ЮЖНО-УРАЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

УТВЕРЖДАЮ:
Заведующий выпускающей кафедрой

ЮУрГУ
Электронный документ, подписанный ПЭП, хранится в системе электронного документооборота Южно-Уральского государственного университета
СВЕДЕНИЯ О ВЛАДЕЛЬЦЕ ПЭП
Кому выдан: Замышляева А. А.
Пользователь: zamyshliaevaaa
Дата подписания: 05.09.2022

А. А. Замышляева

РАБОЧАЯ ПРОГРАММА

дисциплины 1.Ф.П2.03 Программирование на C++
для направления 01.03.02 Прикладная математика и информатика
уровень Бакалавриат
профиль подготовки Компьютерные технологии и разработка программных систем
форма обучения очная
кафедра-разработчик Прикладная математика и программирование

Рабочая программа составлена в соответствии с ФГОС ВО по направлению подготовки 01.03.02 Прикладная математика и информатика, утверждённым приказом Минобрнауки от 10.01.2018 № 9

Зав.кафедрой разработчика, д.физ.-мат.н., проф.

ЮУрГУ
Электронный документ, подписанный ПЭП, хранится в системе электронного документооборота Южно-Уральского государственного университета
СВЕДЕНИЯ О ВЛАДЕЛЬЦЕ ПЭП
Кому выдан: Замышляева А. А.
Пользователь: zamyshliaevaaa
Дата подписания: 21.08.2022

А. А. Замышляева

Разработчик программы, старший преподаватель

ЮУрГУ
Электронный документ, подписанный ПЭП, хранится в системе электронного документооборота Южно-Уральского государственного университета
СВЕДЕНИЯ О ВЛАДЕЛЬЦЕ ПЭП
Кому выдан: Сартасова М. Ю.
Пользователь: sartasovami
Дата подписания: 24.08.2022

М. Ю. Сартасова

Челябинск

1. Цели и задачи дисциплины

Целью преподавания и изучения дисциплины является обучение студентов языку C++, методике разработки программ с использованием технологии объектно-ориентированного программирования. Задачи дисциплины заключаются в том, чтобы студенты получили опыт разработки компьютерных программ на языке C++, могли реализовать на C++ математические алгоритмы; освоили синтаксис и стандартные библиотеки C++

Краткое содержание дисциплины

Введение в язык. Модификаторы типов. Введение в ООП. Перегрузка операторов. Наследование. Шаблоны. Исключения. Аллокаторы. Контейнеры. Итераторы. Move-семантика и rvalue-ссылки. Умные указатели. Вывод типов. Шаблонное метапрограммирование и SFINAE. Функциональные объекты и лямбда-функции. Некоторые особые полезные типы.

2. Компетенции обучающегося, формируемые в результате освоения дисциплины

Планируемые результаты освоения ОП ВО (компетенции)	Планируемые результаты обучения по дисциплине
ПК-1 Способен активно участвовать в разработке системного и прикладного программного обеспечения	Знает: методы и средства разработки программного обеспечения на языке программирования C++ Умеет: программировать на языке C++ Имеет практический опыт: проектирования, кодирования и отладки программного обеспечения, разрабатываемого на языке C++

3. Место дисциплины в структуре ОП ВО

Перечень предшествующих дисциплин, видов работ учебного плана	Перечень последующих дисциплин, видов работ
Нет	Скриптовые языки программирования, Методы трансляции и формальные языки, Анализ требований и проектирование ПО, Базы данных, Web-программирование, Программирование на языке Python, Программирование на C#, Программирование для мобильных устройств

Требования к «входным» знаниям, умениям, навыкам студента, необходимым при освоении данной дисциплины и приобретенным в результате освоения предшествующих дисциплин:

Нет

4. Объём и виды учебной работы

Общая трудоемкость дисциплины составляет 3 з.е., 112 ч., 68,5 ч. контактной работы

Вид учебной работы	Всего часов	Распределение по семестрам в часах
		Номер семестра
		1
Общая трудоёмкость дисциплины	112	112
Аудиторные занятия:	64	64
Лекции (Л)	32	32
Практические занятия, семинары и (или) другие виды аудиторных занятий (ПЗ)	32	32
Лабораторные работы (ЛР)	0	0
Самостоятельная работа (СРС)	39,5	39,5
Изучение теоретического материала для практических занятий и контрольных работ	27	27
Подготовка к экзамену	12,5	12.5
Консультации и промежуточная аттестация	8,5	8,5
Вид контроля (зачет, диф.зачет, экзамен)	-	экзамен

5. Содержание дисциплины

№ раздела	Наименование разделов дисциплины	Объем аудиторных занятий по видам в часах			
		Всего	Л	ПЗ	ЛР
1	Язык C++ и ООП	32	16	16	0
2	Библиотека STL	32	16	16	0

5.1. Лекции

№ лекции	№ раздела	Наименование или краткое содержание лекционного занятия	Кол-во часов
1	1	Введение в язык Общие слова: место языка C++ среди современных языков, актуальные версии этого языка, ключевые люди, связанные с этим языком, официальный стандарт языка Структура программы, функция main, понятие области видимости (scope). Ключевые слова. Ввод-вывод (cin, cout). Объявления (declarations). Идентификаторы. Фундаментальные типы (int, long, long long, float, double, long double, char, bool, модификаторы signed и unsigned). Размеры этих типов, основные операции над ними, неявные преобразования типов между собой. Литералы, литеральные суффиксы для основных типов. Объявления функций, разница между объявлением и определением, one definition rule. Выражения (expressions). Операторы. Арифметические операторы. Побитовые операторы. Логические операторы, особенности их работы. Оператор присваивания и операторы составного присваивания, особенности его работы. Понятие lvalue и rvalue в C++03. Инкремент и декремент, отличие префиксной версии от постфиксной. Операторы сравнения. Тернарный оператор. Оператор “запятая”. Оператор sizeof. Инструкции (statements). Конструкции if...else, for, while, do...while, switch, их синтаксис, правила работы. Инструкции break, continue, return, их действие. Инструкция goto и метки. Понятия ошибки компиляции, ошибки времени выполнения (runtime error), неопределенного поведения (undefined behaviour), отличия между ними, примеры. Виды ошибок компиляции:	2

		лексические, синтаксические, семантические. Понятие segmentation fault и stack overflow.	
2	1	Модификаторы типов Указатели, операции над ними. Операция взятия адреса. Автоматическая память (стек). Массивы, операция [] (квадратные скобки), ее принцип работы. Указатель на void и его особенности. Функции. Перегрузка функций, правила разрешения перегрузки (общая схема, без деталей). Функции с аргументами по умолчанию. Функции с неуказанным количеством аргументов. Указатели на функции, операции над ними, их особенности. Динамическая память. Операторы new и new[], их использование (в стандартной форме). Операторы delete и delete[], их использование (в стандартной форме). Проблема утечек памяти. Проблема двойного удаления. Передача аргументов по значению и по указателю, первая версия функции swap. Дилемма с присваиванием (создавать новое название или копию?). Идея ссылок (references). Отличия ссылок от указателей, правила работы со ссылками, вторая версия функции swap. Проблема, связанная со ссылкой на локальную переменную (“битые ссылки”). Идея констант, ключевое слово const. Понятие константных и неконстантных операций, особенности работы с константами. Константные и неконстантные ссылки. Константные указатели и указатели на константу. Разрешенные и запрещенные присваивания между всеми вышеупомянутыми типами. Виды приведений типов: static_cast, reinterpret_cast, const_cast и C-style cast, их особенности, примеры применения и примеры, когда они не сработают.	2
3	1	Введение в ООП Идея ООП. Понятия класса и структуры, членов класса. Поля и методы, понятие инкапсуляции. Модификаторы доступа. Конструкторы и деструкторы. Конструктор по умолчанию. Перегрузка конструкторов. Конструктор копирования, его сигнатура и схема реализации. Пример, когда необходим нетривиальный конструктор копирования и оператор присваивания. Правила генерации компилятором конструкторов. Ключевые слова default и delete в контексте определения функций-членов. Операторы “точка” и “стрелочка”. Ключевое слово this и пример использования. Оператор присваивания, его сигнатура и схема реализации. “Правило трех”. Проблема с инициализацией констант и ссылок. Решение: списки инициализации в конструкторах. Ключевое слово explicit. Пример с конструктором String(int n). Константные и неконстантные методы, примеры. Ключевое слово mutable, пример применения. Понятие дружественных функций и классов, ключевое слово friend. Проблема вызова конструкторов из других конструкторов. Решение: делегирующие конструкторы. Статические поля и методы, пример. Локальные статические переменные. Указатели на члены и указатели на методы. Синтаксис объявления, пример использования. Операторы “точка со звездочкой” и “стрелочка со звездочкой”.	2
4	1	Перегрузка операторов Общая идея перегрузки операторов. Перегрузка арифметических операторов на примере класса BigInteger: бинарные операторы, составные присваивания с ними, правильное выражение одного через другое. Проблема с корректностью выражений вида “x+y=5;”. Проблема в случае левого операнда - не объекта класса (выражения вида “5+x”). Перегрузка операторов << и >> на примере потокового ввода-вывода. Перегрузка операторов сравнения, правильное выражение одних сравнений через другие. Перегрузка инкремента и декремента (префиксного и постфиксного). Перегрузка оператора [] (квадратные скобки). Правильное соблюдение константности при перегрузке оператора []. Перегрузка оператора “круглые скобки”. Понятие функтора и функционального класса, компаратора. Пример использования в стандартных алгоритмах. Особенности перегрузки операторов “логическое И”, “логическое ИЛИ” и “запятая”. Особенности перегрузки операторов “унарная звездочка”, “унарный амперсанд” и “стрелочка”. Перегрузка операторов приведения типа. Еще	2

		одно применение ключевого слова explicit.	
5	1	Наследование Объявление наследования. Модификатор доступа protected. Разница между приватным, публичным и защищенным наследованием. Разница между наследованием классов и структур. Поиск имен при наследовании. Сокрытие имен наследником. Явный вызов методов родителя у наследника. Использование :: и using. Проблемы с видимостью названий родителей и их полей у потомков в случае двухуровневого наследования, где первый уровень - приватное наследование. Правила действия слова friend в этих случаях. Порядок вызова конструкторов и деструкторов при наследовании. Проблема с инициализацией родителей при определении конструктора наследника, вновь применение списков инициализации. Правила размещения объектов классов-наследников в памяти. Множественное наследование, неоднозначности при нем, проблема ромбовидного наследования. Примеры разрешения неоднозначности с помощью приведений типов и оператора ::, комбинации всего этого с приватным наследованием, сдвиги указателей. Виртуальное наследование. Особенности комбинации виртуального и невиртуального наследования. Приведение типов между родителем и наследником: срезка при копировании, приведение указателей, приведение ссылок. Особенности static_cast, reinterpret_cast между родителями и наследниками (а также указателями или ссылками на них). Оператор dynamic_cast, его отличие от static_cast. Виртуальные функции, их общая идея и отличие от невиртуальных. Особенности размещения в памяти классов с виртуальными функциями, понятие полиморфизма. Полиморфные классы. Понятие о таблице виртуальных функций. Виртуальный деструктор и его предназначение. Абстрактные классы и “чисто виртуальные” (pure virtual) функции, их особенности. Чисто виртуальный деструктор. Ошибка “pure virtual function call” и ее возникновение. Ключевые слова override и final при наследовании, их предназначение. Механизм RTTI. Оператор typeid и динамическое определение типа объекта. Класс std::type_info. Проблема с вызовом виртуальных функций в конструкторах. Проблема с аргументами по умолчанию в виртуальных функциях. Empty base optimization, примеры.	2
6	1	Шаблоны Мотивировка и общая идея шаблонов. Шаблоны классов, шаблоны функций, синтаксис объявления, примеры использования, связь шаблонов и полиморфизма, статический полиморфизм. Специализации шаблонов, принцип “частное предпочтительнее общего” применительно к шаблонам. Частичные и полные специализации. Принцип “лучше точное соответствие, чем приведение типа”. Разница между специализацией и перегрузкой для шаблонных функций. Правила выбора компилятором кандидатов на специализацию и на перегрузку. Ключевое слово typedef, его предназначение. Шаблонные typedef’ы, использование слова using. Проблема с обращением к typedef’ам внутри шаблонных классов. Применение ключевого слова typename для решения этой проблемы. Примеры реализации простейших type_traits с помощью шаблонных структур и typedef’ов внутри них: remove_const, remove_reference. Правила вывода типов для шаблонов. Отбрасывание ссылок при выводе типа. Разбор случаев со ссылками и константами. Параметры шаблонов, не являющиеся типами (пример: массив константной длины). Параметры шаблонов, являющиеся шаблонами (“template template parameters”). Шаблоны с переменным количеством аргументов (variadic templates). Синтаксис использования. “Откусывание” шаблонных аргументов по одному. Оператор “sizeof…”. Функциональные классы и функциональные объекты (функторы), схема использования. Компараторы. Пример: компаратор в std::sort. Стандартные компараторы (std::less, std::greater, std::equal и т. п.), их реализация. Curiously Recurring Template Pattern (CRTP).	2
7	1	Исключения Общая идея, мотивировка использования исключений, оператор	2

		throw и конструкция try...catch. Примеры стандартных операторов, генерирующих исключения. Разница между исключениями и ошибками времени выполнения. Ошибки, не являющиеся исключениями, и исключения, не являющиеся ошибками. Правила ловли и повторного бросания исключений, приведения типов при ловле исключений. Ловля всех исключений. Правила выбора блока catch компилятором в случае, когда подходят разные блоки. Копирование при бросании и ловле исключений, исключения и наследование. Особенности перехвата исключений по значению и по ссылке, по ссылке на базовый класс. Спецификации исключений в старом стиле и их проблемы, unexpected exceptions (неожиданные исключения). спецификации исключений в стиле C++11, оператор и спецификатор noexcept. Условный noexcept. Исключения в конструкторах и проблема утечки памяти при исключениях. Исключения в деструкторах, функция uncaught_exception, функции terminate и set_terminate. Гарантии безопасности при исключениях: базовая и строгая. Function-try блоки, их особенности.	
8	1	Аллокаторы Placement new, его синтаксис, действие и отличие от обычного new. Разница между оператором new и функцией operator new. Более подробный разбор действия оператора new. Перегрузка new для отдельных классов. Перегрузка глобального new. Определение new с произвольными параметрами. То же самое для операторов delete и delete[]. Пример, когда компилятор неявно вызывает delete с нестандартными параметрами. Поведение delete для полиморфных объектов. nothrow оператор new, его синтаксис и особенности. Разбор поведения new в случае нехватки памяти. Функция new_handler, функции set_new_handler и get_new_handler. Понятие аллокатора. Класс std::allocator, его основные методы (allocate, deallocate, construct, destroy) и их примерная реализация. Особенности реализации конструкторов и оператора присваивания у стандартного аллокатора. Класс std::allocator_traits, его предназначение и основные методы. Пример нестандартного аллокатора (PoolAllocator), идея реализации его методов. Проблемы с конструктором копирования и оператором присваивания.	2
9	2	Контейнеры Общие сведения о контейнерах. Класс std::vector, его предназначение, идея реализации, основные методы и их алгоритмическая сложность. Поля класса std::vector. Реализация конструкторов, деструкторов, оператора присваивания с правильным обращением к аллокатору. Реализация метода push_back с правильным обращением к аллокатору. Реализация оператора [] для константных и неконстантных vector. Разница между [] и методом at(). Метод emplace_back, его реализация и отличие от push_back. Методы size(), resize(), capacity(), reserve() и shrink_to_fit(). Особенности работы с аллокатором при копировании вектора. Метод select_on_container_copy_construction. Вопросы на понимание: чему равно sizeof(v), где v - вектор, и что произойдет при вызове delete[] &(v[0])? Класс vector и его отличие от обычного vector, преимущества и недостатки. Внутренний класс BoolProxy. Особенности реализации оператора [] и оператора присваивания для vector по сравнению с обычным vector. Класс std::deque, основные методы и их алгоритмическая сложность. Разница между deque и vector: методы deque, отсутствующие у vector; методы vector, отсутствующие у deque. Адаптеры над контейнерами: std::stack, std::queue и std::priority_queue, их реализации. Компараторы в priority_queue и ее специфичные методы. Класс std::list, основные методы и их алгоритмическая сложность. Идея реализации list’a. Вставка и удаление из произвольного места. Специфичные для list’a методы: splice, sort, merge, reverse. Особенности работы list’a с аллокатором, метод rebind у аллокаторов. Класс std::forward_list, его отличия от обычного list. Ассоциативные контейнеры. Класс std::map, его предназначение, идея реализации. Описание шаблонных параметров класса map. Класс std::pair и функция std::make_pair. Основные	2

		методы map’а и их алгоритмическая сложность. Способы поиска в map’е. Способы вставки в map, особенности работы оператора []. Способы удаления из map’а. Классы std::set, std::multimap и std::multiset, их предназначение, отличия от std::map. Класс std::unordered_map, сходства и различия с обычным std::map. Основные методы и их алгоритмическая сложность. Особые для unordered_map шаблонные параметры: Hasher, Equal. Класс std::hash и его специализации. Особые для unordered_map методы: bucket_count, load_factor, rehash. Классы std::unordered_set, std::unordered_multimap, их идея, отличие от unordered_map.	
10	2	Итераторы Общая идея итераторов. Использование итераторов у стандартных контейнеров. Виды итераторов: input, output, forward, bidirectional, random access. Операции, поддерживаемые каждым видом итераторов. Виды итераторов у стандартных контейнеров. Константные и reverse-итераторы. Методы cbegin, cend, rbegin, rend, crbegin, crend у контейнеров. Реализация класса std::reverse_iterator, метод base. Класс std::iterator, его предназначение. Класс std::iterator_traits, его предназначение. Пример ситуации, когда он необходим (обращение к value_type). Функции std::distance и std::advance. Различие в поведении этих функций для разных видов итераторов, реализация этого различия. Стандартная библиотека алгоритмов, использование стандартных алгоритмов над контейнерами с итераторами. Итераторы для вставок: классы std::insert_iterator, std::back_insert_iterator, их предназначение, реализация. Функции std::inserter, std::back_inserter, их реализация. Правила инвалидации итераторов в стандартных контейнерах. Безопасные и небезопасные операции в контейнерах с точки зрения инвалидации итераторов.	2
11	2	Move-семантика и rvalue-ссылки Проблемы, приводящие к идее move-семантики: неэффективный swap, неэффективный push_back, emplace_back, construct. Применение магической функции std::move. Решение проблемы со swap. Понятие move-конструктора и move-assignment оператора, их реализация, генерация компилятором, “правило пяти”. Реализация функции std::move. Дилемма: что принять в качестве параметра? Понятие rvalue-ссылок. Особенности инициализации rvalue-ссылок, разрешенные и запрещенные присваивания между ссылками (включая проблемы с константностью). Решение проблемы с push_back. Понятия glvalue, lvalue, rvalue, prvalue и xvalue. Связи между ними. Примеры выражений, являющихся тем или иным видом value. Понятие универсальных ссылок, отличие их от rvalue-ссылок. Правила вывода типа шаблонов в случае универсальных ссылок, решение проблемы с типом параметра функции move. Правила сворачивания ссылок (reference collapsing). Проблема прямой передачи (perfect forwarding). Функция std::forward и ее применение. Решение проблем с emplace_back и construct. Реализация std::forward, ее обсуждение. Почему типы у принимаемого параметра и возвращаемого значения именно такие? Новая проблема с push_back: безопасность относительно исключений. Функция std::move_if_noexcept, решение проблемы с ее помощью. Return Value Optimization, условия ее возникновения. Примеры, когда RVO точно произойдет и когда может не произойти. Примеры, когда имеет и когда не имеет смысл писать return std::move(x) вместо return x. Copy Elision, примеры. Ссылочные квалификаторы. Решение проблемы с запретом оператора присваивания для rvalue у кастомных типов. Примеры типов, для которых прямая передача работает некорректно. Особенности поведения универсальных ссылок при разрешении перегрузки. Феномен “поглощения” универсальными ссылками обычных ссылок.	2
12	2	Умные указатели Идея и мотивировка умных указателей. Класс std::auto_ptr как первая неудачная попытка реализовать идею. Класс std::unique_ptr, его концепция. Особенности его конструкторов, деструктора и операторов присваивания. Методы * и ->. Специализация unique_ptr для массивов. Класс	2

		std::shared_ptr, его концепция. Идея реализации счетчика ссылок. Реализация методов. Потенциальная проблема, связанная с прямым вызовом new. Функции std::make_unique и std::make_shared как способ избежать прямого вызова new. Реализация этих функций. Функция std::allocate_shared, ее предназначение и реализация. Исправление реализации конструктора shared_ptr для этого. Проблема закольцованности указателей. Класс std::weak_ptr как решение этой проблемы. Реализация методов этого класса. Проблема с реализацией метода expired(), модификация класса shared_ptr для правильной работы с этим. Кастомные deleter'ы для умных указателей, схема использования. Более правильная реализация деструкторов unique_ptr и shared_ptr. Класс std::enable_shared_from_this, его предназначение и реализация. Еще одна модификация конструктора shared_ptr (проверка на наследника enable_shared_from_this).	
13	2	Вывод типов Проблема с длинными названиями типов. Проблема с возможными ошибками в написании точных названий типов. Ключевое слово auto как решение этих проблем. Правила вывода типов для auto. Особый случай с типом initializer_list. Особенности при auto&&. auto в качестве возвращаемого типа функции. Ключевое слово decltype, правила вывода типов для него. Особенности поведения decltype от выражений (случаи lvalue, xvalue, prvalue). Пример с decltype((x)). Особенности взятия decltype от тернарного оператора. Конструкция decltype(auto). Пример: обертка над обращением к контейнеру по индексу. Трюк для вывода названий выведенных типов на экран (намеренное провоцирование ошибок компиляции).	2
14	2	Шаблонное метапрограммирование и SFINAE Имитация if через шаблоны. Имитация for через шаблоны. Примеры: вычисление чисел Фибоначчи, проверка простоты числа, вывод чисел от 1 до 1000 с помощью шаблонов. Ключевое слово constexpr для функций и для переменных. Отличие constexpr от const. Требования к constexpr-функциям. type_traits. Структуры std::is_const (pointer, reference etc.), std::add_const (pointer, reference etc.), std::remove_const (pointer, reference, extent etc.), их реализации. Структуры std::is_same, std::true_type, std::false_type, std::conjunction, std::disjunction, std::conditional, std::rank, их реализации. Идиома SFINAE. Общая идея. Простейший пример: структура std::enable_if, ее реализация и применение. Структура std::is_class, ее реализация (без реализации std::is_union). Реализация метода std::allocator_traits::construct (проверка, определен ли у аллокатора метод construct). Проблема: невозможность написать T() для произвольного типа T. Функция std::declval, ее особенности. Решение предыдущей проблемы с ее помощью. Структуры std::is_constructible, std::is_convertible, std::is_copy_constructible, std::is_move_constructible etc. Их реализации. Реализация std::is_nothrow_move_constructible. Реализация std::move_if_noexcept через std::is_nothrow_move_constructible. Почему принимаемый и возвращаемый типы именно такие? Понятие неполных типов (incomplete types). Новая проблема с declval (что возвращать), решение проблемы с помощью rvalue-ссылки. Реализация std::is_base_of. Пример применения: проверка is_base_of<enable_shared_from_this> в конструкторе shared_ptr. Реализация std::common_type.	2
15	2	Функциональные объекты и лямбда-функции Лямбда-функции: мотивировка, простой пример (нестандартный компаратор в std::sort). Списки захвата в лямбда-функциях. Захват по ссылке и по значению. Особенности захвата this. Захват с присваиванием и перемещающий захват в C++14. Слово mutable применительно к лямбда-функциям. Явное указание возвращаемого значения. Захват по умолчанию и проблемы, которые он потенциально порождает. Пример с классом и методом getFunction() в нем. Обобщенные лямбда-функции в C++14. Применение auto и decltype в лямбда-функциях. Класс std::function, его предназначение и схема использования. Реализация std::function. Функция std::bind, ее предназначение и схема использования	2

		(без реализации). Placeholder’ы. Класс std::is_invocable. Класс std::invoke_result и функция std::invoke. (без реализации)	
16	2	Некоторые особые полезные типы Юнионы (union), их основная идея. Отличия от классов и структур. Инициализация полей юниона, активный член юниона и его изменение на другой. Класс std::variant, его предназначение и основные методы. Примерное описание реализации этого класса. Класс std::any, его предназначение и основные методы. Примерная реализация этого класса. Класс std::optional, его предназначение и основные методы. Неудачная попытка создать vector. Класс std::reference_wrapper для решения этой и других проблем. Примерная реализация этого класса.	2

5.2. Практические занятия, семинары

№ занятия	№ раздела	Наименование или краткое содержание практического занятия, семинара	Кол-во часов
1	1	Основы языка C++	2
2	1	Модификаторы типов, динамическая память и указатели	2
3	1	Определение классов в C++	2
4	1	Перегрузка операторов	2
5	1	Наследование	2
6	1	Шаблоны	2
7	1	Исключения	2
8	1	Аллокаторы. Контрольная работа	2
9	2	Контейнеры	2
10	2	Итераторы	2
11	2	Move-семантика и rvalue-ссылки	2
12	2	Умные указатели	2
13	2	Вывод типов	2
14	2	Шаблонное метапрограммирование и SFINAE	2
15	2	Функциональные объекты и лямбда-функци	2
16	2	Некоторые особые полезные типы	2

5.3. Лабораторные работы

Не предусмотрены

5.4. Самостоятельная работа студента

Выполнение СРС			
Подвид СРС	Список литературы (с указанием разделов, глав, страниц) / ссылка на ресурс	Семестр	Кол-во часов
Изучение теоретического материала для практических занятий и контрольных работ	ЭУМД,осн.лит.2.с.3-77,осн.лит.3,гл.4,9-11,13-19 или ПУМД,доп.лит.,гл.6-9	1	27
Подготовка к экзамену	ЭУМД,осн.лит.2. с.3-77, осн.лит.3, гл.9-11, 13-19 или ПУМД, доп.лит., гл.6-9	1	12,5

6. Фонд оценочных средств для проведения текущего контроля успеваемости, промежуточной аттестации

Контроль качества освоения образовательной программы осуществляется в соответствии с Положением о балльно-рейтинговой системе оценивания результатов учебной деятельности обучающихся.

6.1. Контрольные мероприятия (КМ)

№ КМ	Се-местр	Вид контроля	Название контрольного мероприятия	Вес	Макс. балл	Порядок начисления баллов	Учи-тыва-ется в ПА
1	1	Текущий контроль	Задание 1	1	10	Подключены необходимые заголовочные файлы и пространство имен std - 1 балл, иначе 0 баллов Корректно выполнен ввод или инициализация и вывод данных - 2 балла, иначе 0 баллов Корректно выполнено выделение и освобождение памяти для массива - 2 балла, иначе 0 баллов Выполнена обработка данных, в программе содержатся необходимое количество циклов и проверок условий - 3 балла, иначе 0 баллов Обработка данных выполнена без ошибок и эффективно - 2 балла, иначе 0 баллов	экзамен
2	1	Текущий контроль	Задание 2	2	20	отчет содержит необходимые пункты - 2 балла, иначе 0 баллов в main содержатся вызовы для всех открытых методов класса - 2 балла, иначе 0 баллов тесты исследуют все возможности класса и не требуют определения порядка действий от человека - 2 балла, иначе 0 баллов есть результаты выполнения main - 2 балла, иначе 0 баллов есть комментарии в интерфейсе класса ко всем полям, методам и функциям - 2 балла, иначе 0 баллов разделение реализации и интерфейса класса выполнено правильно (критерий: в интерфейсе могут быть только реализации методов из одного оператора) - 2 балла, иначе 0 баллов используется явное динамическое выделение и освобождение памяти - 2 балла, иначе 0 баллов конструктор и деструктор написаны правильно - 2 балла, иначе 0 баллов нет ошибок в реализации других методов - 2 балла, иначе 0 баллов реализация методов эффективна - 2 балла, иначе 0 баллов	экзамен
3	1	Текущий	Задание 3	2	20	отчет содержит необходимые пункты -	экзамен

		контроль				2 балла, иначе 0 баллов в main содержатся вызовы для всех открытых методов и других операций класса - 1 балл, иначе 0 баллов тесты исследуют все возможности класса и не требуют определения порядка действий от человека - 2 балла, иначе 0 баллов есть результаты выполнения main - 1 балл, иначе 0 баллов все указанные в задании операции были реализованы (возможно с ошибками) - 2 балла, иначе 0 баллов есть комментарии в интерфейсе класса ко всем полям, методам и функциям - 2 балла, иначе 0 баллов разделение реализации и интерфейса класса выполнено правильно (критерий: в интерфейсе могут быть только реализации методов из одного оператора) - 2 балла, иначе 0 баллов выполняется правило 7 (поведение перегруженных операций должно соответствовать поведению этих операций для стандартных типов данных) - 2 балла, иначе 0 баллов операции для ввода и вывода написаны правильно - 2 балла, иначе 0 баллов нет ошибок в реализации других операций - 2 балла, иначе 0 баллов реализация операций эффективна - 2 балла, иначе 0 баллов	
4	1	Текущий контроль	Задание 4	2	20	отчет содержит необходимые пункты - 2 балла, иначе 0 баллов в main содержатся вызовы для всех открытых методов и других операций класса, используется dynamic_cast (см. пример) - 2 балла, иначе 0 баллов тесты исследуют все возможности класса и не требуют определения порядка действий от человека - 2 балла, иначе 0 баллов есть результаты выполнения main - 2 балла, иначе 0 баллов все указанные в задании операции были реализованы (возможно с ошибками) - 2 балла, иначе 0 баллов есть комментарии в интерфейсе класса ко всем полям, методам и функциям - 2 балла, иначе 0 баллов нет ошибок в наследовании - 2 балла, иначе 0 баллов нет ошибок в реализации методов - 2 балла, иначе 0 баллов разделение реализации и интерфейса класса выполнено правильно	экзамен

						(критерий: в интерфейсе могут быть только реализации методов из одного оператора) - 2 балла, иначе 0 баллов	
5	1	Текущий контроль	Задание 5	2	20	отчет содержит необходимые пункты - 2 балла, иначе 0 баллов в main содержатся вызовы для всех открытых методов и других операций класса - 2 балла, иначе 0 баллов действия приводят к исключительной ситуации не менее 2 раз, имеется обработка исключений (см. пример) - 2 балла, иначе 0 баллов тесты исследуют все возможности класса и не требуют определения порядка действий от человека - 2 балла, иначе 0 баллов есть результаты выполнения main - 2 балла, иначе 0 баллов все указанные в задании операции были реализованы (возможно с ошибками) - 2 балла, иначе 0 баллов есть комментарии в интерфейсе класса ко всем полям, методам и функциям - 2 балла, иначе 0 баллов нет ошибок в реализации методов 4 балла, иначе 0 баллов разделение реализации и интерфейса класса выполнено правильно (критерий: в интерфейсе могут быть только реализации методов из одного оператора) - 2 балла, иначе 0 баллов	экзамен
6	1	Текущий контроль	Контрольная работа	2	20	Контрольная работа проводится на практическом занятии после прохождения темы "Шаблоны" КР содержит 4 небольших задания (от 4 до 10 строк) на 45 минут. Правильное решение задачи оценивается в 5 баллов, оценка снижается на 1 балл за каждую ошибку Максимальная оценка, итого 20 баллов Проверяется знание синтаксиса по темам * ввод-вывод в C++ * динамическое выделение памяти * определение классов и методов * объявление объектов и вызов методов * перегрузка операций * шаблоны функций	экзамен
7	1	Проме-жуточная аттестация	Экзаменационный билет	-	5	Критерии оценки Знает основные термины дисциплины (собеседование по билету) - 1 балл, иначе 0 баллов Правильный ответ на 1 вопрос билета -	экзамен

						2 балла, частичный ответ - 1 балл, иначе 0 баллов Правильный ответ на 2 вопрос билета - 2 балла, частичный ответ - 1 балл, иначе 0 баллов	
8	1	Бонус	Бонус	-	15	Критерии оценки 1) Решение задач на следующих олимпиадах по программированию: - личное первенство ЮУрГУ - квалификация чемпионата мира - ¼ финала чемпионата мира - ½ финала чемпионата мира (по 0,5 балла за решенную задачу, но не более 3 баллов за соревнование). 2) Работа у доски на практических занятиях Учитывается работа студента у доски, активность в обсуждении Самостоятельное решение задания у доски - 1 балл Решение задания у доски с частичными подсказками - 0,5 балла Решение под диктовку - 0 баллов Активность в обсуждении, нахождение ошибки в решении на доске - 0,2 балла 3) Посещение всех занятий Посещение всех занятий (пропуски только по уважительной причине) - 2 балла Посещение от 85% до 99% занятий - 1 балл Посещение менее 85% занятий - 0 баллов Итого в сумме 15 баллов максимум	экзамен

6.2. Процедура проведения, критерии оценивания

Вид промежуточной аттестации	Процедура проведения	Критерии оценивания
экзамен	Возможно определение рейтинга студента по дисциплине по результатам текущего контроля (три контрольных работы в течении семестра) в соответствии с п.2.6. Экзамен проводится в форме письменного ответа по билету и собеседования. После выдачи билета студенту предоставляется 1 час на подготовку ответа по теоретическим вопросам.	В соответствии с пп. 2.5, 2.6 Положения

6.3. Паспорт фонда оценочных средств

Компетенции	Результаты обучения	№ КМ							
		1	2	3	4	5	6	7	8
ПК-1	Знает: методы и средства разработки программного обеспечения на	+	+	+	+	+	+	+	+

	языке программирования С++								
ПК-1	Умеет: программировать на языке С++	+	+	+	+	+	+	+	+
ПК-1	Имеет практический опыт: проектирования, кодирования и отладки программного обеспечения, разрабатываемого на языке С++	+	+	+	+	+		+	+

Типовые контрольные задания по каждому мероприятию находятся в приложениях.

7. Учебно-методическое и информационное обеспечение дисциплины

Печатная учебно-методическая документация

а) основная литература:

Не предусмотрена

б) дополнительная литература:

1. Павловская, Т. А. С/С++. Программирование на языке высокого уровня [Текст] учебник для вузов по направлению "Информатика и вычисл. техника" Т. А. Павловская. - СПб. и др.: Питер, 2020. - 460 с. ил.

в) отечественные и зарубежные журналы по дисциплине, имеющиеся в библиотеке:

Не предусмотрены

г) методические указания для студентов по освоению дисциплины:

1. Методические указания для СРС по выполнению лабораторных работ

из них: учебно-методическое обеспечение самостоятельной работы студента:

1. Методические указания для СРС по выполнению лабораторных работ

Электронная учебно-методическая документация

№	Вид литературы	Наименование ресурса в электронной форме	Библиографическое описание
1	Дополнительная литература	Электронно-библиотечная система издательства Лань	Ашарина, И.В. Объектно-ориентированное программирование в С++: лекции и упражнения. [Электронный ресурс] — Электрон. дан. — М. : Горячая линия-Телеком, 2012. — 320 с. https://e.lanbook.com/book/5115
2	Основная литература	Электронный каталог ЮУрГУ	Демидов, А. К. Объектно-ориентированное программирование на С++ [Электронный ресурс] : учеб. пособие по направлениям 01.03.02 "Приклад. математика и информатика" и 01.03.04 "Приклад. математика" / А. К. Демидов. - Челябинск, 2017. http://www.lib.susu.ac.ru/ftd?base=SUSU_METHOD&key=000557669
3	Основная литература	Электронно-библиотечная система издательства Лань	Липман, С. Язык программирования С++. Полное руководство. [Электронный ресурс] / С. Липман, Ж. Лажойе. — Электрон. дан. — М. : ДМК Пресс, 2006. — 1105 с. https://e.lanbook.com/book/1216

Перечень используемого программного обеспечения:

1. -MinIDE (сборка из SciTE, MinGW C/C++, GDB)(бессрочно)
2. -LibreOffice(бессрочно)
3. -Codeblocks(бессрочно)
4. Microsoft-Visual Studio(бессрочно)

Перечень используемых профессиональных баз данных и информационных справочных систем:

Нет

8. Материально-техническое обеспечение дисциплины

Вид занятий	№ ауд.	Основное оборудование, стенды, макеты, компьютерная техника, предустановленное программное обеспечение, используемое для различных видов занятий
Лекции	332 (3б)	компьютер, проектор
Практические занятия и семинары	333 (3б)	компьютеры, компилятор C++, LibreOffice Write