МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное
образовательное учреждение
высшего профессионального образования
Тихоокеанский государственный университет

УТВЕРЖДАЮ

Проректор по учебной работе

_____ С.В. Шалобанов

«_____» _____20___ г.

ПРОГРАММА ДИСЦИПЛИНЫ

по кафедре Вычислительная техника

ПРОГРАММИРОВАНИЕ

Утверждена научно-методическим советом университета для направлений подготовки Инфокоммуникационные технологии и системы связи (профиль Многоканальные телекоммуникационные системы) и Информатика и вычислительная техника (профиль Вычислительные машины, комплексы, системы и сети)

Хабаровск 2012 г.

Программа разработана в соответствии с требованиями государственного образовательного стандарта, предъявляемыми к минимуму содержания дисциплины и в соответствии с примерной программой дисциплины, утвержденной департаментом образовательных программ и стандартов профессионального образования с учетом особенностей региона и условий организации учебного процесса Тихоокеанского государственного технического университета.

Программу составил (и)
к.т.н., доцент, Сорокин Н.Ю.

Программа рассмотрена и утверждена на заседании кафедры

протокол № ____ от «____»_____20__ г.

Завкафедрой ВТ д.т.н. Сай С.В. «___»_____20__ г

Программа рассмотрена и утверждена на заседании УМК и рекомендована к изданию

протокол № ____ от «____»_____20__ г.

Председатель УМКС _____ «____»_____20__ г.

Декан ФАИТ_____ «___»_____20__ г. Воронин В.В.

Цель и задачи дисциплины

Цели дисциплины

Целью дисциплины «Программирование» является изучение и освоение *базовых понятий, методов и приемов программирования*, применяемых на всех основных этапах жизненного цикла программы. Дисциплина является ***базовой*** в техническом образовании для направления подготовки Инфокоммуникационные технологии и системы связи.

Задачи дисциплины:

- сформировать взгляд на программирование как на систематическую научно-практическую деятельность, носящую массовый характер (производство программ заданного качества в заданные сроки);
- сформировать базовые теоретические понятия (возможно, на элементарном уровне), лежащие в основе процесса конструирования программ, в том числе:
    - заложить основы разработки надёжных программ, акцентируя внимание как на методах аналитической верификации и разработке корректных программ, так и на технике (технологии) их испытания (тестирования и отладки);
    - заложить в основу конструирования и использования сложных (динамических) структур данных модель (парадигму) абстрактного типа данных, с последующим переходом к объектно-ориентированному подходу;
    - дать навыки технологии разработки корректных программ, (относительно) инвариантные к используемому языку программирования высокого уровня и опирающиеся на универсальную модель вычислительной машины;
- научить реализации корректных программ на выбранном рабочем языке программирования с учётом особенностей его конкретной реализации на персональной ЭВМ (конкретной системы программирования);
- продемонстрировать теоретически и на практике целесообразность и возможность конструктивного использования базовых теоретических понятий, методов и приемов (абстрактных схем) программирования;
- сформировать начальные представления и знания об анализе сложности алгоритмов и программ.

Требования к уровню освоению содержания дисциплины

В результате изучения дисциплины студенты должны

(а) *Знать:*

- способы постановки и спецификации задач для решения на ПЭВМ;
- основные современные методы и средства разработки корректных структурированных алгоритмов и программ;

- технологию работы на персональной ЭВМ (ПЭВМ), правила и приемы диалоговой работы на ПЭВМ при программировании типовых задач;
- положения объектно-ориентированного программирования;
- способы записи и документирования алгоритмов и программ;
- способы испытания и отладки программ.

(б) *Уметь:*

- самостоятельно осуществлять постановку и спецификацию задачи для решения на ПЭВМ;
- самостоятельно составлять, отлаживать, тестировать и документировать программы для персональных ЭВМ;
- доказывать корректность ключевых фрагментов составленных алгоритмов и программ.

(в) *Иметь представление о*

- методах и проблемах доказательства правильности программ;
- анализе некоторых задач и алгоритмов;
- проблемах оптимизации программ;
- абстрактных типах данных (на примерах), их спецификации, представлении и реализации в конкретном языке;
- основных понятиях объектно-ориентированного программирования и их реализации в конкретной системе программирования;
- технологии производства программ.

Объем дисциплины и виды учебной работы

Наименование	По учебным планам основной траектории обучения	
	С максимальной трудоемкостью	С минимальной трудоемкостью
Общая трудоемкость дисциплины По ГОС По УП	 216 216	
Изучается в семестрах	2	
Виды итогового контроля по семестрам Зачет Экзамен Курсовой проект (КП) Курсовая работа (КР) **Виды итогового контроля самостоятельной работы без отчетностей** Расчетно-графические работы (РГР) Реферат (РФ) Домашние задания (ДЗ)	 2	
Аудиторные занятия: Всего В том числе: лекции (Л) Лабораторные работы (ЛР) Практические занятия (ПЗ)	 90 36 36 18	
Самостоятельная работа Общий объем часов (С2) В том числе: на подготовку к лекциям на подготовку к ЛР на подготовку к ПЗ на выполнение КП на выполнение КР на выполнение РГР на написание РФ на выполнение ДЗ на экзаменационную сессию	 126 36 54 36	

Содержание дисциплины

Введение
Предмет дисциплины и ее задачи. Содержание и форма проведения занятий. Связь с другими дисциплинами учебного плана специальности.

Тема 1. Основные задачи разработки, анализа и верификации программ
1.1. *Этапы и проблемы решения задач на ПЭВМ.* Решение задач на ПЭВМ. Основные этапы и проблемы конструирования программ. Постановка задачи и спецификация. Разработка алгоритма решения задачи. Характеристики алгоритма, его свойства, запись алгоритма в виде блок-схемы. Программа как продукт.
1.2. *Верификация и анализ алгоритмов и программ.* Пример разработки, верификации и анализа алгоритма. Алгоритм Евклида для нахождения НОД двух чисел. Разработка и запись алгоритма в виде блок-схемы. Метод математической индукции и его применение в доказательстве правильности алгоритма. Метод индуктивных утверждений. Проблемы верификации программ. Цели и методы верификации программ. Аналитическая верификация (доказательство корректности) и испытание программы. Конструктивный подход: конструирование и верификация. Автоматизация верификации программ. Оценка сложности алгоритмов (на примерах).

Тема 2. Общие сведения о языке и системе программирования
2.1. *Общая характеристика языка.* Основные объекты программы. Классификация действий и данных. Программа на языке программирования: синтаксис и семантика. Пример программы: структура программы, описание переменных, присваивание.
2.2. *Система программирования*. Проект как совокупность программного кода и экранных форм. Инспектор свойств объектов. Компоненты системы. Трансляция программ (компиляция и интерпретация). Выполнение программы. Системы поддержки процесса подготовки и выполнения программ.

Тема 3. Представление данных. Концепция типов данных. Стандартные типы
3.1. *Тип данных как совокупность значений и действий*. Принцип строгой типизации. Номенклатура простых типов языка. Скалярные типы. Порядковые типы. Предописанные типы.
3.2. *Целые числа* без знака и со знаком. Принципы представления в памяти. Множество значений, допустимые операции.
3.3. *Вещественные числа*. Принципы представления в памяти. Свойства машинной арифметики. Машинное эпсилон. Примеры. Множество значений, допустимые операции.

3.4. *Символьный тип*. Множество значений, предописанные функции. Особенности реализаций. Способы использования символьного типа в программах.
3.5. *Логический тип*. Множество значений, операции. Особенности логических выражений в языке программирования, приоритет операций.

Тема 4. Основные управляющие структуры и их реализация
4.1. *Основные управляющие структуры* программирования. Составной оператор.
4.2. *Условный оператор*. Оператор множественного выбора.
4.3. *Операторы цикла* с предусловием и с постусловием: сходство, различие, преобразования. Цикл с параметром.

Тема 5. Процедуры и функции
5.1. *Метод последовательных уточнений* (пошаговой декомпозиции) и концепция подпрограммы (модуля) в программировании. Локальные и глобальные переменные, область действия
5.2. *Описание процедур и функций*. Формальные и фактические параметры. Способы передачи параметров. Параметры-значения, параметры-переменные, параметры-константы, процедуральные параметры.
5.3. *Процедуры и программирование действий с массивами.* Массивы в качестве параметров процедур и функций. Процедуры и функции как инструмент определения и реализации действий с массивами. Особенности использования многомерных массивов и их компонент в качестве параметров процедур и функций. Примеры обработки матриц с использованием процедур. Открытые массивы.

Тема 6. Типы данных пользователя.
6.1. *Описание типов* данных пользователя.
6.2. *Перечисляемый тип*: определение типа, константы, переменные, операции, предописанные функции, реализация ввода и вывода.
6.3. *Диапазонный тип*. Базовый тип. Определение диапазонного типа. Операции и предописанные функции. Примеры использования.

Тема 7. Обработка структурированных типов данных
7.1. *Массив* как составной тип данных. Определение типа. Определение переменных. Логическая структура массивов. Доступ к элементам массива (адресация). Примеры работы с массивами.
7.2. *Тип данных запись*. Определения. Синтаксис и семантика. Примеры использования. Оператор присоединения with. Записи с вариантами. Рекомендации по применению.
7.3. *Тип множество*. Определение типа. Конструкторы множеств. Операции над множествами. Примеры использования. Особенности реализации.

7.4. *Строки*. Определение, операции над строками. Представление строк (с явной длиной, с символом ограничителем). Реализация типовых операций над строками.
7.5. *Указатели*. Ссылочные и идентифицированные (динамические) переменные. Действия над ссылками и динамическая память. Применение ссылочных переменных при программировании. Определяемые рекурсивные типы данных. Цепное представление последовательностей. Программирование основных операций.
7.6. *Работа с файлами*. Определение файлов и файловых переменных. Базовые операции с файлами. Текстовые файлы. Типизированные файлы. Типовые действия, особенности работы с разными типами файлов.

Тема 8. Приемы программирования некоторых задач
8.1. *Линейный и бинарный поиск.* Задача поиска элемента массива. Линейный поиск. Варианты решения. Доказательство корректности. Общая схема линейного просмотра. Задача поиска места элемента в упорядоченном массиве. Спецификация задачи. Разработка корректной программы бинарного поиска. Анализ алгоритма бинарного поиска. Дерево бинарного поиска. Оптимальность алгоритма бинарного поиска. Оптимизация программы бинарного поиска. Быстрый вариант с развертыванием цикла.
8.2. *Рекурсия в программировании.* Рекурсивные определения и рекурсивные функции. Рекурсивные алгоритмы. Выполнение рекурсивных алгоритмов: подготовка трассы, стек (магазин). Анализ рекурсивных алгоритмов. Рекуррентные уравнения. Соотношение время-память для рекурсивных алгоритмов. Рекурсивные процедуры и функции в языке программирования. Приемы рекурсивного программирования (нисходящая и восходящая рекурсия, накапливающие параметры). Примеры: простая рекурсия, программы с несколькими рекурсивными вызовами, косвенная рекурсия (взаимно- рекурсивные подпрограммы). Преобразование рекурсивных программ в итеративные (“избавление” от рекурсии). Примеры.
8.3. *Программирование линейных списков.* Линейный однонаправленный список (Л1-список) как абстрактный тип данных. Функциональная спецификация Л1-списка. Ссылочная реализация Л1-списка в связанной памяти. Представление и реализация Л1-списка на языке программирования. Разновидности линейных списков: циклические, двунаправленные (Л2-списки). Примеры.

Тема 9. Элементы объектно-ориентированного программирования
Основные идеи объектно-ориентированного программирования. Объектовый тип и экземпляр объектового типа. Данные и методы объекта. Инкапсуляция. Наследование. Полиморфизм. Пример программы.

Тема 10. Графические объекты.
10.1. Создание элементарных графических изображений. Компьютерная анимация

10.2. Компоненты для включения средств деловой графики: построение графиков функций.

Заключение
Основные тенденции и направления развития методов и языков программирования на ПЭВМ. Инструментальная поддержка и автоматизация конструирования программ.

Разделы дисциплины и виды занятий и работ

№	Раздел дисциплины	Л	ЛР	ПЗ	РГР	С2
1.	Этапы и проблемы решения задач на ПЭВМ	*		*		
2.	Общие сведения о языке и системе программирования	*	*			*
3.	Представление данных. Концепция типов данных. Стандартные типы	*		*		
4.	Основные управляющие структуры и их реализация	*	*	*		*
5.	Процедуры и функции	*	*	*		
6.	Типы данных пользователя	*	*			*
7.	Обработка структурированных типов данных	*	*	*		
8.	Приемы программирования некоторых задач	*	*	*		*
9.	Элементы объектно-ориентированного программирования	*	*	*		*
10.	Графические объекты			*		*
11.	Заключение	*				

Лабораторный практикум

Используемое оборудование. Лабораторные занятия проводятся в аудитории, оснащенной современными персональными компьютерами. Количество компьютеров должно соответствовать количеству студентов на занятии (1 компьютер на 1 студента). На компьютерах должна быть установлена операционная система Windows XP и интегрированная система программирования. Может быть использована любая из версий языка, выбранного кафедрой для преподавания.

Целью лабораторного практикума является выработка и закрепление навыков самостоятельного составления, отладки и тестирования программ на языке программирования.

Оценка результатов работы. В ходе выполнения задания для поставленной задачи составляется и отлаживается программа. Результат работы оценивается в процессе тестирования на наборе контрольных примеров, предложенных как самим студентом-разработчиком, так и преподавателем, проводящем занятие.

Задания для лабораторных работ включают в себя ряд задач по теме занятия. Количество и общий объем приведенного списка лабораторных работ превышает объем часов, отводимых на выполнение лабораторных работ по учебному плану. При разработке рабочей программы из приведенного списка выбираются необходимые работы. Количество заданий в каждой работе также может быть изменено.
Ниже приведены примеры заданий для одного варианта.

№ п/п	Название	Кол-во часов	Описание Лабораторной работы
1	Блок-схемы алгоритма выполнения программы	2	Студентам предлагается разработать блок-схему алгоритма для решения простейшей задачи по варианту. Например, поиск корней квадратного уравнения. Результатом выполнения лабораторной работы является блок – схема алгоритма, в графическом виде выполненная по ГОСТу, для решения задачи.
2	Линейные программы	2	Студентам предлагается написать программу на языке С++ с использованием стандартных математических функций для расчета заданной по варианту формулы. Например $Z = 2\sin^2(3\pi - 2\alpha)^{2/3} + cos^3(5\pi + 2\alpha)$. Результатом выполнения лабораторной работы является блок – схема алгоритма результирующей программы, исходный код результирующей программы. Дополнительным требованием к выполнению программы является: необходимо предусмотреть «защиту от дурака» для вводимых значений(контроль вводимых данных

			пользователем), необходимо проводить контроль корректности выполнения математических операций (деление на ноль, корень из отрицательного числа) и вывод соответствующих сообщений пользователю программы.
3	Разветвляющие вычислительные процессы	4	Студентам предлагается написать программу на языке С++, которая по введенному значению аргумента вычисляет значение функции, заданном в виде графика. Параметр вводится с клавиатуры. Результатом выполнения лабораторной работы является блок – схема алгоритма результирующей программы, исходный код результирующей программы. Дополнительным требованием к выполнению программы является: необходимо предусмотреть «защиту от дурака» для вводимых значений(контроль вводимых данных пользователем), необходимо проводить контроль корректности выполнения математических операций (деление на ноль, корень из отрицательного числа) и вывод соответствующих сообщений пользователю программы.
4	Организация циклов	2	Студентам предлагается написать программу на языке С++ которая позволит вычислить и вывести на экран в виде таблицы значения функции заданной с помощью ряда Тейлора, на интервале от Хнач до Хкон с шагом dx с точностью ε. Таблицу необходимо снабдить заголовком и шапкой. Каждая строка таблицы должна содержать значение аргумента, значение функции, количество просуммированных элементов ряда. Для прорисовки таблицы использовать символы «псевдографики». Результатом выполнения лабораторной работы является блок – схема алгоритма результирующей программы, исходный код результирующей программы. Дополнительным требованием к выполнению программы является: необходимо предусмотреть «защиту от дурака» для вводимых значений(контроль вводимых данных пользователем), необходимо проводить контроль корректности выполнения математических операций (деление на ноль, корень из отрицательного числа) и вывод соответствующих сообщений пользователю программы.
5	Одномерные массивы и указатели	4	Студентам предлагается написать программу на языке С++ которая позволит производить различные операции на данными, находящимися в одномерном массиве из n числовых элементов типа целое, вещественное. Размерность массива n может быть задана константной или задана в качестве параметра из командной строки. Студент должен при написании программы показать умение применять указатели при обращении к элементам массива. Заполнение массива производится поэлементно с командной строки. Например: В одномерном массиве, состоящем из n вещественных элементов, вычислить: Сумму положительных элементов массива. Произведение элементов массива с четными номерами. Результатом выполнения лабораторной работы является блок – схема алгоритма результирующей программы, исходный код результирующей программы.

			Дополнительным требованием к выполнению программы является: необходимо предусмотреть «защиту от дурака» для вводимых значений(контроль вводимых данных пользователем), необходимо проводить контроль корректности выполнения математических операций (деление на ноль, корень из отрицательного числа) и вывод соответствующих сообщений пользователю программы.
6	Двумерные массивы	4	Студентам предлагается написать программу на языке С++ которая позволит производить различные операции на данными, находящимися в двумерном массиве (или двух различных двумерных массивах), представленном прямоугольной матрицей. Размерности массива могут быть заданы как константами, так и вводится пользователем из командной строки. В программе необходимо ввести возможность задавать количество элементов массива в качестве параметра из командной строки. Студент должен при написании программы показать умение применять указатели при обращении к элементам массива. Заполнение массива производится поэлементно с командной строки. Например: Дана целочисленная прямоугольная матрица m*n. Определить количество строк, не содержащих ни одного нулевого элемента. Даны две целочисленные прямоугольные матрицы m*n. Получить и вывести на экран построчно результирующую матрицу, каждый элемент, которой есть отношение соответствующего элемента первой и второй матриц. Результатом выполнения лабораторной работы является блок – схема алгоритма результирующей программы, исходный код результирующей программы. Дополнительным требованием к выполнению программы является: необходимо предусмотреть «защиту от дурака» для вводимых значений(контроль вводимых данных пользователем), необходимо проводить контроль корректности выполнения математических операций (деление на ноль, корень из отрицательного числа) и вывод соответствующих сообщений пользователю программы.
7	Подпрограммы	4	Студентам предлагается написать программу на языке С++ которая позволит выполнять различные математические операции(по варианту) над двумя числами. Алгоритм, реализующий каждую заданную математическую операцию, должен быть реализован в виде подпрограммы и вызываться по запросу из основной программы. Для возвращения подпрограммой результирующего значения необходимо использовать «return». Например: Реализовать программу калькулятор, которая по выбору пользователя реализует арифметические операции сложение, вычитание, деление, умножение над двумя вещественными числами a и b, вводимыми с клавиатуры. Результатом выполнения лабораторной работы является блок – схема алгоритма результирующей программы, исходный код результирующей программы.

			Дополнительным требованием к выполнению программы является: необходимо предусмотреть «защиту от дурака» для вводимых значений(контроль вводимых данных пользователем), необходимо проводить контроль корректности выполнения математических операций (деление на ноль, корень из отрицательного числа) и вывод соответствующих сообщений пользователю программы.
8	Структуры и связные списки	6	Студентам предлагается написать программу на языке С++ которая должна осуществлять операции над объектами типа структура и список. Функции изменения содержания элементов списка предлагается реализовывать как часть элемента списка и производить обращение к ним через указатель на функцию. Например: Разработать программу, обеспечивающую ввод исходных данных об учениках (ФИО ПОЛ ГОД РОЖД.)с клавиатуры в элемент списка представленный структурой данных в памяти. Обеспечить хранение их в памяти в виде связного списка, сортировку списка по алфавитному (Ф) и по числовому параметру (ГОД РОЖД.), изменение содержания указанного элемента списка, добавление элемента списка, удаление элемента списка. Результатом выполнения лабораторной работы является блок – схема алгоритма результирующей программы, исходный код результирующей программы. Дополнительным требованием к выполнению программы является: необходимо предусмотреть «защиту от дурака» для вводимых значений (контроль вводимых данных пользователем) и вывод соответствующих сообщений пользователю программы.
9	Работа с файлами	4	Студентам предлагается написать программу на языке С++ которая должна осуществлять базовые операции с файлами: запись данных в файл, чтение данных из файла, создание файла, удаление файла и тд. Для осуществления операции ввода/вывода в файл предлагается использовать потоковый ввод вывод с применением операторов cin и cout. Например: Разработать программу, которая позволит записать в файл 10 строк введенных пользователем с клавиатуры (сигналом окончанием ввода строки является нажатие клавиши «Enter»), прочитать из записанного файла строку и вывести ее содержание на экран. Номер целевой строки вводится пользователем с клавиатуры, после записи всех строк в файл. После вывода требуемой строки пользователем программа не должна завершаться, а переходит в режим ожидания команды от пользователя на ввод следующих 10 строк или вывод требуемой строки. Результатом выполнения лабораторной работы является блок – схема алгоритма результирующей программы, исходный код результирующей программы. Дополнительным требованием к выполнению программы является: необходимо предусмотреть «защиту от дурака» для вводимых значений (контроль вводимых данных

			пользователем) и вывод соответствующих сообщений пользователю программы.
10	Классы и визуальное программирование	6	Студентам предлагается написать программу на языке C++ реализующую (определяющую) свой «класс» по варианту. А также написать тестовую программу используя средства визуального программирования, показывающую функционирование реализованного класса. Например: Реализовать класс Circle. В классе реализовать методы: рисование окружности по заданным координатам центра и радиуса (координаты центра и радиус являются свойствами класса protected), определения лежит ли заданная точка внутри окружности, заданной координатами центра и радиусом, на границе окружности, за окружностью. Для осуществления вывода графической информации использовать класс MFC CDC. Результатом выполнения лабораторной работы является блок – схема алгоритма результирующей программы, исходный код результирующей программы. Дополнительным требованием к выполнению программы является: необходимо предусмотреть «защиту от дурака» для вводимых значений (контроль вводимых данных пользователем) и вывод соответствующих сообщений пользователю программы.

Лабораторный практикум и
его взаимосвязь с содержанием лекционного курса

№ п/п	№ раздела лекции	Наименование лабораторной работы
1	1, 2, 3	Блок-схемы алгоритма выполнения программы
2	1, 2	Линейные программы
3	3, 4	Разветвляющие вычислительные процессы
4	3, 4	Организация циклов
5	2, 3, 7	Одномерные массивы и указатели
6	2, 3, 7	Двумерные массивы
7	2, 4, 5, 8	Подпрограммы
8	2,3, 5, 6, 7	Структуры и связные списки
9	5, 8, 9	Работа с файлами
10	5, 6, 8, 9	Классы и визуальное программирование

Практические занятия

№ п/п	Название Практической работы	Кол-во часов	Описание
1	Изучение принципов осуществления форматированного ввода/вывода.	2	В процессе выполнения практической работы студенты получают навыки работы с использования семейства функций printf и scanf.
2	Изучение методов динамического выделения памяти.	2	В процессе выполнения практической работы студенты получают навыки работы с операторами new, delete, и функциями malloc, free.
3	Обработка исключений.	2	В процессе выполнения практической работы

			студенты получают навыки по обработке исключений с использованием конструкции try-catch.
4	Работа с указателями	2	В процессе выполнения практической работы студенты получают и закрепляют навыки по работе с указателями: область использования указателей, правила использования указателей, операции над указателями.
5	Визуальная среда программирования VS2010. Приложения типа Документ – Вид.	4	В процессе выполнения практической работы студенты получают навыки по созданию приложения типа Документ Вид
6	Визуальная среда программирования VS2010. Приложения типа Диалог	2	В процессе выполнения практической работы студенты получают навыки по созданию простого приложения типа диалог с элементами управления и использования его в качестве планшета для отображения графической информации
7	Изучение принципов работы с символьной информацией с использованием класса MFC CString.	2	В процессе выполнения практической работы студенты получают навыки работы с методами свойствами класса MFC CString.
8	Изучение принципов работы с файлами с использованием класса MFC CFile.	2	В процессе выполнения практической работы студенты получают навыки работы с методами свойствами класса MFC CFile.

Практические занятия и их
взаимосвязь с содержанием лекционного курса

№ п/п	№ раздела лекции	Наименование лабораторной работы
1	2, 3, 4, 5	Изучение принципов осуществления форматированного ввода/вывода.
2	2, 3,4, 5, 6	Изучение методов динамического выделения памяти.
3	2, 3,4, 5, 6	Обработка исключений.
4	3, 7,	Работа с указателями
5	7.1, 6, 9, 10	Визуальная среда программирования VS2010. Приложения типа Документ – Вид.
6	7.2, 6, 8, 9, 10	Визуальная среда программирования VS2010. Приложения типа Диалог
7	7.6, 6, 8, 9, 10	Изучение принципов работы с символьной информацией с использованием класса MFC CString.
8	7.5, 6, 8, 9, 10	Изучение принципов работы с файлами с использованием класса MFC CFile.

Контроль знаний студентов

Входной контроль знаний студентов

Входной контроль остаточных знаний проводится в форме анкетирования. Курс Алгоритмические языки и программирование базируется на понятиях, изучаемых в предшествующих естественнонаучных дисциплинах математика и информатика средней общеобразовательной школы.

Вопросы входного контроля:

1. Что изучает информатика?
2. По каким признакам и сколько поколений вычислительной техники выделяют в истории ее развития?
3. Какие устройства могут входить в состав ПК?
4. Дайте определение алгоритма.
5. Какие основные формы представления алгоритмов существуют?
6. Какие типы алгоритмов(вычислительных процессов) Вы знаете?
7. Какие основные системы счисления Вы знаете?

8. Каковы единицы измерения количества информации?
9. О каких языках программирования Вам известно?
10. Каковы этапы решения задач с помощью ПК?
11. Что такое программа?
12. Какие программы называются прикладными?
13. Что такое редактор текстов?
14. Что такое база данных?
15. Что такое электронная таблица?
16. Что такое операционная система?

Текущий контроль осуществляется в процессе выполнения студентами заданий лабораторных работ и проводится в форме собеседования преподавателя со студентом по теме занятия.

Кроме того, в течение семестра предусмотрен ряд проверочных работ, целью которых является определение уровня усвоения студентом учебного материала.

Задания для составления алгоритмов:

1. Проверить, является ли одномерный числовой массив упорядоченным по убыванию.
2. Определить в одномерном числовом массиве число соседств из двух положительных чисел.
3. Найти наименьший элемент в одномерном числовом массиве.
4. Найти произведение ненулевых элементов одномерного массива.
5. Определить в одномерном числовом массиве число соседств из двух чисел одного знака.
6. Определить, имеется ли в одномерном числовом массиве хотя бы одна пара совпадающих по величине соседних чисел.
7. Определить в одномерном числовом массиве число соседств из взаимно обратных чисел.
8. Найти наибольший элемент одномерного числового массива.
9. Определить, имеется ли в одномерном числовом массиве хотя бы одна пара взаимно противоположных соседних чисел.
10. Дан числовой массив a1,a2,..aN. Построить массив b1,b2,..bN, в котором b1=a1, b2=a1+a2, b3=a1+a2+a3 и т.д.

Выходной контроль знаний студентов

Дисциплина завершается зачетом и экзаменом в каждом семестре.

Условиями получения зачета являются:

- успешное выполнение всех заданий лабораторных работ, предусмотренных рабочей программой;
- положительные результаты тестирования по вопросам текущего контроля.

Экзаменационные билеты включают в себя теоретические вопросы и практические задания по темам, изученным в течение семестра. Списки вопросов теоретической части и заданий для практической реализации являются составной частью КИМ УМКД и могут варьироваться по решению кафедры.
Допускается проведение итогового тестирования с использованием тестов, составленных в соответствии с содержанием дисциплины.
Для тестирования можно использовать систему AST или другую аналогичную программу. Банк вопросов для тестового контроля входит в состав УМКД и может со временем изменяться.

Экзаменационные вопросы

1. Языки программирования. История. Области применения.
2. Языки программирования. Классификация.
3. Синтаксис и семантика языков программирования. Грамматики.
4. Язык С. Основные характеристики языка.
5. Язык С++. Основные характеристики языка.
6. Языки С/С++. Операции. Приоритет операций.
7. Языки С/С++. Операторы условия, выбора, цикла, перехода.
8. Языки С/С++. Подпрограммы.
9. Языки С/С++. Типы данных.
10. Языки С/С++. Структуры данных.
11. Языки С/С++. Массивы. Выделение памяти.
12. Языки С/С++. Символьные массивы и строки.
13. Языки С/С++. Указатели.
14. Язык С++. Классы. Основные понятия.
15. Язык С++. Классы. Модификаторы доступа к членам класса.
16. Язык С++. Классы. Конструктор, деструктор.
17. Рекурсия. Трассировка рекурсивных функций.
18. Связные списки. Тип данных. Простейшие операции.
19. Функции ввода-вывода библиотеки stdio.h (работа с файлами).
20. Потоковые функции ввода-вывода в языке С++ (работа с файлами).
21. Функции ввода-вывода библиотеки stdio.h (форматированный ввод-вывод).
22. Потоковые функции ввода-вывода в языке С++ (форматированный ввод-вывод).
23. Методы оценки алгоритмов. Сложность.
24. Алгоритмы сортировки. Анализ сложности алгоритмов.
25. Алгоритмы поиска. Анализ сложности алгоритмов.

Учебно-методическое обеспечение дисциплины

Основная литература

1. Страуструп Б. Язык программирования С++. М.: Бином, 1999
2. Архангельский А.Я. С++ Builder 5. М: Бином. 2000.
3. Рей Лишнер С++. Справочник. - СПб.: Питер, 2005
4. Баженова И. Ю. , Сухомлин В. А. Введение в программирование. - М.: Бином, 2007
5. Пратт Т., Зелковиц М. Языки программирования: разработка и реализация. - СПб.: Питер, 2002
6. Семакин И.Г., Шестаков А.П. Основы программирования: Учебник. – М.: Мастерство; НМЦ СПО; Высшая школа, 2001

Методическая литература:

1. Сафьянова Е.Н. Основы алгоритмизации и программирование. Томск: изд-во Томск.госуд.техн.ун-т, 2000.
2. Буч Г. Объектно-ориентированный анализ и проектирование с примерами на С++. М:Бином, 1999
3. Г.С. Иванова, Т.Н. Ничушнина, Е.К. Пугачев Объектно–ориентированное программирование. МГТУ М.Н. Баумана. 2001
4. Усакова О.Ф. и др. Программирование алгоритмов обработки данных. СПб.: БХВ-Петербург, 2003.
5. Окулов С.М. Программирование в алгоритмах. М.: Бином. Лаборатория знаний, 2004.
6. Юркин А. Задачник по программированию. СПб.: Питер, 2002.

Дополнительная литература:

1. Касьянов В.Н., Сабельфельд. Сборник заданий по практикуму на ЭВМ. Учебное пособие для вузов. М.: Наука. Гл.ред.физ.-мат.лит., 1986.
2. Вирт Н. Алгоритмы+структуры данных=программы. М.: Мир, 1985.

Материально-техническое обеспечение дисциплины

Лабораторные занятия проводятся в аудитории, оснащенной современными персональными компьютерами. Количество компьютеров должно соответствовать количеству студентов на занятии (1 компьютер на 1 студента). На компьютерах должна быть установлена операционная система Windows 9x/XP/NT и интегрированная система программирования. Может быть использована любая из последних популярных версий среды, выбранной по решению кафедры.

Методические рекомендации по организации изучения дисциплины

Включение России в единое всемирное информационное пространство, увеличение объема и структурной сложности хранилищ электронных данных, расширение круга пользователей определяют актуальность подготовки квалифицированных специалистов в различных областях по разработке и эксплуатации специализированных средств, базирующихся на современной вычислительной технике.

Дисциплина Алгоритмические языки и программирование является для студентов первой в ряду дисциплин, связанных с использованием современных компьютерных систем для разработки необходимого программного обеспечения. Целью изучения данной дисциплины является подготовка студентов к восприятию специальных средств и методов решения прикладных задач физики с использованием вычислительных машин. Основными задачами дисциплины является обучение студентов приемам и принципам разработки алгоритмов, использованием алгоритмического языка для реализации алгоритма на компьютере, а также формирование у будущих специалистов четкого понимания концепции типизации данных.

По решению кафедры может быть выбран один из трех вариантов: 1) изучается язык Pascal в его современной реализации Delphi; 2) изучается язык C++ (Builder, Visual и др.); 3) изучение основ алгоритмизации начинается на языке Pascal, а затем происходит переход на C. Выбор языка Паскаль обусловлен хорошим соответствием свойств и характеристик языка для задач обучения программированию. В данном языке четко разработана концепция типов данных, которая является одним из центральных моментов при овладении программированием. Кроме того, язык Паскаль является универсальным языком программирования и владением им позволит студентам решать задачи, возникающие при изучении дисциплин специализации, таких, как «Вычислительная математика», «Методы решения задач математической физики», «Обработка экспериментальных данных на ЭВМ» и др. Однако в связи с широким распространением языка C мотивация студентов к изучению именно его особенно высока. Кроме того, при изучении таких дисциплин, как «Операционные системы», «Организация ЭВМ» владение языком C является предпочтительным. Поэтому лучшим

является вариант 3 с последовательным знакомством с обоими языками. Изучение языка С в таком случае проводится в сравнении с языком Pascal.

С другой стороны, при решении задач по информатике студенту необходимо владеть следующим материалом:

Физика (школьный курс). Скорость, ускорение, трение, работа, мощность, энергия.

Алгебра. Матрицы, операции с матрицами. Системы линейных уравнений, методы их решения.

Геометрия. Вектора на плоскости и в пространстве. Скалярное, векторное произведение.

Математический анализ. Пределы. Дифференциальное и интегральное исчисление в объеме школьного курса.

Самостоятельная работа предполагает, что:

1) отдельные темы могут быть отнесены на самостоятельное изучение;
2) на лекциях предлагается значительное количество контрольных вопросов и упражнений, служащих для проверки усвоения теории;
3) для успешного выполнения задач лабораторного практикума регулярно задаются домашние задания, которые проверяют усвоение методов и приемов решения разбираемых на занятиях задач, закрепляют алгоритмические умения и навыки.

Самостоятельная работа не расширяет рамки программы, она призвана закрепить излагаемый на лекциях и лабораторных занятиях материал, а также приучает студентов к самостоятельному овладению новым материалом.

На основании программы кафедрой разрабатываются рабочие учебные программы дисциплины с учетом фактического числа часов, отведенных на ее изучение.

В ходе беседы, проводимой при входном контроле остаточных знаний можно определить уровень школьной подготовки по вопросам, связанным с алгоритмизацией и программированием. Исходя из сложившейся ситуации, при проведении лабораторных работ из приведенного набора заданий лабораторного практикума можно выбирать задания с тем или иным уровнем сложности. Так для студентов, имеющих навыки программирования лабораторные работы 1 и 2 можно сократить, ограничившись заданиями 1,3 из работы 1 и заданием 3 из работы 2. Освободившееся время целесообразно использовать для совершенствования навыков анализа алгоритмов. Например, можно провести сравнительный анализ различных алгоритмов.

Набор задач, приведенных в качестве примеров заданий на лабораторные работы, также является ориентировочным.

Словарь терминов и персоналий

Адрес. Номер конкретного байта оперативной памяти компьютера.

Алгоритм. Заранее заданное понятное и точное предписание возможному исполнителю совершить определенную последовательность действий для получения решения задачи за конечное число шагов.

Алфавит. Фиксированный для данного языка набор основных символов, т.е. "букв алфавита", из которых должен состоять любой текст на этом языке. Никакие другие символы в тексте не допускаются.

Архитектура компьютера. Логическая организация, структура и ресурсы компьютера, которые может использовать программист. Определяет принципы действия, информационные связи и взаимное соединение основных логических узлов компьютера.

База данных. Один или несколько файлов данных, предназначенных для хранения, изменения и обработки больших объемов взаимосвязанной информации.

Байт. Группа из восьми битов, рассматриваемая при хранении данных как единое целое.

Библиотека стандартных подпрограмм. Совокупность подпрограмм, составленных на одном из языков программирования и удовлетворяющих единым требованиям к структуре, организации их входов и выходов, описаниям подпрограмм.

Бит. Наименьшая единица информации в цифровом компьютере, принимающая значения "0" или "1".

Ввод. Считывание информации с внешнего устройства в память компьютера.

Вещественное число. В информатике — тип данных, содержащий числа, записанные с десятичной точкой и (или) с десятичным порядком.

Вывод. Результаты работы программы, выдаваемые компьютером пользователю, другому компьютеру или во внешнюю память.

Выражение. В языке программирования — запись правила для вычисления некоторого значения. Строится из констант, переменных и указателей функций, объединенных знаками операций.

Идентификатор. Символическое имя переменной, которое идентифицирует её в программе.

Инструментальные программные средства. Программы, используемые в ходе разработки, корректировки или развития других программ: редакторы, отладчики, вспомогательные системные программы, графические пакеты и др. По назначению близки системам программирования.

Информатика. Дисциплина, изучающая структуру и общие свойства информации, а также закономерности и методы её создания, хранения, поиска, преобразования, передачи и использования в различных сферах человеческой деятельности. За понятием "информатика" закреплены области, связанные с разработкой, созданием, использованием и материально-техническим обслуживанием систем обработки информации, включая компьютеры и их программное обеспечение, а также организационные, коммерческие, административные и социально-политические аспекты компьютеризации — массового внедрения компьютерной техники во все области жизни людей. Информатика в самом своем существе базируется на компьютерной технике.

Информационная технология. Совокупность методов и устройств, используемых людьми для обработки информации. Охватывает всю вычислительную технику, технику связи и, отчасти, — бытовую электронику, телевизионное и радиовещание.

Информация. Сведения об объектах и явлениях окружающей среды, их параметрах, свойствах и состоянии, которые воспринимают информационные системы (живые организмы, управляющие машины и др.) в процессе жизнедеятельности и работы. Применительно к обработке данных на компьютерах — произвольная последовательность символов, несущих смысловую нагрузку.

Ключевое слово. Слово языка программирования, имеющее определённый смысл для транслятора. Его нельзя использовать для других целей, например, в качестве имени переменной.

Команда. Описание элементарной операции, которую должен выполнить компьютер. Обычно содержит код выполняемой операции, указания по определению операндов (или их адресов), указания по размещению получаемого результата. Последовательность команд образует программу.

Компьютер. Программируемое электронное устройство, способное обрабатывать данные и производить вычисления, а также выполнять другие задачи манипулирования символами. Основу компьютеров образует аппаратура (HardWare), построенная, в основном, с использованием электронных и электромеханических элементов и устройств. Принцип действия компьютеров состоит в выполнении программ (SoftWare) — заранее заданных, чётко определённых последовательностей арифметических, логических и других операций.

Компьютеризация. Задачи массового внедрения компьютеров во все области жизни, стоящие перед странами как необходимое важное условие их прогресса и развития, а также последствия, которые будут вызваны этим массовым внедрением компьютеров. Цель компьютеризации — улучшение качества жизни людей за счёт увеличения производительности и облегчения условий их труда.

Логический тип. Тип данных, представляемый значениями "истина" или "ложь" ("да" или "нет"). Иногда также называется булевским в честь английского математика XIX века Джорджа Буля.

Логическое высказывание. Любое предложение, в отношении которого можно однозначно сказать, истинно оно или ложно.

Массив. Последовательность однотипных элементов, число которых фиксировано и которым присвоено одно имя. Компьютерный эквивалент таблицы. Положение элемента в массиве однозначно определяется его индексами.

Математическая модель. Система математических соотношений — формул, уравнений, неравенств и т.д., отражающих существенные свойства объекта.

Машинный язык. Совокупность машинных команд компьютера, отличающаяся количеством адресов в команде, назначением информации, задаваемой в адресах, набором операций, которые может выполнить машина, и др.

Обработка информации. В информатике — любое преобразование информации из одного вида в другой, производимое по строгим формальным правилам.

Оператор. Фраза алгоритмического языка, определяющая некоторый законченный э тап обработки данных. В состав операторов входят ключевые слова, данные, выражения и др.

Описание. Раздел программы, идентифицирующий структуры данных, которыми должна манипулировать программа, и описывающий их типы.

Основание системы счисления. Количество различных цифр, используемых для изображения чисел в данной системе счисления.

Подпрограмма. Самостоятельная часть программы, которая создаётся независимо от других частей и затем вызывается по имени. Когда имя подпрограммы используется в качестве оператора программы, выполняется вся группа операторов, представляющая тело подпрограммы.

Прикладная программа. Любая конкретная программа, способствующая решению какой-либо задачи в пределах данной проблемной области.

Программирование процесс подготовки задач для решения их на ЭВМ, состоящий из следующих этапов: составление "плана решения" задачи в виде набора операций (алгоритмическое описание задачи); описание "плана решения" на языке программирования (составление программы); трансляция программы с языка программирования на машинный язык (в виде последовательности команд, реализация которых техническими средствами ЭВМ и есть процесс решения задачи). Программированием называют также раздел прикладной математики, изучающий и разрабатывающий методы и средства составления, проверки и улучшения программ для ЭВМ.

Прокрутка. Имитация программистом за столом выполнения программы на конкретном наборе тестовых данных.

Псевдокод. Система обозначений и правил, предназначенная для единообразной записи алгоритмов. Занимает промежуточное место между естественным и формальным языками.

Семантика. Система правил истолкования отдельных языковых конструкций. Определяет смысловое значение предложений языка. Устанавливает, какие последовательности действий описываются теми или иными фразами языка и какой алгоритм определён данным текстом на алгоритмическом языке.

Синтаксис. Набор правил построения фраз языка, позволяющий определить, какие комбинации символов являются осмысленными предложениями в этом языке.

Структурное программирование. Метод разработки программ, в частности, требующий разбиения программы на небольшие независимые части (модули). Обеспечивает возможность проведения строгого доказательства правильности программ, повышает уверенность в правильности конечной программы.

Таблица истинности. Табличное представление логической схемы (операции), в котором перечислены все возможные сочетания значений истинности входных сигналов (операндов) вместе со значением истинности выходного сигнала (результата операции) для каждого из этих сочетаний.

Тест. Некоторая совокупность данных для программы, а также точное описание всех результатов, которые должна выработать программа на этих данных, в том виде, как эти результаты должны быть выданы программой.

Тестирование. Этап решения задачи на компьютере, в процессе которого проверяется работоспособность программы, не содержащей явных ошибок.

Тип данных. Понятие языка программирования, определяющее структуру констант, переменных и других элементов данных, разрешенные их значения и операции, которые можно над ними выполнять.

Транслятор. Программа-переводчик. Преобразует программу, написанную на одном из языков высокого уровня, в программу, состоящую из машинных команд.

Файл. Именованная совокупность любых данных, размещенная на внешнем запоминающем устройстве и хранимая, пересылаемая и обрабатываемая как единое целое. Файл может содержать программу, числовые данные, текст, закодированное изображение и др. Имя файла регистрируется в каталоге.

Цикл. Приём в программировании, позволяющий многократно повторять одну и ту же последовательность команд (операторов).