Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Российский государственный университет им. А.Н. Косыгина
(Технологии. Дизайн. Искусство)»

Институт информационных технологий и цифровой трансформации
Кафедра информационных технологий

РАБОЧАЯ ПРОГРАММА УЧЕБНОЙ ДИСЦИПЛИНЫ

Программная инженерия и гибкие методологии разработки ПО

Уровень образования	бакалавриат
Направление подготовки	01.03.02 Прикладная математика и информатика
Профиль	Программирование и искусственный интеллект
Срок освоения образовательной программы по очной форме обучения	4 года
Форма обучения	очная

Рабочая программа учебной дисциплины «Программная инженерия и гибкие методологии разработки ПО» основной профессиональной образовательной программы высшего образования рассмотрена и одобрена на заседании кафедры, протокол № 9 от 11.04.2024 г.

Разработчик(и) рабочей программы «Программная инженерия и гибкие методологии разработки ПО»:

1. канд. техн. наук, доцент А.А. Семёнов
2. преподаватель М.Ю. Пивненко

Заведующий кафедрой: канд. техн. наук, доцент И.Б. Разин

1. ОБЩИЕ СВЕДЕНИЯ

Учебная дисциплина «Программная инженерия и гибкие методологии разработки ПО» изучается в шестом семестре.
Курсовая работа/Курсовой проект – не предусмотрены.

1.1. Форма промежуточной аттестации: экзамен.

При проведении промежуточной аттестации применяется Методика использования балльно-рейтинговой системы при реализации основных профессиональных образовательных программ высшего образования Института информационных технологий и цифровой трансформации, подписанная 08.04.2024 директором ИИТиЦТ Чикуновым И.М.

1.2. Место учебной дисциплины в структуре ОПОП

Учебная дисциплина «Программная инженерия и гибкие методологии разработки ПО» относится к части, формируемой участниками образовательных отношений.
Основой для освоения дисциплины являются результаты обучения по предшествующим дисциплинам и практикам:

- Программирование;
- Прикладное программирование;
- Алгоритмы и структуры данных.

Результаты обучения по учебной дисциплине используются при изучении следующих дисциплин:

- Технологии разработки мобильных приложений;
- Серверная веб-разработка.

2. ЦЕЛИ И ПЛАНИРУЕМЫЕ РЕЗУЛЬТАТЫ ОБУЧЕНИЯ ПО ДИСЦИПЛИНЕ

Целями изучения дисциплины «Программная инженерия и гибкие методологии разработки ПО» являются:

- изучение способов представления и структурирования информации о явлениях и процессах в окружающем мире применительно к своей профессиональной деятельности;
- освоение методов ориентирования и взаимодействия с ресурсами информационной среды, осуществления выбора различных моделей использования программных средств разработки для информационных и автоматизированных систем:
- изучение методов построения алгоритмов и основных этапов разработки и создания современных программных продуктов с учетом основных требований информационной безопасности и гибких методологий разработки ПО;
- формирование навыков научно-практического подхода к построению эффективных диалоговых интерфейсов, ориентированных на пользователя;
- изучение принципов, методов и средств решения стандартных задач современного объектно-ориентированного и визуального программирования;
- формирование у обучающихся компетенций, установленных образовательной программой в соответствии с ФГОС ВО по данной дисциплине.

Результатом обучения по учебной дисциплине является овладение обучающимися знаниями, умениями, навыками и опытом деятельности, характеризующими процесс формирования компетенций и обеспечивающими достижение планируемых результатов освоения учебной дисциплины.

2.1. Формируемые компетенции, индикаторы достижения компетенций, соотнесённые с планируемыми результатами обучения по дисциплине:

Код и наименование компетенции	Код и наименование индикатора достижения компетенции	Планируемые результаты обучения по дисциплине
ПК-2. Способен реализовывать проекты цифровой трансформации предприятий в самостоятельно выбранной предметной области, в том числе разрабатывать новые информационные и цифровые продукты путем применения существующих информационных и цифровых технологий, а также их адаптации под заданные условия, требования и ограничения	ИД-ПК-2.1 Определение принадлежности задачи профессиональной деятельности заданному классу и предметной области	– Формирует перечень задач в профессиональной деятельности для получения заданного результата (достижения заданной цели). – Самостоятельно использует методы и средства познания, обучения и самоконтроля для приобретения новых знаний и умений с использованием современных информационных технологий. – Оценивает предметную область с использованием сетевыми средствами для обмена данными в глобальной информационной сети Интернет. – Анализирует и обобщает информацию для правильной постановки цели и нахождения способов ее достижения.
	ИД-ПК-2.2 Выбор оптимального набора инструментальных средств и ИТ-методов решения профессиональной задачи в рамках предметной области	– Оценивает набор инструментальных средств решения профессиональных задач с учетом имеющихся ресурсов. – Оценивает качество ИТ-методов решения задач в соответствии с предметной областью. – Прогнозирует зависимость результата достижения цели от качества решения ИТ-задачи. – Самостоятельно использует типовые инструменты контроля решения ИТ-задач.
	ИД-ПК-2.3 Адаптация современных методов и алгоритмов под конкретные задачи выбранной предметной области	– Применяет методики использования программных средств для решения практических задач в информационных и автоматизированных системах. – Умеет разрабатывать современные эффективные интерфейсы «человек - электронно-вычислительная машина». – Анализирует навыки работы с программными средствами для управления информацией и коммуникации на основе базовых принципов современных информационных технологий. – Рационально оценивает и обосновывает принимаемые проектные решения для выбора и установки программных средств.

Код и наименование компетенции	Код и наименование индикатора достижения компетенции	Планируемые результаты обучения по дисциплине
		– Самостоятельно разрабатывает и составляет структурные схемы алгоритмов и реализует их на языках программирования высокого уровня.
	ИД-ПК-2.4 Использование ИТ-инструментов для решения задачи в выбранной предметной области	– Оценивает сущность и значение информационных технологий и инструментов для решения практических задач. – Проводит анализ основных методов, способов и средств получения, хранения, переработки, защиты и визуализации информации. – Самостоятельно осуществляет инсталляцию и практическую реализацию информационно-коммуникационных программ и компьютерных приложений. – Оценивает выполнение тестовых примеров для проверки их корректности и эффективности. – Анализирует методики использования программных средств для решения практических задач в информационных и автоматизированных системах.

3. СТРУКТУРА И СОДЕРЖАНИЕ УЧЕБНОЙ ДИСЦИПЛИНЫ

Общая трудоёмкость учебной дисциплины по учебному плану составляет:

по очной форме обучения –	6	**з.е.**	192	**час.**

3.1. Структура учебной дисциплины для обучающихся по видам занятий (очная форма обучения)

Структура и объем дисциплины			
Объем дисциплины по семестрам	**Общий объем работы обучающегося, час**	**Контактная аудиторная работа, час**	**Самостоятельная работа обучающегося, час**

			лекции, час	практические занятия, час	лабораторные занятия, час	практическая подготовка, час	курсовая работа/ курсовой проект	самостоятельная работа обучающегося, час	промежуточная аттестация, час
6 семестр	экзамен	192	34		34	6		86	32
Всего:		192	34		34	6		86	32

3.2. Структура учебной дисциплины для обучающихся по разделам и темам дисциплины: (очная форма обучения)

Планируемые (контролируемые) результаты освоения: код(ы) формируемой(ых) компетенции(й) и индикаторов достижения компетенций	Наименование разделов, тем; форма(ы) промежуточной аттестации	Виды учебной работы				Самостоятельная работа, час	Виды и формы контрольных мероприятий, обеспечивающие по совокупности текущий контроль успеваемости; формы промежуточного контроля успеваемости
		Контактная работа					
		Лекции, час	Практические занятия, час	Лабораторные работы/ индивидуальные занятия, час	Практическая подготовка, час		
	Шестой семестр						
ПК-2: ИД-ПК-2.1 ИД-ПК-2.2 ИД-ПК-2.3 ИД-ПК-2.4	**Раздел I. Программная инженерия**	4	x	4	x	12	Формы текущего контроля по разделам: 1. Выполнение лабораторных работ. 2. Контроль посещений. 3. Посещение профориентационных мероприятий. 4. Участие (достижения) в профессиональных конкурсах. 5. Научная и/или практическая работа.
	Лекция 1.1. Сущность и методы программной инженерии	2				x	
	Лекция 1.2. Процесс разработки ПО	2				x	
	Лабораторная работа № 1.1. Расчетные задачи			2		6	
	Лабораторная работа № 1.2. Ветвления и циклы			2		6	
ПК-2: ИД-ПК-2.1 ИД-ПК-2.2 ИД-ПК-2.3 ИД-ПК-2.4	**Раздел II. Визуальное моделирование программных систем**	4	x	4	1	12	
	Лекция 2.1. Визуальное моделирование систем	1				x	
	Лекция 2.2. Технология разработки ПО и средства автоматизации	2				x	
	Лекция 2.3. Визуальное моделирование ПО	1				x	
	Лабораторная работа № 2.1. Массивы и строки			1		4	
	Лабораторная работа № 2.2. Функции			2		4	
	Лабораторная работа № 2.3. Указатели и функции. Динамическое выделение памяти			1	1	4	
ПК-2: ИД-ПК-2.1 ИД-ПК-2.2 ИД-ПК-2.3 ИД-ПК-2.4	**Раздел III. Технологии программирования сложных систем**	4	x	4	1	12	
	Лекция 3.1. Технология программирования сложных систем	1				x	
	Лекция 3.2. Программная инженерия / Software Engineering (SE) программных продуктов	1				x	
	Лекция 3.3. Новые подходы к разработке изменяемых программных продуктов	1				x	
	Лекция 3.4. Модели и методы проектирования вариантов систем	1				x	

Планируемые (контролируемые) результаты освоения: код(ы) формируемой(ых) компетенции(й) и индикаторов достижения компетенций	Наименование разделов, тем; форма(ы) промежуточной аттестации	Виды учебной работы				Самостоятельная работа, час	Виды и формы контрольных мероприятий, обеспечивающие по совокупности текущий контроль успеваемости; формы промежуточного контроля успеваемости
		Контактная работа					
		Лекции, час	Практические занятия, час	Лабораторные работы/ индивидуальные занятия, час	Практическая подготовка, час		
	Лабораторная работа № 3.1. Потоковые классы и файлы. Текстовые и бинарные файлы			2		6	
	Лабораторная работа № 3.2. Массивы и функции			2	1	6	
ПК-2: ИД-ПК-2.1 ИД-ПК-2.2 ИД-ПК-2.3 ИД-ПК-2.4	**Раздел IV. Методы спецификации и верификации систем**	4	x	4	1	12	
	Лекция 4.1. Определения характеристик систем методами анализа и извлечения знаний	2					
	Лекция 4.2. Формальные методы спецификации, верификации и доказательства правильности систем	2					
	Лабораторная работа № 4.1. Графика. Структура.			2		6	
	Лабораторная работа № 4.2. Классы. Инкапсуляция			2	1	6	
ПК-2: ИД-ПК-2.1 ИД-ПК-2.2 ИД-ПК-2.3 ИД-ПК-2.4	**Раздел V. Теория и методы проектирования**	4	x	4	1	12	
	Лекция 5.1. Теория и методы проектирования моделей доменов и систем	2				x	
	Лекция 5.2. Методы экспертирования, тестирования и оценки качества ПС	2				x	
	Лабораторная работа № 5.1. Классы. Операторные функции. Реализация класса для перегрузки операций			2		6	
	Лабораторная работа № 5.2. Классы. Приведение типов.			2	1	6	
ПК-2: ИД-ПК-2.1 ИД-ПК-2.2 ИД-ПК-2.3 ИД-ПК-2.4	**Раздел VI. Принципы разработки программного обеспечения**	6	x	6	1	12	
	Лекция 6.1. Принципы разработки программных продуктов	6					
	Лабораторная работа № 6.1. Классы. Наследование			6	1	12	
ПК-2: ИД-ПК-2.1	**Раздел VII. Паттерны и методологии разработки программного обеспечения**	8	x	8	1	14	

Планируемые (контролируемые) результаты освоения: код(ы) формируемой(ых) компетенции(й) и индикаторов достижения компетенций	Наименование разделов, тем; форма(ы) промежуточной аттестации	Виды учебной работы				Самостоятельная работа, час	Виды и формы контрольных мероприятий, обеспечивающие по совокупности текущий контроль успеваемости; формы промежуточного контроля успеваемости
		Контактная работа					
		Лекции, час	Практические занятия, час	Лабораторные работы/ индивидуальные занятия, час	Практическая подготовка, час		
ИД-ПК-2.2 ИД-ПК-2.3 ИД-ПК-2.4	Лекция 7.1. Общие паттерны распределения обязанностей (GRASP)	4					
	Лекция 7.2. Паттерны проектирования GoF	4					
	Лабораторная работа № 7.1. Классы. Наследование при изначальной разработке программы			4		7	
	Лабораторная работа № 7.2. Классы. Отработка задач с указателями this			4	1	7	
	Экзамен	x	x	x	x	32	Компьютерный тест. Промежуточная аттестация производится в рамках балльно-рейтинговой системы. Оценка по дисциплине выставляется в соответствии с Системой оценивания результатов текущего контроля и промежуточной аттестации.
	ИТОГО за шестой семестр	**34**		**34**	**6**	**118**	**Экзамен**
	ИТОГО за весь период	**34**		**34**	**6**	**118**	

3.3. Краткое содержание учебной дисциплины

№ пп	Наименование раздела и темы дисциплины	Содержание раздела (темы)
Раздел I	**Программная инженерия.**	
Лекция 1.1	Сущность и методы программной инженерии	Сущность программной инженерии. Методы программной инженерии. Анализ требований, проектирование, разработка, тестирование, внедрение, поддержка и обслуживание, управление проектами, контроль качества, обеспечение безопасности, обучение и сертификация, управление рисками, реинжиниринг и модернизация, моделирование.
Лекция 1.2	Процесс разработки ПО	Проблемы разработки программного обеспечения. Модели процесса разработки программного обеспечения. Каскадная модель, инкрементная модель, спиральная модель, модель быстрой разработки приложений (RAD), экстремальное программирование (XP), гибкая методология разработки (Agile), DevOps, Scrum, Канбан.
Лабораторная работа № 1.1	Расчетные задачи	Применение расчетных операций для работы с целыми и вещественными типами данных. Реализация расчетных задач согласно вариантам заданий.
Лабораторная работа № 1.2	Ветвления и циклы	Циклические конструкции с условиями и оператором switch. Применение ветвлений и циклов для реализации расчетных задач согласно вариантам заданий.
Раздел II	**Визуальное моделирование программных систем**	
Лекция 2.1	Визуальное моделирование систем	Цели и значение моделирования. Принципы моделирования. Графические нотации моделирования
Лекция 2.2	Технология разработки ПО и средства автоматизации	Методологии разработки ПО. Характеристика и классификация CASE-средств (по функциональности, по типу поддержки, по уровню интеграции, по области применения). Технологии и инструментальные средства. Унифицированный процесс разработки.
Лекция 2.3	Визуальное моделирование ПО	Унифицированный язык моделирования UML (Unified Modeling Language). Визуальные модели и диаграммы программных систем. Виды диаграмм
Лабораторная работа № 2.1	Массивы и строки	Реализация операций с одномерными и двумерными массивами согласно вариантам заданий. Реализация операций со строками согласно вариантам заданий.
Лабораторная работа № 2.2	Функции	Применение функций при разработке приложений согласно вариантам заданий.
Лабораторная работа № 2.3	Указатели и функции. Динамическое выделение памяти	Применение указателей при передаче аргументов в функции на примере обработки одномерных и двумерных динамических массивов при разработке приложений согласно вариантам заданий.
Раздел III	**Технологии программирования сложных систем**	
Лекция 3.1	Технология программирования сложных	Технология программирования сложных систем. Технология модульного проектирования систем.

	систем	Сборочное программирование и перспективы развития.
Лекция 3.2	Программная инженерия / Software Engineering (SE) программных продуктов	Инженерия компьютерных систем. Области знаний ядра SWEBOK. Модели гибкой разработки ПО. Парадигмы программирования SE.
Лекция 3.3	Новые подходы к разработке изменяемых программных продуктов	Инженерия изготовления ПП в Product Line. Инженерия повторного использования КПИ/Reuse. Подходы к созданию вариантов ПП. Моделирование изменяемых систем по К. Чернецки. Применение инженерии SPLE в системе Grid. Фабрики программ и AppFab.
Лекция 3.4	Модели и методы проектирования вариантов систем	Определение вариантов КПИ в ПС и СПС. Базовые модели проектирования систем. Моделирование систем в языке UML. Аспекты управления вариабельностью ПС.
Лабораторная работа № 3.1	Потоковые классы и файлы. Текстовые и бинарные файлы	Работа с текстовыми и бинарными файлами при разработке консольных и визуальных приложений согласно вариантам заданий.
Лабораторная работа № 3.2	Массивы и функции	Обработка массивов посредством функций при разработке консольных и визуальных приложений согласно вариантам заданий.
Раздел IV	**Методы спецификации и верификации систем**	
Лекция 4.1	Определения характеристик систем методами анализа и извлечения знаний	Моделирование функциональных и нефункциональных характеристик программных продуктов. Модели представления знаний. Методы поиска и извлечения знаний.
Лекция 4.2	Формальные методы спецификации, верификации и доказательства правильности систем	Методы и языки формальной спецификации. Формальные методы спецификации моделей систем. Формальные методы доказательства программ. Подходы к верификации моделей систем и характеристик.
Лабораторная работа № 4.1	Графика. Структура.	Работа с графикой с применением структуры фигур. Построение многоугольников по точкам, сохранение их в текстовый файл, очистка холста, загрузка данных из файла и восстановление нарисованных многоугольников. Расчет периметра и площади нарисованных и загруженных из файла многоугольников.
Лабораторная работа № 4.2	Классы. Инкапсуляция	Разработать визуальное приложение, в котором необходимо создать класс, содержащий три поля типа int, предназначенные для хранения часов, минут и секунд. Один из конструкторов класса должен инициализировать поля нулевыми значениями, а другой конструктор – заданным набором значений. Необходимо изучить и повторить представленный иллюстрированный пример, демонстрирующий реализацию класса платной дороги с тарифом за проезд.
Раздел V	**Теория и методы проектирования**	
Лекция 5.1	Теория и методы проектирования моделей доменов и систем	Теория моделирования систем из объектов на уровнях. Операции над классами объектов. Формальные основы объектного анализа. Определение моделей ОМ (Operation Management), ПС (Production Scheduling), МХ (Manufacturing Execution). Теория моделирования ПС из компонентов. Связь компонентной и объектной моделей.

		Технология конфигурационной сборки. Реализация технологии сборки в ИТК.
Лекция 5.2	Методы экспертирования, тестирования и оценки качества ПС	Экспертирование компонентов и систем. Подходы к процессу тестирования ПС и СПС. Оценивание качества ПС и СПС.
Лабораторная работа № 5.1	Классы. Операторные функции. Реализация класса для перегрузки операций	Реализовать приложение с классом MyInt, демонстрирующего перегрузку арифметических операций и операций сравнения для объектов этого класса. В классе нужно реализовать нулевой конструктор, инициализирующий поля нулевыми значениями, и ненулевой, инициализирующий поля класса значениями.
Лабораторная работа № 5.2	Классы. Приведение типов.	Необходимо изучить и повторить проиллюстрированный пример, демонстрирующий реализацию следующей задачи. На основе типа char создайте класс Stroka. Перегрузите операцию приведения строки типа char к типу Stroka и наоборот. Напишите GUI- приложение для проверки этого класса.
Раздел VI	**Принципы разработки программного обеспечения**	
Лекция 6.1	Принципы разработки программных продуктов	Принципы SOLID. Принцип программирования KISS. Принцип программирования DRY. Принцип программирования YAGNI. Чистая архитектура.
Лабораторная работа № 6.1	Классы. Наследование	Реализация базового класса Float и производного класса FloatPr. В примере разрабатывается визуальное приложение (Windows VCL Application), в котором на основе стандартного типа float создается базовый класс Float, имеющий два конструктора, метод вывода на экран и метод для перегрузки арифметической операции +. Используя общее наследование, создается производный класс FloatPr, добавляющий возможность использования операций -, *, /. Далее идет проверка производного класса.
Раздел VII	**Паттерны и методологии разработки программного обеспечения**	
Лекция 7.1	Общие паттерны распределения обязанностей (GRASP)	Проектирование на основе обязанностей и GRASP. Паттерн Создатель (Creator). Паттерн Информационный эксперт (Information Expert). Паттерн Слабая связность (Low Coupling). Паттерн Контроллер (Controller). Паттерн Высокое зацепление (High Cohesion). Паттерн Полиморфизм (Polymorphism). Паттерн Чистая выдумка (Pure Fabrication). Паттерн Посредник (Indirection). Паттерн Устойчивость к изменениям (Protected Variations).
Лекция 7.2	Паттерны проектирования GoF	Порождающие паттерны. Структурные паттерны. Поведенческие паттерны.
Лабораторная работа № 7.1	Классы. Наследование при изначальной разработке программы	Требуется изучить и повторить проиллюстрированный пример, демонстрирующий применение технологии наследования при изначальном проектировании приложения. Используя класс Tovar, в примере создаётся два производных от него класса: 1) TovarProd, добавляющий возможность хранить информацию о сроке хранения и температуре хранения продуктовых товаров; 2) TovarProm, позволяющий хранить информацию в

		соответствии с полями базового класса. Созданное визуальное приложение (Windows VCL Application) должно позволять: вводить информацию либо о продуктовых товарах, либо о промышленных товарах; выводить общую стоимость товаров, имеющихся на складе. Поставленную задачу удобно решать с применением технологии наследования при изначальной разработке приложения.
Лабораторная работа № 7.2	Классы. Отработка задач с указателями this	В железнодорожных кассах хранится информация о свободных местах в поездах на ближайшую неделю в следующем виде: дата выезда, пункт назначения, время отправления, число свободных мест. Поступает запрос на резервирование m мест до города N на k-й день недели с временем отправления поезда не позднее t часов вечера. Вывести время отправления или сообщение о невозможности выполнить заказ в полном объеме. Программу написать с использованием класса и указателей this. Класс описать в отдельном файле. Должна быть возможность добавления в таблицу, сохранения в текстовый файл и загрузки из файла.

3.4. Организация самостоятельной работы обучающихся

Самостоятельная работа студента – обязательная часть образовательного процесса, направленная на развитие готовности к профессиональному и личностному самообразованию, на проектирование дальнейшего образовательного маршрута и профессиональной карьеры.

Самостоятельная работа обучающихся по дисциплине организована как совокупность аудиторных и внеаудиторных занятий и работ, обеспечивающих успешное освоение дисциплины.

Аудиторная самостоятельная работа обучающихся по дисциплине выполняется на учебных занятиях под руководством преподавателя и по его заданию. Аудиторная самостоятельная работа обучающихся входит в общий объем времени, отведенного учебным планом на аудиторную работу, и регламентируется расписанием учебных занятий.

Внеаудиторная самостоятельная работа обучающихся – планируемая учебная, научно-исследовательская, практическая работа обучающихся, выполняемая во внеаудиторное время по заданию и при методическом руководстве преподавателя, но без его непосредственного участия, расписанием учебных занятий не регламентируется.

Внеаудиторная самостоятельная работа обучающихся включает в себя:

– подготовку к лекциям, практическим занятиям, лабораторным работам и экзамену;

– изучение специальной рекомендованной литературы;

– изучение разделов/тем, не выносимых на лекции и практические занятия самостоятельно;

– подготовка к выполнению лабораторных работ;

– участие в рекомендованных контрольно-рейтинговых мероприятиях, в том числе профориентационных;

– подготовка к компьютерному тестированию на промежуточных аттестациях.

Самостоятельная работа обучающихся с участием преподавателя в форме иной контактной работы предусматривает групповую и (или) индивидуальную работу с обучающимися и включает в себя:

– проведение индивидуальных и групповых консультаций по отдельным темам/разделам дисциплины;

– проведение консультаций перед экзаменом, перед зачетом с оценкой;

– консультации по организации самостоятельного изучения отдельных разделов/тем, базовых понятий учебных дисциплин профильного/родственного бакалавриата, которые формировали ОПК и ПК, в целях обеспечения преемственности образования.

Перечень разделов/тем, полностью или частично отнесенных на самостоятельное изучение с последующим контролем:

№ пп	Наименование раздела /темы *дисциплины,* выносимые на самостоятельное изучение	Задания для самостоятельной работы	Виды и формы контрольных мероприятий (учитываются при проведении текущего контроля)	Трудоемкость, час
Раздел I	**Программная инженерия**			
Лабораторная работа № 1.1	Расчетные задачи	Изучение учебной, научной и технической литературы по теме лабораторной работы. Работа с материалами конспекта лекций. Анализ задания к лабораторной работе, выбор способов её выполнения.	Выполнение лабораторной работы.	6
Лабораторная работа № 1.2	Ветвления и циклы	Изучение учебной, научной и технической литературы по теме лабораторной работы. Работа с материалами конспекта лекций. Анализ задания к лабораторной работе, выбор способов её выполнения.	Выполнение лабораторной работы.	6
Раздел II	**Визуальное моделирование программных систем**			
Лабораторная работа № 2.1	Массивы и строки	Изучение научной и технической литературы, нормативных документов, стандартов языков программирования. Работа с материалами конспекта лекций. Анализ задания к лабораторной работе, выбор способов её выполнения. Осваивание методов объектно-ориентированного и визуального программирования. Изучение элементов системы разработки программ и операторов языка для выполнения задания лабораторной работы.	Выполнение лабораторной работы.	4
Лабораторная работа № 2.2	Функции	Изучение научной и технической литературы, нормативных документов, стандартов языков программирования. Работа с материалами конспекта лекций. Анализ задания к лабораторной работе, выбор способов её выполнения. Осваивание методов объектно-ориентированного и визуального программирования.	Выполнение лабораторной работы.	4

		Изучение элементов системы разработки программ и операторов языка для выполнения задания лабораторной работы.		
Лабораторная работа № 2.3	Указатели и функции. Динамическое выделение памяти	Изучение научной и технической литературы, нормативных документов, стандартов языков программирования. Работа с материалами конспекта лекций. Анализ задания к лабораторной работе, выбор способов её выполнения. Осваивание методов объектно-ориентированного и визуального программирования. Изучение элементов системы разработки программ и операторов языка для выполнения задания лабораторной работы.	Выполнение лабораторной работы.	4
Раздел III	**Технологии программирования сложных систем**			
Лабораторная работа № 3.1	Потоковые классы и файлы. Текстовые и бинарные файлы	Изучение научной и технической литературы, нормативных документов, стандартов языков программирования. Работа с материалами конспекта лекций. Анализ задания к лабораторной работе, выбор способов её выполнения. Осваивание методов объектно-ориентированного и визуального программирования. Изучение элементов системы разработки программ и операторов языка для выполнения задания лабораторной работы.	Выполнение лабораторной работы.	6
Лабораторная работа № 3.2	Массивы и функции	Изучение научной и технической литературы, нормативных документов, стандартов языков программирования. Работа с материалами конспекта лекций. Анализ задания к лабораторной работе, выбор способов её выполнения. Осваивание методов объектно-ориентированного и визуального программирования. Изучение элементов системы разработки программ и операторов языка для выполнения задания лабораторной работы.	Выполнение лабораторной работы.	6
Раздел IV	**Методы спецификации и верификации систем**			
Лабораторная работа № 4.1	Графика. Структура.	Изучение научной и технической литературы, нормативных документов, стандартов языков программирования. Работа с материалами конспекта лекций. Анализ задания к лабораторной работе, выбор способов её выполнения. Осваивание методов	Выполнение лабораторной работы.	6

		объектно-ориентированного и визуального программирования. Изучение элементов системы разработки программ и операторов языка для выполнения задания лабораторной работы.		
Лабораторная работа № 4.2	Классы. Инкапсуляция	Изучение научной и технической литературы, нормативных документов, стандартов языков программирования. Работа с материалами конспекта лекций. Анализ задания к лабораторной работе, выбор способов её выполнения. Осваивание методов объектно-ориентированного и визуального программирования. Изучение элементов системы разработки программ и операторов языка для выполнения задания лабораторной работы.	Выполнение лабораторной работы.	6
Раздел V	**Теория и методы проектирования**			
Лабораторная работа № 5.1	Классы. Операторные функции. Реализация класса для перегрузки операций	Изучение научной и технической литературы, нормативных документов, стандартов языков программирования. Работа с материалами конспекта лекций. Анализ задания к лабораторной работе, выбор способов её выполнения. Осваивание методов объектно-ориентированного и визуального программирования. Изучение элементов системы разработки программ и операторов языка для выполнения задания лабораторной работы.	Выполнение лабораторной работы.	6
Лабораторная работа № 5.2	Классы. Приведение типов.	Изучение научной и технической литературы, нормативных документов, стандартов языков программирования. Работа с материалами конспекта лекций. Анализ задания к лабораторной работе, выбор способов её выполнения. Осваивание методов объектно-ориентированного и визуального программирования. Изучение элементов системы разработки программ и операторов языка для выполнения задания лабораторной работы.	Выполнение лабораторной работы.	6
Раздел VI	**Принципы разработки программного обеспечения.**			
Лабораторная работа № 6.1	Классы. Наследование	Изучение научной и технической литературы, нормативных документов, стандартов языков программирования. Работа с материалами конспекта лекций. Анализ задания к лабораторной	Выполнение лабораторной работы.	12

		работе, выбор способов её выполнения. Осваивание методов объектно-ориентированного и визуального программирования. Изучение элементов системы разработки программ и операторов языка для выполнения задания лабораторной работы.		
Раздел VII	**Паттерны и методологии разработки программного обеспечения**			
Лабораторная работа № 7.1	Классы. Наследование при изначальной разработке программы	Изучение научной и технической литературы, нормативных документов, стандартов языков программирования. Работа с материалами конспекта лекций. Анализ задания к лабораторной работе, выбор способов её выполнения. Осваивание методов объектно-ориентированного и визуального программирования. Изучение элементов системы разработки программ и операторов языка для выполнения задания лабораторной работы.	Выполнение лабораторной работы.	7
Лабораторная работа № 7.2	Классы. Отработка задач с указателями this	Изучение научной и технической литературы, нормативных документов, стандартов языков программирования. Работа с материалами конспекта лекций. Анализ задания к лабораторной работе, выбор способов её выполнения. Осваивание методов объектно-ориентированного и визуального программирования. Изучение элементов системы разработки программ и операторов языка для выполнения задания лабораторной работы.	Выполнение лабораторной работы.	7

3.5. Применение электронного обучения, дистанционных образовательных технологий

При реализации программы учебной дисциплины возможно применение электронного обучения и дистанционных образовательных технологий.

Реализация программы учебной дисциплины с применением электронного обучения и дистанционных образовательных технологий регламентируется действующими локальными актами университета.

Применяются следующие разновидности реализации программы с использованием ЭО и ДОТ.

В электронную образовательную среду, по необходимости, могут быть перенесены отдельные виды учебной деятельности:

использование ЭО и ДОТ	**использование ЭО и ДОТ**	**объем, час**	**включение в учебный процесс**
смешанное обучение	лекции	34	в соответствии с расписанием учебных занятий
	лабораторные занятия	34	

4. РЕЗУЛЬТАТЫ ОБУЧЕНИЯ ПО ДИСЦИПЛИНЕ, КРИТЕРИИ ОЦЕНКИ УРОВНЯ СФОРМИРОВАННОСТИ КОМПЕТЕНЦИЙ, СИСТЕМА И ШКАЛА ОЦЕНИВАНИЯ

4.1. Соотнесение планируемых результатов обучения с уровнями сформированности компетенций.

Итоговое количество баллов в 100-балльной системе по результатам текущей и промежуточной аттестации определяется в соответствии с Методикой использования балльно-рейтинговой системы при реализации основных профессиональных образовательных программ высшего образования Института информационных технологий и цифровой трансформации.

Уровни сформированности компетенции(-й)	**Итоговое количество баллов в 100-балльной системе по результатам текущей и промежуточной аттестации**	**Оценка в пятибалльной системе по результатам текущей и промежуточной аттестации**	**Показатели уровня сформированности**		
			универсальной(-ых) компетенции(-й)	**общепрофессиональной(-ых) компетенций**	**профессиональной(-ых) компетенции(-й)**
					ПК-2: ИД-ПК-2.1 ИД-ПК-2.2 ИД-ПК-2.3 ИД-ПК-2.4
высокий	85 – 100	отлично/ зачтено (отлично)/ зачтено			Обучающийся: – исчерпывающе и логически стройно излагает учебный материал, умеет связывать теорию с практикой, справляется с решением задач профессиональной направленности высокого уровня сложности, правильно обосновывает принятые решения по использованию информационных технологий; – способен уверенно использовать современные системы разработки прикладных программ с эффективными графическими интерфейсами и системы коммуникации в сети

					Internet; – показывает творческие способности в понимании и практическом использовании языков высокого уровня, использовании визуальных компонентов разработки приложений графических интерфейсов; – дополняет теоретическую информацию сведениями, самостоятельно полученными из источников научно-технической информации; – способен провести целостный анализ среды разработки современных программ на основе объектно-ориентированного и визуального программирования; – свободно ориентируется в учебной и профессиональной литературе; – дает развернутые, исчерпывающие, профессионально грамотные ответы на вопросы, в том числе, дополнительные.
повышенный	70 – 84	хорошо/ зачтено (хорошо)/ зачтено			Обучающийся: – достаточно подробно, грамотно и по существу излагает изученный материал, приводит и раскрывает в тезисной форме основные понятия информационных технологий; – анализирует современные

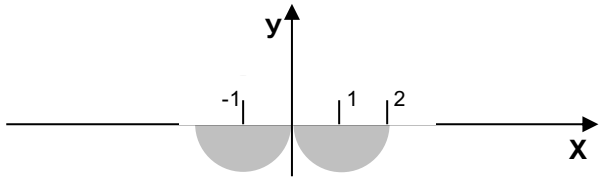
					технология программирования с незначительными пробелами; – способен использовать только основные функциональные возможности систем разработки программ и систем коммуникации в сети Internet; – способен провести анализ основных элементов разработки современных программ на основе объектно-ориентированного и визуального программирования; – допускает единичные негрубые ошибки; – достаточно хорошо ориентируется в учебной и профессиональной литературе; – ответ отражает знание теоретического и практического материала, не допуская существенных неточностей.
базовый	55 – 69	удовлетворительно/ зачтено (удовлетворительно)/ зачтено			Обучающийся: – демонстрирует теоретические знания основного учебного материала дисциплины в объеме, необходимом для дальнейшего освоения ОПОП; – с неточностями излагает принципы и методы разработки современных программ на основе объектно-ориентированного и визуального программирования; – способен использовать отдельны элементы визуальной

					разработки прикладных программ; – анализирует современные технологии программирования с неточностями и ошибками; – демонстрирует фрагментарные знания основной учебной литературы по дисциплине; – ответ отражает знания на базовом уровне теоретического и практического материала в объеме, необходимом для дальнейшей учебы и предстоящей работы по профилю обучения.
низкий	0 – 54	неудовлетворительно/ не зачтено	Обучающийся: – демонстрирует фрагментарные знания теоретического и практического материал, допускает грубые ошибки при его изложении на занятиях и в ходе промежуточной аттестации; – испытывает серьёзные затруднения в применении теоретических положений при решении практических задач профессиональной направленности стандартного уровня сложности, не владеет необходимыми для этого навыками и приёмами; – не способен проанализировать учебно-методическую, техническую и научную литературу; – не владеет основными принципами и навыками работы в современных средах разработки прикладных программ, не умеет пользоваться системами коммуникации (Internet); – выполняет задания только по образцу и под руководством преподавателя; – ответ отражает отсутствие знаний на базовом уровне теоретического и практического материала в объеме, необходимом для дальнейшей учебы.		

5. ОЦЕНОЧНЫЕ СРЕДСТВА ДЛЯ ТЕКУЩЕГО КОНТРОЛЯ УСПЕВАЕМОСТИ И ПРОМЕЖУТОЧНОЙ АТТЕСТАЦИИ, ВКЛЮЧАЯ САМОСТОЯТЕЛЬНУЮ РАБОТУ ОБУЧАЮЩИХСЯ

При проведении контроля самостоятельной работы обучающихся, текущего контроля и промежуточной аттестации по учебной дисциплине «Программная инженерия и гибкие методологии разработки ПО» проверяется уровень сформированности у обучающихся компетенций и запланированных результатов обучения по дисциплине, указанных в разделе 2 настоящей программы.

5.1. Формы текущего контроля успеваемости, примеры типовых заданий:

№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция
1	Выполнение лабораторной работы. Лабораторная работа № 1.1	Расчетные задачи. **Вариант 1.** Подсчитать k-количество цифр в десятичной записи целого неотрицательного числа n. **Вариант 2.** Дано n вещественных чисел. Вычислить разность между максимальным и минимальным из них. **Вариант 3.** Дана непустая последовательность различных натуральных чисел, за которой следует 0. Определить порядковый номер наименьшего из них. **Вариант 4.** Даны целое n>0 и последовательность из n вещественных чисел, среди которых есть хотя бы одно отрицательное число. Найти величину наибольшего среди отрицательных чисел этой последовательности. **Вариант 5.** Дано n вещественных чисел. Определить, образуют ли они возрастающую последовательность. …	ПК-2: ИД-ПК-2.1 ИД-ПК-2.2 ИД-ПК-2.3 ИД-ПК-2.4
	Выполнение лабораторной работы. Лабораторная работа № 1.2	Ветвления и циклы. Часть 1. Единое задание для закрепления пройденного материала. Приложения реализовать в консольном и в визуальном исполнении. 1) Дана заштрихованная область (см. рис.). Написать программу, которая позволяет вводить с клавиатуры точку с координатами (x; y) и определять попадает ли данная точка в область или нет.  2) Написать программу, которая выводит в виде таблицы значения функции y=*f(x)* на интервале $x \in [1;10]$ с шагом $dX = 1$: $\begin{cases} x^2 + 4x - 10, & \text{если } x=5 \text{ или } x=7 \end{cases}$	ПК-2: ИД-ПК-2.1 ИД-ПК-2.2 ИД-ПК-2.3 ИД-ПК-2.4

№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция

$$f(x) = \begin{cases} \sqrt[2]{4x} + \sqrt[3]{16x}, & \text{если } x=4 \\ x^3 - 2, & \text{в остальных случаях} \end{cases}$$

Необходимо реализовать программу в двух версиях. В одной версии следует применить циклическую конструкцию с условиями, а в другой версии – циклическую конструкцию со switch.

Часть 2. Индивидуальное задание по вариантам. Приложение реализовать в консольном и в визуальном исполнении.

Вариант 1. Вычислить значение функции

$$f(X) = \begin{bmatrix} X^2 + 12*X, \text{если } X = 3 \\ 3*X^{3.5} + 0.78*X, \text{если } X = 13 \\ 12*X + 35, \text{в другом варианте } X \end{bmatrix}'' \quad X \in [0,32], \Delta X = 1;$$

Написать два варианта программы - с использованием оператора if и с использованием оператора switch. Результат представить в виде таблицы.

Вариант 2. Вычислить значение функции

$$f(X) = \begin{bmatrix} (X+12)^{3+x} + X, \text{ если } X = 6 \\ 3*X^3 + 0.78*X, \text{ если } X = 2 \\ 0.56*X^{X/24}, \text{ в другом варианте } X \end{bmatrix} X \in [0,10], \Delta X = 2;$$

Написать два варианта программы - с использованием оператора if и с использованием оператора switch. Результат представить в виде таблицы.

Вариант 3. Вычислить значение функции

№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция
		$$f(X)=\begin{bmatrix}(25+X)/(X^2+24), \text{ если } X=5 \text{ и } X=3\\ 3*X^{3.5}+0.78*X, \text{ если } X=10\\ X^{X*2}, \text{ в другом варианте } X\end{bmatrix} X\in[-1,20], \Delta X=1;$$ Написать два варианта программы - с использованием оператора if и с использованием оператора switch. Результат представить в виде таблицы. **Вариант 4.** Вычислить значение функции $$f(X)=\begin{bmatrix}(25+X)/(23*X+24), \text{ если } X=3 \text{ и } X=0\\ X^{3.5}+0.78*\sqrt[3]{2*X}, \text{ если } X=10\\ X^5, \text{в другом варианте } X\end{bmatrix} X\in[0,20], \Delta X=1;$$ Написать два варианта программы - с использованием оператора if и с использованием оператора switch. Результат представить в виде таблицы. **Вариант 5.** Вычислить значение функции $$f(X)=\begin{bmatrix}\sqrt[5]{(X+25)}, \text{ если } X=3 \text{ и } X=0\\ \sqrt{X}+0.78*\sqrt[3]{2*X}, \text{если } X=10\\ X^X, \text{в другом варианте } X\end{bmatrix} X\in[-1,20], \Delta X=1;$$ Написать два варианта программы - с использованием оператора if и с использованием оператора switch. Результат представить в виде таблицы. …	

№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция
	Выполнение лабораторной работы. Лабораторная работа № 2.1	Массивы. Часть 1. Единое задание для закрепления пройденного материала. Приложения реализовать в консольном исполнении. 1) Написать программу, которая позволяет вводить одномерный целочисленный массив из 9 элементов и вычислять: а) сумму положительных элементов массива с чётными номерами; б) количество элементов массива, равных нулю; в) вещественный результат деления максимального элемента массива на минимальный. В случае возникновения исключительных ситуаций программа должна выводить соответствующие сообщения. 2) Написать программу, которая позволяет вводить целочисленную матрицу 3×4, выводить её в наглядном формате и определять номер столбца, сумма элементов которого минимальна. Программа должна позволять производить вычисления столько раз, сколько угодно пользователю. Часть 2. Индивидуальное задание по вариантам. Приложение реализовать в консольном исполнении. **Вариант 1.** Дан массив целых чисел. Написать программу для сортировки массива по возрастанию. **Вариант 2.** Дан массив целых чисел. Написать программу для определения максимальной и минимальной суммы двух соседних элементов массива. **Вариант 3.** Дан массив целых чисел. Написать программу для определения максимальной и минимальной разницы между двумя соседними элементами массива. **Вариант 4.** Дан массив вещественных чисел. Написать программу для сортировки массива по возрастанию целой части его элементов. **Вариант 5.** Дан массив целых чисел. Написать программу для сортировки заданного массива по убыванию суммы цифр элементов. … Строки. Часть 1. Единое задание для закрепления пройденного материала. Приложения реализовать в консольном и в визуальном исполнении. 1) Написать программу, которая позволяет вводить с клавиатуры строку, содержащую более одного слова, и определять: а) количество символов введённой строки; б) содержит ли строка слова одинаковой длины; в) количество запятых.	ПК-2: ИД-ПК-2.1 ИД-ПК-2.2 ИД-ПК-2.3 ИД-ПК-2.4

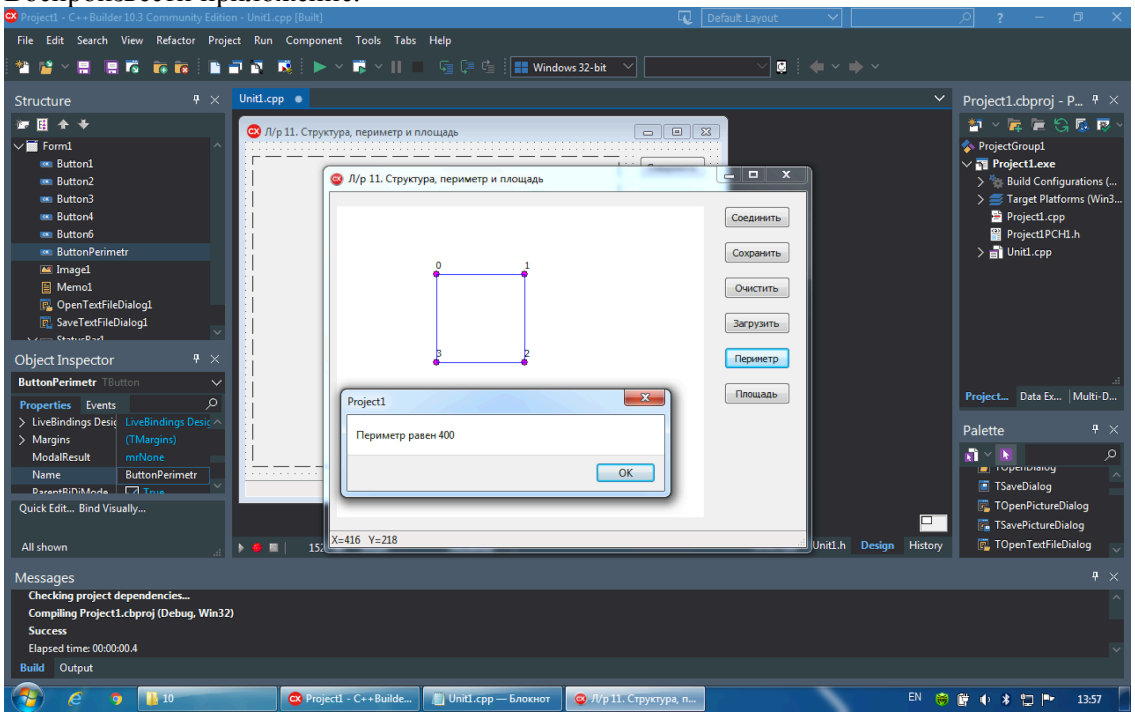
№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция
		Необходимо учесть, что при вводе строки, могут использоваться не только пробелы, но и знаки пунктуации. 2) Напишите программу, которая позволяет вводить с клавиатуры строку, содержащую более одного слова, и определять количество слов, заканчивающихся на заданную пользователем букву. Необходимо учесть, что при вводе строки, могут использоваться не только пробелы, но и знаки пунктуации Часть 2. Индивидуальное задание по вариантам. Приложение реализовать в консольном и/или в визуальном исполнении. **Вариант 1.** В заданном тексте удалить слова, начинающиеся с заданной буквы. При написании программы нельзя пользоваться стандартными функциями обработки строк. **Вариант 2.** Вывести на экран заданный текст, удалив из него лишние пробелы, т.е. из нескольких подряд идущих пробелов оставить только один. При написании программы нельзя пользоваться стандартными функциями обработки строк. **Вариант 3.** Задан текст, заканчивающийся точкой. Вывести на экран сначала все цифры, входящие в него, а затем все остальные литеры. При написании программы нельзя пользоваться стандартными функциями обработки строк. **Вариант 4.** Определить, сколько различных литер входит в заданный текст, заканчивающийся точкой. При написании программы нельзя пользоваться стандартными функциями обработки строк. **Вариант 5.** Заданный текст вывести на экран по строкам, понимая под строкой либо очередные 60 символов, если среди них нет запятой, либо текст до запятой включительно. При написании программы нельзя пользоваться стандартными функциями обработки строк. …	
	Выполнение лабораторной работы. Лабораторная работа № 2.2	Функции. Часть 1. Единое задание для закрепления пройденного материала. Приложения реализовать в визуальном исполнении. 1) Написать функцию, которая возвращает возведённое в степень число, и программу, использующую эту функцию для возведения введённого пользователем числа в заданную им степень. 2) Написать программу, позволяющую сортировать введенный пользователем	ПК-2: ИД-ПК-2.1 ИД-ПК-2.2 ИД-ПК-2.3 ИД-ПК-2.4

№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция
		целочисленный одномерный массив по убыванию или по возрастанию (в зависимости от выбора пользователя). Сортировку по убыванию необходимо реализовать методом пузырька, а сортировку по возрастанию – методом выбора. Указанные виды сортировок нужно оформить в виде отдельных функций. 3) Напишите функцию, которая переворачивает строку (массив типа char). Используйте цикл for, который меняет местами первый и последний символы, затем следующие и т.д. Строка должна передаваться в функцию как аргумент. Напишите программу, которая должна принимать от пользователя строку (содержащую более одного слова), вызывать функцию, а затем выводить полученный результат. 4) Напишите функцию вычисления корней квадратного уравнения. Программа должна принимать от пользователя коэффициенты, вызывать функцию, а затем выводить полученный результат. Параметрами функции должны быть коэффициенты и корни уравнения. Такие аргументы как корни уравнения необходимо передавать по ссылке. Значение, возвращаемое функцией, должно передавать в вызывающую программу информацию о наличии корней: 2 – два разных корня; 1 – корни одинаковые; 0 – уравнение не имеет решения. Кроме того, функция должна проверять корректность исходных данных. Если исходные данные неверные, то функция должна возвращать -1. Часть 2. Индивидуальное задание по вариантам. Приложение реализовать в консольном и/или в визуальном исполнении. **Вариант 1.** Дана строка. Написать функцию для определения, входит ли в строку заданное слово, или нет. При написании программы необходимо использовать стандартные функции обработки строк. **Вариант 2.** Дана строка символов. Написать функцию для определения одинаковы или нет второе и последнее слово в этой строке. При написании программы необходимо использовать стандартные функции обработки строк. **Вариант 3.** Дана строка символов. Написать функцию для определения слов, которые начинаются и заканчиваются на одинаковые буквы. При написании программы необходимо использовать стандартные функции обработки строк. **Вариант 4.** Дана строка символов. Написать функцию для определения, входит ли в эту	

№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция
		строку заданное слово. Если входит, то его нужно удалить. При написании программы необходимо использовать стандартные функции обработки строк. **Вариант 5.** Дана строка символов. Написать функцию для формирования строки с обратным порядком слов. При написании программы необходимо использовать стандартные функции обработки строк. …	
	Выполнение лабораторной работы. Лабораторная работа № 2.3	Указатели и функции. Динамическое выделение памяти. Часть 1. Единое задание для закрепления пройденного материала. Приложения реализовать в консольном исполнении. 1) Напишите программу, которая позволяет вводить динамический одномерный целочисленный массив и посредством вызова функции определять минимальную сумму между двумя соседними элементами массива. При передаче массива в функцию используйте указатели, а размерность передавайте по ссылке. Применение глобальных переменных категорически запрещено. 2) Напишите программу, позволяющую вводить динамический многомерный целочисленный массив и посредством функций осуществлять: 1) вывод введённого массива на экран в наглядном формате; 2) вычисление суммы элементов в тех строках, которые содержат хотя бы один отрицательный элемент; 3) определение минимального из чисел, встречающихся в заданной матрице более одного раза. При передаче аргументов в функции старайтесь использовать указатели. Применение глобальных переменных категорически запрещено Часть 2. Индивидуальное задание по вариантам. Приложение реализовать в консольном исполнении. При передаче массива в функцию используйте указатели, а размерность передавайте по ссылке. Применение глобальных переменных категорически запрещено. **Вариант 1**. Пусть пользователь вводит динамическую матрицу размером NxM. Написать функцию для вычисления суммы диагональных элементов. **Вариант 2.** Пусть пользователь вводит динамическую матрицу размером NxM. Написать функцию для вычисления максимальных и минимальных элементов в столбцах. **Вариант 3.** Пусть пользователь вводит динамическую матрицу размером NxM. Написать функцию для вычисления максимальной и минимальной суммы элементов в строках. **Вариант 4.** Пусть пользователь вводит динамическую матрицу размером NxM. Написать	ПК-2: ИД-ПК-2.1 ИД-ПК-2.2 ИД-ПК-2.3 ИД-ПК-2.4

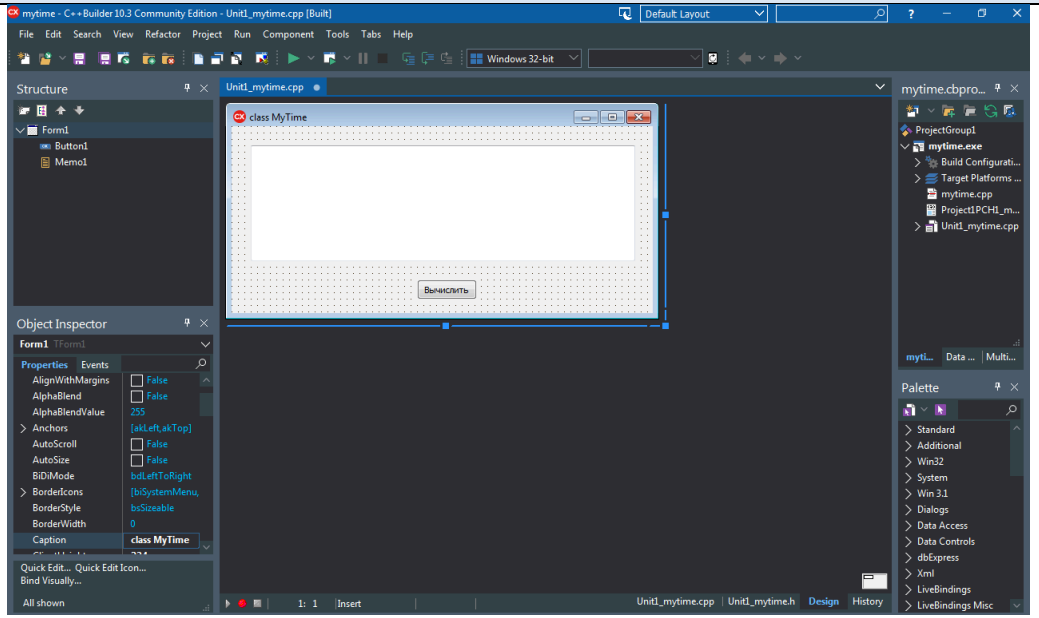
№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция
		функцию для сортировки строк матрицы по убыванию. **Вариант 5.** Пусть пользователь вводит динамическую матрицу размером NxM. С помощью функции определить максимальный элемент. …	
	Выполнение лабораторной работы. Лабораторная работа № 3.1	Потоковые классы и файлы. Текстовые и бинарные файлы. Часть 1. Единое задание для закрепления пройденного материала. Приложение 1 реализовать в консольном и в визуальном исполнении, приложение 2 – в консольном исполнении. 1) Напишите программу, которая позволяет: а) выводить на экран содержимое указанного текстового файла; б) записывать в другой текстовый файл только те предложения, которые содержат введённое с клавиатуры слово. Если указанное слово отсутствует в тексте, программа должна выводить сообщение и не должна создавать выходной файл. По возможности старайтесь использовать динамическое выделение памяти. 2) Напишите программу, которая позволяет создавать список данных железнодорожных поездов и хранить его в бинарном файле (например, …\railwaybase.dat). Каждая запись бинарного файла содержит: номер поезда (sizeof(int) байт); пункт отправления-прибытия (30 байт); количество проданных билетов (sizeof(int) байт); цена одного билета (sizeof(float) байт). Программа должна позволять: а) добавлять в двоичный файл записи, причём столько раз, сколько угодно пользователю; б) показывать все записи в наглядном формате, причём стоимость одного билета должна выводиться с двумя знаками после запятой; в) подсчитывать выручку указанного номера поезда. Пользователь должен иметь возможность в любой момент выбрать необходимое ему действие с помощью меню. Меню реализуйте с помощью функции, возвращающей значение. Предусмотрите обработку исключительных ситуаций. Часть 2. Индивидуальное задание по вариантам. Приложение реализовать в консольном исполнении. **Вариант 1.** Дан текстовый файл, состоящий из 5 строк. Написать функцию для	ПК-2: ИД-ПК-2.1 ИД-ПК-2.2 ИД-ПК-2.3 ИД-ПК-2.4

№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция
		сравнения первого слова второй строки и последнего слова пятой строки. **Вариант 2.** Дан текстовый файл, состоящий из некоторого количества строк. Написать функцию для сравнения первой и последней строки. **Вариант 3.** Дан текстовый файл, состоящий из некоторого количества строк. Написать функцию для создания нового файла с обратным порядком строк. **Вариант 4.** Дан текстовый файл, состоящий из некоторого количества строк. Написать функцию для создания нового файла, содержащего вторые слова строк первого файла. **Вариант 5.** Даны два текстовых файла, состоящие из некоторого количества строк. Написать функцию для сравнения этих файлов. …	
	Выполнение лабораторной работы. Лабораторная работа № 3.2	Массивы (векторы) и функции. **Вариант 1.** Преобразовать одномерный целочисленный массив таким образом, чтобы сначала располагались все положительные элементы, а потом – все отрицательные (элементы, равные 0, считать положительными). **Вариант 2.** Сжать одномерный вещественный массив, удалив из него все элементы, модуль которых не превышает 1. Освободившиеся в конце массива элементы заполнить нулями. **Вариант 3.** Сжать одномерный вещественный массив, удалив из него все элементы, модуль которых находится в интервале [a,b]. Освободившиеся в конце массива элементы заполнить нулями. **Вариант 4.** Преобразовать одномерный вещественный массив таким образом, чтобы сначала располагались все элементы, равные нулю, а потом – все остальные. **Вариант 5.** Преобразовать одномерный целочисленный массив таким образом, чтобы в первой его половине располагались элементы, стоявшие в нечетных позициях, а во второй половине – элементы, стоявшие в четных позициях. …	ПК-2: ИД-ПК-2.1 ИД-ПК-2.2 ИД-ПК-2.3 ИД-ПК-2.4
	Выполнение лабораторной работы. Лабораторная работа	Часть 1. Единое задание для закрепления пройденного материала. Работа с графикой с применением структуры фигур: struct Koordinata {	ПК-2: ИД-ПК-2.1 ИД-ПК-2.2

№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция
	№ 4.1	int x; int y; }; Построение многоугольников по точкам, сохранение их в текстовый файл, очистка холста, загрузка данных из файла и восстановление нарисованных многоугольников. Расчет периметра и площади нарисованных и загруженных из файла многоугольников. Воспроизвести приложение.  Часть 2. Индивидуальное задание по вариантам. Приложение реализовать в консольном и/или в визуальном исполнении. **Вариант 1**. Нарисовать график функции $f(X)= 2*X^3 + 3*X^2$. **Вариант 2**. Нарисовать график функции $f(X)= X^2 + 25*X$.	ИД-ПК-2.3 ИД-ПК-2.4

№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция
		Вариант 3. Нарисовать график функции f(X)= $\sin(X)+\sqrt{\lvert X\rvert}*\cos(23*X)$. **Вариант 4**. Нарисовать график функции f(X)=sin(X). **Вариант 5**. Нарисовать график функции f(X)=cos(X). …	
	Выполнение лабораторной работы. Лабораторная работа № 4.2	Классы. Инкапсуляция. Реализация класса MyTime. Лабораторная работа посвящена изучению классов в ООП. 1) Изучить и повторить самостоятельно иллюстрированный пример (см. ниже), демонстрирующий реализацию следующей задачи. Разработать визуальное приложение, в котором необходимо создать класс с именем MyTime, содержащий три поля типа int, предназначенные для хранения часов, минут и секунд. Один из конструкторов класса должен инициализировать поля нулевыми значениями, а другой конструктор – заданным набором значений. Создайте метод класса, который будет выводить значения полей на экран в формате 23:59:59, и метод, складывающий значения двух объектов типа MyTime, передаваемых в качестве аргументов. В обработчике события ButtonClick следует создать два инициализированных объекта и один неинициализированный объект, затем сложить два инициализированных значения, а результат присвоить третьему объекту и вывести его значение на экран (например, 13:23:50 + 10:52:50 = 0:16:40). Иллюстрированный пример разработки приложения Интерфейс приложения	ПК-2: ИД-ПК-2.1 ИД-ПК-2.2 ИД-ПК-2.3 ИД-ПК-2.4

№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция



Исходный код программы

.h-файл:

```
//---------------------------------------------------------------------------
#ifndef Unit1_mytimeH
#define Unit1_mytimeH
//---------------------------------------------------------------------------
#include <System.Classes.hpp>
#include <Vcl.Controls.hpp>
#include <Vcl.StdCtrls.hpp>
#include <Vcl.Forms.hpp>
//---------------------------------------------------------------------------
class TForm1 : public TForm
```

№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция

		{
		__published: // IDE-managed Components
		    TMemo *Memo1;
		    TButton *Button1;
		    void __fastcall Button1Click(TObject *Sender);
		private: // User declarations
		public: // User declarations
		    __fastcall TForm1(TComponent* Owner);
		};
		
		class MyTime {
		 private:
		    int chas;
		    int min;
		    int sec;
		 public:
		    MyTime() { chas=0; min=0; sec=0; }
		    MyTime(int ch, int m, int s) {
		      chas=ch; min=m; sec=s;
		    }
		    void show();
		    void summa(MyTime t1, MyTime t2);
		};
		//---
		extern PACKAGE TForm1 *Form1;
		//---
		#endif
		
		.cpp-файл:
		
		//---
		#include <vcl.h>
		#pragma hdrstop	

№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция

		#include "Unit1_mytime.h"
		//---
		#pragma package(smart_init)
		#pragma resource "*.dfm"
		TForm1 *Form1;
		
		void MyTime::show(){
		AnsiString s;
		s = IntToStr(chas) + ":" + IntToStr(min) + ":" + IntToStr(sec);
		Form1->Memo1->Lines->Add(s);
		}
		
		void MyTime::summa(MyTime t1, MyTime t2){
		sec = t1.sec + t2.sec;
		min = t1.min + t2.min;
		chas = t1.chas + t2.chas;
		if(sec>=60) { min++; sec-=60; }
		if(min>=60) { chas++; min-=60; }
		if(chas>=24) chas = chas-24;
		}
		
		//---
		__fastcall TForm1::TForm1(TComponent* Owner)
		: TForm(Owner)
		{
		}
		//---
		void __fastcall TForm1::Button1Click(TObject *Sender)
		{
		Memo1->Clear();
		MyTime T1(13,23,50), T2(10,52,50), T3;
		T1.show(); T2.show();	

№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция
		T3.summa(T1, T2); T3.show(); } //----------	

2) Повторив рассмотренный в п.1 пример, необходимо обеспечить возможность ввода пользователем значений переменных T1 и T2. Для этого необходимо дополнительно разместить на форме два компонента LabeledEdit, которые и будут обеспечивать ввод значений для T1 и T2, и реализовать считывание этих значений в обработчике Button1Click.

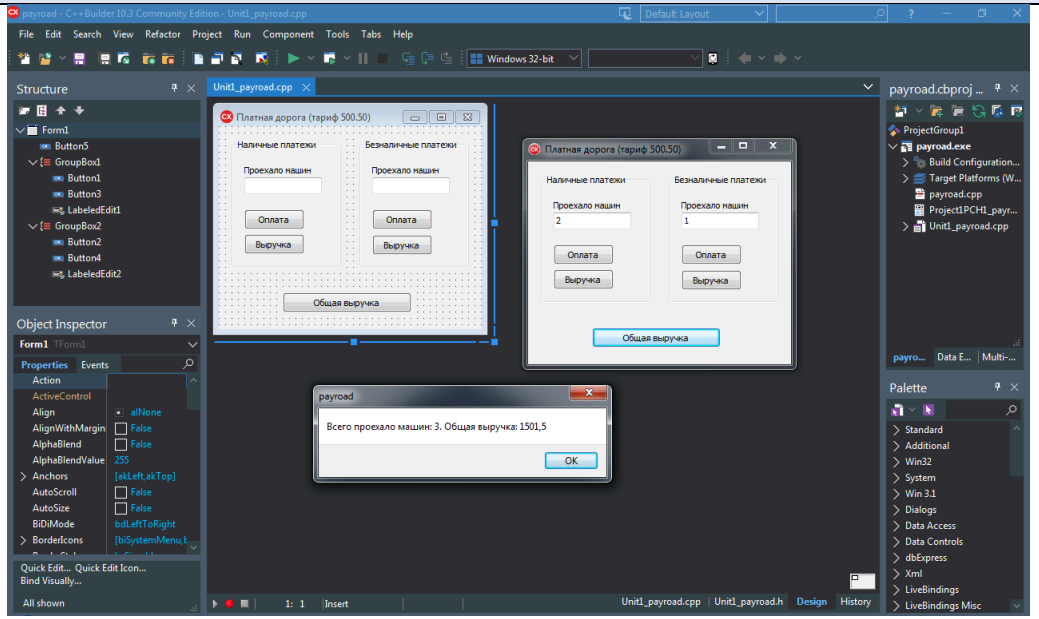
Классы. Инкапсуляция. Реализация класса PayRoad.
Лабораторная работа посвящена изучению классов в ООП. Изучить и повторить проиллюстрированный ниже пример, демонстрирующий реализацию класса платной дороги с тарифом 500,50 за проезд. В примере разрабатывается визуальное приложение (Windows VCL Application), в котором необходимо создать класс с именем PayRoad, содержащий три поля:

int Cars;
float Cash; // наличные
float NonCash; // безналичные

Они предназначены для хранения кол-ва машин, наличных и безналичных платежей. Нулевой конструктор класса должен инициализировать поля нулевыми значениями. Необходимо создать четыре метода класса, которые будут увеличивать счетчик проехавших машин, выводить в LabeledEdit кол-во проехавших машин, выводить суммы платежей (наличных и безналичных), считать и выводить сводные данные.

Иллюстрированный пример разработки приложения

Интерфейс приложения

№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция
			

Здесь в окна LabeledEdit ничего с клавиатуры вводить нельзя, они отображают просто счетчики проехавших машин, которые увеличиваются после нажатия кнопок Оплата.

Исходный код программы

.h-файл:

```
//---------------------------------------------------------------------------
#ifndef Unit1_payroadH
#define Unit1_payroadH
//---------------------------------------------------------------------------
#include <System.Classes.hpp>
#include <Vcl.Controls.hpp>
#include <Vcl.StdCtrls.hpp>
```

№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция

		#include <Vcl.Forms.hpp>
		#include <Vcl.ExtCtrls.hpp>
		//---
		class TForm1 : public TForm
		{
		__published: // IDE-managed Components
		  TGroupBox *GroupBox1;
		  TLabeledEdit *LabeledEdit1;
		  TButton *Button1;
		  TGroupBox *GroupBox2;
		  TLabeledEdit *LabeledEdit2;
		  TButton *Button2;
		  TButton *Button3;
		  TButton *Button4;
		  TButton *Button5;
		  void __fastcall Button1Click(TObject *Sender);
		  void __fastcall Button3Click(TObject *Sender);
		  void __fastcall Button2Click(TObject *Sender);
		  void __fastcall Button4Click(TObject *Sender);
		  void __fastcall Button5Click(TObject *Sender);
		private: // User declarations
		public: // User declarations
		  __fastcall TForm1(TComponent* Owner);
		};
		
		class PayRoad { // класс платной дороги
		  private:
		  int Cars;
		  float Cash;
		  float NonCash;
		  public:
		  PayRoad() { Cars=0; Cash=0.0; NonCash=0.0; }
		  void paying(int flag); // увеличиваем счетчик проехавших	

№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция

		void sum(int flag); // суммируем платежи
		void total(PayRoad car1, PayRoad car2); // считаем итого по нал. и безнал. платежам
		void show(int flag); // выводим showmessage
		};
		//---
		extern PACKAGE TForm1 *Form1;
		//---
		#endif
		
		.cpp-файл:
		
		//---
		#include <vcl.h>
		#pragma hdrstop
		
		#include "Unit1_payroad.h"
		//---
		#pragma package(smart_init)
		#pragma resource "*.dfm"
		TForm1 *Form1;
		
		PayRoad CarCash, CarNonCash, TotalCars;
		
		void PayRoad::paying(int flag){ // увеличиваем счетчик проехавших
		  Cars++;
		  if(flag==1) Cash+=500.50;
		  if(flag==2) NonCash+=500.50;
		}
		
		void PayRoad::show(int flag){
		  if(flag==1) Form1->LabeledEdit1->Text = Cars;
		  if(flag==2) Form1->LabeledEdit2->Text = Cars;
		}	

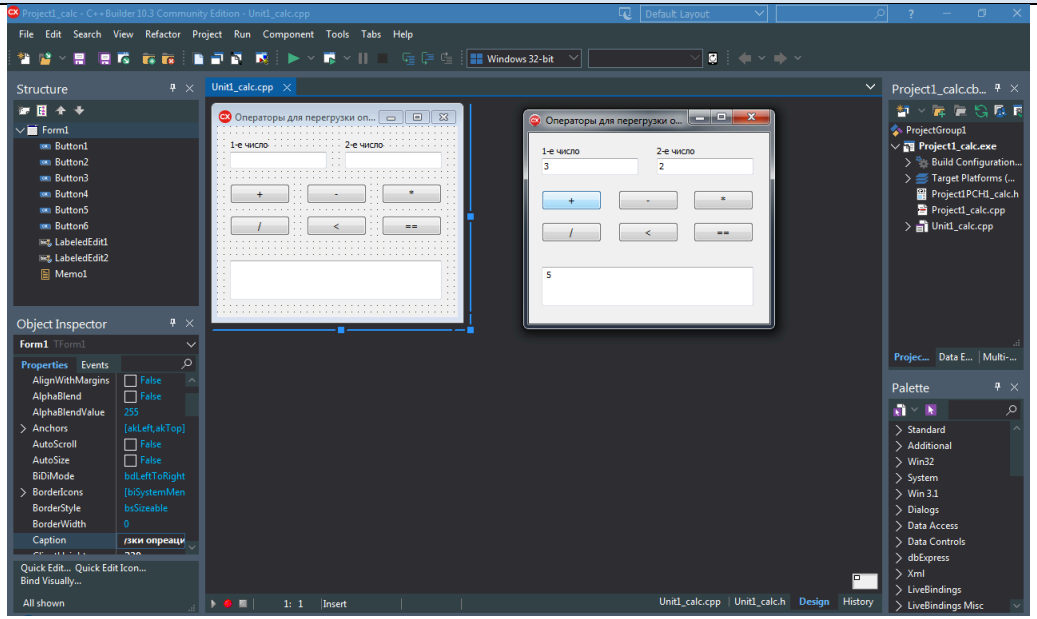
№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция

		void PayRoad::sum(int flag) {
		  if(flag==1) ShowMessage(Cash);
		  if(flag==2) ShowMessage(NonCash);
		}
		
		void PayRoad::total(PayRoad car1, PayRoad car2) {
		  Cars = car1.Cars + car2.Cars;
		  Cash = car1.Cash + car2.Cash;
		  NonCash = car1.NonCash + car2.NonCash;
		  AnsiString s = "Всего проехало машин: " + IntToStr(Cars) + "." + " Общая выручка: " + FloatToStr(Cash+NonCash);
		  ShowMessage(s);
		}
		//---
		__fastcall TForm1::TForm1(TComponent* Owner)
		  : TForm(Owner)
		{
		}
		//---
		void __fastcall TForm1::Button1Click(TObject *Sender)
		{
		  CarCash.paying(1);
		  CarCash.show(1);
		}
		//---
		void __fastcall TForm1::Button3Click(TObject *Sender)
		{
		  CarCash.sum(1);
		}
		//---
		void __fastcall TForm1::Button2Click(TObject *Sender)
		{	

№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция

		CarNonCash.paying(2);
		CarNonCash.show(2);
		}
		//---
		void __fastcall TForm1::Button4Click(TObject *Sender)
		{
		CarNonCash.sum(2);
		}
		//---
		void __fastcall TForm1::Button5Click(TObject *Sender)
		{
		TotalCars.total(CarCash, CarNonCash);
		}	

	Выполнение лабораторной работы. Лабораторная работа № 5.1	Классы. Операторные функции. Реализация класса для перегрузки операций. Лабораторная работа посвящена изучению классов в ООП. Изучить и повторить проиллюстрированный ниже пример, демонстрирующий реализацию класса MyInt, демонстрирующего перегрузку арифметических операций и операций сравнения для объектов этого класса. В примере разрабатывается визуальное приложение (Windows VCL Application), в котором необходимо создать класс с именем MyInt, содержащий одно поле: int I; В классе нужно реализовать нулевой конструктор, инициализирующий поля нулевыми значениями, и ненулевой, инициализирующий поля класса значениями. Необходимо создать семь методов класса, которые будут реализовывать операции +, -, *, /, <, == с объектами класса и выводить результат в Memo. Иллюстрированный пример разработки приложения Интерфейс приложения	ПК-2: ИД-ПК-2.1 ИД-ПК-2.2 ИД-ПК-2.3 ИД-ПК-2.4

№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция
			

Здесь в окна LabeledEdit пользователь вводит числа и нажимает кнопки операций, результат выводится в Memo методом класса show().

Исходный код программы

В предыдущих лабораторных работах описание класса мы размещали в .h-файле, а в этой работе разместим для упрощения в .cpp-файле.

.cpp-файл:

```
//---------------------------------------------------------------------------
#include <vcl.h>
#pragma hdrstop
```

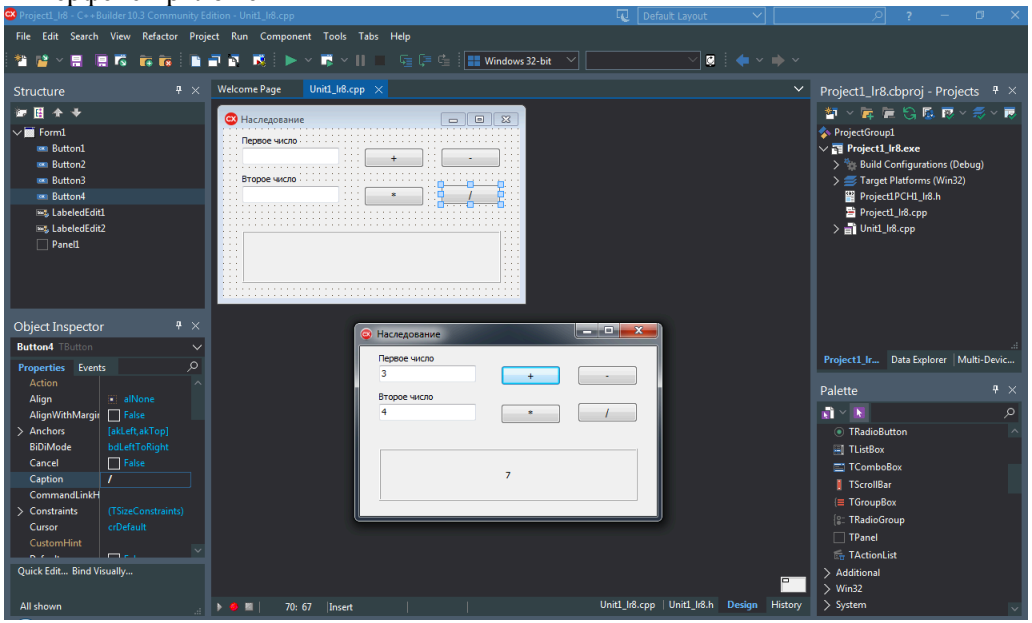
№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция

		#include "Unit1_calc.h"
		//---
		#pragma package(smart_init)
		#pragma resource "*.dfm"
		TForm1 *Form1;
		
		class MyInt {
		  private:
		  int I;
		  public:
		  MyInt() { I=0; }
		  MyInt(int i) { I=i; }
		  void show(){
		  Form1->Memo1->Lines->Add(I);
		  }
		  MyInt operator+(MyInt a2) {
		  return (I + a2.I);
		  }
		  bool operator<(MyInt a2) {
		  if(I < a2.I) return true;
		  else return false;
		  }
		};
		//---
		__fastcall TForm1::TForm1(TComponent* Owner)
		  : TForm(Owner)
		{
		}
		//---
		void __fastcall TForm1::Button1Click(TObject *Sender)
		{
		  MyInt i1, i2, i3;
		  i1 = StrToInt(LabeledEdit1->Text);	

№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция

		i2 = StrToInt(LabeledEdit2->Text);
		i3 = i1 + i2;
		i3.show();
		}
		//---
		void __fastcall TForm1::Button5Click(TObject *Sender)
		{
		MyInt i1, i2, i3;
		i1 = StrToInt(LabeledEdit1->Text);
		i2 = StrToInt(LabeledEdit2->Text);
		if(i1 < i2) Form1->Memo1->Lines->Add("Истина");
		else Form1->Memo1->Lines->Add("ЛОЖЬ");
		}
		//---
		
		Повторив рассмотренный пример, необходимо самостоятельно доделать реализацию кнопок -, *, / (обработать ситуацию деления на ноль), ==. Выполните абстракцию.	

	Выполнение лабораторной работы. Лабораторная работа № 5.2	Классы. Приведение типов. Реализация класса Stroka. Лабораторная работа посвящена изучению классов в ООП. Изучить и повторить проиллюстрированный в видеозаписи пример, демонстрирующий реализацию следующей задачи. На основе типа char создайте класс Stroka. Перегрузите операцию приведения строки типа char к типу Stroka и наоборот. Напишите визуальное приложение (Windows VCL Application) для проверки этого класса.	ПК-2: ИД-ПК-2.1 ИД-ПК-2.2 ИД-ПК-2.3 ИД-ПК-2.4
	Выполнение лабораторной работы. Лабораторная работа № 6.1	Классы. Наследование. Лабораторная работа посвящена изучению классов в ООП. Изучить и повторить проиллюстрированный ниже пример, демонстрирующий реализацию базового класса Float и производного класса FloatPr. В примере разрабатывается визуальное приложение (Windows VCL Application), в котором на основе стандартного типа float создается базовый класс Float, имеющий два конструктора, метод вывода на экран и метод для перегрузки арифметической операции +. Используя общее наследование, создается производный класс FloatPr, добавляющий возможность использования операций -, *, /. Далее идет проверка производного класса.	ПК-2: ИД-ПК-2.1 ИД-ПК-2.2 ИД-ПК-2.3 ИД-ПК-2.4

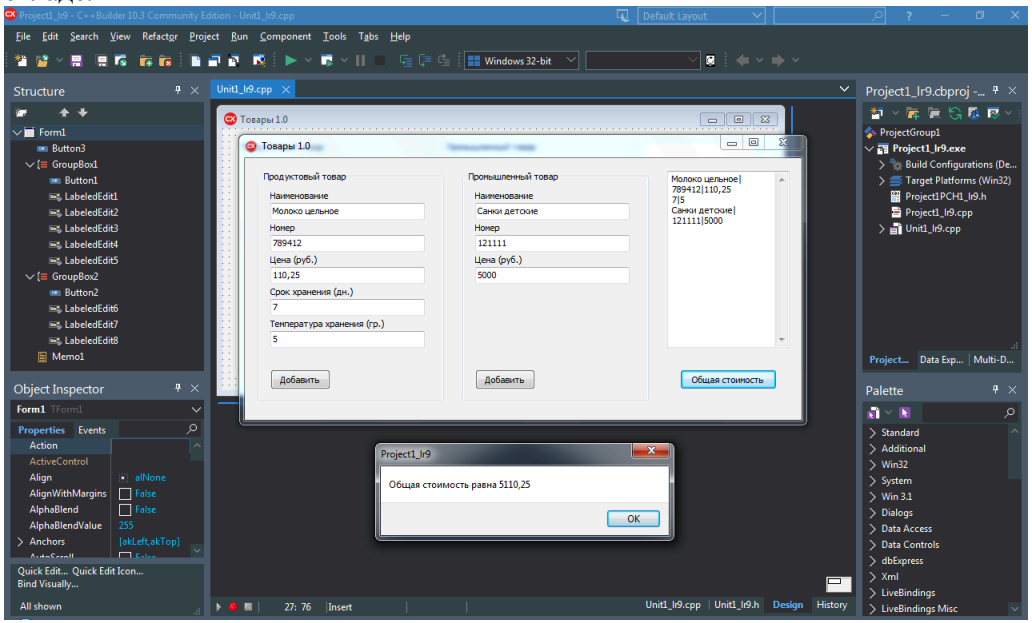
№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция
		Иллюстрированный пример разработки приложения Интерфейс приложения  Исходный код программы В .h-файле для упрощения ничего не прописываем. .cpp-файл: //--- #include <vcl.h> #pragma hdrstop #include "Unit1_lr8.h" //---	

№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция

		#pragma package(smart_init)
		#pragma resource "*.dfm"
		TForm1 *Form1;
		
		class Float { // базовый класс
		  protected:
		  float f;
		  public:
		  Float() { f=0; }
		  Float(float d) { f=d; }
		  void show() { Form1->Panel1->Caption = f; }
		  Float operator+(Float b) { return (f+b.f); }
		};
		
		class FloatPr : public Float { // производный класс (общее наследование)
		  public:
		  FloatPr() : Float() {}
		  FloatPr(float v) : Float(v) {}
		  FloatPr(Float a) : Float(a) {} // для +
		  FloatPr operator-(FloatPr b) { return (f-b.f); }
		  FloatPr operator*(FloatPr b) { return (f*b.f); }
		  FloatPr operator/(FloatPr b) { return (f/b.f); }
		};
		
		//---
		__fastcall TForm1::TForm1(TComponent* Owner)
		  : TForm(Owner)
		{
		}
		//---
		void __fastcall TForm1::Button1Click(TObject *Sender)
		{
		  FloatPr f1, f2, f3;	

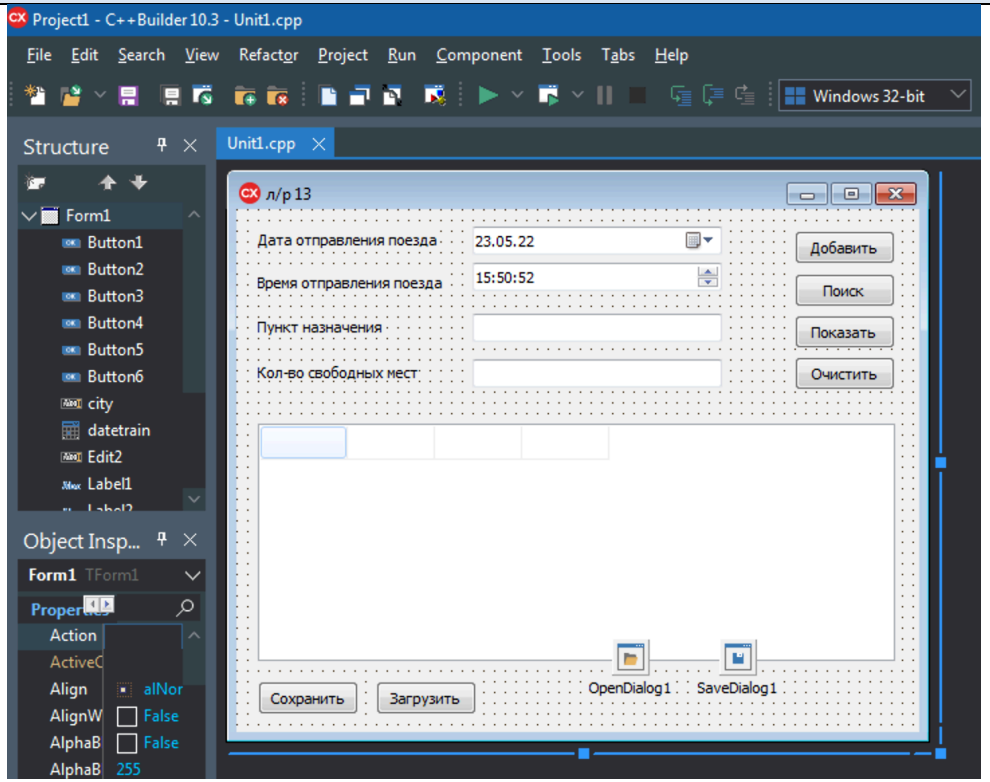
№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция

		f1 = StrToFloat(LabeledEdit1->Text);
		f2 = StrToFloat(LabeledEdit2->Text);
		f3 = f1+f2; // вызывается метод базового класса operator+ и третий конструктор производного класса
		f3.show();
		}
		//---
		void __fastcall TForm1::Button2Click(TObject *Sender)
		{
		FloatPr f1, f2, f3;
		f1 = StrToFloat(LabeledEdit1->Text);
		f2 = StrToFloat(LabeledEdit2->Text);
		f3 = f1-f2; // вызывается метод производного класса operator-
		f3.show();
		}
		//---
		void __fastcall TForm1::Button3Click(TObject *Sender)
		{
		FloatPr f1, f2, f3;
		f1 = StrToFloat(LabeledEdit1->Text);
		f2 = StrToFloat(LabeledEdit2->Text);
		f3 = f1*f2; // вызывается метод производного класса operator*
		f3.show();
		}
		//---
		void __fastcall TForm1::Button4Click(TObject *Sender)
		{
		FloatPr f1, f2, f3;
		f1 = StrToFloat(LabeledEdit1->Text);
		f2 = StrToFloat(LabeledEdit2->Text);
		f3 = f1/f2; // вызывается метод производного класса operator/
		f3.show();
		}	

№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция
		Выполните абстракцию.	
	Выполнение лабораторной работы. Лабораторная работа № 7.1	Классы. Наследование при изначальной разработке программы. Лабораторная работа посвящена изучению классов в ООП. Требуется изучить и повторить проиллюстрированный пример, демонстрирующий применение технологии наследования при изначальном проектировании приложения. Используя известный по лекциям класс Tovar, в примере создаётся два производных от него класса: 1) TovarProd, добавляющий возможность хранить информацию о сроке хранения и температуре хранения продуктовых товаров; 2) TovarProm, позволяющий хранить информацию в соответствии с полями базового класса. Созданное визуальное приложение (Windows VCL Application) должно позволять: вводить информацию либо о продуктовых товарах, либо о промышленных товарах; выводить общую стоимость товаров, имеющихся на складе.  Поставленную задачу удобно решать с применением технологии наследования при изначальной разработке приложения. По условию задачи, у нас две категории товаров, у	ПК-2: ИД-ПК-2.1 ИД-ПК-2.2 ИД-ПК-2.3 ИД-ПК-2.4

№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция
		которых первые три поля (наименование, номер, цена) совпадают. И действие по добавлению записи тоже совпадает. Таким образом, создадим базовый класс Tovar и два производных от него класса для продуктовых товаров и для промышленных. Производных класс TovarProd для продуктовых товаров будет задействовать все поля и методы базового класса и добавлять еще некоторые специфические именно для себя. Производный класс TovarProm вообще не требует никаких доработок, так как ему полностью хватает возможностей базового класса. Далее создадим массивы для работы с классами продуктовых и промышленных товаров. После создадим обработчик события нажатия на кнопку Button1 и обработчик события нажатия на кнопку Button2, прописав в них логику добавления товаров в соответствующие массивы и вывода записей в Memo1. В обработчик события нажатия кнопки Button3 запишем логику подсчета общей стоимости добавленных товаров.	
	Выполнение лабораторной работы. Лабораторная работа № 7.2	Классы. Отработка задач с указателями this. В железнодорожных кассах хранится информация о свободных местах в поездах на ближайшую неделю в следующем виде: дата выезда, пункт назначения, время отправления, число свободных мест. Поступает запрос на резервирование m мест до города N на k-й день недели с временем отправления поезда не позднее t часов вечера. Вывести время отправления или сообщение о невозможности выполнить заказ в полном объеме. Пример интерфейса приложения:	ПК-2: ИД-ПК-2.1 ИД-ПК-2.2 ИД-ПК-2.3 ИД-ПК-2.4

№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция



Программу написать с использованием класса и компонента StringGrid. Класс описать в отдельном файле. Должна быть возможность добавления в таблицу, сохранения в текстовый файл и загрузки из файла.
Исходный код:
unit1.h

```
//---------------------------------------------------------------------------

#ifndef Unit1H
#define Unit1H
```

№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция

		//---
		#include <System.Classes.hpp>
		#include <Vcl.Controls.hpp>
		#include <Vcl.StdCtrls.hpp>
		#include <Vcl.Forms.hpp>
		#include <Vcl.ComCtrls.hpp>
		#include <Vcl.Grids.hpp>
		#include <Vcl.Dialogs.hpp>
		//---
		class TForm1 : public TForm
		{
		__published: // IDE-managed Components
		TLabel *Label1;
		TDateTimePicker *datetrain;
		TLabel *Label2;
		TLabel *Label3;
		TDateTimePicker *timetrain;
		TLabel *Label4;
		TEdit *Edit1;
		TButton *Button1;
		TEdit *city;
		TButton *Button2;
		TStringGrid *StringGrid1;
		TButton *Button3;
		TButton *Button4;
		TSaveDialog *SaveDialog1;
		TOpenDialog *OpenDialog1;
		TButton *Button5;
		TButton *Button6;
		void __fastcall Button1Click(TObject *Sender);
		void __fastcall Button2Click(TObject *Sender);
		void __fastcall Button3Click(TObject *Sender);
		void __fastcall Button4Click(TObject *Sender);	

№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция

		void __fastcall Button5Click(TObject *Sender);
		void __fastcall Button6Click(TObject *Sender);
		
		private: // User declarations
		public: // User declarations
		__fastcall TForm1(TComponent* Owner);
		void update();
		};
		//---
		extern PACKAGE TForm1 *Form1;
		//---
		#endif
		file1.h
		
		class train
		{
		public:
		String date;
		String time;
		String stop_point;
		int free_space;
		
		public:
		train(String date, String time, String stop_point, int free_space);
		void set_train(String date, String time, String stop_point, int free_space);
		train get_train();
		};
		
		train::train(String date, String time, String stop_point, int free_space)
		{
		this->date = date;
		this->time = time;
		this->stop_point = stop_point;	

№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция

		this->free_space = free_space;
		}
		
		void train::set_train(String date, String time, String stop_point, int free_space)
		{
		this->date = date;
		this->time = time;
		this->stop_point = stop_point;
		this->free_space = free_space;
		}
		
		train train::get_train()
		{
		return *this;
		}
		unit1.cpp
		// ---
		
		#include <vcl.h>
		#pragma hdrstop
		
		#include "Unit1.h"
		
		#include "File1.h"
		#include <vector>
		// ---
		#pragma package(smart_init)
		#pragma resource "*.dfm"
		TForm1 *Form1;
		std::vector<train>trains;
		
		// ---
		__fastcall TForm1::TForm1(TComponent* Owner) : TForm(Owner) {	

№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция

```
    Form1->StringGrid1->Cells[0][0] = "Дата";
    StringGrid1->ColWidths[0] = 140;
    Form1->StringGrid1->Cells[1][0] = "Время";
    StringGrid1->ColWidths[1] = 140;
    Form1->StringGrid1->Cells[2][0] = "Место назначения";
    StringGrid1->ColWidths[2] = 185;
    Form1->StringGrid1->Cells[3][0] = "Свободных мест";
    StringGrid1->ColWidths[3] = 100;
}

// ---------------------------------------------------------------------------
void __fastcall TForm1::Button1Click(TObject *Sender) {
    train tr(Form1->datetrain->Date, Form1->timetrain->Time, Form1->city->Text,
        StrToInt(Form1->Edit1->Text));
    trains.push_back(tr);
    StringGrid1->RowCount = 0;
     int row_count = StringGrid1->RowCount;
    for (int i = 0; i < trains.size(); i++,row_count++) {

        StringGrid1->RowCount = row_count + 1;
        Form1->StringGrid1->Cells[0][row_count] = trains[i].get_train().date;
        Form1->StringGrid1->Cells[1][row_count] = trains[i].get_train().time;
        Form1->StringGrid1->Cells[2][row_count] =
            trains[i].get_train().stop_point;
        Form1->StringGrid1->Cells[3][row_count] =
            IntToStr(trains[i].get_train().free_space);
    }
}

void TForm1::update() {

}
```

№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция
		// ---------- void __fastcall TForm1::Button2Click(TObject *Sender) { bool good = false; StringGrid1->RowCount = 1; for (int i = 0; i < trains.size(); i++) { String sdate = Form1->datetrain->Date; if(sdate != trains[i].get_train().date) continue; if (Form1->city->Text != "" && Form1->city->Text != trains[i].get_train ().stop_point) continue; String stime = Form1->timetrain->Time; if (stime < trains[i].get_train().time) continue; if (StrToInt(Form1->Edit1->Text) > trains[i].get_train().free_space) continue; good = true; int row_count = StringGrid1->RowCount; StringGrid1->RowCount = row_count + 1; Form1->StringGrid1->Cells[0][row_count] = trains[i].get_train().date; Form1->StringGrid1->Cells[1][row_count] = trains[i].get_train().time; Form1->StringGrid1->Cells[2][row_count] = trains[i].get_train().stop_point; Form1->StringGrid1->Cells[3][row_count] = IntToStr(trains[i].get_train().free_space); }	

№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция

		if (good == false)
		Application->MessageBox(L"Подходящих поездов нет", L"Внимание",
		MB_OKCANCEL);
		}
		
		// ---
		void __fastcall TForm1::Button3Click(TObject *Sender) {
		
		if (SaveDialog1->Execute()) {
		int f;
		if (FileExists(SaveDialog1->FileName))
		f = FileOpen(SaveDialog1->FileName, fmOpenWrite);
		else
		f = FileCreate(SaveDialog1->FileName);
		if (f != -1) {
		// сохранить таблицу в файле
		for (int i = 0; i < StringGrid1->RowCount; i++) {
		AnsiString st = StringGrid1->Rows[i]->DelimitedText + "\r\n";
		FileWrite(f, st.c_str(), st.Length());
		}
		FileClose(f);
		
		}
		}
		}
		
		int GetLine(int f, AnsiString *st) {
		unsigned char buf[256]; // строка (буфер)
		unsigned char *p = buf; // указатель на строку
		int n; // кол-во прочитанных байт (значение ф-и FileRead)
		int len = 0; // длина строки
		n = FileRead(f, p, 1);
		while (n != 0) {	

№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция

```
        if (*p == '\r') {
            n = FileRead(f, p, 1); // прочитать '\n'
            break;
        }
        len++;
        p++;
        n = FileRead(f, p, 1);
    }
    *p = '\0';
    if (len != 0)
        st->printf("%s", buf);
    return len;
}

// ---------------------------------------------------------------------------
void __fastcall TForm1::Button4Click(TObject *Sender) {
    if (OpenDialog1->Execute()) {
        int inc = 0;
        int f; // дескриптор файла
        AnsiString st; // прочитанная строка
        bool fl = true; // true - чтение первой строки
        if ((f = FileOpen(OpenDialog1->FileName, fmOpenRead)) == -1)
            return;
        // файл открыт
        // загрузить в таблицу
        while (GetLine(f, &st) != 0) {
            // добавить строку в таблицу
            if (fl) {
                StringGrid1->Rows[StringGrid1->Row]->DelimitedText = st;
                fl = false;
                trains.clear();
            }
            else {
```

№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция

```
            inc++;
            StringGrid1->RowCount++;
            StringGrid1->Row = StringGrid1->RowCount - 1;
            StringGrid1->Rows[StringGrid1->Row]->DelimitedText = st;
            train tr(Form1->StringGrid1->Cells[0][inc],Form1->StringGrid1->Cells[1][inc],
            Form1->StringGrid1->Cells[2][inc],StrToInt(Form1->StringGrid1->Cells[3][inc]));
            trains.push_back(tr);

          }
        }
        FileClose(f);

    }

}
void __fastcall TForm1::Button5Click(TObject *Sender)
{
    StringGrid1->RowCount = 1;

    for (int i = 0; i < trains.size(); i++) {

        int row_count = StringGrid1->RowCount;
        StringGrid1->RowCount = row_count + 1;
        Form1->StringGrid1->Cells[0][row_count] = trains[i].get_train().date;
        Form1->StringGrid1->Cells[1][row_count] = trains[i].get_train().time;
        Form1->StringGrid1->Cells[2][row_count] =
            trains[i].get_train().stop_point;
        Form1->StringGrid1->Cells[3][row_count] =
            IntToStr(trains[i].get_train().free_space);
    }
}
//---------------------------------------------------------------------------
```

№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция

		void __fastcall TForm1::Button6Click(TObject *Sender)
		{
		trains.clear();
		StringGrid1->RowCount = 1;
		}
		//---
		Также необходимо попробовать вместо спецификатора public для полей класса train использовать спецификатор private или protected:
		class train {
		public:
		//protected:
		//private:
		Что можно изменить в программном коде, чтобы обойтись без public?	

2	Посещение профориентационных мероприятий	№1. Участие в публичных профориентационных мероприятиях, проводимых на территории РГУ им. А.Н. Косыгина. №2. Участие в публичных профориентационных мероприятиях, проводимых вне территории РГУ им. А.Н. Косыгина.	ПК-2: ИД-ПК-2.1 ИД-ПК-2.2 ИД-ПК-2.3 ИД-ПК-2.4
3	Участие (достижения) в профессиональных конкурсах	Участие или призовое место в хакатоне или ином соревновании с официальным участием РГУ им. А.Н. Косыгина	
4	Научная и/или практическая работа	Участие в научной конференции или ином научном мероприятии в качестве представителя РГУ им. А.Н. Косыгина	

5.2. Критерии, шкалы оценивания текущего контроля успеваемости:

Критерии и шкалы оценивания формируются в соответствии с ограничениями Методикой использования балльно-рейтинговой системы при реализации основных профессиональных образовательных программ высшего образования Института информационных технологий и цифровой трансформации.

Тип	Наименование	Критерии оценивания и правила начисления баллов за КРМ	Балл или

контрольно-рейтингового мероприятия	КРМ	Контрольные сроки и шкала эрозии баллов	Правила начисления баллов	Начисление баллов после завершения аттестации	диапазон баллов
Посещение проф-ориентационных мероприятий	Участие в публичных мероприятиях, проводимых на территории РГУ им. А.Н. Косыгина	Нет	Приказ или Распоряжение о включении мероприятий в учебный процесс, наличие отметки о посещении мероприятия. Подтверждение от директора института о соответствии мероприятия профилю подготовки. Балл за КРМ определяется как отношение количества посещенных мероприятий к проведенным. Мероприятие засчитывается как посещенное при условии активной работы обучающегося на мероприятии: озвучивание вопросов, участие в дискуссиях, проявлении признаков сформированности соответствующих компетенций и т.п. КРМ может быть учтено по всем дисциплинам, использующим БРС.	Нет	1-5
	Участие в публичных мероприятиях, проводимых вне территории РГУ им. А.Н. Косыгина	Нет	Приказ или Распоряжение об участии в мероприятии, наличие подтверждения посещения мероприятия. Подтверждение от директора института о соответствии мероприятия профилю подготовки. Балл за КРМ определяется как отношение количества посещенных мероприятий к проведенным. Мероприятие засчитывается как посещенное при условии активной работы обучающегося на мероприятии: озвучивание вопросов, участие в дискуссиях, проявлении признаков сформированности соответствующих компетенций и т.п. КРМ может быть учтено по всем дисциплинам, использующим БРС.	Нет	1-4
Участие (достижения) в профессиональных конкурсах	Участие или призовое место в хакатоне или ином соревновании с официальным участием РГУ им. А.Н. Косыгина	Нет	Приказ или Распоряжение об организации и/или участии в мероприятии. Документы, подтверждающие участие и результаты участия. Соответствие содержания дисциплины и мероприятия определяет реализующий дисциплину преподаватель. Баллы за мероприятия определяются реализующим дисциплину преподавателем на основании предоставленных документов. КРМ может быть учтено только в одной дисциплине, использующей БРС (по выбору студента).		
			Обучающийся проявил профессиональный подход к выполнению конкурсного задания, занял призовое место или его конкурсная работа выполнена на высоком профессиональном уровне без грубых ошибок.	Да	1-2

Тип контрольно-рейтингового мероприятия	Наименование КРМ	Критерии оценивания и правила начисления баллов за КРМ			Балл или диапазон баллов
		Контрольные сроки и шкала эрозии баллов	Правила начисления баллов	Начисление баллов после завершения аттестации	
			Обучающийся участвовал в конкурсе, выполнил конкурсное задание полностью и в срок. Однако его работа содержит ошибки, помарки или не соответствует тематике дисциплины.		0-1
Научная и/или практическая работа	Участие в научной конференции или ином научном мероприятии в качестве представителя РГУ им. А.Н. Косыгина	Нет	Сертификат или иные документ, подтверждающие участие и результаты участия в научных конференциях или иных научных мероприятиях. Соответствие содержания дисциплины и прошедшего обучения определяет реализующий дисциплину преподаватель. Баллы за мероприятия определяются реализующим дисциплину преподавателем на основании предоставленных документов. КРМ может быть учтено только в одной дисциплине, использующей БРС (по выбору студента).		
			Обучающийся представил актуальную и оригинальную работу, соответствующую тематике дисциплины. Работа отмечена призовым местом, иным знаком отличия или представляет собой интерес в рамках ИТ-направления.	Да	3-4
			Обучающийся представил формальную работу, не имеющей признаки научной работы. Работа содержит ошибки, признаки плагиата или не соответствует научной тематике по формальным признакам.		0-2
Выполнение учебных заданий	Лабораторная работа	Нет	Работа выполнена полностью. Нет ошибок в логических рассуждениях и в реализации задания в виде файла или выполняемой программы. Возможно наличие одной неточности или описки, не являющиеся следствием незнания или непонимания учебного материала и не влияющей на функциональные качества программы. Обучающийся показал полный объем знаний, умений в освоении пройденных тем и применение их на практике. Работа зачтена.	Да	47-55
			Работа выполнена полностью, но выбран неэффективный алгоритм или метод реализации, обоснований шагов решения недостаточно. Допущена одна ошибка или два-три недочета, которые незначительно влияют на качество представленной работы. Работа зачтена.		38-46

Тип контрольно-рейтингового мероприятия	Наименование КРМ	Критерии оценивания и правила начисления баллов за КРМ			Балл или диапазон баллов
		Контрольные сроки и шкала эрозии баллов	Правила начисления баллов	Начисление баллов после завершения аттестации	
			Допущены более одной ошибки или более двух-трех недочетов, которые оказывают значительное влияние на представляемый файл или компьютерную программу, ухудшают их информативность и функциональные возможности. Работа зачтена.		30-37
			Работа выполнена не полностью. Допущены грубые ошибки. Файлы не содержат необходимой информации, компьютерная программа выдаёт неправильные результаты при вычислении тестовых примеров. Работа не зачтена.		0-29
				Итого:	0-70

5.3. Промежуточная аттестация:

Форма промежуточной аттестации	Типовые контрольные задания и иные материалы для проведения промежуточной аттестации:	Формируемая компетенция
Экзамен: Компьютерное тестирование	Вопрос 1. Что такое программная инженерия? а) Разработка программного обеспечения. б) Управление жизненными циклами программного обеспечения. в) Создание надёжных и эффективных компьютерных программ. г) Всё вышеперечисленное. Вопрос 2. Какие основные этапы жизненного цикла программного обеспечения? а) Анализ требований, проектирование, кодирование, тестирование, внедрение и сопровождение. б) Подготовка, планирование, анализ требований, проектирование, кодирование, интеграция, тестирование, внедрение и сопровождение. в) Анализ требований, проектирование, кодирование, тестирование, внедрение и	ПК-2: ИД-ПК-2.1 ИД-ПК-2.2 ИД-ПК-2.3 ИД-ПК-2.4

	сопровождение.
	г) Подготовка, анализ требований, проектирование, кодирование, интеграция, тестирование, внедрение и сопровождение.
	. . .
	Вопрос 14.
	Одной из часто используемых библиотечных функций для работы со строковыми массивами типа char является функция strncmp(s1, s2, n), которая
	o копирует не более n символов из строки s2 в s1 и возвращает s1
	o добавляет не более n символов из строки s2 к s1 и возвращает s1
	o сравнивает строку s1 и первые n символов строки s2
	
	Вопрос 15.
	В определении, в объявлении и при вызове одной и той же функции типы и порядок следования параметров
	o должны совпадать
	o могут частично не совпадать
	o должны различаться
	
	Вопрос 17.
	Способ передачи аргументов, при котором функция создаёт копии передаваемых значений, называется
	o передачей аргументов по ссылке
	o передачей аргументов по указателю
	o передачей аргументов по значению
	
	Вопрос 19.
	Допустим, в программе используется некоторая функция вида void function_1(float& v) { v = v* 2.54; }. Какой вызов этой функции должен быть в главной функции main() для расчета значения вещественной переменной var?
	o function_1(&var);
	o function_1(var);
	
	Вопрос 80.
	Политика основных принципов ООП такова, что, если функция является членом класса, она [[1]] доступ к полям класса.	

	Вопрос 101. Если поле данных класса описано с ключевым словом static, то значение этого поля будет [[1]] для всех объектов данного класса.	

5.4. Критерии, шкалы оценивания промежуточной аттестации учебной дисциплины:

Результат промежуточной аттестации определяется как соответствие суммы набранных рейтинговых баллов за контрольно-рейтинговые мероприятия текущей аттестации и контрольно-рейтинговых баллов, набранных за промежуточную аттестацию. Оценка по дисциплины выставляется в соответствии с Системой оценивания результатов текущего контроля и промежуточной аттестации, описанной в данном документе, а также в соответствии с Методикой использования балльно-рейтинговой системы при реализации основных профессиональных образовательных программ высшего образования Института информационных технологий и цифровой трансформации.

Форма промежуточной аттестации **Наименование оценочного средства**	**Критерии оценивания**	**Шкалы оценивания**		
		100-балльная система	**Пятибалльная система**	
Экзамен: компьютерное тестирование	За выполнение каждого тестового задания испытуемому выставляются баллы. За полностью правильный ответ к каждому заданию с выбором одного правильного варианта выставляется один балл, за неправильный — ноль. За задания с выбором нескольких правильных ответов или в заданиях с сопоставлениями испытуемый может получить менее 1 балла. Например, если правильных ответов в задании два, то за каждый он получает 0,5 балла, если правильных ответов три, то за каждый он получает 0,333 балла и т.п. Правила оценки всего теста: вне зависимости от количества заданий в тесте общая сумма баллов за все правильные ответы пересчитывается тестирующей компьютерной системой в итоговые баллы. 20 итоговых баллов эквивалентны 100% правильных ответов. Для того, чтобы получить отличную, хорошую, удовлетворительную или неудовлетворительную оценки, итоговые баллы за промежуточную аттестацию складываются с баллами за выполненные лабораторные работы и практические задания.	*21 – 30 баллов*	*5*	*85% - 100%*
		11 – 20 баллов	*4*	*70% - 84%*
		6 – 10 баллов	*3*	*55% - 69%*
		0 – 5 баллов	*2*	*54% и менее 54%*

5.5. Система оценивания результатов текущего контроля и промежуточной аттестации.

В соответствии с Методикой использования балльно-рейтинговой системы при реализации основных профессиональных образовательных программ высшего образования Института информационных технологий и цифровой трансформации, оценка по дисциплине выставляется обучающемуся с учётом результатов текущей и промежуточной аттестации.

Форма контроля	100-балльная система	Пятибалльная система
Текущий контроль:		
- Выполнение лабораторных работ	0 - 55 баллов	зачтено/не зачтено
- посещение профориентационных мероприятий	0 – 9 баллов	зачтено/не зачтено
- участие (достижения) в профессиональных конкурсах	0 – 3 балла	зачтено/не зачтено
- научная и/или практическая работа	0 – 3 балла	зачтено/не зачтено
Промежуточная аттестация *экзамен*	0 - 30 баллов	отлично хорошо удовлетворительно неудовлетворительно
Итого за шестой семестр *экзамен*	0 - 100 баллов	

Полученный совокупный результат конвертируется в пятибалльную систему оценок в соответствии с таблицей:

100-балльная система	Пятибалльная система (оценка по дисциплине)
	экзамен
85 – 100 баллов	отлично
70 – 84 баллов	хорошо
55 – 69 баллов	удовлетворительно
0 – 54 баллов	неудовлетворительно

6. ОБРАЗОВАТЕЛЬНЫЕ ТЕХНОЛОГИИ

Реализация программы предусматривает использование в процессе обучения следующих образовательных технологий:

- проблемная лекция;
- проектная деятельность;
- групповые дискуссии;
- анализ ситуаций и имитационных моделей;
- преподавание дисциплины на основе результатов научных исследований;
- поиск и обработка информации с использованием сети Интернет;
- дистанционные образовательные технологии;
- использование на лекционных занятиях видеоматериалов и наглядных пособий;
- самостоятельная работа в системе компьютерного тестирования.

7. ПРАКТИЧЕСКАЯ ПОДГОТОВКА

Практическая подготовка в рамках учебной дисциплины реализуется при проведении практических занятий и лабораторных работ, поскольку они предусматривают передачу учебной информации обучающимся, которая необходима для последующего выполнения практической работы.

8. ОРГАНИЗАЦИЯ ОБРАЗОВАТЕЛЬНОГО ПРОЦЕССА ДЛЯ ЛИЦ С ОГРАНИЧЕННЫМИ ВОЗМОЖНОСТЯМИ ЗДОРОВЬЯ

При обучении лиц с ограниченными возможностями здоровья и инвалидов используются подходы, способствующие созданию безбарьерной образовательной среды: технологии дифференциации и индивидуального обучения, применение соответствующих методик по работе с инвалидами, использование средств дистанционного общения, проведение дополнительных индивидуальных консультаций по изучаемым теоретическим вопросам и практическим занятиям, оказание помощи при подготовке к промежуточной аттестации.

При необходимости рабочая программа дисциплины может быть адаптирована для обеспечения образовательного процесса лицам с ограниченными возможностями здоровья, в том числе для дистанционного обучения.

Учебные и контрольно-измерительные материалы представляются в формах, доступных для изучения студентами с особыми образовательными потребностями с учетом нозологических групп инвалидов:

Для подготовки к ответу на практическом занятии, студентам с ограниченными возможностями здоровья среднее время увеличивается по сравнению со средним временем подготовки обычного студента.

Для студентов с инвалидностью или с ограниченными возможностями здоровья форма проведения текущей и промежуточной аттестации устанавливается с учетом индивидуальных психофизических особенностей (устно, письменно на бумаге, письменно на компьютере, в форме тестирования и т.п.).

Промежуточная аттестация по дисциплине может проводиться в несколько этапов в форме рубежного контроля по завершению изучения отдельных тем дисциплины. При необходимости студенту предоставляется дополнительное время для подготовки ответа на зачете или экзамене.

Для осуществления процедур текущего контроля успеваемости и промежуточной аттестации обучающихся создаются, при необходимости, фонды оценочных средств, адаптированные для лиц с ограниченными возможностями здоровья и позволяющие оценить достижение ими запланированных в основной образовательной программе результатов обучения и уровень сформированности всех компетенций, заявленных в образовательной программе.

9. МАТЕРИАЛЬНО-ТЕХНИЧЕСКОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ

Характеристика материально-технического обеспечения дисциплины соответствует требованиями ФГОС ВО.

Материально-техническое обеспечение дисциплины при обучении с использованием традиционных технологий обучения.

Наименование учебных аудиторий, лабораторий, мастерских, библиотек, спортзалов, помещений для хранения и профилактического обслуживания учебного оборудования и т.п.	**Оснащенность учебных аудиторий, лабораторий, мастерских, библиотек, спортивных залов, помещений для хранения и профилактического обслуживания учебного оборудования и т.п.**
119071, г. Москва, Малый Калужский переулок, дом 1, строение 3	
аудитории для проведения занятий лекционного типа	комплект учебной мебели, технические средства обучения, служащие для представления учебной информации большой аудитории: – компьютерная техника (ноутбук/компьютер); – проектор;

Наименование учебных аудиторий, лабораторий, мастерских, библиотек, спортзалов, помещений для хранения и профилактического обслуживания учебного оборудования и т.п.	Оснащенность учебных аудиторий, лабораторий, мастерских, библиотек, спортивных залов, помещений для хранения и профилактического обслуживания учебного оборудования и т.п.
	– экран.
аудитории для проведения практических занятий, выполнения лабораторных работ, занятий по практической подготовке, групповых и индивидуальных консультаций, текущего контроля и промежуточной аттестации	комплект учебной мебели, технические средства обучения, служащие для представления учебной информации большой аудитории: – компьютерная техника (ноутбук/компьютер); – проектор; – экран; – персональные компьютеры, подключенные к сети Интернет.
Помещения для самостоятельной работы обучающихся	**Оснащенность помещений для самостоятельной работы обучающихся**
читальный зал библиотеки:	– компьютерная техника, подключение к сети «Интернет»

Материально-техническое обеспечение учебной дисциплины при обучении с использованием электронного обучения и дистанционных образовательных технологий.

Необходимое оборудование	Параметры	Технические требования
Персональный компьютер/ ноутбук/планшет, камера, микрофон, динамики, доступ в сеть Интернет	Веб-браузер	Версия программного обеспечения не ниже: Chrome 72, Opera 59, Firefox 66, Edge 79, Яндекс.Браузер 19.3
	Операционная система	Версия программного обеспечения не ниже: Windows 7, macOS 10.12 «Sierra», Linux
	Веб-камера	640x480, 15 кадров/с
	Микрофон	любой
	Динамики (колонки или наушники)	любые
	Сеть (интернет)	Постоянная скорость не менее 192 кБит/с

Технологическое обеспечение реализации программы осуществляется с использованием элементов электронной информационно-образовательной среды университета.

10. УЧЕБНО-МЕТОДИЧЕСКОЕ И ИНФОРМАЦИОННОЕ ОБЕСПЕЧЕНИЕ УЧЕБНОЙ ДИСЦИПЛИНЫ

№ п/п	Автор(ы)	Наименование издания	Вид издания (учебник, УП, МП и др.)	Издательство	Год издания	Адрес сайта ЭБС или электронного ресурса *(заполняется для изданий в электронном виде)*	Количество экземпляров в библиотеке Университета
10.1 Основная литература, в том числе электронные издания							
1	Синаторов С.В.	Информационные технологии	Учебное пособие	М.: Флинта	2021	https://znanium.com/catalog/document?id=374932	-
2	Черткова, Е. А.	Программная инженерия. Визуальное моделирование программных систем	Учебник для вузов	Москва : Издательство Юрайт	2024	https://urait.ru/bcode/534516	-
3	Лаврищева, Е. М.	Программная инженерия. Парадигмы, технологии и CASE-средства	Учебник для вузов	Москва : Издательство Юрайт	2024	https://urait.ru/bcode/537884	-
4	Немцова Т.И., Голова С.Ю., Терентьев А.И.; под ред. Л.Г. Гагариной.	Программирование на языке высокого уровня. Программирование на языке С++	Учебное пособие	М.: ИД ФОРУМ: ИНФРА-М	2024	https://znanium.ru/catalog/document?id=432187	-
10.2 Дополнительная литература, в том числе электронные издания							
1	Плотникова Н.Г.	Информатика и информационно-коммуникационные технологии (ИКТ)	Учебное пособие	М.: РИОР	2021	https://znanium.com/catalog/document?id=370445	-
2	Горбатов С.М., Тарасов Ю.С., Наумова М.Г.	Информационные технологии	Учебное пособие	М.: МИСиС	2016	https://znanium.com/catalog/document?id=371025	-
3	Федотова Е.Л.	Информационные технологии и системы	Учебное пособие	М.: Издательский Дом ФОРУМ	2022	https://znanium.com/catalog/document?id=386738	-
4	М. В. Огнева, Е.	Программирование на языке	Учебное	М.: Издательство	2024	https://urait.ru/bcode/555533	-

	В. Кудрина	C++: практический курс	пособие	Юрайт			
10.3 Методические материалы (указания, рекомендации по освоению дисциплины (модуля) авторов РГУ им. А. Н. Косыгина)							
-	-	-	-	-	-	-	-

11. ИНФОРМАЦИОННОЕ ОБЕСПЕЧЕНИЕ УЧЕБНОГО ПРОЦЕССА

11.1. Ресурсы электронной библиотеки, информационно-справочные системы и профессиональные базы данных:

№ пп	Электронные учебные издания, электронные образовательные ресурсы
1.	*ЭБС «Лань»* http://www.e.lanbook.com/
2.	*«Znanium.com» научно-издательского центра «Инфра-М»* http://znanium.com/
3.	*Электронные издания «РГУ им. А.Н. Косыгина» на платформе ЭБС «Znanium.com»* http://znanium.com/
4.	ЭБС «ИВИС» http://dlib.eastview.com/
5.	Образовательная платформа «ЮРАЙТ» https://urait.ru/
	Профессиональные базы данных, информационные справочные системы
1.	Scopus https://www.scopus.com (международная универсальная реферативная база данных, индексирующая более 21 тыс. наименований научно-технических, гуманитарных и медицинских журналов, материалов конференций примерно 5000 международных издательств);
2.	Научная электронная библиотека eLIBRARY.RU https://elibrary.ru (крупнейший российский информационный портал в области науки, технологии, медицины и образования);
3.	База данных в мире Academic Search Complete - обширная полнотекстовая научно-исследовательская. Содержит полные тексты тысяч рецензируемых научных журналов по химии, машиностроению, физике, биологии. http://search.ebscohost.com

11.2. Перечень программного обеспечения

№п/п	Программное обеспечение	Реквизиты подтверждающего документа/ Свободно распространяемое
1.	Windows 10 Pro, MS Office 2019	контракт № 18-ЭА-44-19 от 20.05.2019
2.	Microsoft Visual Studio	контракт № 18-ЭА-44-19 от 20.05.2019
3.	Embarcadero C++Builder RAD Studio Professional Academic Concurrent License	№ 15-02.01-2459 от 21.12.2021 Embarcadero License Certificate: #546431, #546432, #546433, #546434, #546435
4.	Code::Blocks — свободная кроссплатформенная среда разработки	Свободно распространяемое на условиях GNU General Public License v.3.
5.	Visual Studio Community	Свободно распространяемая среда разработки.
6.	Visual Studio Code	Свободно распространяемая среда разработки.

ЛИСТ УЧЕТА ОБНОВЛЕНИЙ РАБОЧЕЙ ПРОГРАММЫ УЧЕБНОЙ ДИСЦИПЛИНЫ

В рабочую программу учебной дисциплины внесены изменения/обновления и утверждены на заседании кафедры:

№ пп	год обновления РПД	характер изменений/обновлений с указанием раздела	номер протокола и дата заседания кафедры