

## АННОТАЦИЯ К РАБОЧЕЙ ПРОГРАММЕ ДИСЦИПЛИНЫ «Программирование»

по основной профессиональной образовательной программе по направлению подготовки  
09.03.04 «Программная инженерия» (уровень бакалавриата)

**Направленность (профиль):** Разработка программно-информационных систем

**Общий объем дисциплины** – 18 з.е. (648 часов)

**В результате освоения дисциплины у обучающихся должны быть сформированы компетенции с соответствующими индикаторами их достижения:**

- ОПК-2.2: Использует современные информационные технологии и программные средства, в том числе отечественного производства, при решении задач профессиональной деятельности;
- ОПК-4.1: Применяет стандарты, нормы, правила, техническую документацию в профессиональной деятельности;
- ОПК-4.2: Участвует в разработке стандартов, норм и правил, а также технической документации, связанной с профессиональной деятельностью;
- ОПК-5.1: Инсталлирует программное обеспечение согласно инструкциям;
- ОПК-5.2: Коммутирует аппаратное обеспечение в составе информационных и автоматизированных систем;
- ОПК-6.1: Формализует задачу и предлагает алгоритмическое решение;
- ОПК-6.2: Проектирует программные продукты с применением основ информатики;
- ОПК-6.3: Осуществляет разработку и тестирование программных продуктов;

**Содержание дисциплины:**

Дисциплина «Программирование» включает в себя следующие разделы:

**Форма обучения очная. Семестр 1.**

**Объем дисциплины в семестре** – 6 з.е. (216 часов)

**Форма промежуточной аттестации** – Экзамен

**1. ОСНОВЫ ПРОГРАММИРОВАНИЯ НА ЯЗЫКЕ СИ.** 1.1 Основные этапы решения задач на ЭВМ. Жизненный цикл программ. Постановка задачи. Нисходящее и восходящее проектирование. Алгоритм. Способы записи алгоритма. Программа на языке высокого уровня. Тестирование и отладка программ. Интерфейс программ. Документирование

1.2 Типы вычислительных процессов: линейный, разветвляющийся, циклический

1.3 Основные элементы Си-программы

1.4 Стандартные типы данных

1.5 Основные операторы языка Си

1.6 Интегрированная среда разработки Си-программ.

**2. ВВОД-ВЫВОД В СИ. ОПЕРАЦИИ В СИ.** 2.1 Основные типы данных в Си

2.2 Ввод-вывод в Си. Стандартные функции ввода-вывода

2.3 Операции в Си

2.4 Приоритет операций. Порядок выполнения выражений. Приведение типов

2.5 Решение задач с линейным алгоритмом.

**3. ОПЕРАТОРЫ РАЗВЕТВЛЕНИЯ.** 3.1 Условный оператор if. Понятие флажка. Условная операция «?:»

3.2 Оператор переключения switch

3.3 Решение задач с разветвляющимся алгоритмом.

**4. ЦИКЛИЧЕСКИЕ ОПЕРАТОРЫ.** 4.1 Оператор цикла с предусловием while

4.2 Оператор цикла с постусловием do...while

4.3 Оператор цикла for

4.4 Операторы break, continue, goto.

**5. ЗАДАЧ С ЦИКЛИЧЕСКИМ АЛГОРИТМОМ. ВЛОЖЕННЫЕ ЦИКЛЫ.** 5.1 Частичная сумма ряда. Знакопеременные ряды

5.2 Бесконечное суммирование с заданной точностью

5.3 Печать таблиц

- 5.4 Вложенные циклы
- 5.5 Задачи на перебор
- 5.6 Примеры циклических задач.
- 6. ОДНОМЕРНЫЕ МАССИВЫ.** 6.1 Основные понятия: массив, элемент, размерность, количество элементов, базовый тип
- 6.2 Объявление массивов в Си
- 6.3 Типовые задачи по обработке одномерных массивов.
- 7. ДВУМЕРНЫЕ МАССИВЫ.** 7.1 Двумерные массивы
- 7.2 Типовые задачи по обработке двумерных массивов.
- 8. УКАЗАТЕЛИ.** 8.1 Понятие указателя
- 8.2 Адресные операции
- 8.3. Адресная арифметика
- 8.4 Динамическое распределение памяти. Динамические массивы.
- Форма обучения очная. Семестр 2.**
- Объем дисциплины в семестре – 6 з.е. (216 часов)**
- Форма промежуточной аттестации – Экзамен**
- 1. СТРОКИ В СИ.** 1.1 Массивы и указатели
- 1.2 Массивы, указатели и строки
- 1.3 Массивы строк
- 1.4 Многомерные массивы и указатели.
- 2. СТАНДАРТНЫЕ ФУНКЦИИ ДЛЯ РАБОТЫ СО СТРОКАМИ.** 2.1 Функции обработки строк
- 2.2 Типовые задачи по обработке строк.
- 3. ПРОИЗВОДНЫЕ ТИПЫ (СТРУКТУРЫ, ОБЪЕДИНЕНИЯ).** 3.1 Структуры (записи)
- 3.2 Объединения
- 3.4 Переменные структуры
- 3.5 Массивы структур.
- 4. ПОДПРОГРАММЫ В СИ.** 4.1 Технология модульного программирования
- 4.2 Классификация модулей
- 4.3 Определение и описание функций в Си
- 4.4 Управление видимостью функций
- 4.5 Вызов функций. Возвращение результата
- 4.6 Передача параметров
- 4.7 Передача массивов в качестве параметров
- 4.8 Связь функций из разных файлов
- 4.9 Локальные и глобальные данные.
- 5. ДОПОЛНИТЕЛЬНЫЕ ВОЗМОЖНОСТИ ФУНКЦИЙ. КЛАССЫ ПАМЯТИ.** 5.1 Рекурсивные функции
- 5.2 Функции с переменным числом параметров
- 5.3 Функция main() – передача параметров и возвращение результата
- 5.4 Передача имени функции в качестве параметра. Указатель на функцию
- 5.5 Классы памяти.
- 6. ФАЙЛЫ.** 6.1 Файлы и потоки
- 6.2 Определение потока
- 6.3 Стандартные и не стандартные потоки
- 6.4 Текстовые и двоичные потоки.
- 7. ОСНОВНЫЕ ЭТАПЫ РАБОТЫ С ФАЙЛАМИ (С ПОТОКАМИ).** 7.1 Определение файла
- 7.2 Открытие файла
- 7.3 Чтение и запись
- 7.4 Закрытие файла
- 7.5 Дополнительные функции работы с файлами
- 7.6 Примеры организации работы с файлами.
- 8. МЕТОДОЛОГИИ СТРУКТУРНОГО И ОБЪЕКТНО-ОРИЕНТИРОВАННОГО ПРОГРАММИРОВАНИЯ.** 9.1 Технология структурного программирования

9.2 Технология объектно-ориентированного программирования.

**Форма обучения очная. Семестр 3.**

**Объем дисциплины в семестре – 6 з.е. (216 часов)**

**Форма промежуточной аттестации – Экзамен**

**1. Система управления версиями (Version Control System (VCS) или Revision Control System).** Что такое контроль версий, и зачем он нужен. История развития VCS. Git и его дополнительные сервисы..

**2. Абстрактный тип данных (АТД).** "Абстрактность" и "абстракция". Что такое АТД, его интерфейс и внутреннее представление. Что дает программисту использование АТД. Процедурное программирование и АТД. Структуры языка C как одна из реализаций АТД и проблемы такой реализации. Идеи объединения данных и действий над ними..

**3. Введение в ООП (Объектно-Ориентированное Программирование) с примерами на C++..** Основная идея объектно-ориентированного программирования. Классы, экземпляры и объекты. Инкапсуляция и сокрытие. Классы и объекты на C++. Ассоциация, композиция и агрегация (с примерами на C++). Дерево (или граф) объектов. Язык UML. Диаграмма классов UML..

**4. Язык Java и основные реализующие его технологии..** Основы языка Java (Байт код Java, JVM, JDK, JRE). Основные возможности и идеи. Классы, объекты и объектные переменные..

**5. Язык C# и основные реализующие его технологии..** Основы языка C# ( C-подобный синтаксис, NET Framework, CLR, CIL). Основные возможности и идеи. Классы (ссылочные типы) и структуры (значимые типы)..

**6. Создание и уничтожение объектов, конструкторы и деструкторы (с примерами на C++, C# и Java)..** Что такое конструкторы и для чего они необходимы. Конструкторы по умолчанию. Создание конструкторов. Инициализаторы объектов. Списки инициализаторов членов. Конструкторы копий (мелкое и глубокое копирование). Конструкторы перемещения. Общий порядок работы конструктора. Деструкторы, Finalize, оператор delete и сборка мусора..

**7. Особенности реализации некоторых распространенных программных конструкций на языках C++, C# и Java.** Пространство имен и пакеты; Особенности языков C++, Java и C# при работе со строками;

Работа с одномерными и многомерными массивами, особенности работы с массивами объектов; Передача параметров и возврат результата при вызове подпрограмм, особенности передачи объектов в качестве параметров; Перегрузка функций и операторов;

Оператор this, что это такое и для чего он нужен; Дружественные функции и дружественные классы; Статические поля и методы, что это такое и как работают; Перегрузка оператора присваивания; Ошибки времени выполнения, исключения;.

**8. Наследование.** Что такое наследование и чем оно полезно. Базовый и производный класс. Одиночное и множественное наследование. Транзитивное наследование и иерархия наследования. Как это отображается на UML-диаграмме классов. Что не наследуется производными классами от базового. Видимость и доступность. Модификатор protected. Переопределение реализации базового класса, когда можно это делать, и когда это делать обязательно. Неявное наследование. Предотвращение дальнейшего наследования. Практические примеры на C++, C# и Java: Процесс создания базового класса и его производных классов..

**9. Полиморфизм -одна из трех основных парадигм ООП.** Что такое полиморфизм. В каких случаях объекты производного класса могут обрабатываться как объекты базового класса и зачем это нужно. Абстрактные и виртуальные методы. Абстрактные базовые классы, интерфейсы и их реализация. Виртуальный деструктор и виртуальное наследование в C++. Практические примеры на C++, C# и Java:Разработка абстрактных базовых классов и их производных классов..

**10. Шаблоны (templates). Обобщенное программирование(Generics)..** Шаблоны функций и классов (template) на C++ ; Обобщенные методы и классы (generics) на Java и C#; Делегаты на C#..

**11. Примеры применения ООП. Часть 1..** Использование библиотеки STL на C++, контейнер, итератор, алгоритм, дополнительные контейнеры и функциональный объект. Сортировка и поиск с предикатом для контейнера с объектами базового и производного классов.

Коллекции на Java и C#. Реализация интерфейсов IComparable, IComparer. Обобщенные коллекции. Чтение из файла и запись в файл объектов. Сериализация..

**12. Примеры применения ООП. Часть 2 (GUI)..** Обсуждается реализация пяти GUI приложений

на разных платформах (Qt, Windows Form C++, Windows Form C#, Java Swing, Android). В программах используются элементы ввода (строки ввода, надписи, кнопки, меню, списки, таблицы), главная и вспомогательные формы, диалоговые окна, а также работа с графикой (загрузка и вывод изображения, преобразование изображения по отдельным точкам, рисование геометрических фигур и анимация)..

Разработал:

доцент  
кафедры ПМ  
доцент  
кафедры ПМ

Е.В. Егорова

В.С. Троицкий

Проверил:

Декан ФИТ

А.С. Авдеев