
КАКОЕ ИМЯ ПЕРЕМЕННОЙ ЯВЛЯЕТСЯ НЕДОПУСТИМЫМ?
Варианты ответа:

- ○ Age
- ○ _Invalid
- ○ TotalIncome
- ◉ 1color
- ○ R123

Комментарий. Имена переменных в языке С подчиняются следующим требованиям. Имена могут содержать латинские буквы (от a до z и от A до Z), цифры (от 0 до 9) и символ подчеркивания (_). Первым символом имени должна быть буква. Допускается также применение знака подчеркивания, но желательно в начале имени его не использовать. Цифра (от 0 до 9) не может быть первым символом имени. Язык С чувствителен к регистру, т.е. имеет значение, являются ли буквы прописными или строчными, например имена count и Count обозначают две разные переменные. Ключевые слова С, как **if**, **else**, **int**, **float**, и т.д. нельзя использовать в качестве переменных. Во

многих компиляторах длина имени переменной не должна превышать 31 символа (имя может быть и длиннее, но компилятор анализирует из него только 31 символ, начиная с первого). Следует разумно выбирать имена переменных таким образом, чтобы они означали нечто, относящееся к назначению переменных.

Вопрос №3

СКОЛЬКО БАЙТ ПАМЯТИ ПРЕДОСТАВЛЯЕТСЯ ДЛЯ ПЕРЕМЕННОЙ ТИПА **float**?

Варианты ответа:

- ○ 8 байт
- ○ 2 байт
- ◉ 4 байт
- ○ 10 байт
- ○ 1 байт

Комментарий. На разных компьютерных платформах размеры типов данных (количество байт и диапазон значений) могут существенно отличатся. Однако всегда соблюдается следующий набор правил:

- длина переменной типа **char** всегда один байт,
- длина переменной типа **short** меньше или равна длине переменной **int**,
- длина переменной типа **int** меньше или равна длине переменной **long**,
- длина переменной типа **float** меньше или равна длине переменной **double**.

В таблице 2 представлены типы данных, определенные Стандартом С и принятые в системе Borland C++ ver3.1.

Таблица 2.

Тип	Типичный размер в байтах	Минимально допустимый диапазон значений		
char	1	от -128	до	+127
unsigned char	1	от 0	до	255
signed char	1	от -128	до	+127
int	2	от -32768	до	32767
unsigned int	2	от 0	до	65535
signed int	2	от -32768	до	32767
short int	2	от -32768	до	32767
unsigned short	2	от 0	до	65535
int	2	от -32768	до	32767
signed short int	4	от -2147483648	до	2147483647
long int	4	от 0	до	4294967295
unsigned long int	4	от -2147483648	до	2147483647
signed long int	4	от 1.2e-38	до	3.4e+38
float	8	от 2.2e-308	до	1.8e+308
double	10	от 3.4e-4932	до	1.2e+4932
long double				

Вопрос №4

КАКОЙ ДИАПАЗОН ЗНАЧЕНИЙ ИМЕЕТ ПЕРЕМЕННАЯ ТИПА **unsigned int**?

Варианты ответа:

○	0	-	255
○	-32768	-	+32767
○	-128	-	+127
◉	0	-	65535
○	-2147483648	-	+2147483647

Комментарий. Для разных версий языка C/C++ допустимые диапазоны значений переменных различных типов могут различать. В пособии рассматриваются диапазоны типов, принятые в системе Borland C++ ver3.1. Переменная **unsigned int** имеет диапазон от 0 до 65535. См. комментарий к вопросу №3.

Вопрос №5

КАКОЙ ДИАПАЗОН ЗНАЧЕНИЙ ИМЕЕТ ПЕРЕМЕННАЯ ТИПА **double**?

Варианты ответа:

○	0	-	4294967295
◉	2.2e-308	-	1.8e308
○	1.2e-38	-	3.4e+38
○	0	-	65535
○	-2147483648	-	+2147483647

Комментарий. Для разных версий языка C/C++ допустимые диапазоны значений переменных различных типов могут различать. В пособии рассматриваются диапазоны типов, принятые в системе Borland C++ ver3.1. Переменная **double** имеет диапазон от 2.2e-308 до 1.8e+308. См. комментарий к вопросу №3.

Вопрос №6

КАКАЯ КОНСТАНТА ЯВЛЯЕТСЯ СИМВОЛЬНОЙ?

Варианты ответа:

○	123L
○	"X"
◉	'B'
○	0x5A
○	1e-2

Комментарий. Символьная константа - это один символ, заключенный в одинарные кавычки, как, например, 'x'. Значением символьной константы является код этого символа. Например, в коде символов ASCII символьный нуль, или '0', имеет значение 48, очевидно что, это значение отлично от числа 0. Написание '0' вместо численного значения, такого как 48 делает программу не зависящей от конкретного численного представления этого символа в машине.

Вопрос №7
КАКАЯ КОНСТАНТА ЯВЛЯЕТСЯ ШЕСТНАДЦАТЕРИЧНОЙ?
Варианты ответа:

- 1616
- 1e+04
- 0773
- 0XFF
- "1AB"

Комментарий. В языке С разрешено при желании задавать целые константы в шестнадцатеричной системе счисления. Последовательность цифр: 0,1,2,…9,A,B,C,D,E,F перед которой записаны символы 0x или 0X считается шестнадцатеричной константой. Например, десятичное число 31 можно записать как 0X1F в шестнадцатеричной форме.

Вопрос №8
КАКАЯ КОНСТАНТА ЯВЛЯЕТСЯ ВОСЬМЕРИЧНОЙ?
Варианты ответа:

- 0x123
- 123
- 1E+8
- "8"
- 0123

Комментарий. Существует система обозначений для восьмеричных констант. Последовательность цифр, *начинающаяся с 0 и не содержащая десятичных цифр старше 7*, представляет собой восьмеричную константу. Например, десятичное число 31 можно записать как 037 в восьмеричной форме.

Вопрос №9
ЧТО ОЗНАЧАЕТ УПРАВЛЯЮЩАЯ ПОСЛЕДОВАТЕЛЬНОСТЬ ВИДА **\t**?
Варианты ответа:

- ОБРАТНАЯ НАКЛОННАЯ ЧЕРТА
- ГОРИЗОНТАЛЬНАЯ ТАБУЛЯЦИЯ
- НОВАЯ СТРОКА
- СИГНАЛ-ЗВОНОК
- ДВОЙНАЯ КАВЫЧКА

Комментарий. Чтобы представить большинство символьных констант, достаточно заключить соответствующий символ в одинарные кавычки. Но некоторые символы, например символ возврата каретки, требуют специального представления (знака). Имеются и другие неграфические символы, которые должны быть представлены

визуально. Для этих целей используются специальные символьные константы, часто их называют управляющими последовательностями или ESC – последовательностями. В таблице 3 приведены основные управляющие последовательности.

Таблица 3.

Обозначение	Назначение
\n	новая строка
\t	горизонтальная табуляция
\a	сигнал
\b	удаление предыдущего символа
\”	двойная кавычка
\’	одинарная кавычка
\?	знак вопроса

Хотя последовательность выглядит как два символа, на самом деле это один символ.

Вопрос №10

КАКАЯ КОНСТАНТА ЯВЛЯЕТСЯ СТРОКОВОЙ (СТРОКОВЫМ ЛИТЕРАЛОМ)?

Варианты ответа:

- '3'
- "stack"
- 045
- 1.2E+02
- 65.3

Комментарий. Строчная константа иначе строковый литерал - это последовательность, состоящая символов, заключенных в двойные кавычки. Кавычки не являются частью строки, а служат только для ее ограничения. Последовательность может содержать нуль символов, например "" - это нуль-строка.

Вопрос №11

НАЙТИ ПРАВИЛЬНОЕ ОБЪЯВЛЕНИЕ ПЕРЕМЕННЫХ

Варианты ответа:

- double step =1E5, eps =1.E-5;
- int lower =0. upper;
- float eps = 0123;
- DOUBLE lower = x0, EPS =1e+5;
- float EPS = 0x01 step = 200;

Комментарий. Прежде чем воспользоваться переменной в программе на языке С, эту переменную необходимо объявить. Объявление переменной сообщает компилятору ее имя и тип. В объявлении можно также инициализировать переменную, т.е. присвоить ей некоторое начальное значение. Правило (синтаксис) объявления переменной следующее:

тип имя (например **int x; float y; double z;**)

Здесь *тип* указывает тип переменной и должен быть одним из ключевых слов, *имя* – представляет имя переменной. Можно объявить несколько переменных одного и того же типа, разместив их на одной строке и отделив, их друг от друга запятыми, например **double x, y, z;** Правильное объявление переменных дано в первом варианте. Во втором варианте переменные под одним ключевым словом должны отделятся запятой, а не точкой. В третьем варианте переменной типа float присваивается целая восьмеричная константа – это некорректное действие. Остальные объявления неправильны по многим причинам, ключевое слово double записано заглавными буквами, вещественные переменные инициализируются целыми значениями, отсутствует запятая между EPS и step.

Вопрос №12
ЧЕМУ РАВНО ВЫРАЖЕНИЕ 3/4 ?
Варианты ответа:

- 1
- (•) 0
- 0.75
- 0,75
- 0.0

Комментарий. При делении целых дробная часть отбрасывается. Выражение ¾ это деление целых чисел, числитель равен трем, знаменатель четырем, поэтому результат равен нулю.

Вопрос №13
ЧЕМУ РАВНО ВЫРАЖЕНИЕ 1 % 2 ?
Варианты ответа:

- 50.0
- (•) 1
- 0
- 0,5
- 0.5

Комментарий. Арифметической операцией является операция деления по модулю %. Выражение **x % y** дает остаток от деления x на y и, следовательно, равно нулю, когда x делится на y точно. Этот оператор нельзя применять к типам данных с плавающей точной. При целочисленном делении ½ результатом будет 0, а сам остаток, т.е. результат 1%2 равен 1.

Вопрос №14
КАКОЕ ЛОГИЧЕСКОЕ ВЫРАЖЕНИЕ РАВНО НУЛЮ (ЛОЖЬ)?
Варианты ответа:

- 5 >= 5
- 6 > 5
- 6 == 6
- 5 > 6
- 5 != 6

Комментарий. Логическим выражением называется выражение, содержащее операции отношения и/или логические операции. Операциями отношения являются: больше или равно >=, больше >, меньше или равно <=, меньше <, проверки на равенство == и не равно !=. Значением логического выражения является либо истина (обычно 1), либо ложь (всегда 0). Истинность или ложность выражения определяется по правилам алгебры.

Вопрос №15

КАКОЕ ЛОГИЧЕСКОЕ ВЫРАЖЕНИЕ РАВНО ЕДИНИЦЕ (ИСТИНА)?

Варианты ответа:

- 3 != 3
- 3 > 5
- 3 >= 5
- 3 == 5
- 3 < 5

Комментарий. Операции отношения >=, >, <=, < имеют одинаковое старшинство. Непосредственно за ними по уровню старшинства следуют две другие операции отношения: проверки на равенство == и не равно. Истинность или ложность выражения определяется по правилам алгебры.

Вопрос №16

КАКОЕ ЛОГИЧЕСКОЕ ВЫРАЖЕНИЕ РАВНО НУЛЮ (ЛОЖЬ)?

Варианты ответа:

- 1 && 0
- 0 != 1
- 1 || 0
- 0 || 1
- 1 > 0

Комментарий. Логическими операциями являются: логическое И &&, логическое ИЛИ ||, и логическое НЕ !. Чтобы определить истинность или ложность выражения вида "А логическая операция В" необходимо воспользоваться табл. 4.

Таблица 4.

A	B	A&&B	A||B	!A
1	1	1	1	0
1	0	0	1	0
0	1	0	1	1
0	0	0	0	1

Унарная операция отрицания ! преобразует ненулевой или истинный операнд в 0, а нулевой или ложный операнд в 1.

Вопрос №17

КАКОЕ ЛОГИЧЕСКОЕ ВЫРАЖЕНИЕ РАВНО ЕДИНИЦЕ (ИСТИНА)?

Варианты ответа:

- 0 && 0
- 0 > 1
- 0 || 0
- 1 == 0
- 1 || 0

Комментарий. Выражения, связанные операциями && и ||, вычисляются слева направо, причем их рассмотрение прекращается сразу же, как только становится ясно, будет ли результат истиной или ложью. Старшинство операции && выше, чем операции ||, и обе они ниже по старшинству операций отношения и равенства.

Вопрос №18

ПУСТЬ ПЕРВОНАЧАЛЬНО a=1, в=1.

ЧЕМУ РАВНЫ x, a и b ПОСЛЕ ВЫПОЛНЕНИЯ ИНСТРУКЦИИ x = ++a * b--;

Варианты ответа:

- x= 2 , a= 2 , b=1
- x= 0 , a= 2 , b=0
- x= 1 , a= 1 , b=2
- x= 2 , a= 2 , b=2
- x= 2 , a= 2 , b=0

Комментарий. В языке C/C++ имеются две необычные операции для увеличения (инкремента) и уменьшения (декремента) значений переменных. Операция увеличения ++ добавляет 1 к своему операнду, а операция уменьшения -- вычитает 1. Необычность заключается в том, что ++ и -- можно использовать либо как префиксные операции (перед переменной, *++a*), либо как постфиксные (после переменной *a*++). Эффект в обоих случаях состоит в увеличении *a*. Но выражение, *++a* увеличивает переменную *a* до использования ее значения, в то время как *a*++ увеличивает переменную *a* после того, как ее значение было использовано. Если *a* = 1, то выражение x = *a*++; устанавливает x равным 1, в то время как выражение x = ++*a*; устанавливает x равным 2. В обоих случаях *a* становится равным 2.

Вопрос №19

ПУСТЬ ПЕРВОНАЧАЛЬНО a=2, в=1.
ЧЕМУ РАВНЫ x, a и b ПОСЛЕ ВЫПОЛНЕНИЯ ИНСТРУКЦИИ x =--a - b--;
Варианты ответа:

- x= 1 , a= 2 , b=1
- x= 1 , a= 1 , b=0
- x= 0 , a= 1 , b=1
- (●) x= 0 , a= 1 , b=0
- x= 1 , a= 1 , b=1

Комментарий. Операции инкремента и декремента имеют более высокий приоритет выполнения по сравнению с арифметическими операциями: умножения *, деления /, сложения + и вычитания -. Однако в выражениях типа x =--a - b--; последовательность выполнения операторов следующая: вначале выполняется префиксный декремент и *a* устанавливается равным 1, затем осуществляются операции вычитания и присваивания x =0, и наконец выполняется постфиксный декремент, переменная *в* устанавливается равной 0. Операции инкремента и декремента можно применять только к переменным; выражения типа x = (i + j)++ являются ошибочными. В случаях, где нужен только эффект увеличения, как, например, в счетчиках цикла, выбор префиксной (++n) или постфиксной (n++) операции является делом вкуса.

Вопрос №20

ОПЕРАТОРЫ МАНИПУЛИРОВАНИЯ С БИТАМИ НЕ ПРИМЕНИМЫ К ОПЕРАНДАМ ТИПА:?
Варианты ответа:

- short
- short int
- (●) float
- long
- char

Комментарий. В языке C/C++ предусмотрен ряд операций для работы с битами; эти операции нельзя применять к переменным типа float или double.

Вопрос №21

ЧЕМУ РАВЕН РЕЗУЛЬТАТ ПОБИТОВОЙ ОПЕРАЦИИ 10110 & 11011 ?
Варианты ответа:

- 10110
- 11011
- (●) 10010
- 11111
- 01100

Комментарий. Существуют следующие побитовые (поразрядные) операции: побитовое И &, побитовое ИЛИ | , побитовое исключающее ИЛИ ^ и побитовое отрицание ~. Эти операций предназначены для работы с битами. Чтобы определить результат побитовой операции необходимо воспользоваться таблицей истинности (табл.5) для побитовых операторов.

Таблица 5.

A	B	A&B	A | B	A^B	~A
1	1	1	1	0	0
1	0	0	1	1	0
0	1	0	1	1	1
0	0	0	0	0	1

Унарная операция ~ дает дополнение к целому; это означает, что каждый бит со значением 1 получает значение 0 и наоборот.

Следует быть внимательным и отличать побитовые операции & и | от логических связок && и ||, которые подразумевают вычисление значения истинности выражений слева направо. Например, если x=1, а y=2, то значение x & y равно нулю, в то время как значение x && y равно 1.

Вопрос №22

ЧЕМУ РАВЕН РЕЗУЛЬТАТ ПОБИТОВОЙ ОПЕРАЦИИ 10101 | 11000 ?

Варианты ответа:

- 00000
- 10101
- 11111
- 11000
- 11101

Комментарий. См. комментарий к вопросу № 21. В таблице решения 6 каждая клетка – это ячейка для одного бита.

Таблица 6.

Первый операнд	1	0	1	0	1
Второй операнд	1	1	0	0	0
Результат операции побитовое ИЛИ	1	1	1	0	1

Вопрос №23

ЧЕМУ РАВЕН РЕЗУЛЬТАТ ПОБИТОВОЙ ОПЕРАЦИИ 10101 ^ 11000 ?

Варианты ответа:

- 10101
- 01101
- 11000
- 11101
- 00010

Комментарий. См. комментарий к вопросу № 21.

Вопрос №24
ЧЕМУ РАВЕН РЕЗУЛЬТАТ ПОБИТОВОЙ ОПЕРАЦИИ ~10101 ?
Варианты ответа:

- (•) 01010
- ( ) 10101
- ( ) 11111
- ( ) 00010
- ( ) 00000

Комментарий. См. комментарий к вопросу № 21. В таблице решения 7 каждая клетка – это ячейка для одного бита.

Таблица 7.

Операнд	1	0	1	0	1
Результат операции побитовое НЕ	0	1	0	1	0

Вопрос №25
ЧЕМУ РАВЕН РЕЗУЛЬТАТ ПОБИТОВОЙ ОПЕРАЦИИ X<<1, ЕСЛИ X=11001 ?
Варианты ответа:

- (•) 00110010
- ( ) 00011001
- ( ) 00110011
- ( ) 00101010
- ( ) 00111111

Комментарий. Операции << и >> также предназначены для работы с битами. Операции сдвига << и >> осуществляют соответственно сдвиг влево и вправо своего левого операнда на число битовых позиций, задаваемых правым операндом. Таким образом, x<<2 сдвигает x влево на две позиции, заполняя, освобождающиеся биты нулями, что эквивалентно умножению на 4.

Вопрос №26
ЧЕМУ РАВЕН РЕЗУЛЬТАТ ПОБИТОВОЙ ОПЕРАЦИИ X>>2, ЕСЛИ X=11101 ?
Варианты ответа:

- ( ) 00011101
- (•) 00000111
- ( ) 00001110
- ( ) 00111010
- ( ) 00000101

Комментарий. См. также комментарий к вопросу № 25. Сдвиг вправо без знаковой величины всегда сопровождается заполнением освобождающихся разрядов нулями. Сдвиг

вправо знаковой величины заполняет освобождающиеся биты на некоторых машинах содержанием знакового бита 1 (если минус) или 0 (если плюс), а на других машинах 0.

Вопрос №27

ПУСТЬ ПЕРВОНАЧАЛЬНО y=1, x=2. ЧЕМУ РАВЕН x В ВЫРАЖЕНИИ x += y + 1;
Варианты ответа:

- (•) x= 4
- ( ) x= 3
- ( ) x= 2
- ( ) x= 6
- ( ) x= 5

Комментарий. Выражения, типа i = i + 2, в котором стоящая слева переменная повторяется и справа, могут быть записаны в сжатой виде i += 2, используя операцию присваивания вида +=. Большинству бинарных операций (операций подобных +, которые имеют левый и правый операнд) соответствует операция присваивания вида *oper*=, где *oper* - одна из операций типа: +, - , * , / , % , <<, >>, & , | , ^. Если *выр1* и *выр2* - выражения, то запись *выр1 oper= выр2* эквивалентна записи *выр1* =(*выр1*) *oper* (*выр2*). Обратите внимание на круглые скобки вокруг *выр2*.

Вопрос №28

ПУСТЬ ПЕРВОНАЧАЛЬНО y=0, x=2. ЧЕМУ РАВЕН x В ВЫРАЖЕНИИ x >>= y + 1;
Варианты ответа:

- ( ) x= 2
- ( ) x= 3
- (•) x= 1
- ( ) x= 0
- ( ) x= -1

Комментарий. Выражение x >>= y + 1 это сокращенная форма записи выражения x = x >> y +1. В правой части этого выражения стоит оператор сдвига вправо числа x = 2 (т.е. 10 в двоичном коде) на 1 позицию. В результате сдвига вправо двоичное число 10 переходит в 1 и x получает значение 1. См. также комментарий к вопросу № 26.

Вопрос №29

ПУСТЬ ПЕРВОНАЧАЛЬНО y=3, x=1. ЧЕМУ РАВЕН x В ВЫРАЖЕНИИ x |= y - 1;
Варианты ответа:

- ( ) x= -1
- ( ) x= 0
- ( ) x= 1
- ( ) x= 2
- (•) x= 3

Комментарий. Выражение x |= y - 1 это сокращенная форма записи выражения x = x | y -1. В правой части этого выражения стоит побитовый оператор ИЛИ. См. также комментарий к вопросу № 21. Приоритет бинарного оператора " –" выше приоритета оператора "|" (см. табл.8), поэтому выражение x = x | y -1 определено однозначно и y -1 не обязательно заключать в круглые скобки.

Вопрос №30

ПУСТЬ ПЕРВОНАЧАЛЬНО y=1, x=1.ЧЕМУ РАВЕН x В ВЫРАЖЕНИИ x &= y + 1;

Варианты ответа:

- x= 0
- x= 1
- x= 2
- x= -1
- x= -2

Комментарий. Выражение x &= y + 1 это сокращенная форма записи выражения x = x & y +1. В правой части этого выражения стоит побитовый оператор И. См. комментарий к вопросу № 21.

Вопрос №31

ПУСТЬ a=2, в=6. ЧЕМУ РАВНО z В УСЛОВНОМ ВЫРАЖЕНИИ z =(a > b)? a : b

Варианты ответа:

- 6
- 4
- 2
- 1
- 0

Комментарий. Условное выражение записывается с помощью тернарной оператора "? :". В выражении *выр1 ? выр2 : выр3* сначала вычисляется выражение *выр1*. Если оно отлично от нуля (истинно), то вычисляется выражение *выр2*, которое и становится значением условного выражения. В противном случае вычисляется *выр3*, и оно становится значением условного выражения. Таким образом, чтобы положить z равным максимуму из a и b, можно написать z = (a > b) ? a : b; /* z = max(a, b) */

Вопрос №32

ВЫБЕРИТЕ ОПЕРАТОР С НАИБОЛЕЕ НИЗКИМ ПРИОРИТЕТОМ

Варианты ответа:

- [ ]
- =
- &&
- <<
- >>

Комментарий. Операторы языка С/С++ имеют приоритет или старшинство и выполняются в выражениях согласно своему приоритету. В приводимой ниже таблице 8 даны приоритеты и ассоциативности (т.е. порядок выполнения справа налево или слева направо) всех операторов. Операторы, расположенные в одной строке, имеют один и тот же приоритет; строки расположены в порядке убывания приоритета. Так, например, операции * (умножить), / (разделить) и % (найти остаток) имеют одинаковый уровень приоритета, который выше, чем уровень операций + (сложить) и – (вычесть).

Таблица 8.

Приоритет	Порядок вычисления операторов
( ) [ ] -> .	Слева направо
! ~ ++ -- + - * & (тип) sizeof	**Справа налево**
*** / %**	Слева направо
+ -	Слева направо
<< >>	Слева направо
< <= > >=	Слева направо
== !=	Слева направо
&	Слева направо
^	Слева направо
|	Слева направо
&&	Слева направо
||	Слева направо
?:	**Справа налево**
= += -= *= /= %= &= ^= |= <<= >>=	**Справа налево**
, (оператор запятая)	Слева направо

Вопрос №33
КАКОЙ ОПЕРАТОР ВЫПОЛНЯЕТСЯ СПРАВО НАЛЕВО?
Варианты ответа:

- (•) =
- ( ) + СУММА
- ( ) &&
- ( ) ||
- ( ) * УМНОЖЕНИЕ

Комментарий. Из табл.8 следует, что оператор = выполняется справа налево. Остальные операторы, представленные в этом вопросе выполняются наоборот т.е. слева направо.

Вопрос №34
С ИСПОЛЬЗОВАНИЕМ, КАКИХ СКОБОК СОЗДАЕТСЯ БЛОК (СОСТАВНАЯ ИНСТРУКЦИЯ)?
Варианты ответа:

- ( )
- [ ]
- | |
- { }
- < >

Комментарий. В языке C/C++ точка с запятой является признаком конца инструкции. Фигурные скобки { и } используются для объединения описаний и операторов в составной оператор или блок, так что они оказываются эквивалентны одному оператору. Например, этого могут быть фигурные скобки, в которые заключаются операторы, составляющие функцию. Переменные также могут быть описаны внутри любого блока.

Вопрос №35

В КАКОЙ СТРОКЕ УСЛОВНОЙ КОНСТРУКЦИИ НАХОДИТСЯ ОШИБКА?

Фрагмент программы:

- if(n = 0)
- x=y+2;
- else {
- x=y-2;
- z=y+1; }

Комментарий. Стоящее в круглых скобках конструкции **if**(*выражение*) обычно является логическим, выражением отношения или вычисляемым арифметическим выражением, но не может быть оператором присваивания. Ошибка заключается в том, что вместо оператора == (проверки на равенство) находится оператор = (присвоить). Это типичная ошибка программирования.

Вопрос №36

КАКОЙ ЗНАК СТАВИТСЯ ПОСЛЕ ЗАКРЫВАЮЩЕЙ ФИГУРНОЙ СКОБКИ БЛОКА?

Варианты ответа:

- :
- ;
- ,
- НИКАКОЙ
- \\

Комментарий. Точка с запятой никогда не ставится после первой фигурной скобки, которая завершает блок.

Вопрос №37

ПЕРЕМЕННАЯ val = 0. КАКАЯ ФУНКЦИЯ БУДЕТ ВЫПОЛНЯТСЯ В ОПЕРАТОРЕ

```
switch(val){
case 1:   f(); break;
case 2:   g(); break;
default: h(); break;
}
```

Варианты ответа:

- f( )
- g( )
- (•) h( )
- НИКАКАЯ
- f( ) + g( ) + h( )

Комментарий. Оператор switch дает способ выбора одного из многих вариантов выполнения программы, который заключается в проверке совпадения значения выражения с одной из заданных констант и соответствующем ветвлении. Переключатель получает из программы или вычисляет целое выражение в круглых скобках (val) и сравнивает его значение со всеми константами, стоящими после case. Если значение константного выражения, стоящего после case, совпадает со значением целого выражения, то начинается выполнение этой ветви case. Если ни один из случаев не подходит, то выполняется оператор после default. Default является необязательным, элементом конструкции switch, если его нет, и ни один из случаев не подходит, то вообще никакие действия не выполняются. Оператор break приводит к немедленному выходу из переключателя. Поскольку имена case служат только в качестве меток, то если нет явных действий после выполнения операторов, соответствующих одному случаю, происходит переход на следующую ветвь case.

Вопрос №38

ПЕРЕМЕННАЯ count = 3, ЧТО НАПЕЧАТАЕТ ФРАГМЕНТ ПРОГРАММЫ

```
switch(count){
case 1:   printf( "1");
case 2:   printf( "2");
case 3:   printf("3");
default: printf( "4");
}
```

Варианты ответа:

- 1
- 2
- 3
- 4
- (•) 34

Комментарий. См. комментарий к вопросу № 37. Обратите внимание на отсутствие операторов break в ветвях case. Это приводит к проваливанию с одной ветви на другую и выполнению, всех последующих инструкций оператора switch, начиная с case 3.

Вопрос №39

ПЕРЕМЕННАЯ val=2. КАКАЯ ФУНКЦИЯ БУДЕТ ВЫПОЛНЯТЬСЯ В УСЛОВНОЙ ИНСТУКЦИИ

```
if( val==0 )
    f( );
else if( val==1 )
    g( );
else
    h( );
```

Варианты ответа:

- f( )
- g( )
- h( )
- НИКАКАЯ
- f( ) + g( )

Комментарий. Условная инструкция **if - else** используется при необходимости сделать выбор. Синтаксис инструкции имеет вид:

if (*выражение*)
    оператор_1
else
    оператор_2,

где часть **else** является необязательной. Сначала вычисляется выражение; если оно "истинно", т.е. значение выражения отлично от нуля, то выполняется *оператор_1*. Если оно ложно, значение выражения равно нулю, и если есть часть с **else**, то вместо *оператора_1* выполняется *оператор_2*.

Вопрос №40

ПУСТЬ ПЕРЕМЕННАЯ a=6, d=5. ЧЕМУ РАВНА ПЕРЕМЕННАЯ max ПОСЛЕ ВЫПОЛНЕНИЯ УСЛОВНОЙ ИНСТРУКЦИИ

```
if( a<=d )
    max =b;
else
    max =c;
```

Варианты ответа:

- 5
- 6
- c
- b
- d

Комментарий. См. комментарий к вопросу № 39.

Вопрос №41
КАКОЙ ЗАГОЛОВОК ЦИКЛА ЗАПИСАН С ОШИБКОЙ?
Варианты ответа:

- while(c < 7)
- for(i=1; i<n; i++)
- while(c!=255)
- while(i=1; i<n; i++)
- while(c == ' ' || c == '\n')

Комментарий. В языке С/С++ существует три формы оператора цикла, предназначенных для выполнения повторяющихся действий. Цикл состоит из заголовка и тела цикла. В тело цикла включены повторяющиеся операторы. Заголовки различных циклов содержат разные служебные слова **while, for, do – while**. Цикл **while** называется циклом с предисловием и имеет вид:

while*(выражение условия*)
тело_цикла

Работа цикла начинается с вычисления выражения в круглых скобках. Если его значение отлично от нуля, то тело цикла выполняется. Цикл продолжается до тех пор, пока значение *выражение условия* не станет нулем, после чего выполнение цикла заканчивается. Ошибка находится в четвертом заголовке цикла **while** и заключается в правилах записи выражения условия для данного типа цикла.

Вопрос №42
КАКОЙ ЗАГОЛОВОК ЦИКЛА ЗАПИСАН ПРАВИЛЬНО?
Варианты ответа:

- while((c = getchar( )) ==' ' || c=='\n' || c== '\t')
- for(i=1, i<n, i++)
- WHILE (C <7)
- FOR(I=0; I==9; I++)
- for(I=10; I >= N: I++)

Комментарий. Цикл **for** называется параметрическим и имеет вид:
for (*выражение 1; выражение 2; выражение 3*)
тело цикла

Выражение 1 определяет действия, выполняемые до начало цикла, т.е. задает начальные условия цикла. *Выражение 2* – обычно логическое и оно определяет условие окончания или продолжения цикла. Если оно истинно (т.е. не равно 0), то выполняется тело цикла, а затем вычисляется *выражение 3*. После выполнения *выражения 3* вычисляется истинность *выражения 2* и все повторяется. Цикл продолжается до тех пор пока, не станет ложным *выражение2*. Любое из трех выражений может быть опущено, хотя точки с запятой при этом должны оставаться. Если отсутствует *выражение 1*или *выражение 3*, то оно просто выпадает из конструкции. Если же отсутствует проверка, *выражение 2*, то считается, как будто оно всегда истинно, так что цикл вида **for (; ;)** является бесконечным циклом, о котором предполагается, что он будет прерван другими средствами (такими как **break** или **return**).

Вопрос №43

ЦИКЛ for(i=0; isspace(s[i]); i++) ВЫПОЛНЯЕТСЯ ДО ТЕХ ПОР ПОКА

Варианты ответа:

- isspace(s[i])ВОЗВРАЩАЕТ НУЛЬ
- isspace(s[i]) ВОЗВРАЩАЕТ ИСТИНУ
- isspace(s[i]) НИЧЕГО НЕ ВОЗВРАЩАЕТ
- isspace(s[i]) ВОЗВРАЩАЕТ СИМВОЛ i
- isspace(s[i]) ВОЗВРАЩАЕТ СИМВОЛ s

Комментарий. Функция isspace( ) возвращает ненулевое значение, если аргумент функции является пробельным символом. К пробельным символам относятся: пробел, символы горизонтальной и вертикальной табуляции (**\t ,\v**), перевода страницы (**\f**), возврата каретки (**\r**) и новой строки (**\n**). В любом другом случае функция isspace( ) возвращает нуль. Таким образом цикл **for** выполняется до тех пор пока элементы массива s[i] содержат пробельные символы и функция возвращает истину. Функция isspace( ) позволяет проигнорировать левые пробельные символы в массиве символов.

Вопрос №44

ЧЕМУ РАВНЫ ПЕРЕМЕННЫЕ sum И i ПОСЛЕ ВЫПОЛНЕНИЯ ФРАГМЕНТА ПРОГРАММЫ?

```
int i=1,sum= 0;
while( i<=5 ){
    sum+=i;
    i++;
}
```

Варианты ответа:

- sum=10, i=6
- sum=10, i=5
- sum=15, i=5
- sum=15, i=1
- sum=15, i=6

Комментарий. В теле цикла **while** вычисляется сумма чисел от1 до 5, при этом на каждом шаге цикла переменная i , управляющая выполнением цикла увеличивается на единицу (i++). При i=5 тело цикла выполняется, переменная sum =10+5 =15, далее i становится равным 6, условие выполнения цикла оказывается ложным и цикл заканчивается с переменными sum =15 и i =6.

Вопрос №45

ЧЕМУ РАВНЫ ПЕРЕМЕННЫЕ sum И n ПОСЛЕ ВЫПОЛНЕНИЯ ФРАГМЕНТА ПРОГРАММЫ?

```
int n, sum;
sum = 0;
for( n = 1; n<3; n++)
sum -= n;
```

Варианты ответа:

- ○ sum=3, n=3
- ○ sum=3, n=2
- ○ sum=0, n=0
- ○ sum=-3, n=2
- ◉ sum=-3, n=3

Комментарий. Рассмотрим подробно, как работает цикл **for** для нашего случая. Во – первых, следует обратить внимание на то, что переменная sum, в которой будет накапливаться сумма, должна быть обнулена до начала цикла. Это достигается инструкцией sum = 0;

Алгоритм выполнения цикла следующий:

Шаг1.

Пункт1. Переменная n получает значение единица (n =1). Это инициализация счетчика цикла.

Пункт2.Проверка условия n < 3, поскольку на этом шаге 1< 3 , то результатом сравнения будет истина и выполняется тело цикла.

Пункт3. Так как sum -= n эквивалентно sum =sum - n, то на первом шаге sum =0 -1 =-1. Переменная sum получает значение минус единица.

Пункт4. Увеличение счетчика цикла на единицу, выполняется инструкцией n++ , в результате n становится равным 2.

Первый шаг алгоритма цикла закончен , происходит переход ко второму шагу.

Шаг2.

Пункт1. Проверка условия n < 3, теперь 2 <3 , что тоже истина и выполняется тело цикла.

Пункт2. Переменная sum получает значение -3, так как из выражения sum =sum – n, следует, что sum = -1 – 2 =-3.

*Пункт3.*Увеличение счетчика цикла на единицу, выполняется n++, в результате n становится равным 3.

Второй шаг алгоритма цикла закончен, переход к третьему шагу.

Шаг3.
Пункт1. Проверка условия n < 3, теперь уже 3 <3 и это выражение ложно. Следовательно, условие выполнения цикла нарушено, тело цикла не выполняется, цикл заканчивается и управление передается к инструкции следующей за циклом.

В итоге, после завершения цикла for(n = 1; n<3; n++) sum -= n; переменная sum получает значение -3, счетчик цикла n =3.

Вопрос №46
ЧЕМУ РАВНЫ ПЕРЕМЕННЫЕ sum И i ПОСЛЕ ВЫПОЛНЕНИЯ ФРАГМЕНТА ПРОГРАММЫ?

```
int sum = 0;
int i = 5;
do
{
sum += i;
i--;
}
while( i >= 0);
```

Варианты ответа:

- sum=14, i=-1
- sum=14, i=0
- sum=14, i=1
- sum=15, i=0
- sum=15, i=-1

Комментарий. Третий оператор цикла языка С/С++ это цикл с постусловием **do-while**. Особенность цикла заключается в том, что проверка на условие выполнения цикла выполняется в конце каждого прохода и таким образом данный цикл всегда выполняется, по крайней мере, один раз. Синтаксис конструкции цикла имеет вид:

do{
оператор;
}
while (*выражение условия***)**

В цикле **do-while** сначала выполняется оператор, затем вычисляется выражение условия. Если оно истинно, то оператор выполняется снова и т.д. Если выражение становится ложным, цикл заканчивается. Подчеркнем еще раз, что в циклах **while** и **for** проверка окончания цикла осуществляется в начале, а не в конце цикла, в результате циклы **while** и **for** могут ни разу ни выполняться, этим они отличаются от цикла **do-while**.

Вопрос №47
ЧЕМУ РАВНЫ ПЕРЕМЕННЫЕ product И i ПОСЛЕ ВЫПОЛНЕНИЯ ФРАГМЕНТА ПРОГРАММЫ?

```
int  product =1;
int i;
for( i=5; i >0; i--)
product *= i;
```

Варианты ответа:

- product=20, i=1
- product=60, i=2
- product=60, i=1
- product=120, i=0
- product=120, i=1

Комментарий. См. комментарий к вопросу № 42.

Вопрос №48

ЧЕМУ РАВНЫ ПЕРЕМЕННЫЕ sum И i ПОСЛЕ ВЫПОЛНЕНИЯ ФРАГМЕНТА ПРОГРАММЫ?

```
int i, sum= 0;
for( i =1; i<=5; i++)
sum += i;
```

Варианты ответа:

- sum=15, i=6
- sum=15, i=5
- sum=10, i=6
- sum=10, i=5
- sum=10, i=1

Комментарий. См. комментарий к вопросу № 42 и вопросу №45.

Вопрос №49

ЧЕМУ РАВНЫ ПЕРЕМЕННЫЕ product И n ПОСЛЕ ВЫПОЛНЕНИЯ ФРАГМЕНТА ПРОГРАММЫ?

```
int  product=1;
int  n=5;
while( n >= 1){
product *= n;
n -=1;
}
```

Варианты ответа:

- product=60, n=0
- product=60, n=2
- product=60, n=1
- product=120, n=1
- product=120, n=0

Комментарий. В теле цикла **while** вычисляется произведение чисел от 5 до 1, при этом на каждом шаге цикла переменная n , управляющая выполнением цикла уменьшается на единицу т.е. n -=1 эквивалентно n = n – 1. При n = 1 тело цикла выполняется, переменная product=120*1, затем n уменьшается до 0, условие выполнения цикла становится ложным и цикл заканчивается с переменными product =120 и n = 0.

5.2. Вопросы для самоконтроля

Вопрос №1
КАКОЙ ТИП ПЕРЕМЕННОЙ ПРЕДСТАВЛЕН КЛЮЧЕВЫМ СЛОВОМ "**char**"?
Варианты ответа:

- ☐ вещественный с двойной точностью
- ☒ символьный
- ☐ вещественный с одинарной точностью
- ☐ целый

Вопрос №2
КАКОЕ ИМЯ ПЕРЕМЕННОЙ ЯВЛЯЕТСЯ НЕДОПУСТИМЫМ?
Варианты ответа:

- ☐ $dollars
- ☒ _Variable
- ☐ Sum
- ☐ COLORS
- ☐ TeXT

Вопрос №3
СКОЛЬКО БАЙТ ПАМЯТИ ПРЕДОСТАВЛЯЕТСЯ ДЛЯ ПЕРЕМЕННОЙ ТИПА **char**?
Варианты ответа:

- ☐ 8 байт
- ☐ 2 байт
- ☒ 4 байт
- ☐ 10 байт
- ☐ 1 байт

Вопрос №4
КАКОЙ ДИАПАЗОН ЗНАЧЕНИЙ ИМЕЕТ ПЕРЕМЕННАЯ ТИПА **signed char**?
Варианты ответа:

- ☒ 0 - 255
- ☐ -32768 - +32767
- ☐ -128 - +127
- ☐ 0 - 65535
- ☐ -2147483648 - +2147483647

Вопрос №5
КАКАЯ КОНСТАНТА ЯВЛЯЕТСЯ СИМВОЛЬНОЙ?
Варианты ответа:

- ☐ '15'
- ☐ 0XAA
- ☐ 0734
- ☐ 0
- ☒ "56"

Вопрос №6
КАКАЯ КОНСТАНТА ЯВЛЯЕТСЯ ШЕСТНАДЦАТЕРИЧНОЙ?
Варианты ответа:

- ☒ 1616
- ☐ 1e+04
- ☐ 0773
- ☐ 0XFF
- ☐ "1AB"

Вопрос №7
КАКАЯ КОНСТАНТА ЯВЛЯЕТСЯ ВОСЬМЕРИЧНОЙ?
Варианты ответа:

- ☒ 8
- ☐ 088
- ☐ 5.8
- ☐ 045
- ☐ 0x21

Вопрос №8
ЧТО ОЗНАЧАЕТ УПРАВЛЯЮЩАЯ ПОСЛЕДОВАТЕЛЬНОСТЬ ВИДА **\n**?
Варианты ответа:

- ☐ ОБРАТНАЯ НАКЛОННАЯ ЧЕРТА
- ☐ ГОРИЗОНТАЛЬНАЯ ТАБУЛЯЦИЯ
- ☒ НОВАЯ СТРОКА
- ☐ СИГНАЛ-ЗВОНОК
- ☐ ДВОЙНАЯ КАВЫЧКА

Вопрос №9
КАКАЯ КОНСТАНТА ЯВЛЯЕТСЯ СТРОКОВОЙ (СТРОКОВЫМ ЛИТЕРАЛОМ)?
Варианты ответа:

- ☐ 1.2e-3
- ☐ 123U
- ☒ 'B'
- ☐ 0xFA
- ☐ "Hello, world"

Вопрос №10
НАЙТИ ПРАВИЛЬНОЕ ОБЪЯВЛЕНИЕ ПЕРЕМЕННЫХ
Варианты ответа:

- ☐ INT LOWER=1,UPPER=20,STEP;
- ☒ INT LOWER,UPPER,STEP;
- ☐ INT lower, upper, step;
- ☐ int lower=0, upper=20.5, step;
- ☐ int lower,upper,step=20;

Вопрос №11
ЧЕМУ РАВНО ВЫРАЖЕНИЕ 6/5 ?
Варианты ответа:

- ☐ 1
- ☐ 1.2
- ☒ 1,2
- ☐ 1.0
- ☐ 0

Вопрос №12
ЧЕМУ РАВНО ВЫРАЖЕНИЕ 6%5 ?
Варианты ответа:

- ☐ 1,2
- ☐ 1
- ☐ 2.0
- ☒ 1.2
- ☐ 2

Вопрос №13
КАКОЕ ЛОГИЧЕСКОЕ ВЫРАЖЕНИЕ РАВНО НУЛЮ (ЛОЖЬ)?
Варианты ответа:

- ☒ 2 >= 1
- ☐ 2 != 2
- ☐ 2 == 2
- ☐ 1 < 2
- ☐ 2 > 0

Вопрос №14
КАКОЕ ЛОГИЧЕСКОЕ ВЫРАЖЕНИЕ РАВНО ЕДИНИЦЕ (ИСТИНА)?
Варианты ответа:

- ☐ -1 != -1
- ☐ -1 > 1
- ☒ 0 >= -1
- ☐ -1 == 0
- ☐ 1 < 0

Вопрос №15
КАКОЕ ЛОГИЧЕСКОЕ ВЫРАЖЕНИЕ РАВНО НУЛЮ (ЛОЖЬ)?
Варианты ответа:

- ☒ 1 || 0
- ☐ 0 != 1
- ☐ 1 || 1
- ☐ 0 && 1
- ☐ 1 > 0

Вопрос №16
КАКОЕ ЛОГИЧЕСКОЕ ВЫРАЖЕНИЕ РАВНО ЕДИНИЦЕ (ИСТИНА)?
Варианты ответа:

- ☐ -1 != -1
- ☐ -1 > 1
- ☐ 0 || 1
- ☐ -1 == 0
- ☒ 1 && 0

Вопрос №17
ПУСТЬ ПЕРВОНАЧАЛЬНО a=2, в=1.
ЧЕМУ РАВНЫ x, a и b ПОСЛЕ ВЫПОЛНЕНИЯ ИНСТРУКЦИИ x= --a * b--;
Варианты ответа:

- ☐ x= 2 , a= 2 , b=1
- ☒ x= 1 , a= 1 , b=0
- ☐ x= 2 , a= 1 , b=1
- ☐ x= 1 , a= 2 , b=0
- ☐ x= 1 , a= 1 , b=1

Вопрос №18
ПУСТЬ ПЕРВОНАЧАЛЬНО a=2, в=1.
ЧЕМУ РАВНЫ x,a и b ПОСЛЕ ВЫПОЛНЕНИЯ ИНСТРУКЦИИ x= --a + b--;
Варианты ответа:

- ☐ x= 1 , a= 2 , b=1
- ☐ x= 0 , a= 2 , b=0
- ☒ x= 2 , a= 1 , b=1
- ☐ x= 2 , a= 1 , b=0
- ☐ x= 1 , a= 0 , b=1

Вопрос №19
ЧЕМУ РАВЕН РЕЗУЛЬТАТ ПОБИТОВОЙ ОПЕРАЦИИ 11111 & 10001?
Варианты ответа:

- ☐ 00000
- ☐ 11011
- ☐ 11111
- ☐ 01110
- ☒ 10001

Вопрос №20
ЧЕМУ РАВЕН РЕЗУЛЬТАТ ПОБИТОВОЙ ОПЕРАЦИИ 11111 | 10001?
Варианты ответа:

- ☒ 11111
- ☐ 10001
- ☐ 01110
- ☐ 00000
- ☐ 10001

Вопрос №21
ЧЕМУ РАВЕН РЕЗУЛЬТАТ ПОБИТОВОЙ ОПЕРАЦИИ 11101 ^ 00101?
Варианты ответа:

- ☒ 00111
- ☐ 01101
- ☐ 00101
- ☐ 11000
- ☐ 11101

Вопрос №22
ЧЕМУ РАВЕН РЕЗУЛЬТАТ ПОБИТОВОЙ ОПЕРАЦИИ ~11101?
Варианты ответа:

- ☐ 11101
- ☐ 00000
- ☐ 11111
- ☒ 00010
- ☐ 01010

Вопрос №23
ЧЕМУ РАВЕН РЕЗУЛЬТАТ ПОБИТОВОЙ ОПЕРАЦИИ X<< 2, ЕСЛИ X=10001?
Варианты ответа:

- ☒ 00010001
- ☐ 00010000
- ☐ 00011100
- ☐ 01101100
- ☐ 01000100

Вопрос №24
ОПЕРАТОРЫ МАНИПУЛИРОВАНИЯ С БИТАМИ НЕ ПРИМЕНИМЫ К ОПЕРАНДАМ ТИПА:?
Варианты ответа:

- ☐ char
- ☐ signed char
- ☐ unsigned char
- ☒ double
- ☐ long unsigned int

Вопрос №25
ЧЕМУ РАВЕН РЕЗУЛЬТАТ ПОБИТОВОЙ ОПЕРАЦИИ X>>1, ЕСЛИ X=11111?
Варианты ответа:

- ☐ 00001111
- ☐ 00000111
- ☐ 00001110
- ☒ 00111110
- ☐ 00011111

Вопрос №26
ПУСТЬ ПЕРВОНАЧАЛЬНО y=2, x=2. ЧЕМУ РАВЕН x В ВЫРАЖЕНИИ x *= y+1;
Варианты ответа:

- ☐ x= 2
- ☐ x= 3
- ☐ x= 5
- ☒ x= 6
- ☐ x= 4

Вопрос №27
ПУСТЬ ПЕРВОНАЧАЛЬНО y=1, x=1. ЧЕМУ РАВЕН x В ВЫРАЖЕНИИ x <<= y;
Варианты ответа:

- ☐ x= -1
- ☐ x= 0
- ☒ x= 1
- ☐ x= 2
- ☐ x= 3

Вопрос №28
ПУСТЬ ПЕРВОНАЧАЛЬНО y=1, x=1. ЧЕМУ РАВЕН x В ВЫРАЖЕНИИ x &= y+1;
Варианты ответа:

- ☐ x= 0
- ☒ x= 1
- ☐ x= 2
- ☐ x= -1
- ☐ x= -2

Вопрос №29
ПУСТЬ ПЕРВОНАЧАЛЬНО y=1, x=2. ЧЕМУ РАВЕН x В ВЫРАЖЕНИИ x |= y+1;
Варианты ответа:

- ☐ x= -1
- ☐ x= 0
- ☐ x= 1
- ☒ x= 2
- ☐ x= 3

Вопрос №30
ПУСТЬ ПЕРВОНАЧАЛЬНО y=1, x=1. ЧЕМУ РАВЕН x В ВЫРАЖЕНИИ x ^= y+1;
Варианты ответа:

- ☐ x= -1
- ☐ x= 0
- ☒ x= 3
- ☐ x= 2
- ☐ x= 1

Вопрос №31
ПУСТЬ a=6, в=2. ЧЕМУ РАВНО z В УСЛОВНОМ ВЫРАЖЕНИИ z =(a > b)? a : b
Варианты ответа:

- ☐ 0
- ☐ 1
- ☒ 2
- ☐ 4
- ☐ 6

Вопрос №32
ВЫБЕРИТЕ ОПЕРАТОР С НАИБОЛЕЕ ВЫСОКИМ ПРИОРИТЕТОМ
Варианты ответа:

- ☐ & ПОБИТОВОЕ И
- ☐ ^ ПОБИТОВОЕ ИСКЛЮЧАЮЩЕЕ ИЛИ
- ☒ && ЛОГИЧЕСКОЕ И
- ☐ ?: УСЛОВНОЕ ВЫРАЖЕНИЕ
- ☐ & ВЗЯТИЕ АДРЕСА

Вопрос №33
КАКОЙ ОПЕРАТОР ВЫПОЛНЯЕТСЯ СПРАВО НАЛЕВО?
Варианты ответа:

- ☐ % ДЕЛЕНИЕ ПО МОДУЛЮ
- ☐ - РАЗНОСТЬ
- ☐ << СДВИГ ВЛЕВО
- ☐ * РАЗЫМЕНОВАНИЯ
- ☒ * УМНОЖЕНИЕ

Вопрос №34
В КАКОЙ СТРОКЕ УСЛОВНОЙ КОНСТРУКЦИИ НАХОДИТСЯ ОШИБКА?
Фрагмент программы:

- ☒ if(n != 0)
- ☐ ++x; z=y+1;
- ☐ else
- ☐ x=y-2;

Вопрос №35
ДАН ФРАГМЕНТ ПРОГРАММЫ. ЧЕМУ РАВНО z, ЕСЛИ x=10?

```
if( x <= 0 )
    z = abs(x);
else if ( x < 10 )
    z = x;
else
    z =-x;
```

Варианты ответа:

- ☐ 0
- ☐ -10
- ☒ 10
- ☐ 5
- ☐ +5

Вопрос №36
В КАКОЙ СТРОКЕ КОНСТРУКЦИИ ПЕРЕКЛЮЧАТЕЛЯ НАХОДИТСЯ ОШИБКА?
Варианты ответа:

- ☐ switch(c) {
- ☐ case 'a': x = 4: break;
- ☐ case 'b': x = 2; break;
- ☐ case 'c': x = 1; break;
- ☒ default: x = 0;
- ☐ }

Вопрос №37
КАКОЙ ЗАГОЛОВОК ЦИКЛА ЗАПИСАН ПРАВИЛЬНО?
Варианты ответа:

- ☐ for(; ;)
- ☒ for(i=1, i<n, i++)
- ☐ for(i=10:i>=1:i--)
- ☐ for(i=0 i==9 i++)
- ☐ for(i=10;i>=n:i++)

Вопрос №38
КАКОЙ ЗАГОЛОВОК ЦИКЛА ЗАПИСАН ПРАВИЛЬНО?
Варианты ответа:

- ☒ for(n = strlen(s)-1; n>=0; n--)
- ☐ While(0)
- ☐ for(c<=7)
- ☐ FOR(j=0; j<=9; j++)
- ☐ while(I =10; I >=0; I--)

Вопрос №39
ЦИКЛ for(n=0; isdigit(s[i]); i++) ВЫПОЛНЯЕТСЯ ДО ТЕХ ПОР ПОКА
Варианты ответа:

- ☐ n > i
- ☐ n < i
- ☒ ФУНКЦИЯ isdigit(s[i]) ВОЗВРАЩАЕТ НУЛЬ
- ☐ ФУНКЦИЯ isdigit(s[i]) ВОЗВРАЩАЕТ НЕ НУЛЬ
- ☐ i ЕСТЬ ИСТИНА

Вопрос №40

ЧЕМУ ДОЛЖНА БЫТЬ РАВНА ПЕРЕМЕННАЯ count, ЧТОБЫ ВЫПОЛНЯЛАСЬ ФУНКЦИЯ g( ) В ПЕРЕКЛЮЧАТЕЛЕ

```
switch( count ){
case 1:   f(); break;
case 2:   g(); break;
case 3:   h(); break;
}
```

Варианты ответа:

- ☐ 0
- ☐ 1
- ☒ 2
- ☐ 3
- ☐ 4

Вопрос №41

ПЕРЕМЕННАЯ count = 3, ЧТО НАПЕЧАТАЕТ ФРАГМЕНТ ПРОГРАММЫ

```
switch(count){
case 1:   printf( "1"); break;
case 2:   printf( "2"); break;
case 3:   printf( "3"); break;
case 4:   printf( "4"); break;
}
```

Варианты ответа:

- ☐ 1
- ☐ 2
- ☒ 3
- ☐ 4
- ☐ 23

Вопрос №42

ЧЕМУ ДОЛЖНА БЫТЬ РАВНА ПЕРЕМЕННАЯ count, ЧТОБЫ ФУНКЦИЯ ВЫПОЛНЯЛАСЬ g( ).

```
if( count ==1 )
      h( );
else if( count ==5) g( );
else  f( );
```

Варианты ответа:

- ☐ 1
- ☐ 2
- ☐ 3
- ☐ 4
- ☒ 5

Вопрос №43

ПЕРВОНАЧАЛЬНО ПЕРЕМЕННАЯ x=0, a=1, b=1. ЧЕМУ РАВНА ПЕРЕМЕННАЯ x ПОСЛЕ ВЫПОЛНЕНИЯ УСЛОВНОЙ ИНСТРУКЦИИ

```
if( x > a)        x = a;
if( x < b )
      x = -b;     x = 0;
```

Варианты ответа:

- ☐ -1
- ☐ 0
- ☒ 1
- ☐ a
- ☐ -b

Вопрос №44

ЧЕМУ РАВНЫ ПЕРЕМЕННЫЕ sum И i ПОСЛЕ ВЫПОЛНЕНИЯ ФРАГМЕНТА ПРОГРАММЫ?

```
int i=1,sum= 0;
do
{
   sum+=i;      i++;
}
while( i<=5 );
```

Варианты ответа:

- ☐ sum=15, i=1
- ☐ sum=10, i=6
- ☐ sum=15, i=6
- ☐ sum=10, i=5
- ☒ sum=15, i=5

Вопрос №45

ЧЕМУ РАВНЫ ПЕРЕМЕННЫЕ product И i ПОСЛЕ ВЫПОЛНЕНИЯ ФРАГМЕНТА ПРОГРАММЫ?

```
int i=1, product=1;
for( ; i<=5;  i++)
product *= i;
```

Варианты ответа:

- ☐ product=5, i=5
- ☐ product=24, i=5
- ☐ product=24, i=6
- ☒ product=120, i=5
- ☐ product=120, i=6

Вопрос №46

ЧЕМУ РАВНЫ ПЕРЕМЕННЫЕ product И i ПОСЛЕ ВЫПОЛНЕНИЯ ФРАГМЕНТА ПРОГРАММЫ?

```
int  product=1, i = 5;
for( ; i>0; i--)
product *= i;
```

Варианты ответа:

- ☐ product=0, i=0
- ☒ product=120, i=0
- ☐ product=120, i=1
- ☐ product=60, i=0
- ☐ product=20, i=1

Вопрос №47

ЧЕМУ РАВНЫ ПЕРЕМЕННЫЕ sum И n ПОСЛЕ ВЫПОЛНЕНИЯ ФРАГМЕНТА ПРОГРАММЫ?

```
int  n= 1,  sum= 0;
while( n < 3 ) {
sum -= n;
n += 1;  }
```

Варианты ответа:

- ☐ sum= 6, n=3
- ☐ sum=-6, n=3
- ☐ sum= 3, n=2
- ☐ sum= -3, n=3
- ☒ sum= -3, n=2

Вопрос №48

ЧЕМУ РАВНЫ ПЕРЕМЕННЫЕ product И n ПОСЛЕ ВЫПОЛНЕНИЯ ФРАГМЕНТА ПРОГРАММЫ?

```
int  product =1;
for( n=1; n<3; n++)
product *= n;
```

Варианты ответа:

- ☐ product=2, n=2
- ☒ product=2, n=3
- ☐ product=6, n=2
- ☐ product=6, n=3
- ☐ product=3, n=1

Вопрос №49

ЧЕМУ РАВНЫ ПЕРЕМЕННЫЕ sum И n ПОСЛЕ ВЫПОЛНЕНИЯ ФРАГМЕНТА ПРОГРАММЫ?

```
int sum=0;
int n;
for( n=5; n>=0;  n--)
sum += n;
```

Варианты ответа:

- ☐ sum=15, n=0
- ☐ sum=15, n= -1
- ☐ sum=14, n=0
- ☒ sum=14, n= -1
- ☐ sum=14, n=1

5.3. Вопросы для самостоятельного решения

Вопрос №1
КАКОЙ ТИП ПЕРЕМЕННОЙ ПРЕДСТАВЛЕН КЛЮЧЕВЫМ СЛОВОМ "**double**"?
Варианты ответа:

- ☐ вещественный с двойной точностью
- ☐ символьный
- ☐ вещественный с одинарной точностью
- ☐ целый

Вопрос №2
КАКОЕ ИМЯ ПЕРЕМЕННОЙ ЯВЛЯЕТСЯ НЕДОПУСТИМЫМ?
Варианты ответа:

- ☐ NameOfVariable
- ☐ _Variable
- ☐ Name_of_variable
- ☐ nv1
- ☐ int

Вопрос №3
СКОЛЬКО БАЙТ ПАМЯТИ ПРЕДОСТАВЛЯЕТСЯ ДЛЯ ПЕРЕМЕННОЙ ТИПА **long double?**
Варианты ответа:

- ☐ 8 байт
- ☐ 2 байт
- ☐ 4 байт
- ☐ 10 байт
- ☐ 1 байт

Вопрос №4
КАКОЙ ДИАПАЗОН ЗНАЧЕНИЙ ИМЕЕТ ПЕРЕМЕННАЯ ТИПА **unsigned long int?**
Варианты ответа:

- ☐ 0 - 4294967295
- ☐ 2.2e-308 - 1.8e+308
- ☐ 1.2e-38 - 3.4e+38
- ☐ 0 - 65535
- ☐ -2147483648 - +2147483647

Вопрос №5
КАКОЙ ДИАПАЗОН ЗНАЧЕНИЙ ИМЕЕТ ПЕРЕМЕННАЯ ТИПА **float?**
Варианты ответа:

- ☐ 0 - 4294967295
- ☐ 2.2e-308 - 1.8e+308
- ☐ 1.2e-38 - 3.4e+38
- ☐ 0 - 65535
- ☐ -2147483648 - +2147483647

Вопрос №6
КАКАЯ КОНСТАНТА ЯВЛЯЕТСЯ СИМВОЛЬНОЙ ?
Варианты ответа:

- ☐ 1.2E+02
- ☐ "stack"
- ☐ 65.3
- ☐ '3'
- ☐ 045

Вопрос №7
КАКАЯ КОНСТАНТА ЯВЛЯЕТСЯ ШЕСТНАДЦАТЕРИЧНОЙ?
Варианты ответа:

- ☐ 0x0
- ☐ 0
- ☐ 0738
- ☐ 0.0
- ☐ "000"

Вопрос №8
КАКАЯ КОНСТАНТА ЯВЛЯЕТСЯ ВОСЬМЕРИЧНОЙ?
Варианты ответа:

- ☐ 77
- ☐ 01
- ☐ 0E+7
- ☐ '34'
- ☐ 0X63

Вопрос №9
ЧТО ОЗНАЧАЕТ УПРАВЛЯЮЩАЯ ПОСЛЕДОВАТЕЛЬНОСТЬ ВИДА \\?
Варианты ответа:

- ☐ ОБРАТНАЯ НАКЛОННАЯ ЧЕРТА
- ☐ ГОРИЗОНТАЛЬНАЯ ТАБУЛЯЦИЯ
- ☐ НОВАЯ СТРОКА
- ☐ СИГНАЛ-ЗВОНОК
- ☐ ДВОЙНАЯ КАВЫЧКА

Вопрос №10
КАКАЯ КОНСТАНТА ЯВЛЯЕТСЯ СТРОКОВОЙ (СТРОКОВЫМ ЛИТЕРАЛОМ)?
Варианты ответа:

- ☐ 0
- ☐ 0111
- ☐ "Welcome to C"
- ☐ '9'
- ☐ 0XFF

Вопрос №11
НАЙТИ ПРАВИЛЬНОЕ ОБЪЯВЛЕНИЕ ПЕРЕМЕННЫХ
Варианты ответа:

- ☐ float lower upper step;
- ☐ float lower=0., LIMIT=300.;
- ☐ float eps = 0,25 step=20;
- ☐ FLOAT lower=0; LIMIT=1.5;
- ☐ float EPS=1E-20, LOWER=0;

Вопрос №12
ЧЕМУ РАВНО ВЫРАЖЕНИЕ 1/4 ?
Варианты ответа:

- ☐ 0.0
- ☐ 0,25
- ☐ 1
- ☐ 0
- ☐ 0.25

Вопрос №13
ЧЕМУ РАВНО ВЫРАЖЕНИЕ 1%4 ?
Варианты ответа:

- ☐ 2.5
- ☐ 4
- ☐ 1
- ☐ 0
- ☐ 0.25

Вопрос №14
КАКОЕ ЛОГИЧЕСКОЕ ВЫРАЖЕНИЕ РАВНО НУЛЮ (ЛОЖЬ)?
Варианты ответа:

- ☐ 1 >= 0
- ☐ 0 != 1
- ☐ 1 == 0
- ☐ 0 < 1
- ☐ 1 > 0

Вопрос №15
КАКОЕ ЛОГИЧЕСКОЕ ВЫРАЖЕНИЕ РАВНО ЕДИНИЦЕ (ИСТИНА)?
Варианты ответа:

- ☐ 0 != 0
- ☐ 0 > 1
- ☐ 1 >= 0
- ☐ 1 == 0
- ☐ 1 < 0

Вопрос №16
КАКОЕ ЛОГИЧЕСКОЕ ВЫРАЖЕНИЕ РАВНО НУЛЮ (ЛОЖЬ)?
Варианты ответа:

- ☐ 2 && 1
- ☐ 2 || 2
- ☐ 2 == 2
- ☐ 1 < 2
- ☐ 1 > 2

Вопрос №17
КАКОЕ ЛОГИЧЕСКОЕ ВЫРАЖЕНИЕ РАВНО ЕДИНИЦЕ (ИСТИНА)?
Варианты ответа:
- ☐ 0 != 0
- ☐ 0 && 1
- ☐ 1 && 0
- ☐ 0 == 0
- ☐ 1 < 0

Вопрос №18
ПУСТЬ ПЕРВОНАЧАЛЬНО a=1, в=2.
ЧЕМУ РАВНЫ x, a и b ПОСЛЕ ВЫПОЛНЕНИЯ ИНСТРУКЦИИ x= ++a - b++;
Варианты ответа:
- ☐ x= -1 , a= 2 , b=2
- ☐ x= -1 , a= 1 , b=2
- ☐ x= 0 , a= 1 , b=3
- ☐ x= 0 , a= 2 , b=2
- ☐ x= 0 , a= 2 , b=3

Вопрос №19
ПУСТЬ ПЕРВОНАЧАЛЬНО a=0, в=1.
ЧЕМУ РАВНЫ x, a и b ПОСЛЕ ВЫПОЛНЕНИЯ ИНСТРУКЦИИ x= ++a * b--;
Варианты ответа:
- ☐ x= 0 , a= 1 , b=0
- ☐ x= 1 , a= 0 , b=0
- ☐ x= 1 , a= 1 , b=1
- ☐ x= 1 , a= 1 , b=0
- ☐ x= 1 , a= 2 , b=0

Вопрос №20
ОПЕРАТОРЫ МАНИПУЛИРОВАНИЯ С БИТАМИ НЕ ПРИМЕНИМЫ К ОПЕРАНДАМ ТИПА:?
Варианты ответа:
- ☐ float
- ☐ int
- ☐ char
- ☐ long
- ☐ unsigned int

Вопрос №21
ЧЕМУ РАВЕН РЕЗУЛЬТАТ ПОБИТОВОЙ ОПЕРАЦИИ ~10001?
Варианты ответа:
- ☐ 10001
- ☐ 00000
- ☐ 01110
- ☐ 11111
- ☐ 11011

Вопрос №22
ЧЕМУ РАВЕН РЕЗУЛЬТАТ ПОБИТОВОЙ ОПЕРАЦИИ X>>2, ЕСЛИ X=10001?
Варианты ответа:
- ☐ 00010001
- ☐ 00000000
- ☐ 00000100
- ☐ 00000111
- ☐ 00000001

Вопрос №23
ЧЕМУ РАВЕН РЕЗУЛЬТАТ ПОБИТОВОЙ ОПЕРАЦИИ X<<1, ЕСЛИ X=10001?
Варианты ответа:
- ☐ 00111111
- ☐ 00010001
- ☐ 00100010
- ☐ 00101010
- ☐ 00111110

Вопрос №24
ПУСТЬ ПЕРВОНАЧАЛЬНО y=3, x=1. ЧЕМУ РАВЕН x В ВЫРАЖЕНИИ x -= y + 1;
Варианты ответа:
- ☐ x= 2
- ☐ x= 3
- ☐ x= 5
- ☐ x= -3
- ☐ x= -2

Вопрос №25
ПУСТЬ ПЕРВОНАЧАЛЬНО y=3, x=1. ЧЕМУ РАВЕН x В ВЫРАЖЕНИИ x %= y + 1;
Варианты ответа:

- ☐ x= 4
- ☐ x= 0.25
- ☐ x= 0.33333...
- ☐ x=0
- ☐ x= 1

Вопрос №26
ПУСТЬ ПЕРВОНАЧАЛЬНО y=3, x=2. ЧЕМУ РАВЕН x В ВЫРАЖЕНИИ x += y;
Варианты ответа:

- ☐ x= 2
- ☐ x= 1
- ☐ x= 0
- ☐ x= 4
- ☐ x= 5

Вопрос №27
ПУСТЬ ПЕРВОНАЧАЛЬНО y=3, x=2. ЧЕМУ РАВЕН x В ВЫРАЖЕНИИ x &= y - 1;
Варианты ответа:

- ☐ x= 0
- ☐ x= 1
- ☐ x= 2
- ☐ x= -1
- ☐ x= -2

Вопрос №28
ПУСТЬ ПЕРВОНАЧАЛЬНО y=0, x=3. ЧЕМУ РАВЕН x В ВЫРАЖЕНИИ x ^= y + 1;
Варианты ответа:

- ☐ x= 3
- ☐ x= 2
- ☐ x= 1
- ☐ x= 0
- ☐ x= -1

Вопрос №29
ПУСТЬ ПЕРВОНАЧАЛЬНО y=3, x=2. ЧЕМУ РАВЕН x В ВЫРАЖЕНИИ x /= y - 1.0;
Варианты ответа:

- ☐ x= 3
- ☐ x= 2
- ☐ x= 1
- ☐ x= 0
- ☐ x= -1

Вопрос №30
ПУСТЬ ПЕРВОНАЧАЛЬНО y=3, x=2. ЧЕМУ РАВЕН x В ВЫРАЖЕНИИ x /= y + 1.0;
Варианты ответа:

- ☐ x= 0.5
- ☐ x= 1
- ☐ x= 1.5
- ☐ x= 2
- ☐ x= 2.5

Вопрос №31
ПУСТЬ a=3, в=2. ЧЕМУ РАВНО z В УСЛОВНОМ ВЫРАЖЕНИИ z =(a < b)? a : b
Варианты ответа:

- ☐ 0
- ☐ 1
- ☐ 2
- ☐ 3
- ☐ 5

Вопрос №32
ВЫБЕРИТЕ ОПЕРАТОР С НАИБОЛЕЕ ВЫСОКИМ ПРИОРИТЕТОМ
Варианты ответа:

- ☐ &&
- ☐ ( )
- ☐ бинарный +
- ☐ унарный +
- ☐ = =

Вопрос №33
КАКОЙ ОПЕРАТОР ВЫПОЛНЯЕТСЯ СЛЕВА НАПРАВО?
Варианты ответа:
- ☐ ^=
- ☐ <<=
- ☐ >>=
- ☐ =
- ☐ = =

Вопрос №34
В КАКОЙ СТРОКЕ УСЛОВНОЙ КОНСТРУКЦИИ НАХОДИТСЯ ОШИБКА?
Фрагмент программы:
- ☐ if(x <= v[mid])
- ☐ high = mid -1;
- ☐ else;
- ☐ low = mid + 1 ;

Вопрос №35
ДАН ФРАГМЕНТ ПРОГРАММЫ. ЧЕМУ РАВНО s, ЕСЛИ a=5?

```
if( a >= 10 )  s = 0;
else if ( a < -5 )
      s = 1;
else  s = 2;
```

Варианты ответа:
- ☐ 0
- ☐ 1
- ☐ 2
- ☐ -5
- ☐ 10

Вопрос №36
В КАКОЙ СТРОКЕ КОНСТРУКЦИИ ПЕРЕКЛЮЧАТЕЛЯ НАХОДИТСЯ ОШИБКА?
Варианты ответа:
- ☐ switc (x){
- ☐ case '0': ++nother; break;
- ☐ case '1': --nwite; break;
- ☐ case '2': ndigit+2; break;
- ☐ default: x = 0; }

Вопрос №37
КАКОЙ ЗАГОЛОВОК ЦИКЛА ЗАПИСАН С ОШИБКОЙ?
Варианты ответа:
- ☐ for(i=0; i<10; i++)
- ☐ for(i=13;i>1; i--)
- ☐ for(; ;)
- ☐ for(i=1; i<=9; i--)
- ☐ for(i=0; i<=9; i++)

Вопрос №38
КАКОЙ ЗАГОЛОВОК ЦИКЛА ЗАПИСАН С ОШИБКОЙ?
Варианты ответа:
- ☐ for(n= strlen(s)-1; n>=0; n--)
- ☐ for(i=1; i<n; i++)
- ☐ for(;;)
- ☐ while((c= getchar( ))==' ' || c= = '\n' || c== '\t')
- ☐ while( )

Вопрос №39
ПЕРЕМЕННАЯ n ЦЕЛОЕ ЦИСЛО.
ЦИКЛ while((n /=10) >0) ВЫПОЛНЯЕТСЯ ДО ТЕХ ПОР ПОКА
Варианты ответа:
- ☐ n = 10
- ☐ ОСТАТОК ОТ ДЕЛЕНИЯ n/10 БОЛЬШЕ НУЛЯ
- ☐ n = 100
- ☐ РЕЗУЛЬТАТ ДЕЛЕНИЯ n/10 БОЛЬШЕ НУЛЯ
- ☐ 10 >0

Вопрос №40
ПЕРЕМЕННАЯ val = 1 , ЧТО НАПЕЧАТАЕТ ФРАГМЕНТ ПРОГРАММЫ

```
switch( val ){
case 1:   printf( "1");
case 2:   printf( "2");
case 3:   printf( "3");
default: printf( "4"); }
```

Варианты ответа:

- ☐ 1
- ☐ 2
- ☐ 3
- ☐ 4
- ☐ 1234

Вопрос №41

ПЕРЕМЕННАЯ val =1 , ЧТО НАПЕЧАТАЕТ ФРАГМЕНТ ПРОГРАММЫ

```
switch( val ){
case 1:   printf( "1");
case 2:   printf( "2");
case 3:   printf( "3");
case 4:   printf( "4"); break;
}
```

Варианты ответа:

- ☐ 1234
- ☐ 1
- ☐ 2
- ☐ 3
- ☐ 4

Вопрос №42

ПУСТЬ ПЕРЕМЕННАЯ a=5, d=6. ЧЕМУ РАВНА ПЕРЕМЕННАЯ max ПОСЛЕ ВЫПОЛНЕНИЯ УСЛОВНОЙ ИНСТРУКЦИИ

```
if( a<=d )
      max=b;
else
      max=a;
```

Варианты ответа:

- ☐ 5
- ☐ 6
- ☐ a
- ☐ b
- ☐ d

Вопрос №43

ПЕРВОНАЧАЛЬНО ПЕРЕМЕННАЯ x=1, a=0.5, b=-1. ЧЕМУ РАВНА ПЕРЕМЕННАЯ x ПОСЛЕ ВЫПОЛНЕНИЯ УСЛОВНОЙ ИНСТРУКЦИИ

```
if( x < a )     x= -a;
if( x > b )     x= b;   x= 1;
```

Варианты ответа:

- ☐ -1
- ☐ 0
- ☐ 0.5
- ☐ -0.5
- ☐ 1

Вопрос №44

ЧЕМУ РАВНЫ ПЕРЕМЕННЫЕ product И i ПОСЛЕ ВЫПОЛНЕНИЯ ФРАГМЕНТА ПРОГРАММЫ?

```
int  product=1, i= 0;
do
{
product  *= i;  i++;
}
while(i<=5);
```

Варианты ответа:

- ☐ product=120, i=6
- ☐ product=120, i=5
- ☐ product=24, i=5
- ☐ product=24, i=6
- ☐ product=1,   i=5

Вопрос №45

ЧЕМУ РАВНЫ ПЕРЕМЕННЫЕ sum И i ПОСЛЕ ВЫПОЛНЕНИЯ ФРАГМЕНТА ПРОГРАММЫ?

```
int i=5, sum= 0;
while( i >= 0) {
sum += i;
i--; }
```

Варианты ответа:

- ☐ sum=15, i=0
- ☐ sum=15, i=-1
- ☐ sum=14, i=1
- ☐ sum=14, i=0
- ☐ sum=14, i=-1

Вопрос №46

ЧЕМУ РАВНЫ ПЕРЕМЕННЫЕ sum И i ПОСЛЕ ВЫПОЛНЕНИЯ ФРАГМЕНТА ПРОГРАММЫ?

```
int i, sum= 0;
for( i=5; i >=0; i--)
sum += i;
```

Варианты ответа:

- ☐ sum=15, i=0
- ☐ sum=14, i=0
- ☐ sum=15, i=-1
- ☐ sum=14, i=-1
- ☐ sum=15, i=1

Вопрос №47

ЧЕМУ РАВНЫ ПЕРЕМЕННЫЕ product И n ПОСЛЕ ВЫПОЛНЕНИЯ ФРАГМЕНТА ПРОГРАММЫ?

```
int  n=1, product=1;
while( n < 3 ){
product *= n;  n += 1;
}
```

Варианты ответа:

- ☐ product=2, n=2
- ☐ product=2, n=3
- ☐ product=2, n=1
- ☐ product=6, n=2
- ☐ product=6, n=3

Вопрос №48

ЧЕМУ РАВНЫ ПЕРЕМЕННЫЕ sum И n ПОСЛЕ ВЫПОЛНЕНИЯ ФРАГМЕНТА ПРОГРАММЫ?

```
int n=5, sum=0;
while( n >= 0 ){
sum += n;
n -= 1;  }
```

Варианты ответа:

- ☐ sum=14, n=0
- ☐ sum=14, n=-1
- ☐ sum=15, n=-1
- ☐ sum=15, n=0
- ☐ sum=15, n=1

Вопрос №49

ЧЕМУ РАВНЫ ПЕРЕМЕННЫЕ product И n ПОСЛЕ ВЫПОЛНЕНИЯ ФРАГМЕНТА ПРОГРАММЫ?

```
int n, product=1;
for( n=5; n>=1; n--)
product *= n;
```

Варианты ответа:

- ☐ product=120, n=1
- ☐ product=120, n=0
- ☐ product=60, n=2
- ☐ product=15, n=1
- ☐ product=15, n=0

6.Ответы к вопросам для самоконтроля

Таблица 9.

№ Вопроса	Правильный ответ	№ Вопроса	Правильный ответ
1	2	26	4
2	1	17	4
3	5	28	1
4	3	29	4
5	1	30	3
6	4	31	5
7	4	32	5
8	3	33	4
9	5	34	2
10	5	35	2
11	1	36	2
12	2	37	1
13	2	38	1
14	3	39	4
15	4	40	3
16	3	41	3
17	2	42	5
18	4	43	2
19	5	44	3
20	1	45	5
21	4	46	2
22	4	47	4
23	5	48	2
24	4	49	2
25	1		

7.Список литературы

Основная литература.

1. Керниган Б., Ритчи Д. Язык программирования Си, 3-е изд. СПб.: «Невский Диалект», 2001, 352с.
2. Шаммас Н.К. Основы С++ и объектно-ориентированного программирования. Киев, Изд –во BHV, 1996, 433с.
3. Подбельский В.В., Фомин С.С. Программирование на языке Си. М.: Финансы и статистика, 2001, 600с.
4. Крупник А. Б. Изучаем Си. СПб.: Питер, 2001, 256с.
5. Крупник А. Б. Изучаем Си++. СПб.: Питер, 2003, 251с.
6. Джамса К. 1001 совет по С/С++. М.: Бином-Пресс, 1997, 784с.
7. Джамса К. Учимся программировать на языке С++. М.: Мир,1999, 320с.
8. Либерти Дж. Освой самостоятельно С++ за 21 день. М.: Изд. Дом «Вильямс», 2001, 816с.

Дополнительная литература

9. Шилдт Г. Самоучитель С++. СПб.: BHV –Санкт-Петербург, 1997, 511с.
10. Дейтел Х.М., Дейтел П.Дж. Как программировать на Си.3-е изд. М.: Бином-Пресс, 2002, 1168с.
11. Прата С. Язык программирования Си. Лекции и упражнения. СПб.: ООО «ДиаСофтЮП», 2002, 896с.
12. Костюкова Н.И. Программирование на языке Си. Методические рекомендации и задачи по программированию. Новосибирск: Сиб. унив. изд-во, 2003, 158с.
13. Ашарина И.В. Основы программирования на языках С и С++. М.: Горячая линия-Телеком, 2002, 207с.
14. Киммел П. И др. BorlandC++5. СПб.: BHV –Санкт-Петербург, 2001, 976с.
15. Болски М.И. Язык программирования Си. Справочник. М.: Радио и связь, 1988, 96с.

8. Рабочая программа дисциплины “Информатика и ПАЯ” семестр 1

№ п/п	Раздел дисциплины	Лекции
1.	Введение в языки программирования C/C++. Среда программирования Borland C++ ver3.1. Простейшая программа Hello, world.	**1**
2.	Обзор основных конструкций языка C на примере программы перевода градусов Фаренгейта в градусы Цельсия. Версии программы: с переменными типа int, с типом float, для повышения точности расчета, компактная с циклом for, “легко читаемая” с именованными константами.	**1**
3.	Комментарий. Имена переменных. Типы и размеры данных, модификаторы типов. Типы констант: целая десятичная, восьмеричная, шестнадцатеричная, с плавающей точкой. Суффиксы L(l), U(u), F(f) констант. Символьная константа и управляющая последовательность. Строковая константа (строковый литерал). Константы перечисления.	**1**
4.	Объявления переменных. Арифметические операторы. Операторы отношения и логические операторы. Преобразования типов. Операторы инкремента и декремента. Общие сведения о приоритетах операторов. Программа squeeze (K&R)*.	**1**
5.	Управление. Условные конструкции: if, if-else, вложенные конструкции else-if. Переключатель switch. Программа binsearch (K&R).	**1**
6.	Циклы while, for, do-while. Множественная инициализация и приращение счетчиков цикла for. Инструкции break и continue. Инструкция goto и метки. Почему следует избегать оператора goto. Программа trim, atoi , shellsort (K&R).	**1**
7.	Побитовые операторы (побитовое И, ИЛИ, исключающее ИЛИ, НЕ, сдвиг влево, сдвиг вправо). Упрощенная форма оператора присваивания. Программа getbits, bitcount (K&R).	**1**
8.	Условное выражение: (z = (a > b)? a :b). Приоритет и очередность вычислений операторов. Программы подсчета символов, строк и слов (K&R).	**1**
9.	Си-препроцессор. Подключение заголовочных файлов директивой #include. Макроподстановки с помощью #define и отмена макроподстановки #undef. Директива #define с параметрами, оператор конкатенации ##. Директивы условной компиляции #if, #elif, #else, #endif. Конструкция #if !defined (имя макроса) … #endif.	**1**

* Примечание. Ссылка на книгу Керниган Б., Ритчи Д. Язык программирования Си, 3-е изд. СПб.: «Невский Диалект», 2001, 352с.

ЛАБОРАТОРНЫЙ ПРАКТИКУМ

№ п/п	Наименование лабораторных работ
	СЕМЕСТР 1
1.	Запуск интегрированной среды программирования (IDE) из Windows NT. Ознакомление с работой IDE Borland C++ ver 3.1. Составление программы Hello, world. Освоение функций редактирования, компилирования и запуска исполняемого кода. Визуализация результатов работы программы.
2.	Составление, отладка и запуск программы перевода градусов Фаренгейта в градусы Цельсия. Повышение эффективности кода и удобочитаемости составленной программы.
3.	Составление, отладка и запуск программ, вычисляющих простые арифметические выражения на основе стандартной математической библиотеки С.
4.	Составление, отладка и анализ программ с линейными алгоритмами для вычисления высоты, биссектрисы и медианы произвольного треугольника с использованием теорем синусов и косинусов.
5.	Составление, отладка и запуск программ с линейными алгоритмами для перевода углов, данных в градусах, минутах, секундах в вещественное число радианов и обратно.
6.	Составление, отладка и запуск программ на различные типы циклов, включая вложенные, while, for, do – while для нахождения различных сумм и произведений.
7.	Составление, отладка и запуск программ на простейшие типы прямой и обратной геодезической задачи.
8.	Приемка домашней расчетно-графической работы №1. Анализ типовых ошибок, выявленных в программах. Анализ вариантов домашнего типового задания.
9.	Тестирование знаний основ языка С/С++ за первый семестр с помощью программы ТЕСТ.

Содержание