

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

УТВЕРЖДАЮ

Проректор по учебной работе
и образовательным инновациям

О.Г. Прохоренко

05 июля 2023 г.

Регистрационный № УД –1875/6.

ОСНОВЫ И МЕТОДОЛОГИИ ПРОГРАММИРОВАНИЯ

Учебная программа учреждения-образования
по учебной дисциплине для специальности:

6-05-0533-10 Информатика

2023 г.

Учебная программа составлена на основе ОСВО 6-05-0533-10-2023; примерного учебного плана, регистрационный № 6-05-05-021/пр. от 20.12.2022, учебных планов БГУ: № 6-5.3-58/01 от 15.05 2023 г., № 6-5.3-58/02 от 15.05 2023 г., № 6-5.3-58/03 от 15.05 2023 г., № 6-5.3-58/04 от 15.05 2023 г., № 6-5.3-58/05 от 15.05 2023 г.

СОСТАВИТЕЛЬ:

О.Г. Казанцева – старший преподаватель кафедры многопроцессорных систем и сетей факультета прикладной математики и информатики Белорусского государственного университета.

РЕЦЕНЗЕНТЫ:

Е.П. Соболевская – доцент кафедры дискретной математики и алгоритмики факультета прикладной математики и информатики Белорусского государственного университета, кандидат физико-математических наук, доцент;

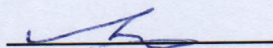
Н.В. Лапицкая – заведующая кафедрой программного обеспечения информационных систем Учреждения образования «Белорусский государственный университет информатики и радиоэлектроники», кандидат технических наук, доцент.

РЕКОМЕНДОВАНА К УТВЕРЖДЕНИЮ:

Кафедрой многопроцессорных систем и сетей
(протокол № 13 от 06.04.2023);

Научно-методическим советом БГУ
(протокол № 9 от 29.06.2023).

Заведующий кафедрой



С.В.Марков

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

Цели и задачи учебной дисциплины

Учебная дисциплина «Основы и методологии программирования» (ОиМП), ориентирована на обучение студентов базовым знаниям, умениям и навыкам в области программирования. Изучаемые темы базируются на использовании современных информационных технологий, новейшего программного и технического обеспечения компьютеров.

Основой для обучения программированию является предмет «Информатика», изучаемый в средней школе.

Цель учебной дисциплины «ОиМП» – подготовка специалиста:

1. имеющего теоретические знания в области разработки программного обеспечения следуя трем основным парадигмам: структурное, объектно-ориентированное и функциональное программирование;
2. имеющего практические навыки использования методов алгоритмизации и программирования, умеющего проектировать эффективные алгоритмы решения поставленной задачи и выбирать наиболее подходящие структуры данных, программные и технические средства их реализации;
3. имеющего практические навыки конструирования программных приложений сложной структуры с использованием языка высокого уровня C++, отвечающие современным требованиям и новейшим компьютерным технологиям с учетом операционного окружения.

Для этого необходимо формирование систематизированных знаний и навыков в области программирования: изучение классических методов программирования; освоение навыков разработки программ, их отладки, тестирования и документирования; формирование у студентов алгоритмического мышления, системного подхода к основам моделирования и формализации посредством создания компьютерных моделей. В итоге обеспечивается база знаний, необходимая для усвоения материала последующих учебных дисциплин в области информатики и успешной дальнейшей работы. При изложении учебной дисциплины важно показать возможности использования инструментария программирования при решении прикладных задач, возникающих в различных областях науки, техники, экономики и производства.

Дисциплина «Основы и методологии программирования» подготавливает студентов к дальнейшему изучению других языков программирования и развивает их компетенции в области решения практических задач, что важно как для будущего обучения (например, в учебной дисциплине «Алгоритмы и структуры данных»), так и для будущей профессиональной деятельности.

Задачи учебной дисциплины:

1. Изучение основных конструкций современного языка программирования высокого уровня.

2. Овладение методологией анализа и проектирования типовых алгоритмов (линейных, условных, циклических, рекурсивных) и оценки их быстродействия.

3. Овладение методологией разработки программ на основе императивного и объектно-ориентированного подхода.

4. Приобретение навыков программирования на языке высокого уровня с использованием современных интегрированных сред разработки (IDE) и инструментальных средств.

5. Овладение основами навыков разработчика программного обеспечения (структурирование программного кода, документирование программного кода, реализация принципа модульности, получение общего представления об организации пользовательского интерфейса).

6. Формирование навыков мышления программиста и использование их при разработке программного обеспечения для решения различных профессиональных, исследовательских и прикладных задач.

Место учебной дисциплины. В системе подготовки специалиста с высшим образованием учебная дисциплина относится **к модулю «Программирование»** государственного компонента.

Учебная программа составлена с учетом межпредметных **связей** и программ по дисциплинам: «Математический анализ» модуля «Математический анализ», «Основы высшей алгебры» модуля «Геометрия и алгебра».

Методы, излагаемые в указанных дисциплинах, используются для проектирования алгоритмов и программных приложений.

Сформированные при изучении дисциплины «ОиМП» компетенции являются основой для дальнейшего изучения дисциплин:

- модуля «Программирование»: «Разработка кросс-платформенных приложений» и «Машинно-ориентированное программирование»;
- модуля «Дискретные структуры и алгоритмы»: «Алгоритмы и структуры данных»;
- модуля «Компьютерные системы»: «Операционные системы».

Знания, полученные в учебной дисциплине, используются при изучении всех дисциплин специализации, при выполнении курсовых и дипломных работ, а также используются как инструментарий для моделирования и компьютерного решения задач ряда математических дисциплин, изучаемых на старших курсах.

Требования к компетенциям

Освоение учебной дисциплины «Основы и методологии программирования» должно обеспечить формирование следующих *базовых профессиональных компетенций*:

БПК. Применять при проектировании приложений такие парадигмы программирования как структурное, объектно-ориентированное и функциональное программирование.

БПК. Разрабатывать программное обеспечение в интегрированных средах разработки.

В результате освоения учебной дисциплины студент должен:

знать:

- базовые понятия и принципы обработки информации, этапы решения автоматизируемых задач;
- принципы проектирования алгоритмов и их реализации на языке программирования;
- уровни представления данных, модели данных и методы обработки данных;
- основные методологии и средства эффективной разработки программного обеспечения;
- методы тестирования и отладки программ.

уметь:

- проектировать эффективные алгоритмы решения поставленной задачи;
- выбирать наиболее подходящие структуры данных, методологии разработки программ, программные и технические средства реализации алгоритма;
- разрабатывать программные приложения с заданной функциональностью и операционным окружением.

иметь навык:

- разработки алгоритмов на основе разных методов алгоритмизации задач;
- тестирования и отладки программ;
- разработки и сопровождения программ в конкретных средах разработки.

Структура учебной дисциплины

Дисциплина изучается в 1 семестре. Всего на изучение учебной дисциплины «Основы и методологии программирования» отведено:

– в очной форме получения высшего образования: 216 часов, в том числе 132 аудиторных часа, из них: лекции – 64 часа, лабораторные занятия – 60 часов, управляемая самостоятельная работа – 8 часов.

Трудоемкость учебной дисциплины составляет 6 зачетных единиц.

Форма промежуточной аттестации – зачет, экзамен.

СОДЕРЖАНИЕ УЧЕБНОГО МАТЕРИАЛА

Раздел 1. Основы программирования

Тема 1.1 Методологии программирования

Императивное и декларативное программирование. Структурное программирование. Объектно-ориентированное программирование. Функциональное программирование. Логическое программирование.

Основные этапы разработки и сопровождения программ. Алгоритм и его свойства. Этапы и принципы разработки алгоритмов.

Тема 1.2. Системы счисления в программировании

Позиционные системы счисления. Двоичная, восьмеричная, шестнадцатеричная системы счисления. Преобразования между ними.

Представление целых чисел (прямой, обратный, дополнительный код) в двоичной системе счисления. Представление вещественных чисел в двоичной системе счисления.

Тема 1.3. Современное положение языка C++

История, эволюция и сферы применения языка C++. Основные этапы развития. C++ – мультипарадигмальный язык программирования. Достоинства и недостатки C++. Международный стандарт языка C++.

Тема 1.4. Базовые понятия языка C++

Структура программы. Библиотеки и заголовочные файлы. Компилирование и выполнение программ. Ошибки компиляции. Этапы компиляции. Компиляторы и среды разработки.

Алфавит, синтаксис, семантика языка C++. Типы данных C++ (фундаментальные типы данных и другие типы данных). Объявление, инициализация, присваивание переменных. Константность переменных (const, constexpr). Операторы языка C++. Приоритет операторов.

Пространство имен std. using-объявления и using-директивы.

Инструкции (statements) C++. Логический блок. Область видимости переменных.

Функции. Объявление и определение функции. Формальные и фактические параметры. Оператор return. Тип void.

Сигнатура main функции. Передача параметров в функцию main. Понятие о командной строке. Компиляция и запуск программы с командной строки.

Алгоритмы целочисленной арифметики.

Потоковый ввод-вывод. Механизмы форматирования. Экспоненциальная запись. Конвертация чисел в экспоненциальную запись. Точность и диапазон типов с плавающей точкой. Ошибки округления. Специальные значения NAN и INF.

Преобразования типов данных. Явное/неявное приведение типов. Безопасные/небезопасные преобразования. Операторы преобразования типов static_cast, const_cast.

Правила именования переменных и функций языка, правила записи констант. Понятие ключевого слова, список ключевых слов C++.

Практики оформления кода. Признаки хорошего стиля, основы Code Style для языка C++.

Раздел 2. Структурное программирование на языке C++

Тема 2.1. Составные типы данных и введение в STL

Введение в STL. Контейнеры STL: `std::vector`, `std::array`, `std::set`, `std::map` и др. Пары `std::pair` и кортежи `std::tuple`. Строки `std::string`.

Основы работы с итераторами.

Алгоритмы внутренних сортировок.

Структуры. Перечисления (`enum`). Класс `enum`. Объявление `using-enum`. Размер структур и перечислений. Выравнивание. Оператор `sizeof`.

Объединения. `std::variant`.

Библиотека алгоритмов. Знакомство с лямбда-выражениями в C++. Алгоритмы, принимающие предикат. Лямбда-выражения и реализация удобного механизма обратного вызова (`callback-функций`) на C++.

Тема 2.2. Ссылки, указатели, константность

Ссылки. Категории значений выражений: `glvalue`, `prvalue`, `xvalue`, `lvalue`, `rvalue` и ссылки на них.

Понятия «адрес», «указатель». Операции с указателями, адресная арифметика. Значения `nullptr`, `std::nullptr_t`. Тип `void*`. Константы и указатели.

Операторы `new`, `delete`.

Статические и динамические массивы. Одномерные и двумерные массивы.

Указатели и массивы. Операторы `new[]`, `delete[]`.

Преобразования указателя: операторы преобразования типов `static_cast`, `reinterpret_cast`.

Функции. Способы передачи параметров: по значению, по ссылке. Константные параметры. Возвращение указателей и ссылок.

Указатели на функцию. `std::function`. Передача функции в качестве параметра (`callback-функции`).

Тема 2.3. Многомодульные приложения

Многофайловые проекты. Отделение объявления от реализации. Структура программы. Директивы препроцессора. Условная компиляция. Страж включения.

Понятие `namespace`. Объявление пространства имен. Использование идентификаторов внутри пространства имен. Глобальная инициализация.

Правила использования директивы `using`.

Статическая функция. Статическая переменная. Статическая функция. Ключевое слово `extern`.

Рекурсивные функции. Прямая и косвенная рекурсия. Примеры рекурсивных алгоритмов. Рекурсивные сортировки. Итеративные функции.

Перегрузка функций.

Шаблоны функций. Экземпляры шаблонов функций. Заменитель возвращаемого типа `decltype(auto)`. Не типизированные параметры шаблона. Перегрузка шаблонных функций.

Шаблонные структуры.

Тема 2.4. Обработка исключительных ситуаций

Исключительные ситуации. Системные и пользовательские исключения. Генерация исключения. Оператор `throw`.

Обработка исключений. Перехват сообщений, оператор `try ... catch`. Виды блоков `catch`.

Стандартные классы исключений в C++. Использование стандартного класса `std::exception` и его производные классы. Метод `what()`.

Тема 2.5. Типы строк в C++ и организация ввода-вывода

Библиотека строк. Два типа строк в C++: `std::string`, нуль-терминированные строки (или C-строки). Функции класса `std::string`. Функции для работы с нуль-терминированными строками.

Текстовые и бинарные файлы. Последовательный и прямой метод доступа. Организация ввода/вывода с использованием потоковых классов. Форматированный вывод.

Строковые потоки.

Библиотека регулярных выражений.

Использование библиотеки `nlohmann/json` для работы с JSON файлами.

Раздел 3. Инструменты современной разработки приложений

Тема 3.1. Введение в тестирование приложений

Отладка программы. Отладка программ в IDE. Стек вызовов и отслеживание переменных.

Инструкции `assert` и `static_assert`.

Введение в тестирование. Подготовка `test-cases`. Ручное тестирование.

Разработка через тестирование (TDD). Библиотеки для `unit`-тестирования, их характеристики (`gTest` и др.).

Ключевые понятия библиотеки для `unit`-тестирования. Утверждения. Классы-тесты (фикстуры).

Тема 3.2. Логирование событий

Понятие логирования событий. Библиотеки логирования. Уровни логирования, форматы сообщений, направления вывода сообщений.

Тема 3.3. Система контроля версий

Отслеживание изменений кода. Системы управления версиями и их модели.

Обзор системы контроля версий `Git`. Инспекция кода (`code review`). Рабочие процессы при версионировании (`workflow`).

Тема 3.4. Профилирование программ

Измерение производительности приложений (профилирование времени выполнения).

Профилирование памяти в программе. Нахождение утечек памяти.

Инструменты профилирования.

Тема 3.5. Язык UML

Введение в унифицированный язык моделирования (UML). Диаграмма вариантов использования, диаграмма последовательности, диаграмма состояний, деятельности, диаграмма классов.

Раздел 4. Объектно-ориентированное программирование на языке C++

Тема 4.1. Основы объектно-ориентированного программирования

Принципы объектно-ориентированного программирования (ООП): абстракция, наследование, полиморфизм, инкапсуляция. Преимущества использования ООП.

Класс – пользовательский тип данных. Сравнение ключевых слов `class` и `struct`. Объекты класса. Члены класса, управление доступом. Реализация членов-функций класса. Указатель `this`. Время жизни объекта. Анонимные объекты.

Внешнее и внутреннее объявление классов. Вложенные классы. Инициализация вложенных классов.

Конструкторы. Список инициализаторов. Конструктор по умолчанию. Перегрузка конструкторов. Делегирующие конструкторы.

Список инициализации `std::initializer_list`. Его использование для создания объектов.

Константные/неконстантные члены класса. Статические члены класса.

Конструктор копирования. Оператор присваивания копированием. Поверхностное и глубокое копирование. Конструктор перемещения и оператор присваивания перемещением.

Конструктор и обработка ошибок. Выброс исключений в методах класса.

Деструкторы. Правило трех. Правило пяти.

Жизненный цикл объекта. Идиома RAII.

Умные указатели.

Принципы проектирования приложений: SOLID, KISS, DRY, WET, YAGNI.

Тема 4.2. Перегрузка операторов

Перегрузка операторов. Ограничения в перегрузке операторов. Способы перегрузки операторов: через члены-функции класса, через обычные функции, через дружественные функции.

Объявление и определение дружественных функций.

Перегрузка бинарных операторов, оператора присваивания. Особенности перегрузки унарных операторов. Перегрузка оператора приведения типа, оператора вызова функции и операторов ввода-вывода.

Дружественные классы.

Константные объекты классов. Константные методы классов. Константные ссылки и классы. Перегрузка константных и не константных членов-функций классов.

Тема 4.3. Реализация пользовательских структур данных

Абстрактные типы данных: стек, дек, очередь, список. Реализация линейного, кольцевого, однонаправленного и двунаправленного списка, стека, очереди, дека на основе массивов, списков и на основе других структур.

Реализация деревьев и поисковых деревьев.

Тема 4.4. Шаблоны классов

Шаблоны классов. Обращение к элементам и перегрузка по константности.

Специализация шаблона: полная и частичная специализация шаблона класса.

Тема 4.5 Наследование и полиморфизм

Наследование. Абстрактные и конкретные классы. Полиморфизм: статический и динамический – раннее и позднее связывание.

Полиморфизм через указатели или ссылки на базовый класс. Виртуальные функции. Таблица виртуальных функций.

Обрезка объектов.

Виртуальный деструктор. Динамическая идентификация типов (механизм RTTI): операторы `dynamic_cast` и `typeid`.

Множественное наследование. Ромбовидное наследование. Виртуальный базовый класс.

Вложенные классы и наследование.

Модификаторы `override` и `final`. Ковариантный тип возврата.

Раздел 5. Стандартная библиотека языка C++

Тема 5.1. Структура стандартной библиотеки и стандартной библиотеки шаблонов

Составные части стандартной библиотеки языка C++ (C++ Standard Library).

Составные части стандартной библиотеки шаблонов (Standard Template Library – STL).

Библиотека контейнеров. Контейнерные классы. Последовательные контейнеры. Адаптеры контейнеров. Ассоциативные контейнеры. Общие методы для работы с контейнерными классами.

Библиотека итераторов. Категории итераторов.

Тема 5.2. Элементы функционального программирования

Библиотека алгоритмов. Алгоритмы и встроенные функции.

Функциональные объекты. Лямбда-выражения и их использование для алгоритмов STL.

Функциональные объекты, которые представляют общие арифметические и логические операции в C++. Обертки функций. Частичное применение

функций. Отрицатели. Поисковые функции. Объектов функций ограниченного сравнения.

Тема 5.3. Библиотеки ввода-вывода и библиотека файловой системы

Библиотека потокового ввода-вывода в стиле ООП, семейство функций вывода на печать (с C++23) и стандартный набор функций ввода-вывода в стиле C.

Средства для выполнения операций с файловыми системами и их компонентами, такими как пути, обычные файлы и каталоги.

УЧЕБНО-МЕТОДИЧЕСКАЯ КАРТА УЧЕБНОЙ ДИСЦИПЛИНЫ

Очная форма получения высшего образования с применением дистанционных образовательных технологий (ДОТ)

Номер раздела, темы	Название раздела, темы	Количество аудиторных часов					Количество часов УСР	Форма контроля знаний
		Лекции	Практические занятия	Семинарские занятия	Лабораторные занятия	Иное		
1	2	3	4	5	6	7	8	9
1	Основы программирования	12			12			
1.1	Методологии программирования	2						Устный опрос
1.2	Системы счисления в программировании	1			2			Контрольная работа (компьютерный тест) №1
1.3	Современное положение языка C++	1						Устный опрос
1.4	Базовые понятия языка C++	8			10			Отчёт по лабораторным работам с их устной защитой. Контрольная работа №2
2	Структурное программирование на языке C++	18			20			

2.1	Составные типы данных и введение в STL	4			6			Отчёт по лабораторным работам с их устной защитой.
2.2	Ссылки, указатели, константность	4			4			Собеседование
2.3	Многомодульные приложения	5			4			Отчёт по лабораторным работам с их устной защитой.
2.4	Обработка исключительных ситуаций	1			2			Контрольная работа (компьютерный тест) №3
2.5	Типы строк в C++ и организация ввода-вывода	4			4			Контрольная работа №4
3	Инструменты современной разработки приложений	8			6		2	
3.1	Введение в тестирование приложений	1			2			Отчёт по лабораторным работам с их устной защитой. Дискуссия
3.2	Логирование событий	1			2			Отчёт по лабораторным работам с их устной защитой. Дискуссия

3.3	Система контроля версий	2					2	Отчеты по лабораторным работам с их устной защитой
3.4	Профилирование программ	2			2			Отчёт по лабораторным работам с их устной защитой. Собеседование
3.5	Язык UML	2						Устный опрос. Контрольная работа (компьютерный тест) №5
4	Объектно-ориентированное программирование на языке C++	18			14		6	
4.1	Основы объектно-ориентированного программирования	6			4			Отчёт по лабораторным работам с их устной защитой.
4.2	Перегрузка операторов	4			4			Отчёт по лабораторным работам с их устной защитой. Контрольная работа №6
4.3	Реализация пользовательских структур данных	3					6	Отчеты по лабораторным работам с их устной защитой

4.4	Шаблоны классов	1			2			Отчёт по лабораторным работам с их устной защитой.
4.5	Наследование и полиморфизм	4			4			Отчёт по лабораторным работам с их устной защитой. Контрольная работа (компьютерный тест) №7
5	Стандартная библиотека языка C++	8			8			
5.1	Структура стандартной библиотеки и стандартной библиотеки шаблонов	2			2			Дискуссия.
5.2	Элементы функционального программирования	4			4			Отчёт по лабораторным работам с их устной защитой. Контрольная работа №8
5.3	Библиотеки ввода-вывода и библиотека файловой системы	2			2			Отчёт по лабораторным работам с их устной защитой.

ИНФОРМАЦИОННО-МЕТОДИЧЕСКАЯ ЧАСТЬ

Перечень основной литературы

1. Барков, И. А. Объектно-ориентированное программирование: учебник [для вузов] / И. А. Барков. - Изд. 2-е, стер. - Санкт-Петербург; Москва; Краснодар: Лань, 2023. - 698 с. - URL: <https://e.lanbook.com/book/329549>.
2. Конова, Е. А. Алгоритмы и программы. Язык C++: учеб. пособие для студ., обуч. по напр. «Прикладная информатика» / Е. А. Конова, Г. А. Поллак. - Изд. 7-е, стер. - Санкт-Петербург; Москва; Краснодар: Лань, 2023. - 384 с. - URL: <https://reader.lanbook.com/book/297002>.
3. Лафоре, Роберт. Объектно-ориентированное программирование в C++ / Р. Лафоре; [пер. с англ.: А. Кузнецов, М. Назаров, В. Шрага]. - 4-е изд. - Санкт-Петербург; Москва; Минск: Питер, 2022. - 923 с. - URL: <https://ibooks.ru/bookshelf/376836>.
4. Павловская, Т. А. C/C++. Программирование на языке высокого уровня: для магистров и бакалавров: учебник для студентов высших учебных заведений, / Т. А. Павловская. - Санкт-Петербург [и др.]: Питер, 2020. - 460 с. - URL: <https://ibooks.ru/bookshelf/376844>.
5. Программирование. Сборник задач: учебное пособие / [О. Г. Архипов и др.]; под ред. М. М. Марана. - Санкт-Петербург; Москва; Краснодар: Лань, 2019. - 137 с. - URL: <https://reader.lanbook.com/book/206768>.

Перечень дополнительной литературы

1. Программирование: принципы и практика с использованием C++/ Страуструп, Бьярне. 2-е изд.: Пер. с англ. – М.: ООО «И.Д. Вильямс», 2016. – 1328 с.
2. Эффективный и современный C++: 42 рекомендации по использованию C++11 и C++14 / Мейерс, Скотт.: Пер. с англ. – М.: ООО «И.Д. Вильямс», 2019. – 304 с.
3. Программирование на C++ в примерах и задачах/ А. Васильев. – Эксмо, 2018. – 368 с.
4. Пикус Ф. Идиомы и паттерны проектирования в современном C++ /Ф. Пикус. – Москва: ДМК, 2020. - 452 с.
5. Вайсфельд, М. Объектно-ориентированный подход = The Object-Oriented Thought Process / М. Вайсфельд; [пер. с англ. И. Сигаилук]. - 5-е междунар. изд. - Санкт-Петербург [и др.]: Питер, 2020. - 254 с. - URL: <https://ibooks.ru/bookshelf/371700>.
6. Шилдт, Герберт. C/C++ : справочник программиста : [пер. с англ.] / Герберт Шилдт. - 3-е изд. - Москва ; Санкт-Петербург : Диалектика, 2019. - 429 с.
7. C/C++ в задачах и примерах/ Н. Культин. – Москва: BHV, 2019. – 272 с.

8. С++ за 24 часа/ Р. Кейденхед, Либерти Джесс. – «Вильямс», 2019. – 448 с.

9. Технология программирования на С++. Начальный курс/ Н. Литвиненко – BHV, 2005. – 288 с.

Электронные ресурсы

1. Образовательный портал БГУ [Электронный ресурс]. – Режим доступа: <https://edufpmi.bsu.by/course/view.php?id=8> – Дата доступа: 02.03.2023.

2. News, Status & Discussion about Standard C++ [Электронный ресурс]. – Режим доступа: <https://isocpp.org/> – Дата доступа: 02.03.2023.

3. cppreference.com [Электронный ресурс]. – Режим доступа: <https://en.cppreference.com/w/> – Дата доступа: 02.03.2023.

4. Документация по С++ [Электронный ресурс]. – Режим доступа: <https://learn.microsoft.com/ru-ru/cpp/cpp/?view=msvc-170> – Дата доступа: 02.03.2023.

5. С и С++ в Visual Studio [Электронный ресурс]. – Режим доступа: <https://learn.microsoft.com/ru-ru/cpp/overview/visual-cpp-in-visual-studio?view=msvc-170> – Дата доступа: 02.03.2023.

6. GoogleTest User's Guide [Электронный ресурс]. – Режим доступа: <https://google.github.io/googletest/> – Дата доступа: 02.03.2023.

7. gabime/spdlog: Fast C++ logging library [Электронный ресурс]. – Режим доступа: <https://github.com/gabime/spdlog> – Дата доступа: 02.03.2023.

Перечень рекомендуемых средств диагностики и методика формирования итоговой отметки

Объектом диагностики компетенций студентов являются знания, умения, полученные ими в результате изучения учебной дисциплины. Выявление учебных достижений студентов осуществляется с помощью мероприятий текущего контроля и промежуточной аттестации.

Для диагностики компетенций в рамках учебной дисциплины рекомендуется использовать следующие формы:

1. Устная форма: собеседование, дискуссия.
2. Письменная форма: контрольные работы.
3. Устно-письменная форма: отчёты по лабораторным работам с их устной защитой.
4. Техническая форма: компьютерные тесты.

В качестве рекомендуемых технических средств диагностики используется Образовательная платформа на платформе Moodle (<https://edufpmi.bsu.by>).

Формой промежуточной аттестации по дисциплине «Основы и методологии программирования» учебными планами предусмотрены **зачет и экзамен**.

При формировании итоговой отметки используется рейтинговая система оценки знаний студента, дающая возможность проследить и оценить динамику процесса достижения целей обучения. Рейтинговая система предусматривает использование весовых коэффициентов в ходе проведения контрольных мероприятий текущей аттестации.

Примерные весовые коэффициенты, определяющие вклад текущей аттестации в отметку при прохождении промежуточной аттестации:

Формирование отметки за текущую аттестацию:

- Отчёт по лабораторным работам с их устной защитой – 30 %.
- контрольные работы – 70 %.

Итоговая отметка по дисциплине рассчитывается на основе отметки текущей аттестации (рейтинговой системы оценки знаний) – 40% и экзаменационной отметки – 60%.

Примерный перечень заданий для управляемой самостоятельной работы

Управляемая самостоятельная работа предлагается в виде индивидуальных заданий.

Тема 3.3. Система контроля версий (2 ч).

Создание репозитория для кода программ. Управление ветками, коммитами.

Тема задания: Создание и управление хранилищем кода проекта.

Форма контроля – отчёты по лабораторным работам с их устной защитой.

Тема 4.3. Реализация пользовательских структур данных (6 ч).

Абстрактные типы данных: стек, дек, очередь, список. Их реализация на массивах, в виде последовательности связанных компонент.

Тема задания: Реализация последовательных структур данных.

Форма контроля – отчёты по лабораторным работам с их устной защитой.

Примерная тематика лабораторных занятий

Занятие 1-2. Системы счисления в программировании (*выполнение теста в системе <https://edufpmi.bsu.by>*).

Базовые понятия языка C++. Структура программы. Компиляторы и среды разработки.

Алфавит, синтаксис, семантика языка C++. Фундаментальные типы данных C++. Переменные. Операторы языка C++. Приоритет операторов.

Лабораторная работа №1. Линейные алгоритмы (выполнение индивидуальных заданий).

Занятие 3-4. Пространство имен std. using-объявления и using-директивы. Логический блок. Область видимости переменных.

Инструкции ветвления, тернарный оператор.

Функции. Функция main.

Основы Code Style для языка C++.

Лабораторная работа №2. Алгоритмы ветвления (выполнение индивидуальных заданий).

Занятие 5-6. Типы данных std::vector, std::string. Инструкции while, do-while, for, range-for. Алгоритмы целочисленной арифметики.

Потоковый ввод-вывод.

Преобразования типов данных.

Лабораторная работа №3. Циклы и массивы (выполнение индивидуальных заданий).

Занятие 7-8. Введение в STL. Контейнеры и другие классы. Основы работы с итераторами.

Библиотека алгоритмов. Знакомство с лямбда-выражениями в C++.

Лабораторная работа №4. Введение в STL (выполнение индивидуальных заданий).

Занятие 9-10. Тестирование приложений. Логирования событий. Реализация алгоритмов внутренних сортировок.

Лабораторная работа №5. Тестирование и логирование приложений (выполнение индивидуальных заданий).

Занятие 11. Использование системы контроля версий для отслеживания изменений кода.

Структуры. Перечисления. Объединения. std::variant.

Лабораторная работа №6. Использование структур и перечислений (выполнение индивидуальных заданий).

Занятие 12-13. Ссылки. Адресная арифметика. Константы и указатели. Преобразования указателя.

Статические и динамические массивы. Одномерные и двумерные массивы.

Функции. Способы передачи параметров. Возвращение указателей и ссылок. Указатели на функцию.

Лабораторная работа №7. Адресная арифметика. Динамическое выделение памяти (выполнение индивидуальных заданий).

Занятие 14-15. Многофайловые проекты. Отделение объявления от реализации. Понятие namespace.

Ключевое слово static, extern.

Рекурсивные и итеративные функции.

Перегрузка функций.

Шаблоны функций. Шаблонные структуры.

Лабораторная работа №8. Перегрузка функций. Шаблоны функций (выполнение индивидуальных заданий).

Занятие 16-17. Исключительные ситуации. Обработка исключений.

Профилирование программ. Инструменты профилирования.

Лабораторная работа №9. Исключительные ситуации. Профилирование программ (выполнение индивидуальных заданий).

Занятие 18-19. Библиотека строк. Текстовые и бинарные файлы. Строковые потоки.

Библиотека регулярных выражений.

Лабораторная работа №10. Текстовые и бинарные файлы (выполнение индивидуальных заданий).

Занятие 20-21. Принципы объектно-ориентированного программирования (ООП).

Классы. Конструкторы. Инициализация объектов. Деструкторы. Правило трех. Правило пяти. Идиома RAII. Умные указатели.

Лабораторная работа №11. Объектно-ориентированное программирование. Конструкторы. Идиома RAII (выполнение индивидуальных заданий).

Занятие 22-23. Перегрузка операторов. Дружественные функции. Дружественные классы.

Лабораторная работа №12. Перегрузка операторов (выполнение индивидуальных заданий).

Занятие 24. Шаблоны классов. Использование стека для вычисления выражений на основе обратной польской записи.

Реализация деревьев и поисковых деревьев.

Лабораторная работа №13. Шаблоны классов (выполнение индивидуальных заданий).

Занятие 25-26. Наследование и полиморфизм. Множественное наследование.

Лабораторная работа №14. Наследование и полиморфизм (выполнение индивидуальных заданий).

Занятие 27-30. Стандартная библиотека языка C++. Библиотека контейнеров. Библиотека итераторов. Библиотека алгоритмов.

Функциональные объекты. Лямбда-выражения и их использование для алгоритмов STL.

Библиотеки ввода-вывода и библиотека файловой системы.

Лабораторная работа №15. Элементы функционального программирования (выполнение индивидуальных заданий).

Примерная тематика контрольных работ

1) Контрольная работа (компьютерный тест) №1 «Системы счисления в программировании».

2) Контрольная работа № 2. «Базовые понятия языка C++».

3) Контрольная работа (компьютерный тест) №3. «Обработка исключительных ситуаций».

4) Контрольная работа № 4. «Типы строк в C++ и организация ввода-вывода».

5) Контрольная работа (компьютерный тест) №5. «Язык UML».

6) Контрольная работа № 6. «Перегрузка операторов».

7) Контрольная работа (компьютерный тест) №7. «Наследование и полиморфизм».

8) Контрольная работа № 8. «Элементы функционального программирования».

Примерный вариант контрольной работы

Контрольная работа № 4.

«Типы строк в C++ и организация ввода-вывода»

Задание 1:

Дан текстовый файл, в первой строке которого задано целое число k , т.ч. $0 \leq k \leq 31$, во второй строке - последовательность целых положительных чисел, числа разделены пробелом.

Из исходной последовательности чисел сформировать последовательность, в которой для каждого числа последовательности выполняется условие: последовательность цифр 11 в двоичном представлении числа встречается k раз (например, в двоичном представлении 11110111 оно встречается 5 раз).

Построенную коллекцию сохранить в файл.

Задание 3:

Дана программа на C++ (входной файл, например, main.cpp), содержащая корректно отформатированный, компилируемый код на языке C++. В программе могут присутствовать однострочные комментарии.

Записать в выходной файл построчно все числовые константы, встречающиеся в этой программе, вместе с номером строки программы, где эта константа встречается.

Указание: Часть строки после // игнорировать и не искать в них числовые константы.

Пример:

	Входной файл	Выходной файл
1	<code>#include <iostream> int main() {</code>	3 -100
2	<code>int x = -100;</code>	4 2
3	<code>int y = x - 2; // = 87</code>	5 1
4	<code>int z = 1 + ++x + 5 + 99;</code>	5 5
5	<code>std::cout << "z + 5 = " << z << '\n';</code>	5 99
6		9 2
7	<code>int a1 = x + y;</code>	9 -7
8	<code>int b = 2 + a1 + -7;</code>	11 10
9	<code>char c = 0x41;</code>	
10	<code>std::cout << "c = " << c << ", c + 10 = " <<</code>	
11	<code>(char)(c + 10) << '\n';</code>	
12	<code>}</code>	

Описание инновационных подходов и методов к преподаванию учебной дисциплины

При организации образовательного процесса используются следующие методы:

– **метод учебной дискуссии**, который предполагает участие студентов в целенаправленном обмене мнениями, идеями для предъявления и/или согласования существующих позиций по определенной проблеме. Использование метода обеспечивает появление нового уровня понимания изучаемой темы, применение знаний (теорий, концепций) при решении проблем, определение способов их решения.

– **метод группового обучения**, который представляет собой форму организации учебно-познавательной деятельности обучающихся, предполагающую функционирование разных типов малых групп, работающих как над общими, так и специфическими учебными заданиями.

В качестве технических средств для организации работы в рамках учебной дисциплины рекомендуется использовать Образовательный портал БГУ (<https://edufpmi.bsu.by>) – инструмент с эффективной функциональностью контроля, тренинга и самостоятельной работы.

Методические рекомендации по организации самостоятельной работы обучающихся

Для организации самостоятельной работы студентов по учебной дисциплине следует использовать современные информационные ресурсы: разместить на образовательном портале комплекс учебных и учебно-методических материалов (учебно-программные материалы, учебное издание для теоретического изучения дисциплины, методические указания к лабораторным занятиям, материалы текущего контроля и промежуточной аттестации, позволяющие определить соответствие учебной деятельности обучающихся требованиям образовательного стандарта высшего образования и учебно-программной документации, в т.ч. вопросы для подготовки к зачету, экзамену, задания, тесты, вопросы для самоконтроля, список рекомендуемой литературы, информационных ресурсов и др.).

При составлении общих и индивидуальных заданий по учебной дисциплине необходимо предусмотреть возрастание их сложности: от заданий, формирующих достаточные знания по изученному учебному материалу на уровне узнавания, к заданиям, формирующим компетенции на уровне воспроизведения, и далее к заданиям, формирующим компетенции на уровне применения полученных знаний.

Таким образом, задания по учебной дисциплине рекомендуется делить на три модуля:

– **задания, формирующие достаточные знания по изученному учебному материалу на уровне узнавания:**

Задание 1. Задания на знакомство с синтаксисом:

- Объявление переменных разных типов (целочисленных, вещественных, символьных).

- Использование операторов присваивания.

- Ввод и вывод данных с помощью `std::cin` и `std::cout`.

- Знакомство с комментариями (однострочными и многострочными).

Задание 2. Задания на работу с операторами:

- Арифметические операции (сложение, вычитание, умножение, деление).

- Операторы сравнения (равенство, неравенство, больше, меньше).

- Логические операторы (и, или, не).

Задание 3. Задания на условные операторы:

- Использование инструкций `if`, `else if`, `else`.

- Создание простых условий на основе значений переменных.

Задание 4. Задания на циклы:

- Использование инструкций `for`, `while`, `do-while`.

- Простые циклы для вывода чисел от 1 до `n`.

Задание 5. Задания на работу с функциями:

- Объявление и вызов функций.

- Передача параметров в функции.

- Возврат значений из функций.

Задание 6. Задания на работу с массивами:

- Объявление и инициализация массивов.

- Циклы для обхода элементов массива.

Задание 7. Задания на работу с строками:

- Использование строковых литералов.

- Функции для работы со строками (например, `strlen`, `strcpy`, `strcat`).

Задание 8. Задания на базовые алгоритмы:

- Нахождение суммы элементов массива.

- Поиск максимального/минимального значения в массиве.

– **задания, формирующие компетенции на уровне воспроизведения:**

Задание 1. Найти требуемый элемент в одномерном массиве. Какие из изученных алгоритмов можно применить, если известно, что массив а) упорядочен; б) неупорядочен?

Задание 2. Перечислите известные вам алгоритмы внутренней сортировки. Какие из этих алгоритмов реализованы в стандартных функциях языка C++? Как применить стандартные функции сортировки для массива, содержащего объекты нестандартного типа?

Задание 3. Перечислите известные вам функции обработки строк? Какие из них понадобятся, чтобы в строке поменять местами слово максимальной и минимальной длины? Убрать повторяющиеся символы? Найти все слова, являющиеся целыми числами?

Задание 4. Реализовать линейный однонаправленный список на последовательности связанных компонент, упорядоченный по возрастанию ключей. Имя входного файла: `input.txt`. Имя выходного файла: `output.txt`.

Формат входных данных. Входной файл содержит последовательность чисел – ключи в порядке добавления в список. Ключи задаются в формате по одному в строке.

Формат выходных данных. Выходной файл должен содержать последовательность ключей, полученную при проходе по списку в прямом порядке.

– задания, формирующие компетенции на уровне применения полученных знаний:

Задание 1. Реализовать объектно-ориентированное приложение для решения задачи объединения двух линейных однонаправленных списка, упорядоченных по возрастанию ключей (ключи в списке не повторяются). При объединении списков одинаковые ключи включаются в итоговый список один раз. Имя входного файла: input.txt. Имя выходного файла: output.txt.

Формат входных данных. Входной файл содержит две строки, в каждой – последовательность чисел – ключи в порядке добавления их в первый и второй список соответственно. Числа разделены пробелами.

Формат выходных данных. Выходной файл должен содержать последовательность ключей, полученную при проходе по итоговому списку в прямом порядке.

Задание 2. Реализовать объектно-ориентированное приложение для сортировки транспортных средств по убыванию цены. Реализовать класс Repo (репозиторий) для хранения транспортных средств класса Vehicle. Иерархия транспортных средств: базовый класс Vehicle, представляющий транспортное средство. Его производные классы Car и Bicycle.

Имя входного файла: input.txt. Имя выходного файла: output.txt.

Формат входных данных. Входной файл содержит информацию о транспортном средстве в строковом виде: название_ТС модель_ТС цена_ТС.

Формат выходных данных. Выходной файл должен содержать строки – упорядоченную последовательность транспортных средств в формате: цена_ТС название_ТС модель_ТС.

Примерный перечень заданий к зачету

1. Описать структуру программы на языке C++. Указать как объявлять и инициализировать переменные и константы.

2. Описать размерность фундаментальных типов данных в C++. Привести примеры различных вариантов приведения типов.

3. Привести примеры использования унарных и бинарных арифметических операторов в C++, операторов сравнения при вычислении выражений.

4. Реализовать вычисления с использованием операторов сдвига, побитовых и логических операторов C++. Описать особенности вычисления логических выражений – неполное вычисление логических выражений.

5. Написать код, демонстрирующий использование операторов присваивания, инкремента и декремента, тернарного оператора.

6. Описать особенности использования оператора запятая.
7. Реализовывать алгоритм с использованием инструкции выбора if и switch.
8. Реализовывать циклический алгоритм с использованием инструкций: while, do-while, for, range-for.
9. Описать многомодульное приложение с объявлением и определением функции в C++. Описать разнице между формальными и фактическими параметрами.
10. Реализовывать функции с использованием разных способов передачи параметров (по значению и по ссылке).
11. Разработать и реализовать рекурсивный алгоритм решения задачи.
12. Реализовать приложение с перегрузкой функций.
13. Описать понятия ссылка, указатель. Провести код приложения для демонстрации их особенностей.
14. Разработать и реализовать алгоритм решения задачи с использованием шаблонов функций.
15. Разработать и реализовать алгоритм решения задачи с использованием статических одномерных и двумерных массивов.
16. Описать возможности и способы применения контейнеров `std::vector` и `std::array`.
17. Разработать и реализовать алгоритм решения задачи с использованием динамического выделения памяти под одномерные и двумерные массивы.
18. При реализации алгоритма задачи осуществлять передачу одномерных и двумерных массивов в качестве параметров функции.
19. Реализовать алгоритмы сортировки: выбором, вставками (прямой алгоритм, двоичный), пузырьком, шейкерной.
20. Описать способы представления строковых данных: нуль-терминированные строки и класс `std::string`.
21. Описать основные функции для обработки C-строк. Привести примеры.
22. Продемонстрировать основные функции класса `std::string`.
23. Разработать приложение для работы с текстовыми и бинарными файлами.
24. Разработать приложение с использованием потоков ввода-вывода, и возможностью форматированного ввода-вывода.
25. При разработке приложения выполнить обработку и создания исключений.
26. Разработать классы для реализации линейных структур данных: стек, дек, очередь, список на массиве и на последовательности связанных компонент.
27. Описать способы объявления и реализации класса с использованием ключевых слов `struct` и `class`, указать их сходства и отличия.
28. На примере кода программы продемонстрировать способы доступа к членам класса с учетом спецификатора доступа и с учетом способа объявления объектов.

29. Разработать приложение, в котором используются разные способы создания объектов с помощью конструкторов.

30. Разработать приложение, в котором реализованы конструкторы копирования и перемещения.

31. Описать идиому RAII и ее использование при проектировании классов.

32. Разработать приложения для демонстрации возможностей использования виртуальных деструкторов.

33. Описать различные способы перегрузки операторов для при разработке пользовательских классов.

Примерный перечень вопросов к экзамену

1. История и предназначение C++.
2. Алгоритм и его свойства. Формализации понятия «алгоритм».
3. Этапы и принципы разработки алгоритмов.
4. О стилях кодирования. Code Style на примере QT C++ Style и требований в рамках курса. Переменные, функции и их именование, константы, комментарии, заголовочные файлы и др.
5. Позиционные системы счисления. Двоичная, восьмеричная, шестнадцатеричная системы счисления. Преобразования между ними.
6. Представление целых чисел (прямой, обратный, дополнительный код) в 2с\с.
7. Представление вещественных чисел в 2с\с.
8. Структура программ и заголовочные файлы. Базовые типы данных.
9. Типы данных с плавающей точкой. Экспоненциальная запись. Конвертация чисел в экспоненциальную запись. Точность и диапазон типов с плавающей точкой. Ошибки округления. nan и inf.
10. Библиотека cmath. Основные функции.
11. Приоритет операций. Оператор запятой.
12. Арифметические операции. Битовые операции.
13. Оператор присваивания. Присваивание и инициализация. const и constexpr. Ссылка.
14. Безопасность типов. Безопасные\небезопасные преобразования.
15. Пространство имен. Пространство имен std. Логический блок. Область видимости переменных.
16. Консольный ввод и вывод. Файловый ввод\вывод. Форматированный вывод.
17. Логические операции. Инструкции выбора (if, if-else, вложенные if). Тернарный оператор.
18. Инструкции выбора (switch). break.
19. Циклы (for, while, do-while, range-for). Ключевые слова auto. Инструкции break, continue.

20. Основные алгоритмы сортировки и их улучшения: выбором, вставками (прямой, двоичный), пузырьком (шейкерная) и др.
21. Функции. Объявление и определение. Встроенные функции (inline). Передача параметров по ссылке и по значению. Параметры по умолчанию. Константность аргументов функции.
22. Возврат значения функции по ссылке и по значению. Ключевое слово `const` для возвращаемого значения из функции.
23. Перегрузка функций.
24. Рекурсия прямая и косвенная. Примеры рекурсивных алгоритмов. Рекурсивные сортировки. Итеративные функции.
25. Шаблоны функций.
26. Указатель на функцию. Массив указателей на функцию. `std::function`.
27. Модульное программирование. Многофайловые проекты. Отделение объявления от реализации. Пространство имен. Глобальная инициализация. Статические переменные, статические функции. Ключевое слово `extern`.
28. Компиляция, линковка. Директивы условной компиляции.
29. Отладка приложения. Типы ошибок. Обработка ошибок, `cerr` и `exit()`. `assert` и `static_assert`.
30. Возникновение, создание исключений. Обработка исключений. Блок `try-catch`. `catch(...)`.
31. Введение в тестирование. Подготовка `test-cases`. Ручное тестирование.
32. Разработка через тестирование (TDD). Библиотеки для `unit-тестирования`, их характеристики (`gtest` и др.). Ключевые понятия. Утверждения (`ASSERT_`, `EXPECT_`).
33. Классы-тесты (фикстуры). Пример класса-теста для тестирования класса в приложении (например класса `List` и др.).
34. Логирование событий. На примере библиотеки `spdlog`: уровни журналирования, направления вывода сообщений, шаблон вывода сообщения. Особенности вывода сообщения в файл.
35. Потоки ввода-вывода. Основные методы ввода/вывода. Обработка ошибок ввода-вывода. Форматированные потоки и манипуляторы. Форматирование вывода.
36. Работа с файловыми потоками. Режимы открытия файлов. Позиционирование. Текстовые и бинарные файлы. Чтение и запись файла.
37. Строковые потоки (`istream`, `ostream`).
38. Память, адреса и указатели. Объявление и инициализация указателя. Указатели и ссылки. `nullptr`. Преобразование указателя.
39. Арифметические операции над указателями. Оператор `sizeof`.
40. Динамически распределяемая память. Операторы `new` и `delete`. Утечка памяти. Деструкторы и динамическая память.

41. Связь между указателями и массивами. Статические и динамические массивы. Работа с C-массивами. Многомерные массивы в стиле C. Операторы `new[]` и `delete[]`.

42. Указатели на объекты класса и ссылки на объекты класса. Указатели и ссылки на объекты класса, как параметры функций и как возвращаемые значения. Ключевое слово `const` для возвращаемого значения функции.

43. Тип `void*` и операторы приведения типов.

44. Указатели и `const`. Ссылки и `const`. Оператор доступа к членам через указатель. Указатели на указатели.

45. Парадигмы программирования (процедурное, объектно-ориентированное, функциональное).

46. Принципы ООП (абстракция, наследование, полиморфизм, инкапсуляция). Преимущества использования ООП.

47. Принципы SOLID, KISS, DRY, WET, YAGNI.

48. UML. Диаграмма классов, отношение между классами. Примеры реализаций.

49. Типы, определенные пользователем (структура, класс, объединение (`union`)).

50. Перечисления (`enum`). Класс `enum`.

51. Различия в `struct` и `class`. Объявление и определение классов. Размер структуры. Классы и члены класса. Модификаторы доступа `public` и `private`, `protected`. Инициализация объектов классов. Инициализация нестатических членов классов. Присваивание значений членам структур. Значения по умолчанию.

52. Список инициализации `std::initializer_list`. Его использование для создания объектов.

53. Конструкторы. Конструкторы по умолчанию, Параметризованные конструкторы. Деструкторы. Правило трех. Правило пяти. Конструктор копирования. Перегрузка оператора присваивания. Поверхностное и глубокое копирование. Копирование перемещения и Оператор присваивания перемещением.

54. Внешнее и внутреннее объявление классов. Вложенные классы. Инициализация вложенных классов.

55. Ключевые слова `static` и `extern`. Инициализация статических членов класса.

56. Константные объекты классов. Константные методы классов. Константные ссылки и классы. Перегрузка константных и не константных функций.

57. Указатель `this`. Время жизни объекта. Анонимные объекты.

58. Перегрузка операторов. Ограничения в перегрузке операторов. Перегрузка операторов через дружественные функции, через обычные

функции, через члены-функции класса. Дружественные функции, дружественные классы.

59. Перегрузка операторов сравнения. Перегрузка арифметических бинарных операторов, перегрузка операторов << и >>.

60. Префиксный/постфиксный инкремент/декремент, унарных операторов плюс, минус и логического НЕ.

61. Перегрузка оператора [], перегрузка оператора (). Перегрузка оператора приведения типа.

62. Абстрактные типа данных (АТД) стек, очередь. Базовые операции.

63. Абстрактные типы данных (АТД) дек, список. Базовые операции.

64. Варианты реализации структуры данных Стек: на массиве, через последовательность связанных компонент.

65. Варианты реализации структуры данных Очередь: на массиве, через последовательность связанных компонент.

66. Варианты реализации структуры данных Дек: на массиве, через последовательность связанных компонент.

67. Варианты реализации структуры данных Однонаправленный список: на массиве, через последовательность связанных компонент.

68. Варианты реализации структуры данных Двухнаправленный список: на массиве, через последовательность связанных компонент.

69. Наследование классов. Интерфейсы. Абстрактные классы. Защищенный доступ (protected). Обрезка объектов.

70. Множественное наследование. Проблема ромбовидного наследования. Виртуальный базовый класс.

71. Вложенные классы и наследование.

72. Статический и динамический полиморфизм. Раннее и позднее связывания. Указатели/ссылки и наследование. Виртуальные функции. Чистые виртуальные функции. Таблица виртуальных функций.

73. Виртуальные деструкторы и присваивание.

74. Вызов и переопределение методов родительского класса. Модификаторы override и final. Ковариантный тип возврата.

75. Базовый класс и наследник – приведение типов по иерархии, динамическое приведение типов. Оператор dynamic_cast.

76. Преобразования типов: неявные преобразования, const_cast, static_cast, reinterpret_cast, explicit_cast, dynamic_cast.

77. Шаблоны функций. Экземпляры шаблонов функций. Параметр non-type в шаблоне.

78. Шаблоны классов и шаблоны членов-функций классов. Конкретизация и явная специализация. Явная специализация шаблона функции. Явная специализация шаблона класса. Частичная специализация шаблона класса. Частичная специализация шаблонов классов для типов указателей.

79. Стандартная библиотека C++ и ее компоненты. Стандартная библиотека шаблонов (STL) и ее компоненты.

80. Библиотека контейнеров STL. Возможности контейнеров и операции над контейнерами. Аллокатор.

81. Библиотека Итераторы STL. Категории и итераторов. Адаптеры итераторов.

82. Библиотека алгоритмов. Категории алгоритмов STL.

83. Функциональные объекты. Функциональные объекты. Лямбда-выражения и их использование для алгоритмов STL.

84. Функциональные объекты, которые представляют общие арифметические и логические операции в C++. Обертки функций. Частичное применение функций. Отрицатели. Поисковые функции. Объектов функций ограниченного сравнения.

85. Библиотека строк. Символьные типы данных. Классы `std::basic_string`, `std::basic_string_view` и их специализации. Основные функции этих классов, примеры использования.

86. Библиотека строк. С-строки. Функции работы с ними. Библиотека `cctype`.

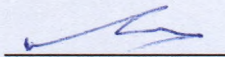
87. Библиотека потокового ввода-вывода в стиле ООП, семейство функций вывода на печать (с C++23) и стандартный набор функций ввода-вывода в стиле C.

88. Средства для выполнения операций с файловыми системами и их компонентами, такими как пути, обычные файлы и каталоги.

ПРОТОКОЛ СОГЛАСОВАНИЯ УЧЕБНОЙ ПРОГРАММЫ УО

Название учебной дисциплины, с которой требуется согласование	Название кафедры	Предложения об изменениях в содержании учебной программы учреждения высшего образования по учебной дисциплине	Решение, принятое кафедрой, разработавшей учебную программу (с указанием даты и номера протокола)
1. Разработка кросс-платформенных приложений	Многопроцессорных систем и сетей	нет	Изменений не требуется (протокол № 13 от 06.04.2023)
2. Машинно-ориентированное программирование	Многопроцессорных систем и сетей	нет	Изменений не требуется (протокол № 13 от 06.04.2023)
3. Алгоритмы и структуры данных	Дискретной математики и алгоритмики	нет	Изменений не требуется (протокол № 13 от 06.04.2023)
4. Операционные системы	Многопроцессорных систем и сетей	нет	Изменений не требуется (протокол № 13 от 06.04.2023)

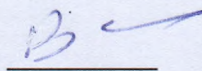
Заведующий кафедрой МСС
к. физ.-мат. наук, доцент



С.В.Марков

06.04.2023

Заведующий кафедрой ДМА
доктор физ.-мат. наук, профессор



В.М.Котов

06.04.2023

**ДОПОЛНЕНИЯ И ИЗМЕНЕНИЯ К УЧЕБНОЙ ПРОГРАММЕ ПО
ИЗУЧАЕМОЙ УЧЕБНОЙ ДИСЦИПЛИНЕ**

на ____ / ____ учебный год

№ п/п	Дополнения и изменения	Основание

Учебная программа пересмотрена и одобрена на заседании кафедры
_____ (протокол № ____ от _____ 202_ г.)

Заведующий кафедрой

УТВЕРЖДАЮ
Декан факультета
