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 Институт информационных технологий и цифровой трансформации
Кафедра Информационных технологий

РАБОЧАЯ ПРОГРАММА УЧЕБНОЙ ДИСЦИПЛИНЫ

Технологии обработки информации

Уровень образования	бакалавриат
Направление подготовки	09.03.02 Информационные системы и технологии
Профиль	Информационные технологии в цифровых системах управления производством
Срок освоения образовательной программы по очной форме обучения	4 года
Форма обучения	очная

Рабочая программа учебной дисциплины «Технологии обработки информации» основной профессиональной образовательной программы высшего образования, рассмотрена и одобрена на заседании кафедры, протокол № 7 от 28.02.2023 г.

Разработчик рабочей программы «Технологии обработки информации»:

канд. техн. наук, доцент А. А. Семенов

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

1. ОБЩИЕ СВЕДЕНИЯ

Учебная дисциплина «Технологии обработки информации» изучается в четвертом семестре.
Курсовая работа/Курсовой проект – предусмотрена в 4 семестре.

1.1. Формы промежуточной аттестации:

четвертый семестр - экзамен

1.2. Место учебной дисциплины в структуре ОПОП

Учебная дисциплина «Технологии обработки информации» относится к части, формируемой участниками образовательных отношений.
Основой для освоения дисциплины являются результаты обучения по предшествующим дисциплинам и практикам:

- Введение в профессию;
- Информатика;
- Технология программирования.

Результаты обучения по учебной дисциплине, используются при изучении следующих дисциплин:

- Компьютерная геометрия и графика;
- Управление данными;
- Инфокоммуникационные системы и сети.

2. ЦЕЛИ И ПЛАНИРУЕМЫЕ РЕЗУЛЬТАТЫ ОБУЧЕНИЯ ПО ДИСЦИПЛИНЕ

Целями изучения дисциплины «Технологии обработки информации» 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 информационных и коммуникационных технологий в управлении производством;
- изучение методов построения алгоритмов и основных этапов разработки и создания современных программных продуктов;
- освоение подходов к построению рациональных диалоговых интерфейсов, ориентированных на пользователя;
- изучение базовых правил и принципов современного объектно-ориентированного и визуального программирования;
- изучение методов поиска, анализа и обработки информации;
- 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-3 Способен разрабатывать компоненты системных программных продуктов	ИД-ПК-3.2 Использование современных языков программирования для разработки программных модулей	- Оценивает сущность и значение информационных технологий в развитии современного общества; владеет основными методами, способами и средствами получения, хранения и обработки информации; - Использует навыки разработки программных модулей для обработки информации с соблюдением основных требований к отладке и тестированию; - Применяет методики использования программных средств для решения практических задач в профессиональной деятельности; - Умеет разрабатывать современные эффективные интерфейсы «человек - электронно-вычислительная машина»; - Рационально оценивает и обосновывает принимаемые проектные решения, составляет блок-схемы алгоритмов и реализует их на языках программирования высокого уровня; - Осуществляет практическую реализацию информационно-коммуникационных программ и компьютерных приложений, выполняет тестовые примеры для проверки их корректности и эффективности.
	ИД-ПК-3.3 Использование современных информационных технологий для отладки и тестирования программных модулей	

3. СТРУКТУРА И СОДЕРЖАНИЕ УЧЕБНОЙ ДИСЦИПЛИНЫ

Общая трудоёмкость учебной дисциплины по учебному плану составляет:

по очной форме обучения –	5	**з.е.**	180	**час.**

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

Структура и объем дисциплины									
Объем дисциплины по семестрам	**форма промежуточной аттестации**	**всего, 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**

4 семестр	экзамен, курсовая работа	180	28		54	10	18	34	36
Всего:		180	28		54	10	18	34	36

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		
	Четвертый семестр						
ПК-3: ИД-ПК-3.2 ИД-ПК-3.3	**Раздел I. Интегрированная среда разработки и технологии программирования**	4	x	4	1	4	
	Лекция 1.1. Интегрированная среда разработки как инструмент для создания приложений. Технологии программирования для обработки информации	4					Контроль посещаемости.
	Лабораторная работа № 1.1. Графика. Структура Figura {};			4	1		Выполнение лабораторной работы.
ПК-3: ИД-ПК-3.2 ИД-ПК-3.3	**Раздел II. Структуры**	3	x	4	1	4	
	Лекция 2.1. Структуры в C++	3					Контроль посещаемости.
	Лабораторная работа № 2.1. Сортировки Яндекса			4	1		Выполнение лабораторной работы.
ПК-3: ИД-ПК-3.2 ИД-ПК-3.3	**Раздел III. Динамические структуры данных**	3	x	4	1	4	
	Лекция 3.1. Динамические структуры данных в C++	3					Контроль посещаемости.
	Лабораторная работа № 3.1. Визуализация сортировки Яндекса			4	1		Выполнение лабораторной работы.
ПК-3: ИД-ПК-3.2 ИД-ПК-3.3	**Раздел IV. Классы**	4	x	8	1	6	
	Лекция 4.1. Классы в C++	4					Контроль посещаемости.
	Лабораторная работа № 4.1. Классы. Инкапсуляция. Реализация класса MyTime			4			
	Лабораторная работа № 4.2. Классы. Инкапсуляция. Реализация класса PayRoad			4	1		Выполнение лабораторной работы.

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		
ПК-3: ИД-ПК-3.2 ИД-ПК-3.3	**Раздел V. Массивы и классы**	3	x	4	1	4	
	Лекция 5.1. Массивы и классы в C++	3					Контроль посещаемости.
	Лабораторная работа № 5.1. Классы. Операторные функции. Реализация класса MyInt для перегрузки операций			4	1		Выполнение лабораторной работы.
ПК-3: ИД-ПК-3.2 ИД-ПК-3.3	**Раздел VI. Перегрузка операций**	3	x	10	2	4	
	Лекция 6.1. Перегрузка операций	3					Контроль посещаемости.
	Лабораторная работа № 6.1. Классы. Приведение типов. Реализация класса Stroka			4	1		
	Лабораторная работа № 6.2. Классы. Реализация класса авиарейсов			6	1		Выполнение лабораторной работы.
ПК-3: ИД-ПК-3.2 ИД-ПК-3.3	**Раздел VII. Наследование**	4	x	14	2	4	
	Лекция 7.1. Наследование в C++	4					Контроль посещаемости.
	Лабораторная работа № 7.1. Классы. Наследование			4			
	Лабораторная работа № 7.2. Классы. Наследование при изначальной разработке программы			4	1		
	Лабораторная работа № 7.3. Классы. Наследование и отработка взаимодействия с компонентом StringGrid			6	1		Выполнение лабораторной работы.
ПК-3:	**Раздел VIII. Виртуальные и дружественные функции**	4	x	6	1	4	

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-3.2 ИД-ПК-3.3	Лекция 8.1. Виртуальные и дружественные функции в C++	4					Контроль посещаемости.
	Лабораторная работа № 8.1. Классы. Наследование и виртуальные методы			6	1		Выполнение лабораторной работы.
	Выполнение курсовой работы	x	x	x	x	18	Защита курсовой работы
	Экзамен	x	x	x	x	36	Электронное тестирование.
	ИТОГО за четвертый семестр	**28**		**54**	**10**	**88**	**Экзамен**
	ИТОГО за весь период	**28**		**54**	**10**	**88**	

3.3. Краткое содержание учебной дисциплины

№ пп	Наименование раздела и темы дисциплины	Содержание раздела (темы)
Четвертый семестр		
Раздел I	**Интегрированная среда разработки и технологии программирования**	
Лекция 1.1	Интегрированная среда разработки как инструмент для создания приложений. Технологии программирования для обработки информации.	Введение. Интегрированная среда разработки как инструмент для создания приложений. Характеристика основных технологий программирования. Структуры данных. Правила кодирования, документирования и основные этапы создания программного обеспечения.
Лабораторная работа № 1.1	Графика. Структура Figura {};	Лабораторная работа посвящена изучению взаимодействия с геометрическими примитивами и отработке структур C++. В рамках данной работы реализуется визуальное приложение согласно представленному интерфейсу.
Раздел II	**Структуры**	
Лекция 2.1	Структуры в C++.	Структуры, структуры и функции, массивы структур, поиск в массиве структур, вложенность структур, рекурсия, алгоритм быстрой сортировки, массивы структур и бинарные файлы.
Лабораторная работа № 2.1	Сортировки Яндекса.	Лабораторная работа посвящена изучению основных алгоритмов сортировок и способов их реализации на примере Яндекса (см. https://academy.yandex.ru/posts/osnovnye-vidy-sortirovok-i-primery-ikh-realizatsii). Лабораторную работу необходимо реализовать в виде визуального приложения. Нужно изучить представленный иллюстрированный пример реализации на C++ основных видов сортировок. Необходимо воспроизвести реализацию проекта и по аналогии доделать пирамидальную сортировку и сортировку слиянием. Выполнить оптимизацию кода и абстракцию.
Раздел III	**Динамические структуры данных**	
Лекция 3.1	Динамические структуры данных в C++	Понятие и предназначение динамических структур данных (ДСД). Характеристика и синтаксис таких ДСД, как линейные списки, стеки, очереди и бинарные деревья. Возможные области применения и операции над ДСД.
Лабораторная работа № 3.1	Визуализация сортировки Яндекса	Лабораторная работа посвящена визуализации алгоритмов сортировки Яндекса (см. https://academy.yandex.ru/posts/osnovnye-vidy-sortirovok-i-primery-ikh-realizatsii). Лабораторную работу необходимо выполнить в виде визуального приложения. Разрабатывается визуальное приложение, позволяющее визуализировать процесс сортировки целочисленного массива из 12 случайных элементов с анимацией. Допускается использовать подходящий стандартный компонент, либо конструкцию из прямоугольников. Визуализируемую сортировку пользователь выбирает в списке (компонент ComboBox). Замедление просто реализовать с помощью связки sleep() + ProcessMessages().
Раздел IV	**Классы**	
Лекция	Классы в C++	Основные свойства ООП (инкапсуляция, наследование и

4.1		полиморфизм). Понятие и элементы класса. Отличие структур от классов. Синтаксис описания класса. Спецификаторы доступа. Методы класса, отличие методов от функций. Доступ к методам класса. Конструкторы. Определение методов класса вне класса. Методы, возвращающие значения. Классы и память. Статические данные класса. Константные методы. Деструкторы.
Лабораторная работа № 4.1	Классы. Инкапсуляция. Реализация класса MyTime	Лабораторная работа посвящена изучению классов в ООП на C++. Требуется: 1) Изучить и повторить представленный иллюстрированный пример, 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). 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.

Лабораторная работа № 4.2	Классы. Инкапсуляция. Реализация класса PayRoad	Лабораторная работа посвящена изучению классов в ООП на C++. Необходимо изучить и повторить представленный иллюстрированный 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.

Раздел V	**Массивы и классы**	
Лекция	Массивы и классы в C++	Массивы и классы. Обработка массивов объектов.

5.1		Совмещение объектно-ориентированной технологии и процедурной. Строковый класс string и его характеристика. Строковый тип AnsiString и его характеристика.
Лабораторная работа № 5.1	Классы. Операторные функции. Реализация класса MyInt для перегрузки операций	Лабораторная работа посвящена изучению классов в ООП на C++. Нужно изучить и повторить проиллюстрированный 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. В окна LabeledEdit пользователь вводит числа и нажимает кнопки операций, результат выводится в Memo методом класса show(). Повторив рассмотренный пример, необходимо самостоятельно доделать реализацию кнопок -, *, / (обработать ситуацию деления на ноль), ==.
Раздел VI	**Перегрузка операций**	
Лекция 6.1	Перегрузка операций	Синтаксис и принцип работы операторных функций. Правила перегрузки. Перегрузка унарных, арифметических операций, операций сравнения. Перегрузка операции приведения типа. Преобразование объектов в основные типы и наоборот. Преобразование объектов одного класса в объекты другого класса.
Лабораторная работа № 6.1	Классы. Приведение типов. Реализация класса Stroka	Лабораторная работа посвящена изучению классов в ООП на C++. 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 визуальное приложение (Windows VCL Application) для проверки этого класса.
Лабораторная работа № 6.2	Классы. Реализация класса авиарейсов	Лабораторная работа посвящена изучению классов в ООП на C++. Необходимо изучить и повторить проиллюстрированный пример, демонстрирующий реализацию класса Reis, который положен в основу реализации списка записей авиарейсов. В примере разрабатывается визуальное приложение (Windows VCL Application), в котором необходимо создать класс с именем Reis, содержащий поля: char name[N]; char type[N]; int kol; float price;

		В приложении реализовываются возможности добавления записей в таблицу StringGrid, записи их в текстовый файл, считывания оттуда и сортировки рейсов. Здесь в окна LabeledEdit пользователь вводит наименование рейса, тип самолета, кол-во билетов, цену одного билета и нажимает кнопки операций. Такие компоненты, как SaveDialog и OpenDialog настройте как для текстовых файлов (DefaultExt: txt; Filter: Текстовые файлы *.txt). Остальные компоненты можно особо не настраивать и оставить настройки по умолчанию. Для StringGrid изменение настроек делается программно в коде. Повторив рассмотренный пример, необходимо самостоятельно исправить самую основную недоработку этой программы. Дело в том, что приложение не оптимизировано для ввода наименований рейсов и типов самолетов с пробелами. Если вводить их с пробелами, то процедуры считывания из файла и сортировки работают некорректно. Также в программе имеется ошибка, которая не позволяет считать последнюю строку из файла и вывести её в StringGrid. Эту ошибку надо тоже исправить.
Раздел VII	**Наследование**	
Лекция 7.1	Наследование в C++	Понятие и предназначение наследования. Синтаксис наследования. Спецификаторы доступа при наследовании. Конструкторы производных классов. Перегрузка функций. Применение наследования при первоначальной разработке объектно-ориентированной программы. Иерархия классов. Общее и частное наследование, комбинации доступа. Множественное наследование. Включение, классы в классах.
Лабораторная работа № 7.1	Классы. Наследование	Лабораторная работа посвящена изучению классов в ООП на C++. Требуется изучить и повторить проиллюстрированный 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.
Лабораторная работа № 7.2	Классы. Наследование при изначальной разработке программы	Лабораторная работа посвящена изучению классов в ООП на C++. 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 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.
Лабораторная работа № 7.3	Классы. Наследование и отработка взаимодействия с компонентом StringGrid	Лабораторная работа посвящена изучению классов в C++. Необходимо изучить и повторить проиллюстрированный пример, демонстрирующий применение технологии наследования при изначальном проектировании приложения. Взяв за основу проект из предыдущей лабораторной работы, мы создадим его улучшенную версию, которая позволит добавлять товары не в компонент Memo, а в компонент StringGrid. Также реализуем сохранение данных в текстовый файл и считывание их из него в компонент StringGrid. Поставленную задачу будем решать с применением технологии наследования при изначальной разработке приложения. Создадим базовый класс и распишем производные классы. 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 StirngGrid1. В обработчик события нажатия кнопки Button3 запишем логику подсчета общей стоимости добавленных товаров. Далее, создадим обработчик события OnCreate для формы и обработчики нажатия кнопок для сохранения данных в текстовый файл и их загрузки из файла. В последней процедуре Button5Click, создается объект SL, у которого тип данных - StringList. Это строковый список. Удобная конструкция для построчного считывания из файла и заполнения строк таблицы StrinGrid этими данными. Реализовав рассмотренный пример, необходимо добавить возможность правильной работы кнопки "Общая стоимость", если данные не добавлялись вручную, а были загружены из файла. Сейчас же, если запустить программу и загрузить данные из файла кнопкой "Загрузить", то

		программа выдаст сообщение "Нет товаров!". Это верно, так как при заполнении табличной части компонента StringGrid, мы не записываем данные в массивы товаров, соответственно массивы остаются пустыми. Это надо исправить. Дополните процедуру Button5Click и создайте дополнительный метод класса, позволяющий добавлять данные из StringGrid. То есть, можно сделать по аналогии с методом, который добавляет записи из эдитов:
		void dobav_zap() { // метод для добавления записей
		if(VidTovara==0) {
		strcpy(name, AnsiString(Form1->LabeledEdit1->Text).c_str());
		number = StrToInt(Form1->LabeledEdit2->Text);
		price = StrToFloat(Form1->LabeledEdit3->Text);
		}
		if(VidTovara==1) {
		strcpy(name, AnsiString(Form1->LabeledEdit6->Text).c_str());
		number = StrToInt(Form1->LabeledEdit7->Text);
		price = StrToFloat(Form1->LabeledEdit8->Text);
		}
		}
		
		Необходимо разработать метод, в котором они добавлялись бы из StringGrid.

Раздел VIII	**Виртуальные и дружественные функции**	
Лекция 8.1	Виртуальные и дружественные функции в C++	Предназначение и синтаксис виртуальных функций. Указатели на базовый класс. Наследование и массивы указателей на базовый класс, применение таких массивов. Абстрактные классы и чистые виртуальные функции. Виртуальные деструкторы. Виртуальные базовые классы и устранение неоднозначности при множественном наследовании. Предназначение и синтаксис дружественных функций. Применение процедурной технологии создания программ в объектно-ориентированной программе. Дружественные классы. Указатель this.
Лабораторная работа № 8.1	Классы. Наследование и виртуальные методы	Лабораторная работа посвящена изучению классов в ООП на C++. Необходимо изучить и повторить пример, демонстрирующий применение технологии наследования с виртуальными методами. Взяв за основу проект из предыдущей лабораторной работы, мы создадим его улучшенную версию, которая позволит работать не с двумя массивами, а с одним универсальным, который может хранить данные как продуктовых товаров, так и промышленных. Также реализуем сохранение данных в текстовый файл и считывание их из него в компонент StringGrid (по аналогии с предыдущей л/р). Также реализуем нормальное функционирование кнопки "Общая стоимость", которая сможет рассчитывать общую стоимость как добавленных вручную товаров, так и загруженных из файла. Реализовав рассмотренный пример, необходимо добавить возможности: а) сортировки товаров по наименованию и цене; б) поиска (по всем свойствам товаров).

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;
– подготовка к практическим занятиям*;*
– подготовка к компьютерному тестированию на промежуточных аттестациях;
– выполнение индивидуальных заданий;
– выполнение курсовых работ;
– подготовка к промежуточной аттестации в течение семестра.

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	**Интегрированная среда разработки и технологии программирования**			
Лаборатор	Графика. Структура	Изучение научной и технической	Выполнение	4

ная работа № 1.1	Figura {};	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.	лабораторной работы.	
Раздел II	Структуры			
Лабораторная работа № 2.1	Сортировки Яндекса	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	**Динамические структуры данных**			
Лабораторная работа № 3.1	Визуализация сортировки Яндекса	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 IV	**Классы**			
Лабораторная работа № 4.1	Классы. Инкапсуляция. Реализация класса MyTime.	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	Выполнение лабораторной работы.	3

		разработки программ и операторов языка для выполнения задания лабораторной работы.		
Лабораторная работа № 4.2	Классы. Инкапсуляция. Реализация класса PayRoad	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.	3
Раздел V	**Массивы и классы**			
Лабораторная работа № 5.1	Классы. Операторные функции. Реализация класса MyInt 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.	4
Раздел VI	**Перегрузка операций**			
Лабораторная работа № 6.1	Классы. Приведение типов. Реализация класса Stroka	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.	2
Лабораторная работа № 6.2	Классы. Реализация класса авиарейсов	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	Выполнение лабораторной работы.	2

		визуального программирования. Изучение элементов системы разработки программ и операторов языка для выполнения задания лабораторной работы.		
Раздел VII	**Наследование**			
Лабораторная работа № 7.1	Классы. Наследование	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.	1
Лабораторная работа № 7.2	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.	2
Лабораторная работа № 7.3	Классы. Наследование и отработка взаимодействия с компонентом StringGrid	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.	1
Раздел VIII	**Виртуальные и дружественные функции**			
Лабораторная работа № 8.1	Классы. Наследование и виртуальные методы	Изучение научной и технической литературы, нормативных документов, стандартов языков программирования. Работа с материалами конспекта лекций. Анализ задания к лабораторной	Выполнение лабораторной работы.	4

		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.		

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	28	в соответствии с расписанием учебных занятий
	лабораторные занятия	54	

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

4.1. Соотнесение планируемых результатов обучения с уровнями сформированности компетенций.

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-3: ИД-ПК-3.2 ИД-ПК-3.3
высокий		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; – 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		хорошо/ зачтено (хорошо)/ зачтено			Обучающийся: – достаточно подробно, грамотно и по существу излагает изученный материал, приводит и раскрывает в тезисной форме основные понятия; – 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		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 отдельные элементы визуальной разработки прикладных программ; – анализирует современные технологии программирования с неточностями и ошибками; – демонстрирует

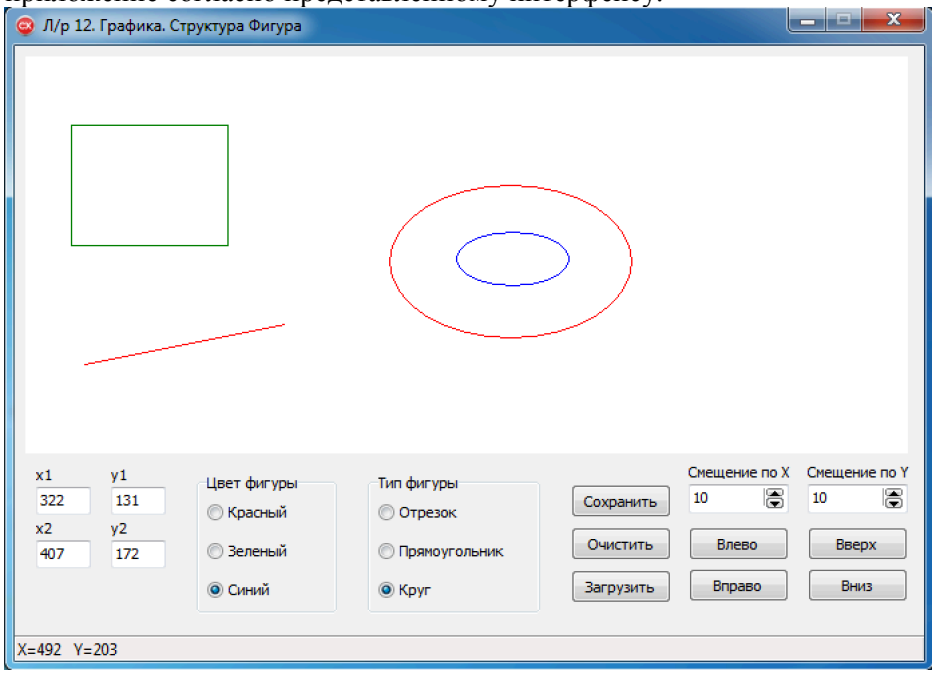
					фрагментарные знания основной учебной литературы по дисциплине; – ответ отражает знания на базовом уровне теоретического и практического материала в объеме, необходимом для дальнейшей учебы и предстоящей работы по профилю обучения.
низкий		неудовлетворительно/ не зачтено	Обучающийся: – демонстрирует 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.		

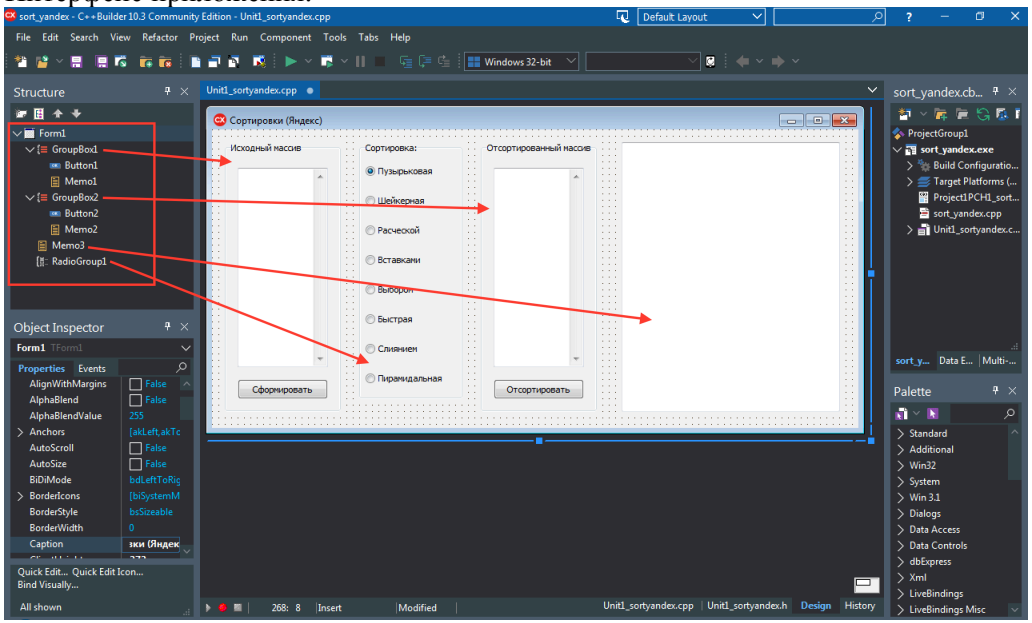
5. ОЦЕНОЧНЫЕ СРЕДСТВА ДЛЯ ТЕКУЩЕГО КОНТРОЛЯ УСПЕВАЕМОСТИ И ПРОМЕЖУТОЧНОЙ АТТЕСТАЦИИ, ВКЛЮЧАЯ САМОСТОЯТЕЛЬНУЮ РАБОТУ ОБУЧАЮЩИХСЯ

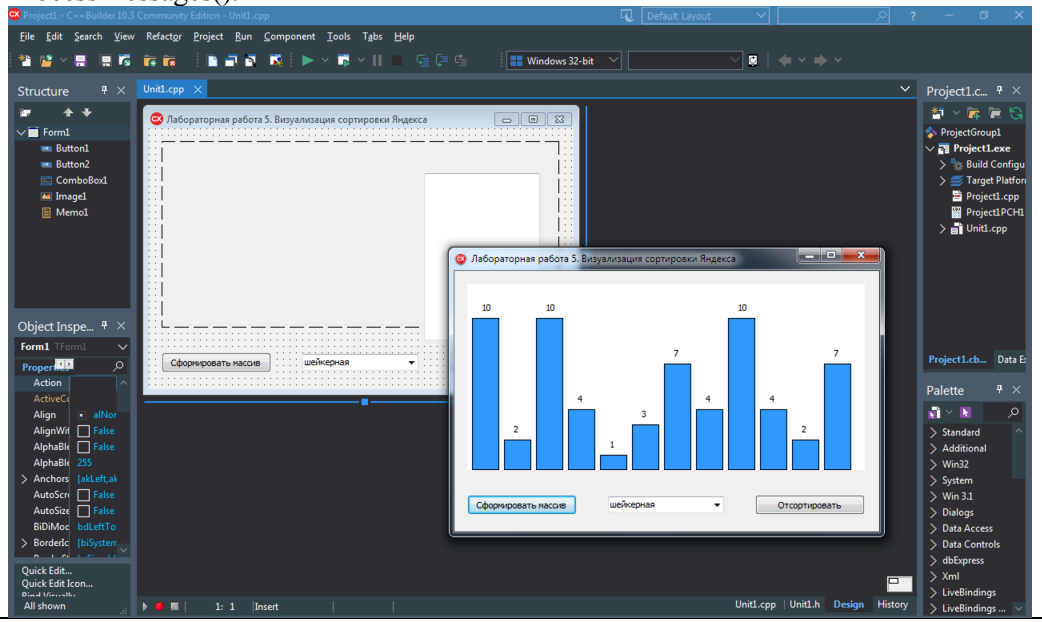
При проведении контроля самостоятельной работы обучающихся, текущего контроля и промежуточной аттестации по учебной дисциплине «Технологии обработки информации» проверяется уровень сформированности у обучающихся компетенций и запланированных результатов обучения по дисциплине, указанных в разделе 2 настоящей программы.

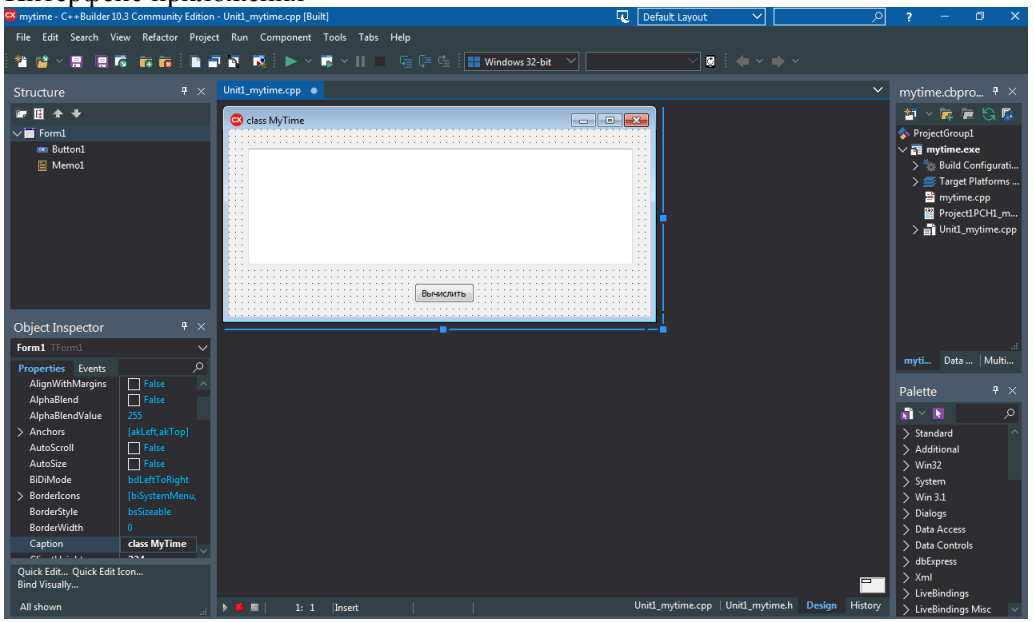
5.1. Формы текущего контроля успеваемости, примеры типовых заданий:

№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция
Лабораторная работа № 1.1	Выполнение лабораторной	Графика. Структура Figura {};. Лабораторная работа посвящена изучению взаимодействия с геометрическими	ПК-3: ИД-ПК-3.2

№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция
	работы.	примитивами и отработке структур C++. В рамках данной работы реализуется визуальное приложение согласно представленному интерфейсу. 	ИД-ПК-3.3
Лабораторная работа № 2.1	Выполнение лабораторной работы.	Сортировки Яндекса. Лабораторная работа посвящена изучению основных алгоритмов сортировок и способов их реализации на примере Яндекса (см. https://academy.yandex.ru/posts/osnovnye-vidy-sortirovok-i-primery-ikh-realizatsii). Лабораторную работу необходимо реализовать в виде визуального приложения. Нужно изучить представленный иллюстрированный пример реализации на C++ основных видов сортировок. Пример реализации на C++ основных видов сортировок: 1) Кнопка 1 формирует случайным образом массив целых чисел из 20000 элементов и выводит, например, в MEMO1. 2) НА ФОРМЕ ИМЕЕТСЯ радиогруп (8 радиоточек для каждой из сортировок).	ПК-3: ИД-ПК-3.2 ИД-ПК-3.3

№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция
		3) Кнопка 2 - сортирует массив случайных чисел (вызывается функция для выбранной пользователем сортировки). 4) Выводим отсортированный массив и время в миллисекундах, затраченное в ходе сортировки. Интерфейс приложения:  Необходимо воспроизвести реализацию проекта и по аналогии доделать пирамидальную сортировку и сортировку слиянием. Выполнить оптимизацию кода (для ускорения операций) и абстракцию.	
Лабораторная работа № 3.1	Выполнение лабораторной работы.	Визуализация сортировки Яндекса. Лабораторная работа посвящена визуализации алгоритмов сортировки Яндекса (см. https://academy.yandex.ru/posts/osnovnye-vidy-sortirovok-i-primery-ikh-realizatsii). Лабораторную работу необходимо выполнить в виде визуального приложения. Разрабатывается визуальное приложение, позволяющее визуализировать процесс сортировки целочисленного массива из 12 случайных элементов с анимацией. Допускается использовать подходящий стандартный компонент, либо конструкцию из	ПК-3: ИД-ПК-3.2 ИД-ПК-3.3

№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция
		прямоугольников. Визуализируемую сортировку пользователь выбирает в списке (компонент ComboBox). Замедление просто реализовать с помощью связки sleep() + ProcessMessages(). 	
Лабораторная работа № 4.1	Выполнение лабораторной работы.	Классы. Инкапсуляция. Реализация класса MyTime. Лабораторная работа посвящена изучению классов в ООП на C++. 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	ПК-3: ИД-ПК-3.2 ИД-ПК-3.3

№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция
		значение на экран (например, 13:23:50 + 10:52:50 = 0:16:40). Иллюстрированный пример разработки приложения Интерфейс приложения  Исходный код программы .h-файл: //--- #ifndef Unit1_mytimeH #define Unit1_mytimeH //--- #include <System.Classes.hpp>	

№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция

		#include <Vcl.Controls.hpp>
		#include <Vcl.StdCtrls.hpp>
		#include <Vcl.Forms.hpp>
		//---
		class TForm1 : public TForm
		{
		__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	

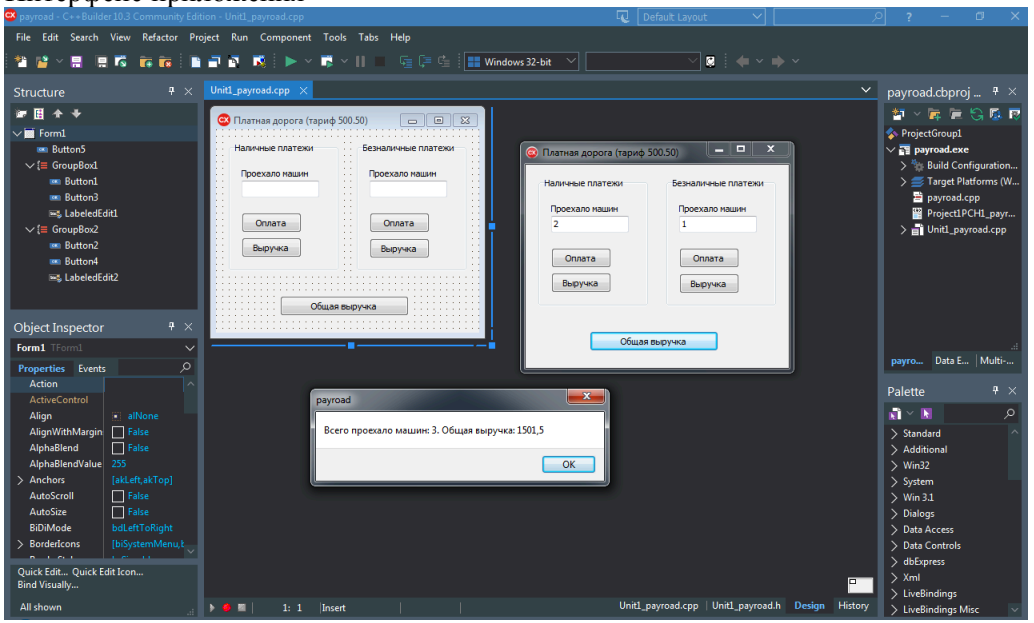
№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция

		.cpp-файл:
		
		//---
		#include <vcl.h>
		#pragma hdrstop
		
		#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)
		{
		}
		//---	

№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция

		void __fastcall TForm1::Button1Click(TObject *Sender)
		{
		Memo1->Clear();
		MyTime T1(13,23,50), T2(10,52,50), T3;
		T1.show(); T2.show();
		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.	

Лабораторная работа № 4.2	Выполнение лабораторной работы.	Классы. Инкапсуляция. Реализация класса PayRoad.
		Лабораторная работа посвящена изучению классов в ООП на С++. 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
			ПК-3:
			ИД-ПК-3.2
			ИД-ПК-3.3

№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция
		Интерфейс приложения  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>	

№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция

		#include <Vcl.Controls.hpp>
		#include <Vcl.StdCtrls.hpp>
		#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:	

№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция

```
        PayRoad() { Cars=0; Cash=0.0; NonCash=0.0; }
        void paying(int flag); // увеличиваем счетчик проехавших
        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;
```

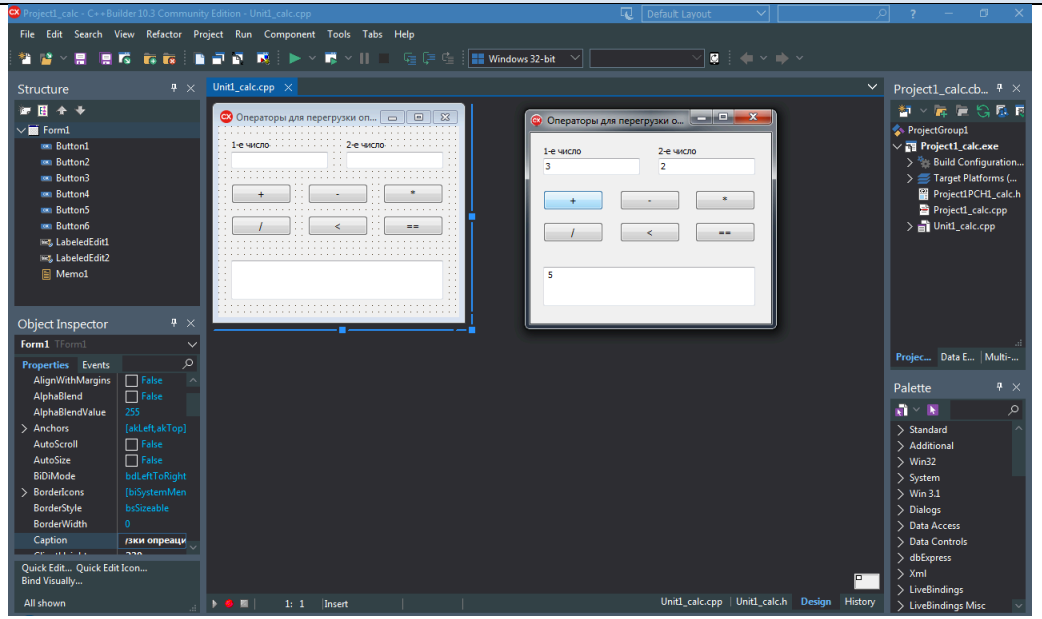
№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция

		if(flag==2) Form1->LabeledEdit2->Text = Cars;
		}
		
		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);
		}
		//---	

№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция

		void __fastcall TForm1::Button2Click(TObject *Sender)
		{
		  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);
		}	

Лабораторная работа № 5.1	Выполнение лабораторной работы.	Классы. Операторные функции. Реализация класса MyInt для перегрузки операций. Лабораторная работа посвящена изучению классов в ООП на C++. 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-3: ИД-ПК-3.2 ИД-ПК-3.3

№ 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	

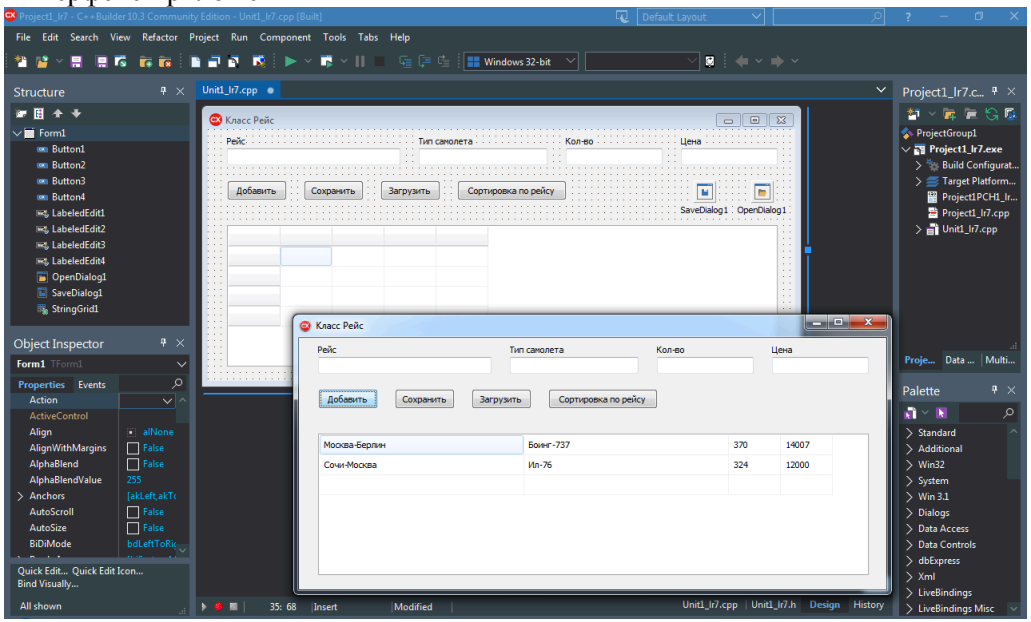
№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция

		#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.	

Лабораторная работа № 6.1	Выполнение лабораторной работы.	Классы. Приведение типов. Реализация класса Stroka. Лабораторная работа посвящена изучению классов в ООП на С++. 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-3: ИД-ПК-3.2 ИД-ПК-3.3
Лабораторная работа № 6.2	Выполнение лабораторной работы.	Классы. Реализация класса авиарейсов. Лабораторная работа посвящена изучению классов в ООП на С++. Изучить и повторить проиллюстрированный ниже пример, демонстрирующий реализацию класса Reis, который положен в основу реализации списка записей авиарейсов. В примере разрабатывается визуальное приложение (Windows VCL Application), в котором необходимо создать класс с именем Reis, содержащий поля: char name[N]; char type[N]; int kol;	ПК-3: ИД-ПК-3.2 ИД-ПК-3.3

№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция
		float price; В приложении реализовываются возможности добавления записей в таблицу StringGrid, записи их в текстовый файл, считывания оттуда и сортировки рейсов. Иллюстрированный пример разработки приложения Интерфейс приложения  Здесь в окна LabeledEdit пользователь вводит наименование рейса, тип самолета, кол-во билетов, цену одного билета и нажимает кнопки операций. Такие компоненты, как SaveDialog и OpenDialog настройте так, как мы всегда с вами делали для текстовых файлов (DefaultExt: txt; Filter: Текстовые файлы *.txt). Остальные компоненты можно особо не настраивать и оставить настройки по умолчанию. Для StringGrid изменение настроек делается программно в коде.	

№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция

		Исходный код программы
		
		В .h-файле ничего не прописываем.
		
		.cpp-файл:
		
		//---
		#include <vcl.h>
		#include <fstream.h>
		#pragma hdrstop
		
		#include "Unit1_lr7.h"
		//---
		#pragma package(smart_init)
		#pragma resource "*.dfm"
		TForm1 *Form1;
		
		int i = 0; // счетчик позиций (строк в таблице, записей в массиве)
		const int N = 255; // размер строк с наименованием рейса и типом самолета
		class Reis {
		  private:
		  char name[N];
		  char type[N];
		  int kol;
		  float price;
		  public:
		  void dobav_zap() { // метод для добавления записи
		  strcpy(name, AnsiString(Form1->LabeledEdit1->Text).c_str());
		  strcpy(type, AnsiString(Form1->LabeledEdit2->Text).c_str());
		  kol = StrToInt(Form1->LabeledEdit3->Text);
		  price = StrToFloat(Form1->LabeledEdit4->Text);
		  }	

№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция

		void show() { // метод для добавления в таблицу
		AnsiString s;
		s = AnsiString(name) + " " + AnsiString(type) + " " + IntToStr(kol)+ " " + FloatToStr(price);
		Form1->StringGrid1->Cells[0][i] = AnsiString(name);
		Form1->StringGrid1->Cells[1][i] = AnsiString(type);
		Form1->StringGrid1->Cells[2][i] = kol;
		Form1->StringGrid1->Cells[3][i] = price;
		}
		void clear() { // метод для очистки эдитов
		Form1->LabeledEdit1->Clear();
		Form1->LabeledEdit2->Clear();
		Form1->LabeledEdit3->Clear();
		Form1->LabeledEdit4->Clear();
		}
		} ;
		
		const int K = 100; // кол-во записей в массиве
		Reis* reis_samolet[K]; // массив класса
		//---
		__fastcall TForm1::TForm1(TComponent* Owner)
		: TForm(Owner)
		{
		}
		//---
		void __fastcall TForm1::Button1Click(TObject *Sender) // Добавить
		{
		reis_samolet[i] = new Reis;
		reis_samolet[i]->dobav_zap();
		reis_samolet[i]->show();
		reis_samolet[i]->clear();
		i++;
		Form1->StringGrid1->RowCount++; // добавим строчку в таблицу StringGrid	

№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция

		}
		//---
		
		void __fastcall TForm1::FormCreate(TObject *Sender) // при создании формы
		{
		  //отключим шапочные строку и столбец таблицы
		  Form1->StringGrid1->FixedCols = 0;
		  Form1->StringGrid1->FixedRows = 0;
		  //кол-во столбцов таблицы сделаем равным кол-ву полей класса
		  Form1->StringGrid1->ColCount = 4;
		  //кол-во строк таблицы сделаем равным одной
		  Form1->StringGrid1->RowCount = 1;
		  //зададим ширину первых двух столбцов
		  Form1->StringGrid1->ColWidths[0]=N;
		  Form1->StringGrid1->ColWidths[1]=N;
		
		}
		//---
		
		void __fastcall TForm1::Button2Click(TObject *Sender) // Сохранить
		{
		  if(SaveDialog1->Execute()) {
		  char *fn = AnsiString(SaveDialog1->FileName).c_str();
		  ofstream AllOrder(fn, ios::app); //создаем поток к файлу
		  // удобнее, конечно, работать с массивом reis_samolet, но
		  // отработаем взаимодействие со StringGrid
		  for(int e = 0; e < StringGrid1->RowCount; e++) {
		  StringGrid1->Rows[e]->Delimiter = ' ';
		  AnsiString s = StringGrid1->Rows[e]->DelimitedText;
		  char *st=s.c_str();
		  AllOrder << st << endl;
		  }
		  AllOrder.close();	

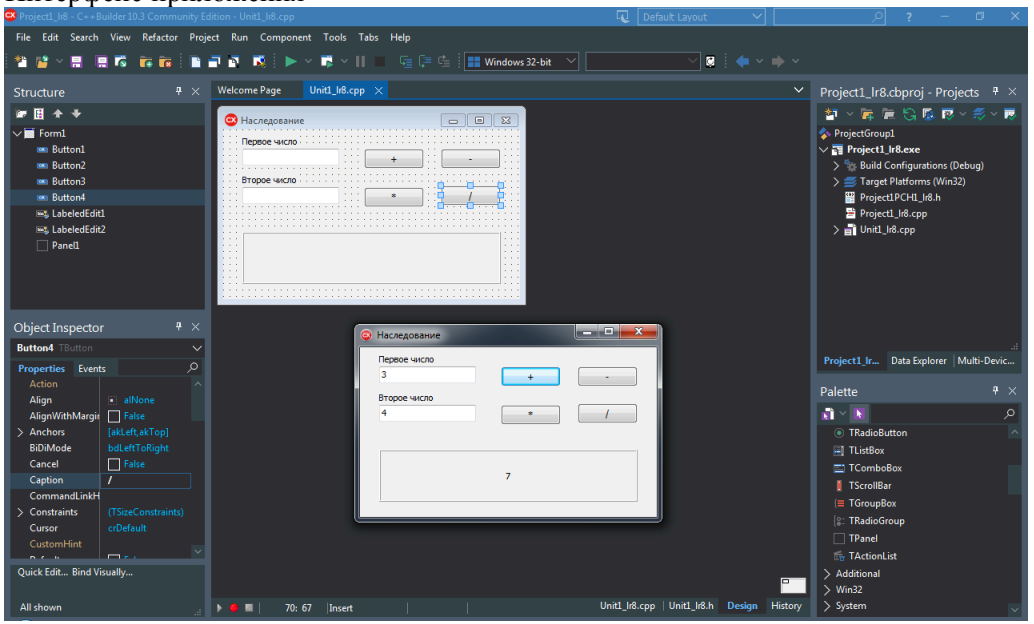
№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция

		}
		}
		//---
		
		void __fastcall TForm1::Button3Click(TObject *Sender) // Загрузить
		{
		  if(OpenDialog1->Execute()) {
		  // из выбранного пользователем файла заполняем данными StringGrid
		  int e=0;
		  char *fn = AnsiString(OpenDialog1->FileName).c_str();
		  ifstream fin2(fn);
		  char name_f[N];
		  char type_f[N];
		  int kol_f;
		  float price_f;
		  while(!fin2.eof()) {
		  fin2 >> name_f >> type_f >> kol_f >> price_f;
		  Form1->StringGrid1->Cells[0][e] = AnsiString(name_f);
		  Form1->StringGrid1->Cells[1][e] = AnsiString(type_f);
		  Form1->StringGrid1->Cells[2][e] = kol_f;
		  Form1->StringGrid1->Cells[3][e] = price_f;
		  e++;
		  }
		
		  fin2.close();
		  }
		}
		//---
		
		void __fastcall TForm1::Button4Click(TObject *Sender) // Сортировка по рейсу
		{
		  TStringList *SL = new TStringList;
		  AnsiString S;	

№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция

		int g=0;
		for (int i=0; i < StringGrid1->RowCount; i++){
		S="";
		//if(StringGrid1->Cells[0][i].IsEmpty()) ShowMessage(StringGrid1->Cells[0][i]);
		for (int j=0; j < StringGrid1->ColCount; j++) S += StringGrid1->Cells[j][i] + " ";
		//ShowMessage(S.Length());
		if(S.Length()!=4) { SL->Add(S); g++; }
		
		}
		SL->Sort();
		
		for (int i=0; i < g; i++){
		
		StringGrid1->Rows[i]->DelimitedText = SL->Strings[i];
		}
		delete SL;
		}
		//---
		
		Повторив рассмотренный пример, необходимо самостоятельно исправить самую основную недоработку этой программы. Дело в том, что приложение не оптимизировано для ввода наименований рейсов и типов самолетов с пробелами. Если вводить их с пробелами, то процедуры считывания из файла и сортировки работают некорректно.
		Также в программе имеется ошибка, которая не позволяет считать последнюю строку из файла и вывести её в StringGrid. Эту ошибку надо тоже исправить.	

Лабораторная работа № 7.1	Выполнение лабораторной работы.	Классы. Наследование. Лабораторная работа посвящена изучению классов в ООП на С++. 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 -, *, /.	ПК-3: ИД-ПК-3.2 ИД-ПК-3.3

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

№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция

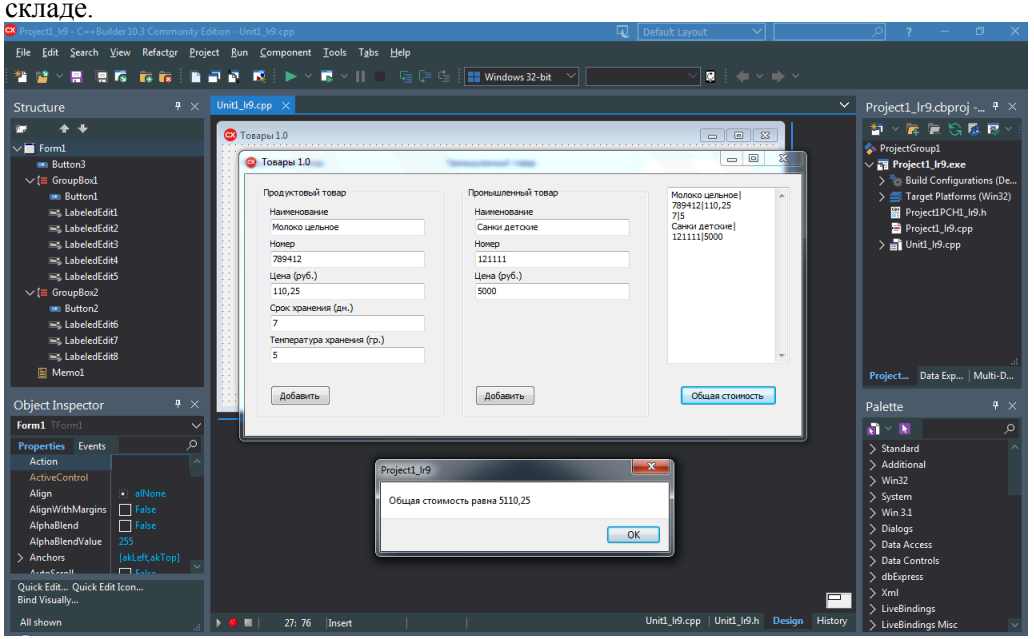
		#include "Unit1_lr8.h"
		//---
		#pragma package(smart_init)
		#pragma resource "*.dfm"
		TForm1 *Form1;
		
		class Float { // базовый класс
		  protected:
		  float f;
		  public:
		  Float() { f=0; }
		  Float(float d) { f=d; }
		  void show() { Form1->Panel1->Caption = f; }
		  Float operator+(Float 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)	

№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция

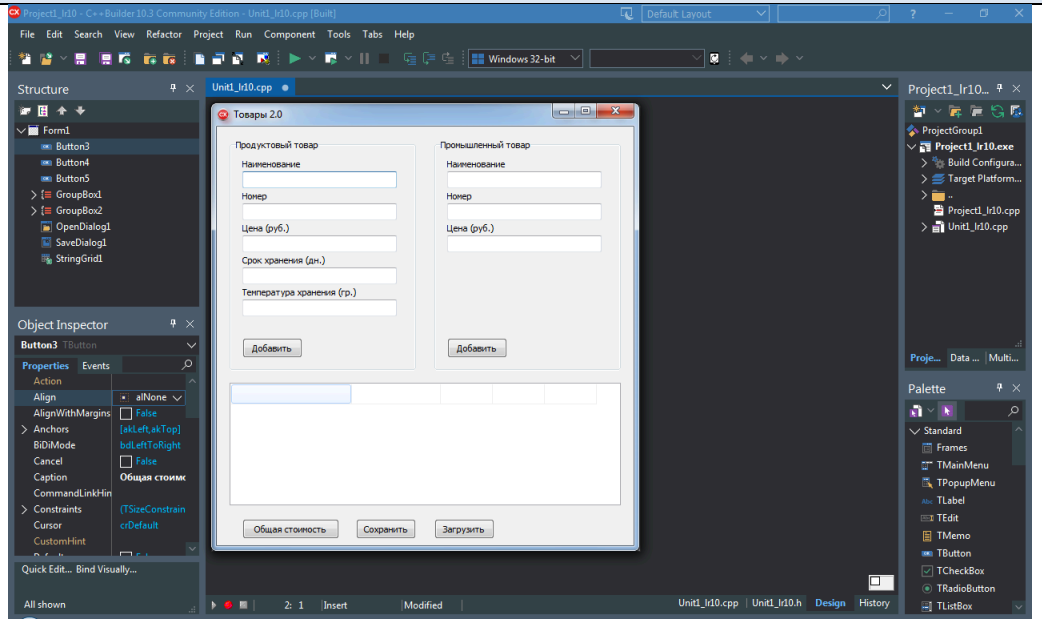
		{
		FloatPr f1, f2, f3;
		f1 = StrToFloat(LabeledEdit1->Text);
		f2 = StrToFloat(LabeledEdit2->Text);
		f3 = f1+f2; // вызывается метод базового класса operator+ и третий конструктор производного класса
		f3.show();
		}
		//---
		void __fastcall TForm1::Button2Click(TObject *Sender)
		{
		FloatPr f1,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

№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция

		f3.show();
		}
		Выполните абстракцию.	

Лабораторная работа № 7.2	Выполнение лабораторной работы.	Классы. Наследование при изначальной разработке программы. Лабораторная работа посвящена изучению классов в ООП на С++. 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. 	ПК-3: ИД-ПК-3.2 ИД-ПК-3.3

№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция
		Поставленную задачу удобно решать с применением технологии наследования при изначальной разработке приложения. По условию задачи, у нас две категории товаров, у 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.	
Лабораторная работа № 7.3	Выполнение лабораторной работы.	Классы. Наследование и отработка взаимодействия с компонентом StringGrid. Лабораторная работа посвящена изучению классов в C++. Необходимо изучить и повторить проиллюстрированный пример, демонстрирующий применение технологии наследования при изначальном проектировании приложения. Взяв за основу проект из предыдущей лабораторной работы, мы создадим его улучшенную версию, которая позволит добавлять товары не в компонент Memo, а в компонент StringGrid. Также реализуем сохранение данных в текстовый файл и считывание их из него в компонент StringGrid.	ПК-3: ИД-ПК-3.2 ИД-ПК-3.3

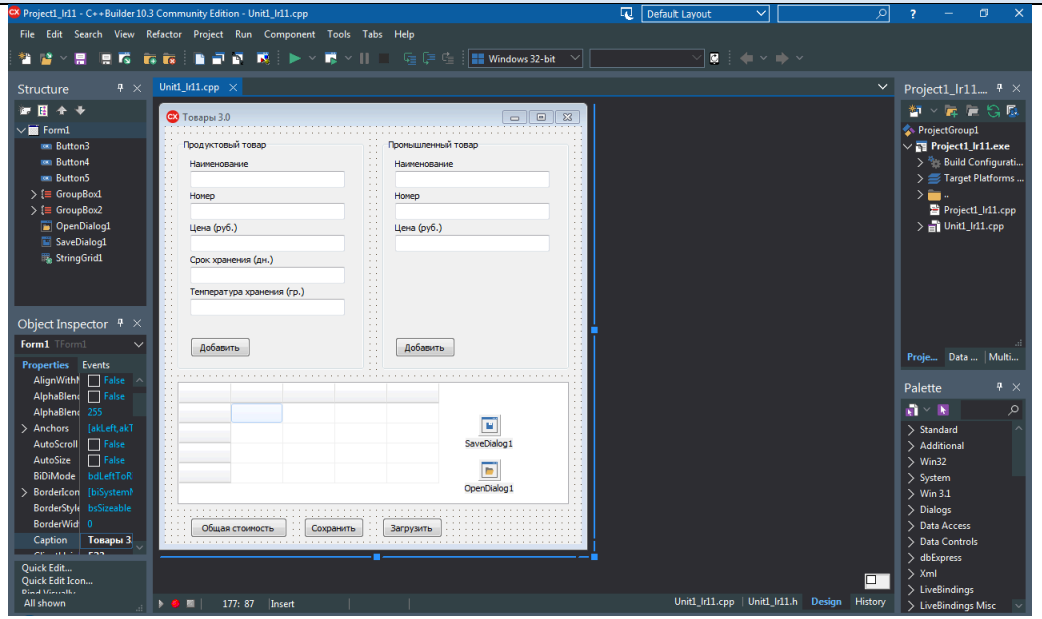
№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция
		 Поставленную задачу будем решать с применением технологии наследования при изначальной разработке приложения. Создадим базовый класс и распишем производные классы. Создадим массивы для работы с классами продуктовых и промышленных 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 StirngGrid1. В обработчик события нажатия кнопки Button3 запишем логику подсчета общей стоимости добавленных товаров. Далее, создадим обработчик события OnCreate для формы и обработчики нажатия кнопок для сохранения данных в текстовый файл и их загрузки из файла. В последней процедуре Button5Click, создается объект SL, у которого тип данных - StringList. Это строковый список. Удобная конструкция для построчного считывания из файла и заполнения строк таблицы StrinGrid этими данными. Реализовав рассмотренный пример, необходимо добавить возможность правильной работы кнопки "Общая стоимость", если данные не добавлялись вручную, а были загружены из файла. Сейчас же, если запустить программу и загрузить данные из файла кнопкой "Загрузить", то программа выдаст сообщение "Нет товаров!". Это верно,	

№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция

		так как при заполнении табличной части компонента StringGrid, мы не записываем данные в массивы товаров, соответственно массивы остаются пустыми. Это надо исправить. Дополните процедуру Button5Click и создайте дополнительный метод класса, позволяющий добавлять данные из StringGrid. То есть, можно сделать по аналогии с методом, который добавляет записи из эдитов:
		void dobav_zap() { // метод для добавления записей
		if(VidTovara==0) {
		strcpy(name, AnsiString(Form1->LabeledEdit1->Text).c_str());
		number = StrToInt(Form1->LabeledEdit2->Text);
		price = StrToFloat(Form1->LabeledEdit3->Text);
		}
		if(VidTovara==1) {
		strcpy(name, AnsiString(Form1->LabeledEdit6->Text).c_str());
		number = StrToInt(Form1->LabeledEdit7->Text);
		price = StrToFloat(Form1->LabeledEdit8->Text);
		}
		}
		
		Необходимо разработать метод, в котором они добавлялись бы из StringGrid.	

Лабораторная работа № 8.1	Выполнение лабораторной работы.	Классы. Наследование и виртуальные методы. Лабораторная работа посвящена изучению классов в ООП на C++. Необходимо изучить и повторить пример, демонстрирующий применение технологии наследования с виртуальными методами. Взяв за основу проект из предыдущей лабораторной работы, мы создадим его улучшенную версию, которая позволит работать не с двумя массивами, а с одним универсальным, который может хранить данные как продуктовых товаров, так и промышленных. Также реализуем сохранение данных в текстовый файл и считывание их из него в компонент StringGrid (по аналогии с предыдущей л/р). Также реализуем нормальное функционирование кнопки "Общая стоимость", которая сможет рассчитывать общую стоимость как добавленных вручную товаров, так и загруженных из файла. Разместим на форме необходимые для UI компоненты как на скриншоте ниже.	ПК-3: ИД-ПК-3.2 ИД-ПК-3.3

№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция

		
		В .h-файле пропишем классы (базовый и производные). Методы базового класса объявим виртуальными. Выдержка из лекции: "Виртуальные функции позволяют решать прямо в процессе выполнения программы, какую именно функцию вызывать. Виртуальные функции дают большую гибкость при выполнении одинаковых действий над разнородными объектами. В частности, они разрешают использование функций, вызванных из массива указателей на базовый класс, который на самом деле содержит указатели (или ссылки) на множество порождённых классов. Это пример полиморфизма." Переключимся на unit.cpp и распишем все методы и глобальные переменные, как в программном коде ниже.
		int VidTovara = 0; // 0-продуктовые, 1-промышленные
		int i=0, ij=0; // глобальные переменные счетчики
		void Tovar::dobav_zap() { // метод для добавления записей
		if(VidTovara==0) {
		strcpy(name, AnsiString(Form1->LabeledEdit1->Text).c_str());	

№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция

		number = StrToInt(Form1->LabeledEdit2->Text);
		price = StrToFloat(Form1->LabeledEdit3->Text);
		}
		if(VidTovara==1) {
		strcpy(name, AnsiString(Form1->LabeledEdit6->Text).c_str());
		number = StrToInt(Form1->LabeledEdit7->Text);
		price = StrToFloat(Form1->LabeledEdit8->Text);
		}
		}
		void Tovar::dobav_zap_iz_SG() {
		strcpy(name, AnsiString(Form1->StringGrid1->Cells[0][ij]).c_str());
		number = StrToInt(Form1->StringGrid1->Cells[1][ij]);
		price = StrToFloat(Form1->StringGrid1->Cells[2][ij]);
		}
		void Tovar::show() { // метод для показа записей в стринггриде
		Form1->StringGrid1->Cells[0][i] = AnsiString(name);
		Form1->StringGrid1->Cells[1][i] = number;
		Form1->StringGrid1->Cells[2][i] = price;
		}
		float Tovar::get_price() { // чтобы не использовать прямой доступ к полю price
		return price;
		}
		void Tovar::clear() { // метод очистки эдитов
		if(VidTovara == 0) {
		Form1->LabeledEdit1->Clear();
		Form1->LabeledEdit2->Clear();
		Form1->LabeledEdit3->Clear();
		}
		if(VidTovara == 1) {
		Form1->LabeledEdit6->Clear();
		Form1->LabeledEdit7->Clear();
		Form1->LabeledEdit8->Clear();
		}	

№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция

		}
		
		void TovarProd::dobav_zap() {
		Tovar::dobav_zap(); // вызов метода базового класса для первых трех полей
		srok = StrToInt(Form1->LabeledEdit4->Text);
		temp = StrToInt(Form1->LabeledEdit5->Text);
		}
		void TovarProd::dobav_zap_iz_SG() {
		Tovar::dobav_zap_iz_SG(); // вызов метода базового класса
		srok = StrToInt(Form1->StringGrid1->Cells[3][ij]);
		temp = StrToInt(Form1->StringGrid1->Cells[4][ij]);
		}
		void TovarProd::show() {
		Tovar::show(); // вызов метода базового класса для первых трех полей
		Form1->StringGrid1->Cells[3][i] = srok;
		Form1->StringGrid1->Cells[4][i] = temp;
		}
		void TovarProd::clear() {
		Tovar::clear();
		Form1->LabeledEdit4->Clear();
		Form1->LabeledEdit5->Clear();
		}
		
		const int K = 100; // максимальное число товаров
		Tovar *prod_and_prom[K]; // !!! массив может хранить продтовары и промтовары !!! - одно из главных достоинств виртуальных методов
		Tovar *new_prom_and_prod[K]; // новый массив пром и прод товаров (для стрингрида, заполненного из файла)
		
		Распишите обработчики добавления товаров. Далее, загружаем данные из файла в список StringList, из которого заполняем StringGrid. Разбираем считываемые данные на принадлежность к промышленным или продуктовым товарам и заполняем новый массив, который создали предварительно именно для этих целей.	

№ пп	Формы текущего контроля	Примеры типовых заданий	Формируемая компетенция
		Реализовав рассмотренный пример, необходимо добавить возможности: а) сортировки товаров по наименованию и цене; б) поиска (по всем свойствам товаров).	

5.2. Критерии, шкалы оценивания текущего контроля успеваемости:

Наименование оценочного средства (контрольно-оценочного мероприятия)	Критерии оценивания	Шкалы оценивания	
		100-балльная система	Пятибалльная система
Лабораторная работа	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.		5
	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.		4
	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.		3
	Работа выполнена не полностью. Допущены грубые ошибки. Файлы не содержат необходимой информации, компьютерная программа выдаёт неправильные результаты при вычислении тестовых примеров. Работа не зачтена.		2
	Работа не выполнена.		

5.3. Промежуточная аттестация:

Форма промежуточной аттестации	Типовые контрольные задания и иные материалы для проведения промежуточной аттестации:	Формируемая компетенция
Экзамен: Компьютерное тестирование	Вопрос 1. Основная часть языка программирования С является подмножеством С++, а большинство программ, написанных на С, являются также программами на С++. o Обратное утверждение верно o Обратное утверждение неверно 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	ПК-3: ИД-ПК-3.2 ИД-ПК-3.3

	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:

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. 10 итоговых баллов эквивалентны 100% правильных ответов. Для того, чтобы получить отличную, хорошую, удовлетворительную или неудовлетворительную оценки, итоговые баллы за промежуточные аттестации каждого семестра складываются с баллами за выполненные лабораторные работы.		*5*	*85% - 100%*
			4	*65% - 84%*
			3	*41% - 64%*
			2	*40% и менее 40%*

5.5. Примерные темы курсовой работы:

1. Разработка приложения для подбора платья по личным предпочтениям и параметрам.
2. Chrome Embedded Framework как инструмент для разработки внутриигровых интерфейсов.
3. Разработка приложения для администратора кафе.
4. Реализация обмена данными посредством XML-файлов.
5. Разработка бот-ассистента для Telegram каналов.

5.6. Критерии, шкалы оценивания курсовой работы/курсового проекта

Форма промежуточной аттестации	**Критерии оценивания**	**Шкалы оценивания**	
		100-балльная система	**Пятибалльная система**
защита курсовой работы	– работа выполнена самостоятельно, носит творческий характер, возможно содержание элементов научной новизны; – собран, обобщен и проанализирован достаточный объем литературных источников; – при написании и защите работы продемонстрированы: высокий уровень сформированности профессиональных компетенций, теоретические знания и наличие практических навыков; – работа правильно оформлена и своевременно представлена на кафедру, полностью соответствует требованиям, предъявляемым к содержанию и оформлению курсовых работ; – на защите освещены все вопросы исследования, ответы на вопросы профессиональные, грамотные, исчерпывающие, результаты исследования подкреплены статистическими критериями;		*5*
	– тема работы раскрыта, однако выводы и рекомендации не всегда оригинальны и / или не имеют практической значимости, есть неточности при освещении отдельных вопросов темы; – собран, обобщен и проанализирован необходимый объем профессиональной литературы, но не по всем аспектам исследуемой темы сделаны выводы и обоснованы практические рекомендации; – при написании и защите работы продемонстрирован: средний уровень сформированности профессиональных компетенций, наличие теоретических		*4*

Форма промежуточной аттестации	Критерии оценивания	Шкалы оценивания	
		100-балльная система	Пятибалльная система
	знаний и достаточных практических навыков; – работа своевременно представлена на кафедру, есть отдельные недостатки в ее оформлении; – в процессе защиты работы были даны неполные ответы на вопросы;		
	– тема работы раскрыта частично, но в основном правильно, допущено поверхностное изложение отдельных вопросов темы; – в работе недостаточно полно была использована профессиональная литература, выводы и практические рекомендации не отражали в достаточной степени содержание работы; – при написании и защите работы продемонстрирован удовлетворительный уровень сформированности профессиональных компетенций, поверхностный уровень теоретических знаний и практических навыков; – работа своевременно представлена на кафедру, однако не в полном объеме по содержанию и / или оформлению соответствует предъявляемым требованиям; – в процессе защиты недостаточно полно изложены основные положения работы, ответы на вопросы даны неполные;		*3*
	– содержание работы не раскрывает тему, вопросы изложены бессистемно и поверхностно, нет анализа практического материала, основные положения и рекомендации не имеют обоснования; – работа не оригинальна, основана на компиляции публикаций по теме; – при написании и защите работы продемонстрирован неудовлетворительный уровень сформированности профессиональных компетенций; – работа несвоевременно представлена на кафедру, не в полном объеме по содержанию и оформлению соответствует предъявляемым требованиям; – на защите показаны поверхностные знания по исследуемой теме, отсутствие представлений об актуальных проблемах по теме работы, даны неверные ответы на вопросы.		*2*

5.7. Система оценивания результатов текущего контроля и промежуточной аттестации.

Оценка по дисциплине выставляется обучающемуся с учётом результатов текущей и промежуточной аттестации.

Форма контроля	100-балльная система	Пятибалльная система
Текущий контроль:		
Выполнение лабораторной работы		зачтено/не зачтено
Промежуточная аттестация экзамен		отлично хорошо удовлетворительно неудовлетворительно
Итого за семестр (дисциплину) экзамен		

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

100-балльная система	пятибалльная система	
	зачет с оценкой/экзамен	зачет
	отлично зачтено (отлично)	
	хорошо зачтено (хорошо)	
	удовлетворительно зачтено (удовлетворительно)	
	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 лабораторных работ и выполнении курсовых работ, предусматривающих участие обучающихся в выполнении отдельных элементов работ, связанных с будущей профессиональной деятельностью.

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 требованиям 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	Шитов В.Н.	Информатика и информационно-коммуникационные технологии в профессиональной деятельности	Учебное пособие	М: НИЦ ИНФРА-М	2022	https://znanium.com/catalog/document?id=388696	-
3	Шуляк О.А.	Основы программирования	Учебно-методическая литература	М.: Флинта	2021	https://znanium.com/catalog/document?id=390158	-
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	2021	https://znanium.com/catalog/document?id=363426	-
10.2 Дополнительная литература, в том числе электронные издания							
1	Гутгарц Р. Д.	Проектирование автоматизированных систем обработки информации и управления	Учебное пособие	М.: Издательство Юрайт	2022	https://urait.ru/bcode/494408	-
2	Плотникова Н.Г.	Информатика и информационно-	Учебное пособие	М.: РИОР	2021	https://znanium.com/catalog/document?id=370445	-

		коммуникационные технологии (ИКТ)					
3	Горбатов С.М., Тарасов Ю.С., Наумова М.Г.	Информационные технологии	Учебное пособие	М.: МИСиС	2016	https://znanium.com/catalog/document?id=371025	-
4	Федотова Е.Л.	Информационные технологии и системы	Учебное пособие	М.: Издательский Дом ФОРУМ	2022	https://znanium.com/catalog/document?id=386738	-
5	М. В. Огнева, Е. В. Кудрина	Программирование на языке С++: практический курс	Учебное пособие	М.: Издательство Юрайт	2022	https://urait.ru/bcode/492984	-
10.3 Методические материалы (указания, рекомендации по освоению дисциплины (модуля) авторов РГУ им. А. Н. Косыгина)							
1	Семенов А.А.	Основы объектно-ориентированного программирования в среде C++Builder	Методическое пособие	М.: ИИЦ МГУДТ	2010	локальная сеть университета	5

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 на C++.	Свободно распространяемое на условиях GNU General Public License v.3.
5.	Visual Studio Community	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