Министерство образования и науки Российской Федерации

федеральное государственное автономное образовательное
учреждение высшего образования
«Санкт-Петербургский политехнический университет Петра Великого»
(ФГАОУ ВО «СПбПУ»)
Институт среднего профессионального образования

Е.А. Кураева

МЕТОДИЧЕСКИЕ УКАЗАНИЯ И ЗАДАНИЯ ПО ВЫПОЛНЕНИЮ ДОМАШНЕЙ КОНТРОЛЬНОЙ РАБОТЫ

для студентов специальности
09.02.03 «Программировнаие в компьютерных системах»

ПО ДИСЦИПЛИНЕ «СИСТЕМНОЕ ПРОГРАММИРОВАНИЕ»

Санкт-Петербург
2019/2020

СОДЕРЖАНИЕ

Теоретические сведения

В соответствии с учебным планом, реализация индивидуальных заданий производится на языке C++.

C++ — компилируемый, статически типизированный язык программирования общего назначения.

Поддерживает такие парадигмы программирования, как процедурное программирование, объектно-ориентированное программирование, обобщённое программирование. Язык имеет богатую стандартную библиотеку, которая включает в себя распространённые контейнеры и алгоритмы, ввод-вывод, регулярные выражения, поддержку многопоточности и другие возможности. C++ сочетает свойства как высокоуровневых, так и низкоуровневых языков.В сравнении с его предшественником — языком C, — наибольшее внимание уделено поддержке объектно-ориентированного и обобщённого программирования.

C++ широко используется для разработки программного обеспечения, являясь одним из самых популярных языков программирования. Область его применения включает создание операционных систем, разнообразных прикладных программ, драйверов устройств, приложений для встраиваемых систем, высокопроизводительных серверов, а также игр. Существует множество реализаций языка C++, как бесплатных, так и коммерческих и для различных платформ. Например, на платформе x86 это GCC, Visual C++, Intel C++ Compiler, Embarcadero (Borland) C++ Builder и другие. C++ оказал огромное влияние на другие языки программирования, в первую очередь на Java и C#.

Синтаксис C++ унаследован от языка C. Одним из принципов разработки было сохранение совместимости с C. Тем не менее, C++ не является в строгом смысле надмножеством C; множество программ, которые могут одинаково успешно транслироваться как компиляторами C, так и компиляторами C++, довольно велико, но не включает все возможные программы на C.

Стандарт C++ состоит из двух основных частей: описание ядра языка и описание стандартной библиотеки.

Первое время язык развивался вне формальных рамок, спонтанно, по мере встававших перед ним задач. Развитию языка сопутствовало развитие кросс-компилятора cfront. Новшества в языке отражались в изменении номера версии кросс-компилятора. Эти номера версий кросс-компилятора распространялись и на сам язык, но применительно к настоящему времени речь о версиях языка C++ не ведут. Лишь в 1998 году язык стал стандартизированным.

C++ поддерживает как комментарии в стиле C (/* комментарий */), так и однострочные (// вся оставшаяся часть строки является комментарием), где // обозначает начало комментария, а ближайший последующий символ новой строки, который не предварён символом \ (либо эквивалентным ему обозначением ??/), считается окончанием комментария. Плюс этого комментария в том, что его не обязательно заканчивать, то есть обозначать окончание комментария.

Спецификатор inline для функций. Функция, определённая внутри тела класса, является inline по умолчанию. Данный спецификатор является подсказкой компилятору и может встроить тело функции в код вместо её непосредственного вызова.

Квалификаторы const и volatile. В отличие от C, где const обозначает только доступ на чтение, в C++ переменная с квалификатором const должна быть инициализирована. volatile используется в описании переменных и информирует компилятор, что значение данной переменной может быть изменено способом, который компилятор не в состоянии отследить. Для переменных, объявленных volatile, компилятор не должен применять средства оптимизации, изменяющие положение переменной в памяти (например, помещающие её в регистр) или полагающиеся на неизменность значения переменной в промежутке между двумя присваиваниями ей значения. В многоядерной системе volatile помогает избегать барьеров памяти 2-го типа[источник не указан 2345 дней].

Пространства имён (namespace). Пример:

```
namespace Foo
{
  const int x=5;
}
const int y = Foo::x;
```

Специальным случаем является безымянное пространство имён. Все имена, описанные в нём, доступны только в текущей единице трансляции и имеют локальное связывание. Пространство имён std содержит в себе стандартные библиотеки C++.

Для работы с памятью введены операторы new, new[], delete и delete[]. В отличие от библиотечных malloc и free, пришедших из C, данные операторы производят инициализацию объекта. Для классов это вызов конструктора, для POD типов инициализацию можно либо не проводить(new Pod;), либо провести инициализацию нулевыми значениями (new Pod(); new Pod{};).

Типы

В C++ доступны следующие встроенные типы. Типы C++ практически полностью повторяют типы данных в C:

Символьные: char, wchar_t (char16_t и char32_t, в стандарте C++11).

Целочисленные знаковые: signed char, short int, int, long int (и long long, в стандарте C++11).

Целочисленные беззнаковые: unsigned char, unsigned short int, unsigned int, unsigned long int(и unsigned long long, в стандарте C++11).

С плавающей точкой: float, double, long double.

Логический: bool, имеющий значения true или false.

Операции сравнения возвращают тип bool. Выражения в скобках после if, while приводятся к типу bool.[13]

Язык ввёл понятие ссылок, а начиная с одиннадцатой версии стандарта rvalue-ссылки и передаваемые ссылки (англ. forwarding reference). (см. Ссылка (C++))

C++ добавляет к C объектно-ориентированные возможности. Он вводит классы, которые обеспечивают три самых важных свойства ООП: инкапсуляцию, наследование и полиморфизм.

В стандарте C++ под классом (class) подразумевается пользовательский тип, объявленный с использованием одного из ключевых слов class, struct или union, под структурой (structure) подразумевается класс, определённый через ключевое слово struct, и под объединением (union) подразумевается класс, определённый через ключевое слово union.

В теле определения класса можно указать как объявления функций, так и их определение. В последнем случае функция является встраиваемой (inline)). Нестатические функции-члены могут иметь квалификаторы const и volatile, а также ссылочный квалификатор (& или &&).

Наследование

C++ поддерживает множественное наследование. Базовые классы (классы-предки) указываются в заголовке описания класса, возможно, со спецификаторами доступа. Наследование от каждого класса может быть публичным, защищённым или закрытым:

По умолчанию базовый класс наследуется как private.

В результате наследования класс-потомок получает все поля классов-предков и все их методы; можно сказать, что каждый экземпляр класса-потомка содержит подэкземпляр каждого из классов-предков. Если один класс-предок наследуется несколько раз (это возможно, если он является предком нескольких базовых классов создаваемого класса), то экземпляры класса-потомка будет включать столько же подэкземпляров данного класса-предка. Чтобы избежать такого эффекта, если он нежелателен, C++ поддерживает концепцию виртуального наследования. При наследовании базовый класс может объявляться виртуальным; на все виртуальные вхождения класса-предка в дерево наследования класса-потомка в потомке создаётся только один подэкземпляр.

Полиморфизм

C++ поддерживает динамический полиморфизм и параметрический полиморфизм.

Параметрический полиморфизм представлен:

Аргументами по умолчанию для функций. К примеру, для функции void f(int x, int y=5, int z=10), вызовы f(1), f(1,5) и f(1,5,10) эквивалентны.

Перегрузка функций: функция с одним именем может иметь разное число и разные по типу аргументы. Например :

void Print(int x);

void Print(double x);

void Print(int x, int y);

Частным случаем перегрузки функций можно считать перегрузку операторов.

Механизмом шаблонов

Динамический полиморфизм реализуется с помощью виртуальных методов и иерархии наследования. Полиморфным в C++ является тип имеющий хотя бы один виртуальный метод. Пример иерархии:

```
class Figure
{
public:
   virtual void Draw() = 0; // чистый виртуальный метод
   virtual ~Figure();          // при наличии хотя бы одного виртуального метода деструктор следует сделать виртуальным
};

class Square : public Figure
{
public:
   void Draw() override;
};

class Circle : public Figure
{
public:
   void Draw() override;
};
```

Здесь класс Figure является абстрактным (и, даже, интерфейсным), так как метод Draw не определён. Объекты данного класса нельзя создать, зато можно использовать ссылки или указатели с типом Figure. Выбор реализации метода Draw будет производиться во время выполнения исходя из реального типа объекта.

Инкапсуляция

Инкапсуляция в C++ реализуется через указание уровня доступа к членам класса: они бывают публичными (открытыми, public), защищёнными (protected) и приватными (закрытыми, private). В C++ структуры формально отличаются от классов лишь тем, что по умолчанию уровень доступа к членам класса и тип наследования у структуры публичные, а у класса — приватные.

Доступ	private	protected	public
Сам класс	да	да	да
Друзья	да	да	да
Наследники	нет	да	да
Извне	нет	нет	да

Проверка доступа происходит во время компиляции, попытка обращения к недоступному члену класса вызовет ошибку компиляции.

Друзья

Функции-друзья — это функции, не являющиеся функциями-членами и тем не менее имеющие доступ к защищённым и закрытым членам класса. Они должны быть объявлены в теле класса как friend. Например:

```
class Matrix {
   friend Matrix Multiply(Matrix m1, Matrix m2);
};
```

Здесь функция Multiply может обращаться к любым полям и функциям-членам класса Matrix.

Дружественным может быть объявлен как весь класс, так и функция-член класса. Четыре важных ограничения, накладываемых на отношения дружественности в C++:

Дружественность не транзитивна. Если A объявляет другом B, а B, в свою очередь, объявляет другом C, то C не становится автоматически другом для A. Для этого A должен явно объявить C своим другом.

Дружественность не взаимна. Если класс A объявляет другом класс B, то он не становится автоматически другом для B. Для этого должно существовать явное объявление дружественности A в классе B.

Дружественность не наследуется. Если A объявляет класс B своим другом, то потомки B не становятся автоматически друзьями A. Для этого каждый из них должен быть объявлен другом A в явной форме.

Дружественность не распространяется на потомков. Если класс A объявляет B другом, то B не становится автоматически другом для классов-потомков A. Каждый потомок, если это нужно, должен объявить B своим другом самостоятельно.

В общем виде это правило можно сформулировать следующим образом: «Отношение дружественности существует только между теми классами (классом и функцией), для которых оно явно объявлено в коде, и действует только в том направлении, в котором оно объявлено».

Специальные функции

Класс по умолчанию может иметь шесть специальных функций: конструктор по умолчанию, конструктор копирования, конструктор перемещения, деструктор, оператор присваивания копированием, оператор присваивания перемещением. Также можно явно определить их все (см. Правило трёх).

```
class Array {
public:
	Array() = default; // компилятор создаст конструктор по-умолчанию сам
	Array(size_t _len) :
			len(_len) {
		val = new double[_len];
	}
	Array(const Array & a) = delete;  // конструктор копирования явно удалён
	Array(Array && a); // конструктор перемещения
	~Array() {
		delete[] val;
	}
	Array& operator=(const Array& rhs); // оператор присваивания копированием
	Array& operator=(Array&& rhs); // оператор присваивания перемещением
	double& operator[](size_t i) {
		return val[i];
```

```
    }
    const double& operator[](size_t i) const {
        return val[i];
    }

protected:
    std::size_t len {0}; // инициализация поля
    double* val {nullptr};
};
```

Конструктор вызывается для инициализации объекта (соответствующего типа) при его создании, а деструктор — для уничтожения объекта. Класс может иметь несколько конструкторов, но деструктор может иметь только один. Конструкторы в C++ не могут быть объявлены виртуальными, а деструкторы — могут, и обычно объявляются для всех полиморфных типов, чтобы гарантировать правильное уничтожение доступного по ссылке или указателю объекта независимо от того, какого типа ссылка или указатель. При наличии хотя бы у одного из базовых классов виртуального деструктора, деструктор класса потомка автоматически становится виртуальным.

Шаблоны

Основная статья: Шаблоны C++

Шаблоны позволяют порождать функции и классы, параметризованные определённым типом или значением. Например, предыдущий класс мог бы реализовывать массив для любого типа данных:

```
template <typename T>
class Array {
  ...
  T& operator[](size_t i) {
        return val[i];
    }
protected:
    std::size_t len {0}; // инициализация поля
    T* val {nullptr};
};
```

Стандартная библиотека

Общая структура

Стандартная библиотека C++ включает в себя набор средств, которые должны быть доступны для любой реализации языка, чтобы обеспечить программистам удобное пользование языковыми средствами и создать базу для разработки как прикладных приложений самого широкого спектра, так и специализированных библиотек. Стандартная библиотека C++ включает в себя часть стандартной библиотеки C. Стандарт C++ содержит нормативную ссылку на стандарт C от 1990 года и не определяет самостоятельно те функции стандартной библиотеки, которые заимствуются из стандартной библиотеки C.

Доступ к возможностям стандартной библиотеки C++ обеспечивается с помощью включения в программу (посредством директивы #include) соответствующих стандартных заголовочных файлов. Всего в стандарте C++11 определено 79 таких файлов. Средства стандартной библиотеки объявляются как входящие в пространство имён std. Заголовочные файлы, имена которых соответствуют шаблону «cX», где X — имя заголовочного файла стандартной библиотеки C без расширения (cstdlib, cstring, cstdio и пр.), содержат объявления, соответствующие данной части стандартной библиотеки C.

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

Тематический план и содержание учебной дисциплины

ОФОРМЛЕНИЕ КОНТРОЛЬНОЙ РАБОТЫ

Контрольная работа сдается в печатном виде. Образец титульного листа приводится в (Приложении № 1)

Наиболее распространённая ошибка, которую допускают студенты, - несоответствие содержания теоретической части теме контрольной работы. В этом случае работа не зачитывается, и студент не допускается к дифференцированному зачёту.

Контрольная работа оформляется в прозрачной папке с файлами.

При оформлении должно быть наличие:

1. титульного листа (см. Приложение 1)
2. текста работы (введение, термины, аналитическая часть, заключение);
3. нумерации контрольной работы арабскими цифрами. Первой страницей считается титульный лист, но на нем не ставят номер страницы. Нумерация начинается со второй страницы.
4. списка использованной литературы (должно быть не менее пяти источников), списки литературы по психологии и этики не старше 7 лет, то есть источники 2010, 2011, 2012,2013, 2014, 2015, 2016 и 2017 года.
5. листа для рецензии преподавателя.

Содержание должно включать:

1. формулировку актуальности темы;
2. объём работы – не менее 7 страниц печатного текста, шрифт должен быть Times New Roman 14;
3. полное раскрытие содержания каждого раздела;
4. необходимые примечания, пояснения, схемы, таблицы;

Работа распечатывается на компьютере на листах формата А 4;

- междустрочный интервал равен 1,5;
- интервал между заголовком и следующим за ним текстом равен 3;
- текст выравнивается по ширине.

Размер полей:- левое 30 мм; - правое 10 мм; - верхнее и нижнее 20 мм. Абзацный отступ - 1,25 см.

Каждая *структурная часть работы начинается с новой страницы*. Заголовок располагается посредине строки, точка не ставится. Также не допускается подчеркивание заголовка и переносы в словах заголовка. Номера страниц проставляются внизу в середине страницы. Титульный лист включается в общую нумерацию, но номер страницы не проставляется. Содержание будет являться 2 стр. Вашей ДКР.

Реферируемый источник, списки использованной литературы, а также все ссылки на литературные работы должны быть оформлены следующим образом: фамилия и инициалы автора, название источника, место и год издания; для журнальных статей необходимо указать название журнала, год издания и номер.

Структура работы:

- титульный лист
- оглавление
- введение
- практическая часть (вопрос 1, вопрос 2)
- заключение
- список литературы
- приложения- если это нужно (таблицы и т.д.)

Заголовки структурных элементов контрольной работы (содержание, название разделов, список литературы, приложения) печатаются заглавными буквами без точки на конце.

Содержание, разделы, список литературы, приложения начинаются с новой страницы.

Домашняя контрольная работа для студентов-заочников являются итогом их самостоятельной работы над учебным материалом, а также средством самоконтроля.

Каждый заочник обязан выполнить в установленный срок и привезти в колледж на проверку контрольную работу. По всем вопросам, возникающим в ходе ее выполнения, следует обращаться к ведущему преподавателю данной дисциплины -Никитиной М.В. (аудитория 119 по адресу: СПб, Приморский пр. д.63, на заочном отделении уточнить дни, когда преподаватель в колледже). Или по средам в течении учебного года на площадке пр. Энгельса д.23, на заочном отделении уточнить есть ли преподаватель в колледже.

Из вышесказанного следует, что домашняя контрольная работа – это своеобразный письменный экзамен, который требует серьёзной подготовки.

ВАРИАНТЫ КОНТРОЛЬНЫХ РАБОТ

ВАРИАНТ 1

Напишите программу, которая запрашивает у пользователя номер месяца и затем выводит соответствующее название времени года. В случае, если пользователь введёт недопустимое число, программа должна вывести сообщение об ошибке.

Пример выполнения программы:

```
Введите номер месяца (число от 1 до 12): 12
Зима
```

ВАРИАНТ 2

Написать программу реализующую игру «Угадай число». Компьютер загадывает число от `0` до `999` (используйте генерацию случайных чисел), а пользователь угадывает его. На каждом шаге угадывающий делает предположение, а задумавший число — говорит сколько цифр из числа угаданы и сколько из угаданных цифр занимают правильные позиции в числе. Например, если задумано число `725` и выдвинуто предположение, что задумано число `523`, то угаданы две цифры (`5` и `2`) и одна из них занимает верную позицию. Например:

```
Компьютер загадал трёхзначное число. Вы должны его
отгадать. После очередного числа вам будет сообщено,
сколько цифр угадано и сколько из них находятся на
своих местах.
Ваш вариант: 123
Угадано: 0. На своих местах: 0
Ваш вариант: 456
Угадано: 1. На своих местах: 0
Ваш вариант: 654
Угадано: 2. На своих местах: 2
Ваш вариант: 657
Угадано:  2. На своих местах: 2
Ваш вариант: 658
Угадано: 3. На своих местах: 3
***Вы угадали число 658!***
```

ВАРИАНТ 3

Напишите программу-телеграф, которая принимает от пользователя сообщение и выводит его на экран в виде последовательности точек и тире. Вывод точек и тире можно сопровождать звуковым сигналом соответствующей длительности. Азбука Морзе для букв русского алфавита приведена ниже.

Буква	Код	Буква	Код	Буква	Код	Буква	Код
А	·–	Б	–···	В	·––	Г	––·
Д	–··	Е	·	Ж	···–	З	––··
И	··	Й	·–––	К	–·–	Л	·–··
М	––	Н	–·	О	–––	П	·––·
Р	·–·	С	···	Т	–	У	··–
Ф	··–·	Х	····	Ц	–·–·	Ч	–––·
Ш	––––	Щ	––·–	Ъ	·––·–·	Ы	–·––
Ь	–··–	Э	··–··	Ю	··––	Я	·–·–

ВАРИАНТ 4

Написать программу, которая объединяет два упорядоченных по возрастанию массива в один, также упорядоченный, массив.

Пример выполнения программы:

```
Введите элементы первого массива: 1 3 5 7 9
Введите элементы второго массива: 2 4 6 8 10
Массив-результат: 1 2 3 4 5 6 7 8 9 10
```

ВАРИАНТ 5

Напишите программу, которая содержит текущую информацию о заявках на авиабилеты. Каждая заявка должна иметь:

пункт назначения;

номер рейса;

ФИО пассажира;

желаемую дату вылета.

Программа должна обеспечивать:

хранение всех заявок в виде списка;

добавление и удаление заявок;

вывод всех заявок.

ВАРИАНТ 6

Напишите программу учёта оценок студентов. Для этого создайте текстовый файл с именем `input_data.txt`, содержащий список из 10 студентов и их оценки по трём предметам: математике, физике, информатике. Содержимое файла:

в первой строке находится общее количество студентов;

в каждой последующей строке находится ФИО студента и три целых числа (оценки);

данные в строке разделены пробелами, а оценки варьируются в диапазоне от 1 до 5.

Затем создайте класс, с помощью которого вы будете считывать данные с файла. На экран выведите ФИО студентов с оценками в порядке убывания их среднего бала.

ВАРИАНТ 7

Рейтинг бакалавра заочного отделения при поступлении в магистратуру определяется средним баллом по диплому, умноженным на коэффициент стажа работы по специальности, который равен: нет стажа — 1, меньше 2-ух лет — 13, от 2 до 5 лет — 16. Написать программу расчёта рейтинга студента при заданном среднем балле диплома (от 3 до 5) и вывести сообщение о приёме в магистратуру (при проходном балле 45).

ВАРИАНТ 8

Используя пять вариантов наборов чисел:

Набор №1: `6, 7, 8`

Набор №2: `7, 8, 9`

Набор №3: `6, 9, 10`

Набор №4: `6, 9, 8`

Набор №5: `7, 6, 10`

Сыграйте с компьютером в игру. Введите с клавиатуры свой вариант и сравните с вариантом компьютера, который выбирается рандомно из 5 допустимых наборов. Если сумма цифр вашего варианта больше суммы цифр варианта компьютера, то вы выиграли и наоборот.

ВАРИАНТ 9

Описать структуру с именем `STUDENT`, содержащую поля:

`Name` — фамилия и инициалы;

`Year` — курс;

`Rating` — успеваемость (массив из пяти элементов).

Написать программу, выполняющую:

ввод с клавиатуры данных в массив `STUD`, состоящий из 10 структур типа `STUDENT`, записи должны быть упорядочены по алфавиту;

вывод на экран записей, упорядоченного списка студентов, средний бал которых превышает общий средний бал;

если таких студентов нет — выдать соответствующее сообщение.

ВАРИАНТ 10

Известно, что сейф открывается при правильном вводе кода из 3-х цифр в диапазоне от 0 до 9. Задайте код и затем откройте сейф, используя метод перебора с помощью цикла for.

Пример выполнения программы:

```
Откроем сейф методом перебора:
код = 738, потребовалось 3026 испытаний
```

ВАРИАНТ 11

Напишите программу, которая вычисляет среднее арифметическое вводимой пользователем с клавиатуры последовательности дробных чисел.

Пример выполнения программы:

```
Введите последовательность дробных чисел: 5.4 7.8 3.0
1.5 2.3
Среднее арифметическое введенной последовательности:
4.0
```

ВАРИАНТ 12

Напишите программу, которая проверяет, является ли год високосным (кратным 4) в пределах от 2000 лет
до нашей эры и до 2000 лет нашей эры.

Пример выполнения программы:

```
Введите год и эру: 656 год нашей эры
Этот год является високосным
```

ВАРИАНТ 13

Напишите программу, моделирующую бросание монеты с помощью генерации случайных чисел. После каждого броска монеты, программа должна записывать в файл результат: `Орёл` или `Решка`. Выполните бросок монеты 100 раз и подсчитайте, сколько раз появилась каждая сторона монеты.

ВАРИАНТ 14

Напишите программу, которая считывает из файла целые числа,
которые рандомно генерируются в диапазоне от 1 до 72. Для каждого считанного числа ваша программа должна вывести строку, содержащую соответствующее количество звёздочек. Например, если ваша программа считала из файла число `7`, то она должна вывести 7 звёздочек: `*******`.

Пример выполнения программы:

```
Числа из файла: 3 17 48 52 46 58 59 64 57
Результат:
Число №1 = 3 ***
Число №2 = 17 ****************
Число №3 = 48 ************************************************
Число №4 = 52 ****************************************************
Число №5 = 46 **********************************************
Число №6 = 58 ******************************************************
Число №7 = 59 ***********************************************
Число №8 = 64 ************************
Число №9 = 57 *********************************************
```

ВАРИАНТ 15

Возьмите любое слово, например, «корова». Используя генерацию случайных чисел, переставьте буквы этого слова в рандомном порядке. Делайте это до тех пор, пока полученное слово не совпадёт с начальным словом. Используя массив, укажите при перестановке букв их индексы. Программа должна корректно работать с любым словом.

Пример выполнения программы:

```
Введите слово: корова
[0] 435021 воакро      [1] 430215 вокроа      [2]
521340 ароовк
[3] 025134 краоов      [4] 104532 окваор      [5]
024531 крваоо
[6] 214305 ровока      [7] 152034 оарков      [8]
130542 оокавр
[9] 031524 кооарв     [10] 503421 аковро     [11]
310425 ооквра
[12] 412035 воркоа    [13] 140532 овкаор     [14]
402513 вкраоо
[15] 124530 орваок    [16] 452130 варook     [17]
423105 вроока
[18] 134520 ооварк    [19] 104352 оквоар     [20]
302415 окрвоа
[21] 203541 ркоаво    [22] 231504 рооакв     [23]
023541 кроаво
[24] 504132 аквоор    [25] 423015 врокоа     [26]
514320 аоворк
[27] 012345 корова
```

ВАРИАНТ 16

На первом курсе `M = 40` студентов. Каждый из них в понедельник получает оценку по программированию, во вторник — оценку по математике, в среду — по физике в пределах от 2 до 5 каждая. Всего в году `N = 35` недель. Лучшим считается студент, который наибольшее количество недель продержался без троек (т.е. получал не ниже 4). Сформируйте три целых массива нужного размера. Задайте оценки с помощью генерации случайных чисел. Найдите лучшего студента.

ВАРИАНТ 17

Контрольно-обучающая система. Напишите интерактивный учебник биологии. Он должен спрашивать у пользователя в случайном порядке 5 вопросов по биологии. Например:

```
Вопрос №1: "Что такое курица?"
Варианты ответа:
1) Рыба
2) Насекомое
3) Птица
4) Земноводное
5) Растение
Ваш выбор: 3
Верно! Правильный ответ — "Птица".
```

После опроса поставьте испытуемому оценку.

ВАРИАНТ 18

В поезде 18 вагонов, в каждом из которых 36 мест. Информация о проданных на поезд билетах хранится в двумерном массиве, номера строк которых соответствуют номерам вагонов, а номера столбцов — номерам мест. Если билет на то или иное место продан, то соответствующий элемент массива имеет значение `1`, в противном случае — `0`. Напишите программу, определяющую число свободных мест в любом из вагонов поезда.

ВАРИАНТ 19

Написать программу проверки знания таблицы умножения. Программа должна вывести 10 примеров и выставить оценку: за 10 правильных ответов — «отлично», за 8 или 9 правильных ответов — «хорошо», за 6 или 7 правильных ответов — «удовлетворительно», остальные варианты — «плохо».

ВАРИАНТ 20

Игра «100 спичек». Из кучки, первоначально содержащей 100 спичек, двое играющих поочерёдно берут по несколько спичек: не менее одной и не более десяти. Проигрывает тот, кто взял последнюю спичку. Количество спичек, которое берёт компьютер, определите с помощью генерации случайных чисел.

ВАРИАНТ 21

Напишите генератор паролей. Составьте три уровня сложности генерации паролей (вместе с их длиной) и спрашивайте у пользователя, какой уровень сложности ему нужен. Проявите свою изобретательность: надёжные пароли должны состоять из сочетания строчных букв, прописных букв, цифр и символов.

Пароли должны генерироваться случайным образом каждый раз, когда пользователь запрашивает новый пароль.

ВАРИАНТ 22

Напишите игру «Корова и быки». Правила:

программа генерирует случайным образом 4-значное число;

пользователю предлагают угадать сгенерированное программой число;

за каждую угаданную пользователем цифру в её правильном положении, он получает «корову»;

за каждую угаданную пользователем цифру в неправильном месте, он получает «быка»;

после каждого предположения пользователю должно выводиться количество «коров» и «быков», которые он заработал;

игра окончена тогда, когда пользователь угадал все цифры.

Например, компьютер загадал число `9978`:

```
Добро пожаловать в игру «Коровы и быки»!
Введите число:
9965
2 коровы, 0 быков
9989
2 коровы, 1 бык
...
```

ВАРИАНТ 23

Вы, в качестве пользователя, загадываете число в своей голове от 0 до 100. Программа должна его угадать, делая предположения, а вы должны сообщить ей, является ли её число слишком большим, слишком маленьким или `Правильно, угадал!`.

В конце программа должна вывести на экран количество предположений, которые ей потребовались для того, чтобы угадать ваше число.

Примечание: Вам, как программисту, придётся выбирать стратегию угадывания компьютером числа пользователя. Самая простая стратегия заключается в переборе чисел от 0 до 100 (например: `1`, `2`, `3` и т.д.), но это очень долго и нелепо. Лучшим вариантом было бы делить диапазон на 2:

начинаем с `50`;

если число пользователя больше, то опять делим диапазон оставшихся чисел на 2 + добавляем к предыдущему предположению, получая, таким образом, `75`;

если число пользователя меньше, то указываем `25`;

и, таким образом, делим диапазон до тех пор, пока не доберёмся к верному результату.

У вас также может быть и другая/своя стратегия

ВАРИАНТ 24

Давайте напишем популярную игру, которая называется «Виселица». В игре вам нужно угадать слово, которое загадала программа, буква за буквой. Игрок угадывает одну букву за раз и может ошибиться только 6 раз (после этого он сразу же проигрывает).

Необходимый функционал вашей программы:

создайте массив слов (например, поместите туда 40 слов) и, рандомным образом, выберите 1 слово для угадывания;

программа должна выводить длину всего слова и отображать буквы, которые угадал игрок;

после каждого неудачного угадывания, программа должна сообщить игроку, сколько у него осталось попыток неверно указать букву, прежде чем он проиграет;

если человек указал букву, которую ранее уже угадывал, и она не дублируется в слове, то не наказывайте его, а просто предоставьте возможность угадать букву ещё раз.

Например, компьютер загадал слово `INTERESTING`:

```
Добро пожаловать в игру "Виселица"!
Слово - _ _ _ _ _ _ _ _ _ _
Угадайте букву: S
Верно - _ _ _ _ _ _ S _ _ _ _
Угадайте следующую букву: F
Неверно! Такой буквы нет, у вас осталось 5 попыток
неверно указать букву!
...
```

ВАРИАНТ 25

Напишите программу, которая запрашивает у пользователя строку, содержащую несколько слов. Затем выведите пользователю ту же строку, но в обратном порядке. Например:

```
Введите строку:
Меня зовут Анатолий!
Результат:
!Анатолий зовут Меня
```

ВАРИАНТ 26

Напишите программу, которая при вводе пользователем числа из диапазона от 1 до 99, добавляет к нему слово `копейка` в правильной форме.

Пример выполнения программы:

```
Введите число из диапазона от 1 до 99: 25
25 копеек
Введите число из диапазона от 1 до 99: 4
4 копейки
```

ВАРИАНТ 27

Напишите программу, которая выводит на экран работающие «электронные часы», которые работают в течение, например, трёх минут или до тех пор, пока пользователь не нажмёт любую клавишу.

ВАРИАНТ 28

Напишите программу, которая вычисляет длину введенной пользователем строки без использования стандартной функции определения длины.

ВАРИАНТ 29

Написать программу вычисления стоимости поездки на автомобиле на дачу (туда и обратно). Исходными данными являются:

расстояние до дачи (км);

количество бензина, которое потребляет автомобиль на 100 км пробега;

цена одного литра бензина.

Пример выполнения программы:

```
Расстояние до дачи (км): 67
Расход бензина (литров на 100 км пробега): 8.5
Цена литра бензина (руб.): 6.5
Поездка на дачу и обратно обойдётся в 74.04 руб.
```

ВАРИАНТ 30

Написать программу, вычисляющую скорость, с которой бегун пробежал дистанцию.

Пример выполнения программы:

```
Введите длину дистанции (м): 1000
Введите время (минут.секунд): 3.25
Вы бежали со скоростью 17.56 км/час
```

ЛИТЕРАТУРА

1. Campbell Parallel Programming with Microsoft® Visual C++® / Campbell. - Москва: Гостехиздат, 2011. - 784 с.
2. Альфред, В. Ахо Компиляторы. Принципы, технологии и инструментарий / Альфред В. Ахо и др. - Москва: Высшая школа, 2015. - 882 с.

3. Балена, Франческо Современная практика программирования на Microsoft Visual Basic и Visual C# / Франческо Балена , Джузеппе Димауро. - М.: Русская Редакция, 2015. - 640 с.
4. Боровский, А. C++ и Pascal в Kylix 3. Разработка интернет-приложений и СУБД / А. Боровский. - М.: БХВ-Петербург, 2015. - 544 с.
5. Давыдов, В. Visual C++. Разработка Windows-приложений с помощью MFC и API-функций / В. Давыдов. - М.: БХВ-Петербург, 2014. - 576 с.
6. Довбуш, Галина Visual C++ на примерах / Галина Довбуш , Анатолий Хомоненко. - М.: БХВ-Петербург, 2012. - 528 с.
7. Зиборов, В. MS Visual C++ 2010 в среде .NET / В. Зиборов. - М.: Питер, 2012. - 320 с.
8. Кетков, Юлий Практика программирования: Visual Basic, C++ Builder, Delphi. Самоучитель (+ дискета) / Юлий Кетков , Александр Кетков. - М.: БХВ-Петербург, 2012. - 464 с.
9. Мешков, А. Visual C++ и MFC / А. Мешков, Ю. Тихомиров. - М.: БХВ-Петербург, 2013. - 546 с.
10. Неформальное введение в C++ и Turbo Vision. - Москва: ИЛ, 2010. - 384 с.
11. Панюкова, Т. А. Языки и методы программирования. Создание простых GUI-приложений с помощью Visual C++. Учебное пособие / Т.А. Панюкова, А.В. Панюков. - Москва: Мир, 2015. - 144 с.
12. Пахомов, Б. C/C++ и MS Visual C++ 2010 для начинающих / Б. Пахомов. - М.: БХВ-Петербург, 2011. - 736 с.
13. Пахомов, Борис C/C++ и MS Visual C++ 2012 для начинающих / Борис Пахомов. - Москва: СИНТЕГ, 2015. - 518 с.
14. Пахомов, Борис C/C++ и MS Visual C++ 2012 для начинающих / Борис Пахомов. - М.: "БХВ-Петербург", 2013. - 502 с.
15. Полубенцева, М. C/C++. Процедурное программирование / М. Полубенцева. - М.: БХВ-Петербург, 2014. - 448 с.
16. Поляков, А. Методы и алгоритмы компьютерной графики в примерах на Visual C++ / А. Поляков, В. Брусенцев. - М.: БХВ-Петербург, 2011. - 560 с.
17. Понамарев, В. Программирование на C++/C# в Visual Studio .NET 2003 / В. Понамарев. - М.: БХВ-Петербург, 2015. - 917 с.
18. Роберт, С. Сикорд Безопасное программирование на C и C++ / Роберт С. Сикорд. - Москва: РГГУ, 2014. - 496 с.
19. Секунов, Н. Программирование на C++ в Linux / Н. Секунов. - М.: БХВ-Петербург, 2016. - 425 с.
20. Сидорина, Татьяна Самоучитель Microsoft Visual Studio C++ и MFC / Татьяна Сидорина. - М.: "БХВ-Петербург", 2014. - 848 с.

ОБРАЗЕЦ ТИТУЛЬНОГО ЛИСТА ДЛЯ КОНТРОЛЬНОЙ РАБОТЫ

МИНОБРНАУКИ РОССИИ
федеральное государственное автономное образовательное учреждение
высшего образования
«Санкт-Петербургский государственный политехнический университет»
(ФГАОУ ВО «СПбПУ»)
Институт среднего профессионального образования

КОНТРОЛЬНАЯ РАБОТА

По дисциплине:

СИСТЕМНОЕ ПРОГРАММИРОВАНИЕ

ПМ 01. «Программировнаие в компьютерных системах»

_____
Номер специальности и группы

Студент (-ка) _____ Иванова А.А.
Подпись Фамилия И.О.

Руководитель _____ Кураева Е.А.
Подпись Фамилия И.О.

Дата поступления ДКР

Оценка работы

Дата проверки

Подпись преподавателя

Рецензия (замечания) по ДКР

Дата поступления ДКР (повторно)

Число, месяц и год сдачи ДКР

Санкт-Петербург
2019 год