



МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Магнитогорский государственный технический университет им. Г.И. Носова»



УТВЕРЖДАЮ
Директор ИЭиАС
С.И. Лукьянов

26.02.2020 г.

РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ (МОДУЛЯ)

ОБЪЕКТНО-ОРИЕНТИРОВАННОЕ ПРОГРАММИРОВАНИЕ

Направление подготовки (специальность)
09.03.01 Информатика и вычислительная техника

Направленность (профиль/специализация) программы
Проектирование и разработка Web-приложений

Уровень высшего образования - бакалавриат

Форма обучения
очная

Институт/ факультет	Институт энергетики и автоматизированных систем
Кафедра	Вычислительной техники и программирования
Курс	2
Семестр	4

Магнитогорск
2020 год

Рабочая программа составлена на основе ФГОС ВО - бакалавриат по направлению подготовки 09.03.01 Информатика и вычислительная техника (приказ Минобрнауки России от 19.09.2017 г. № 929)

Рабочая программа рассмотрена и одобрена на заседании кафедры
Вычислительной техники и программирования
19.02.2020 г. протокол № 5

Зав. кафедрой _____ О.С. Логунова

Рабочая программа одобрена методической комиссией ИЭ и АС
26.02.2020 г. протокол № 5

Председатель _____ С.И. Лукьянов

Рабочая программа составлена:
ст. преподаватель кафедры ВТиП,

_____ М.В. Зарецкий

Рецензент:

Начальник отдела технологических платформ
ООО «Компас Плюс», канд. техн. наук

_____ Д.С. Сафонов

Лист актуализации рабочей программы

Рабочая программа пересмотрена, обсуждена и одобрена для реализации в 2021 - 2022 учебном году на заседании кафедры Вычислительной техники и программирования

Протокол от _____ 20__ г. № ____
Зав. кафедрой _____ О.С. Логунова

Рабочая программа пересмотрена, обсуждена и одобрена для реализации в 2022 - 2023 учебном году на заседании кафедры Вычислительной техники и программирования

Протокол от _____ 20__ г. № ____
Зав. кафедрой _____ О.С. Логунова

Рабочая программа пересмотрена, обсуждена и одобрена для реализации в 2023 - 2024 учебном году на заседании кафедры Вычислительной техники и программирования

Протокол от _____ 20__ г. № ____
Зав. кафедрой _____ О.С. Логунова

Рабочая программа пересмотрена, обсуждена и одобрена для реализации в 2024 - 2025 учебном году на заседании кафедры Вычислительной техники и программирования

Протокол от _____ 20__ г. № ____
Зав. кафедрой _____ О.С. Логунова

1 Цели освоения дисциплины (модуля)

Целями освоения дисциплины «Объектно-ориентированное программирование» являются:

- формирование у студентов понятия об объектно-ориентированной парадигме моделирования бизнес-процессов и разработки WEB-приложений;
- освоение объектно-ориентированной методологии проектирования, разработки и отладки программного обеспечения;
- выработка компетенций, позволяющих определять применимость данной парадигмы в конкретных условиях;
- выработка компетенций, позволяющих находить оптимальные методы применения объектно-ориентированной парадигмы.

2 Место дисциплины (модуля) в структуре образовательной программы

Дисциплина Объектно-ориентированное программирование входит в часть учебного плана формируемую участниками образовательных отношений образовательной программы.

Для изучения дисциплины необходимы знания (умения, владения), сформированные в результате изучения дисциплин/ практик:

Математическая логика и дискретная математика

Средства программирования Web-приложений

Философия

Прикладная математика

Программирование

Информатика

Знания (умения, владения), полученные при изучении данной дисциплины будут необходимы для изучения дисциплин/практик:

Базы данных Web-приложений

Программное обеспечение Front-End в Web разработке

Программное обеспечение Back-End в Web разработке

Проектирование интерфейсов Web-приложений

Алгоритмы обработки больших данных

3 Компетенции обучающегося, формируемые в результате освоения дисциплины (модуля) и планируемые результаты обучения

В результате освоения дисциплины (модуля) «Объектно-ориентированное программирование» обучающийся должен обладать следующими компетенциями:

Код индикатора	Индикатор достижения компетенции
ПК-5	Способность к формализации и алгоритмизации поставленных задач, к написанию программного кода с использованием языков программирования, определения и манипулирования данными и оформлению программного кода в соответствии установленными требованиями
ПК-5.1	Оценивает качество математической модели при формализации задачи предметной области
ПК-5.2	Оценивает качество разработанных алгоритмов для последующего кодирования
ПК-5.3	Оценивает выбор программных средств для программирования и манипулирования данными в соответствии установленными требованиями
ПК-1	Способность к анализу проблемной ситуации, разработке требований к системе, постановке целей создания, разработке концепции и технического задания на создание

Web-приложения, представления концепции, технического задания на Web-приложение и изменений в них заинтересованным лицам	
ПК-1.1	Анализирует требования к разработке Web-приложений и базам данных
ПК-1.2	Оценивает качество разработки технических спецификаций на Web-приложения
ПК-1.3	Оценивает качество проекта на разработку Web-приложения и баз данных

4. Структура, объём и содержание дисциплины (модуля)

Общая трудоемкость дисциплины составляет 4 зачетных единиц 144 академических часов, в том числе:

- контактная работа – 72 академических часов;
- аудиторная – 68 академических часов;
- внеаудиторная – 4 академических часов
- самостоятельная работа – 36,3 академических часов;
- подготовка к экзамену – 35,7 академических часов

Форма аттестации - экзамен

Раздел/ тема дисциплины	Семестр	Аудиторная контактная работа (в академических часах)			Самостоятельная работа студента	Вид самостоятельной работы	Форма текущего контроля успеваемости и промежуточной аттестации	Код компетенции
		Лек.	лаб. зан.	практ. зан.				
1. Объектный подход к программированию								
1.1 Объектный подход к программированию	4	4	4		4	Самостоятельное изучение учебной и научной литературы. Подготовка к лабораторному занятию.	Беседа – обсуждение. Устный опрос	ПК-1.1
1.2 Абстрагирование и контрактная модель, инкапсуляция, модульность, иерархичность, типизация, параллелизм. Объект: состояние, поведение, индивидуальность; отношения между объектами.		4	4	4/2И		4	Самостоятельное изучение учебной и научной литературы. Подготовка к лабораторному занятию. Выполнение лабораторной работы.	Беседа – обсуждение. Анализ программного кода. Устный опрос
Итого по разделу		8	8/2И		4			
2. Классы								
2.1 Сущность класс, данные, методы, доступ, наследование свойств. Отношения между классами.	4	4	4/2И		4	Самостоятельное изучение учебной и научной литературы. Подготовка к лабораторному занятию. Выполнение лабораторной работы.	Беседа – обсуждение. Анализ программного кода. Устный опрос	ПК-5.2

2.2	Одиночное наследование. Множественное наследование объектов и Метаклассы.	Системы классов.	4	4/2И	4	Самостоятельное изучение учебной и научной литературы. Подготовка к лабораторному занятию. Выполнение лабораторной работы.	Беседа – обсуждение. Анализ программного кода. Устный опрос	ПК-5.2
Итого по разделу			8	8/4И	8			
3. Проектирование на основе объектной парадигмы								
3.1	Объектно-ориентированный анализ. Классификация.		4	4/2И	4	Самостоятельное изучение учебной и научной литературы. Подготовка к лабораторному занятию. Выполнение лабораторной работы.	Беседа – обсуждение. Анализ программного кода. Устный опрос	ПК-5.3
3.2	Объектно-ориентированные системы: методы, языки и способы программирования. Разработка интерфейса		4	4/2И	4	Самостоятельное изучение учебной и научной литературы. Подготовка к лабораторному занятию. Выполнение лабораторной работы.	Беседа – обсуждение. Анализ программного кода. Устный опрос	ПК-1.2
Итого по разделу			8	8/4И	8			
4. Современные средства анализа и проектирования бизнес-процессов и WEB ресурсов								
4.1	Язык UML. Пакеты. Диаграммы.		4	4/2И	8	Самостоятельное изучение учебной и научной литературы. Подготовка к лабораторному занятию. Выполнение лабораторной работы.	Беседа – обсуждение. Анализ программного кода. Устный опрос	ПК-1.3
4.2	Программные реализации		6	6/2И	8,3	Самостоятельное изучение учебной и научной литературы. Подготовка к лабораторному занятию. Выполнение лабораторной работ	Беседа – обсуждение. Анализ программного кода. Устный опрос	ПК-1.2, ПК-1.3
Итого по разделу			10	10/4И	16,3			
5. Контроль								

5.1 Внеаудиторная контактная работа	4					Анализ программного кода	Проверка программного кода	ПК-5.1, ПК-5.2, ПК-5.3, ПК-1.1, ПК-1.2, ПК-1.3
5.2 Контроль						Подготовка к экзамену	Экзамен	ПК-5.1, ПК-5.2, ПК-5.3, ПК-1.1, ПК-1.2, ПК-1.3
Итого по разделу								
Итого за семестр	34	34/14И		36,3			экзамен	
Итого по дисциплине	34	34/14И		36,3			экзамен	

5 Образовательные технологии

1. Традиционные образовательные технологии ориентируются на организацию образовательного процесса, предполагающую прямую трансляцию знаний от преподавателя к студенту (преимущественно на основе объяснительно-иллюстративных методов обучения). Учебная деятельность студента носит в таких условиях, как правило, репродуктивный характер.

Формы учебных занятий с использованием традиционных технологий:

Информационная лекция – последовательное изложение материала в дисциплинарной логике, осуществляемое преимущественно вербальными средствами (монолог преподавателя).

Семинар – беседа преподавателя и студентов, обсуждение заранее подготовленных сообщений по каждому вопросу плана занятия с единым для всех перечнем рекомендуемой обязательной и дополнительной литературы.

Практическое занятие, посвященное освоению конкретных умений и навыков по предложенному алгоритму.

Лабораторная работа – организация учебной работы с реальными материальными и информационными объектами, экспериментальная работа с аналоговыми моделями реальных объектов.

2. Технологии проблемного обучения – организация образовательного процесса, которая предполагает постановку проблемных вопросов, создание учебных проблемных ситуаций для стимулирования активной познавательной деятельности студентов.

3. Интерактивные технологии – организация образовательного процесса, которая предполагает активное и нелинейное взаимодействие всех участников, достижение на этой основе лично значимого для них образовательного результата. Наряду со специализированными технологиями такого рода принцип интерактивности прослеживается в большинстве современных образовательных технологий. Интерактивность подразумевает субъект - субъектные отношения в ходе образовательного процесса и, как следствие, формирование саморазвивающейся информационно-ресурсной среды.

Формы учебных занятий с использованием специализированных интерактивных технологий:

Лекция «обратной связи» – лекция–провокация (изложение материала с заранее запланированными ошибками), лекция-беседа, лекция-дискуссия, лекция–пресс-конференция.

4. Информационно-коммуникационные образовательные технологии – организация образовательного процесса, основанная на применении специализированных программных сред и технических средств работы с информацией.

6 Учебно-методическое обеспечение самостоятельной работы обучающихся

Представлено в приложении 1.

7 Оценочные средства для проведения промежуточной аттестации

Представлены в приложении 2.

8 Учебно-методическое и информационное обеспечение дисциплины (модуля)

а) Основная литература:

1. Программная инженерия. Учебник для вузов. 5-е издание обновленное и дополненное. Стандарт третьего поколения. [Электронный ресурс]. / С.А. Орлов – СПб.: Питер, 2016. – 624 с. Режим доступа <http://ibooks.ru/product.php?productid=351445>. Заглавие с экрана ISBN 978-5-496-01917-0

2. Объектно-ориентированное программирование. [Электронный ресурс]. Режим доступа <https://pythonworld.ru/osnovy/obektno-orientirovannoe-programmirovaniye-obshhee-predstavleniye.html>

б) Дополнительная литература:

1. Ашарина, И.В. Объектно-ориентированное программирование в С++: лекции и упражнения [Электронный ресурс]. Учеб. пособие для вузов / И.В. Ашарина – М.: Горячая линия ТЕЛЕКОМ, 2010. – 320 с. Режим доступа <http://ibooks.ru/reading.php?productid=333386> . Заглавие с экрана ISBN 978-5-9912-0082-0

2. Хорев, П.Б. Объектно-ориентированное программирование: учеб. пособие для студ. учреждений высш. проф. образования / П.Б. Хорев. – 3-е изд. испр. – М.: Издательский центр «Академия», 2011. – 448с

3. Буч, Г. Объектно-ориентированный анализ и проектирование с примерами приложений на С++, 2 е изд. / Г. Буч – М.: «Издательство Бином», СПб.: «Невский диалект», 1998. – 560 с.

4. Кью, Дж. Объектно-ориентированное программирование. Учебный курс / Дж Кью, М Джеанини – СПб.: Питер, 2005. – 238 с.

5. Леоненков, А.В. Объектно-ориентированный анализ и проектирование с использованием UML и Rational Rose. Учебное пособие / А.В. Леоненков. – М.: Интернет – Университет Информационных Технологий; БИНОМ. Лаборатория знаний, 2006. – 320 с.

в) Методические указания:

1. Зарецкий М.В. Методические указания для выполнения самостоятельных работ по курсу «Объектно-ориентированное программирование» для студентов специальности 220400[Текст] / М.В. Зарецкий, Ю.Б. Кухта Магнитогорск. МГТУ, 2005 17 с.

г) Программное обеспечение и Интернет-ресурсы:

Программное обеспечение

Наименование ПО	№ договора	Срок действия лицензии
MS Office 2007 Professional	№ 135 от 17.09.2007	бессрочно
Scilab Computation Engine	свободно распространяемое ПО	бессрочно
Anaconda Python	свободно распространяемое ПО	бессрочно
Deductor Studio Academic	Соглашение о сотрудничестве №06-2901\08 от 29.01.2008	бессрочно
WordPress	свободно распространяемое ПО	бессрочно
NotePad++	свободно распространяемое ПО	бессрочно

Профессиональные базы данных и информационные справочные системы

Название курса	Ссылка
Поисковая система Академия Google (Google Scholar)	URL: https://scholar.google.ru/

9 Материально-техническое обеспечение дисциплины (модуля)

Материально-техническое обеспечение дисциплины включает:

Лекционная аудитория ауд. 282 – Мультимедийные средства хранения, передачи и представления информации;

Компьютерные классы Центра информационных технологий ФГБОУ ВПО «МГТУ им. Г.И. Носова» – Персональные компьютеры, объединенные в локальные сети с выходом в Internet, оснащенные современными программно-методическими комплексами для решения задач в области информатики и вычислительной техники;

Аудитории для самостоятельной работы: компьютерные классы; читальные залы библиотеки – ауд. 282 и классы УИТ и АСУ;

Помещения для самостоятельной работы обучающихся, оснащенных компьютерной техникой с возможностью подключения к сети «Интернет» и наличием доступа в электронную информационно-образовательную среду организации – классы УИТ и АСУ;

Помещения для хранения и профилактического обслуживания учебного оборудования – Центр информационных технологий – ауд. 372.

Приложение 1. Учебно-методическое обеспечение самостоятельной работы обучающихся.
Задание к лабораторной работе по теме:

Понятие о парадигме программирования. Виды парадигм программирования. Объектно-ориентированная парадигма. Объектно-ориентированные языки: классификация, архитектура, выразительные средства, технология применения.

В программе на языке Python используется свойство переменной. Определить тип переменной, для которого данное свойство может быть применено и написать его аналог на языках VBA и C+ (n — имя переменной):

1. n.bit_length;
2. n.conjugate;
3. n.denominator;
4. n.from_bytes;
5. n.imag;
6. n.numerator;
7. n.real;
8. n.to_bytes;
9. n.__divmod__;
10. n.__dir__

Задание к лабораторной работе по теме:

Абстрагирование и контрактная модель, инкапсуляция, модульность, иерархичность, типизация, параллелизм. Объект: состояние, поведение, индивидуальность; отношения между объектами.

Создать обработчики событий на VBA и C++.

1. Написать обработчик события для выбора языка из списка, помещенного в ListBox.
2. Написать обработчик события для выбора названия цветка из списка, помещенного в ComboBox.
3. Написать обработчик события для выбора одного из способов настройки алгоритма оптимизации с помощью активизации одной из кнопок группы

OptionButton.

4. Написать обработчик события для выбора одного из методов решения системы линейных алгебраических уравнений из списка, помещенного в ListBox.
5. Написать обработчик события для выбора одного из методов решения нелинейного уравнения из списка, помещенного в ComboBox.
6. Написать обработчик события для выбора коэффициента настройки шага оптимизации с помощью SpinButton.
7. Написать обработчик события для выбора одного из методов вычисления определенного интеграла из списка, помещенного в ComboBox.
8. Написать обработчик события для выбора одного из методов решения задачи Коши для дифференциального уравнения из списка, помещенного в ComboBox.
9. Написать обработчик события для выбора одного из методов решения краевой задачи для уравнения из списка, помещенного в ComboBox.
10. Написать обработчик события для выбора архитектуры искусственной нейронной сети из списка, помещенного в ComboBox.

Задание к лабораторной работе по теме:

Сущность класс, данные, методы, доступ, наследование свойств. Отношения между классами

Реализовать иерархию наследования на C++ и Python:

1. Транспортное средство (абстрактное). Наследники: транспортное средство, использующее живую силу, механическое транспортное средство. Наследники транспортного средства, использующего живую силу: животное для верховой езды, упряжка.
2. Плоды (абстрактные). Наследники: плоды съедобные, плоды несъедобные. Наследники съедобных плодов: яблоки, огурцы.
3. Аудиоаппаратура (абстрактная). Наследники: проигрыватели, магнитофоны. Наследники проигрывателей: проигрыватели цифровых записей, проигрыватели аналоговых записей.
4. Животные (абстрактные). Наследники: хищные животные, травоядные животные. Наследники травоядных животных: лошади, овцы.
5. Верхняя одежда абстрактная. Наследники: пальто, куртки. Наследники курток: куртки меховые, ветровки.
6. Обувь абстрактная. Наследники: ботинки, туфли. Наследники туфель: кроссовки, балетки.
7. Птицы абстрактные. Наследники: птицы летающие, птицы нелетающие. Наследники птиц летающих: воробьи, синицы.
8. Пища абстрактная. Наследники: блюда холодные, блюда горячие. Наследники блюд горячих: борщ, куллама.
9. Литературный текст абстрактный. Наследники: прозаический текст, поэтический текст. Наследники поэтического текста: стихи, написанные ямбом; стихи, написанные амфибрахием.
10. Средства связи абстрактные. Наследники: средства связи стационарные, средства связи мобильные. Наследники средств связи мобильных: автономные приемно-передающие устройства; средства сотовой связи.

Задание к лабораторной работе по теме:

Одиночное наследование. Множественное наследование. Системы объектов и классов. Метаклассы.

Реализовать иерархию наследования на C++ и Python:

1. На верхнем уровне классы: «средство передвижения», «загрязнитель окружающей среды». Создать класс «автомобиль» — наследник обоих классов верхнего уровня.
2. На верхнем уровне классы: «средство мобильной связи», «показатель статуса». Создать класс «смартфон для претенциозной личности» — наследник обоих классов верхнего уровня.
3. На верхнем уровне классы «средство обработки информации», «источник опасности для зрения». Создать класс «персональный компьютер» — наследник обоих классов верхнего уровня.
4. На верхнем уровне классы «устройство для приготовления пищи», «источник пожарной опасности». Создать класс «электрическая плита» — наследник обоих классов верхнего уровня.
5. На верхнем уровне классы: «транспортное средство», «налогооблагаемое имущество». Создать класс «автомобиль» — наследник обоих классов верхнего уровня.
6. На верхнем уровне классы «аудиоаппаратура», «источник шумового загрязнения окружающей среды». Создать класс «аудиосистема» — наследник обоих классов верхнего уровня.
7. На верхнем уровне классы «электронное устройство», «бытовая техника». Создать класс «ноутбук» — наследник обоих классов верхнего уровня.
8. На верхнем уровне классы «средство связи», «рекламируемый объект». Создать класс «смартфон» — наследник обоих классов верхнего уровня.
9. На верхнем уровне классы «средство регистрации изображений», «товар, на который дается гарантия». Создать класс «фотоаппарат» — наследник обоих классов верхнего уровня.
10. На верхнем уровне классы «растение», «объект для украшения интерьера». Создать класс «комнатное растение» — наследник обоих классов верхнего уровня.

Задание к лабораторной работе по теме:

Объектно-ориентированный анализ. Классификация.

Выполнить объектно-ориентированный анализ неформально поставленной задачи:

1. Младенец умеет: есть, пить, плакать, когда голоден, когда хочет пить, плакать, когда надо сменить пеленки, Кроме того, он может плакать из солидарности с другим младенцем (когда слышит плач более 3 минут), плакать просто так. Если младенец плачет более 5 минут, пеленки обязательно будут мокрые. Младенец начинает плакать в случайный момент времени. Мать умеет: кормить младенца, поить младенца, менять ему пеленки, безошибочно определять причину плача, успокаивать одновременно несколько младенцев. Метода стирки у нее нет, поэтому при нехватке пеленок ей придется использовать в этом качестве простыни, наволочки и все, что под руку попадет. Отец умеет: стирать и гладить пеленки. Делать он это может только 1 раз в сутки.
2. Кошка умеет: охотиться на мышей (вероятность поимки мыши — p), воровать у хозяина мясо (вероятность успеха — q), воровать у хозяина молоко (вероятность успеха — r). Мышь умеет: прятаться от кошки, прятаться от хозяина, воровать у хозяина муку (вероятность успеха — s), воровать у хозяина крупу (вероятность успеха — t). Хозяин умеет: отгонять кошку от еды, ловить мышь в капкан, покупать все перечисленные ранее продукты. В произвольные моменты времени хозяин отлучается из дому.
3. Крестьянин выращивает в поле зерно. Птицы поедают зерно (m граммов в день). Кроме того, они поедают жучков (n жучков в день). Жучки поедают зерно (s граммов в день). Если крестьянин уничтожит или прогонит всех птиц, жучки, не

- имея естественных врагов, будут беспрепятственно съедать зерно. Вероятность для крестьянина прогнать птицу — p , вероятность для птицы съесть жучка — q
4. Медицинская страховая компания работает с N клиентами. За каждого клиента она получает ежемесячно w_i ($i = 1, \dots, N$) рублей. Вероятность заболеть в течение года для каждого клиента равна p_i ($i = 1, \dots, N$). Стоимость его излечения является равномерно распределенной случайной величиной в диапазоне от 1000 до 100000 рублей.
 5. Процессинговый центр обслуживает расчеты по дебетовым карточкам. Существует вероятность p попытки получения суммы S неправомерным путем. С вероятностью q эта попытка будет пресечена, а злоумышленник наказан, в противном случае процессинговый центр заплатит банку штраф в размере $2S$.
 6. Системный администратор компьютерной сети умеет определять пользователям права доступа, изымать вредные записи, брать плату за работу в сети, лишать нарушителей прав доступа, брать штрафы за нарушения. Добросовестный пользователь может вносить записи в отведенный ему раздел, стирать свои записи, платить за пользование, извлекать доходы или нести убытки из-за невостребованности информации или деятельности хакеров. За 1 килобайт пользователь платит K рублей, использование 1 килобайта информации приносит N рублей дохода, если информация окажется востребованной (вероятность — p). Хакер может все то, что и добросовестный пользователь. Кроме того, он может с вероятностью q определять чужие пароли и переадресовывать счета за работу в сети другим пользователям (как добросовестным, так и хакерам). В случае обнаружения недобросовестного поведения, он может быть подвергнут штрафу в размере десятикратно превышающем доход от недобросовестной деятельности.
 7. Спамер рассылает по сети Internet недобросовестную рекламу товара (услуги) ценой в G рублей. С вероятностью p_i i -й пользователь купит рекламируемый товар (или услугу), q_i — не отреагирует на спам, с вероятностью r_i заставит спамера заплатить штраф, величина которого указывается далее. За каждую покупку спамер получает 1% комиссионных, при изобличении платит штраф, в 10 раз превышающий размер потенциальных комиссионных (то есть комиссионных, которые бы он получил при условии, что все адресаты купят рекламируемый товар).
 8. Дистрибьютор высококачественной компьютерной техники работает с сетью дилеров. Он покупает у производителей продукцию n наименований по цене k_i за единицу и поставляет ее дилерам с наценкой в $p\%$. Добросовестные дилеры торгуют только фирменным товаром с оговоренной в соглашении с дистрибьютором наценкой в $q\%$. Недобросовестные дилеры могут поступать так же, как и добросовестные. Кроме того, они могут под видом фирменных продавать компьютеры низкого качества с наценкой $r\% > q\%$. В случае обнаружения подлога недобросовестный дилер возмещает обманутым клиентам разницу в цене и платит штраф в размере утроенной неправомерно полученной суммы. Покупатель не имеет возможности отличить добросовестного дилера от недобросовестного. Количество денег у покупателя задается случайным образом. Решение о покупке принимается случайным образом при наличии у покупателя достаточной суммы свободных денег
 9. В библиотеке имеются книги N наименований. Количество экземпляров каждой из этих книг k_i ($i = 1, \dots, N$). В библиотеку записано M читателей. Каждый из читателей может одновременно держать у себя не более L книг. Книги выдаются

на срок до 30 дней. Считаем, что читатель с одинаковой вероятностью может запросить любую из книг. Если запрошенной книги нет в наличии, читателя ставят в очередь на ее получение. Когда читатель возвращает книгу, она достается первому в очереди. Если читатель возвращает книгу после указанного срока, он лишается права получить книги в течение 60 дней. Длительность нахождения книги у читателя является случайной величиной.

10. Фирма распространяет лицензионное программное обеспечение и занимается его сопровождением. За каждую установку программного обеспечения фирма получает k рублей прибыли. В случае возникновения проблем с использованием установленного программного обеспечения фирма должна в течение 2 часов с момента получения заявки направить специалиста для решения проблемы. Вероятность возникновения в течение суток проблемы, требующей вмешательства специалиста, равна p . Специалист устраняет проблему за время, являющееся случайной величиной, равномерно распределенной в интервале $[0, T]$. В том случае, когда специалист опаздывает, фирма платит штраф в размере r рублей за каждый час опоздания. Каждый специалист получает s рублей в месяц.

Задание к лабораторной работе по теме:

Объектно-ориентированные системы: методы, языки и способы программирования. Разработка интерфейса.

Разработать удобный в использовании двуязычный интерфейс для следующих задач.

1. Снежинка имеет форму правильного многоугольника. Падает вниз. Под воздействием ветра может лететь вверх или в сторону. Может слипаться с другими снежинками. Комок снежинок имеет форму шара. Снежинка может таять. Изобразить снегопад, метель оттепель.
2. Схематично изображенный самолет может стоять на стоянке, выруливать по рулежным дорожкам на взлетно-посадочную полосу, разогнаться и взлетать, лететь по прямой траектории, снижаться, совершать посадку и уходить по рулежным дорожкам с взлетно-посадочной полосы на стоянку.
3. Схематично изображенный вертолет может стоять на стоянке, вертикально взлетать, лететь по прямой траектории, к нему может быть прицеплен груз на внешней подвеске. Вертолет может перевозить груз на внешней подвеске, устанавливать груз на указанное место, приземляться.
4. Заготовка проходит между двумя вращающимися валками, при этом уменьшается ее высота (обжатие), увеличивается ширина (уширение) и длина. Прошедшую между валками заготовку поворачивают на 90° (кантовка) и направляют в обратном направлении. Процесс продолжается до тех пор, пока не будут достигнуты требуемые параметры заготовки.
5. Схематично изображенный грузовик может привозить груз к подъемному крану и отвозить груз от подъемного крана. Схематично изображенный подъемный кран разгружает и загружает грузовики.
6. Схематично изображенный надувной шар может быть сплюснутым, его можно надувать (его радиус увеличивается), при излишнем надувании он может лопнуть, он может лопнуть, если его проткнут иголкой. Шары можно соединять в гирлянду, отделять от гирлянды.
7. Схематично изображенная ветряная мельница вращает лопасти. Если ветра нет, лопасти неподвижны. При очень сильном ветре лопасти отваливаются.
8. Схематично изображенный катер может стоять у пристани, в катер может быть помещен груз. Катер может отчаливать, двигаться по фарватеру, причаливать,

- проводить выгрузку.
9. Схематично изображенный трамвай может стоять в депо, ехать от остановки до остановки по прямой и с поворотами, останавливаться на остановках, разворачиваться на конечных остановках.
 10. Схематично изображенный автомобиль может стоять у тротуара, двигаться по прямой, перестраиваться из ряда в ряд, поворачивать. При перестроениях и поворотах у автомобиля должен быть включен соответствующий световой указатель. При нарушении правил автомобиль может быть удален из транспортного потока. Светофор поочередно включает в каждом направлении красный, желтый и зеленый свет

Задание к лабораторной работе по теме:

Язык UML. Пакеты. Диаграммы.

Описать средствами UML следующие задачи:

1. Банк может: привлекать вклады (под $p\%$ годовых), давать кредиты (под $q\%$ годовых). Существует вероятность s того, что банк не сможет расплатиться с вкладчиком. Вкладчик умеет: зарабатывать деньги, вносить вклады, получать по вкладам проценты. Должник умеет зарабатывать деньги, обслуживать долг (выплачивать проценты), возвращать долг. Существует вероятность r того, что должник окажется неплатежеспособным.
2. Экстрасенс умеет: морочить голову пациентам, вымогать у них деньги. Врач умеет: вылечивать пациентов (с вероятностью p). Пациент умеет: зарабатывать деньги на лечение у врача или экстрасенса, принимать решение, у кого лечиться — врача или экстрасенса, погибать от несвоевременного лечения. Заработок пациента задается случайным образом. Случайным образом задается решение о том, у кого лечиться
3. Производитель программного обеспечения (ПО) позиционирует свою продукцию в классе ShareWare. Затраты на выпуск ПО составляют $\$U$. Получив его продукцию, пользователь заплатит $\$t$ с вероятностью p .
4. Провайдер Интернет предлагает пользователям k тарифных планов. Для привлечения новых пользователей он использует звонки потенциальным покупателям услуги по телефону и обход их жилищ. Вероятность приобретения тарифного плана после звонка по телефону равна $p_i, i = 1, \dots, k$, вероятность приобретения тарифного плана после посещения жилища равна $q_i, i = 1, \dots, k$. Себестоимость привлечения одного клиента с использованием телефона равна v , с использованием обхода жилищ — w . Прибыль от одного клиента, пользующегося услугами провайдера равна $t_i, i = 1, \dots, k$.
5. Продовольственный магазин торгует товарами N наименований. У каждого товара есть оптовая цена за единицу b_i , розничная цена за единицу c_i , минимальный объем оптовой закупки k_i , срок реализации t_i (везде $i = 1, \dots, N$). Если товар не реализован своевременно, он подлежит списанию. Ежедневно магазин посещает M покупателей. Каждый из них с вероятностью $p_i, i = 1, \dots, N$ покупает один из товаров, причем величина покупки также является случайной. Покупка может быть совершена только при наличии товара в магазине. В магазине учитываются покупки, потери от списания непроданного товара, а также недополученная прибыль от неудовлетворенного спроса.
6. Открывается компьютерная школа. В ней желающие будут изучать Windows и Microsoft Office. За изучение Windows ученик будет платить K рублей, за

- изучение Microsoft Office — L рублей. В штате состоят два преподавателя, для проведения экзаменов привлекается независимый эксперт. Имеется K людей, желающих получить компьютерное образование. Вероятность того, что они пойдут учиться в эту школу равна p . Преподаватели получают за работу по m рублей (независимо от количества учеников). Эксперту платят по n рублей с каждого экзаменуемого.
7. Компьютерный хитрец добывает в Интернете рефераты для ленивых студентов. Нахождение одного реферата обходится хитрецу в l рублей. Студент платит за один реферат m рублей. Если реферат не будет зачитан, студент получает от хитреца k рублей ($k < m$). Вероятность сдать реферат равна p . Имеется R ленивых студентов и S компьютерных хитрецов. Каждому ленивому студенту может потребоваться от 0 до 10 рефератов. Количество рефератов — равномерно распределенная случайная величина.
 8. Сетевой торговец ходит с мешком товара по квартирам. Цена единицы товара — s рублей. С вероятностью p ему не откроют дверь. Если ему дверь откроют, с вероятностью q он сможет продать одну единицу товара. Всего торговцев — N , потенциальных покупателей — K . Если покупатель что-либо уже купил у одного торговца, то никому из них он в течение 10 дней не откроет дверь. Для продолжения работы торговец» должен продавать в день товара на сумму не менее t рублей. Если он в течение 10 дней не продает товара на эту сумму, ему приходится прекращать свою торговую деятельность.
 9. Инвестиционный строительный фонд пирамидальной конструкции покупает своим старым пайщикам жилье за счет части взносов новых пайщиков. Так продолжается до тех пор, пока имеется достаточное количество новых пайщиков. Когда их становится недостаточно, фонд объявляет о банкротстве. Цена жилища — S рублей, суммарный взнос пайщика — R рублей, на покупку жилья фонд направляет $f\%$ денежных поступлений. В городе имеется K жителей, нуждающихся в жилье и имеющих достаточно средств. Вероятность привлечь в фонд одного жителя — p . Каждый человек вступает только один раз только в один фонд. Пусть количество фондов — N .
 10. Провайдер кабельного телевидения предлагает пользователям k пакетов. Для привлечения новых пользователей он использует рекламу в социальных сетях, реклама в эфирных каналах телевидения. Вероятность приобретения пакета в результате воздействия рекламы в социальных сетях равна $p_i, i = 1, \dots, k$, вероятность приобретения тарифного плана после воздействия рекламы в эфирных каналах телевидения равна $q_i, i = 1, \dots, k$. Себестоимость привлечения одного клиента с использованием рекламы в социальных сетях v , с использованием рекламы в эфирных каналах телевидения — w . Прибыль от одного клиента, пользующегося услугами провайдера равна $t_i, i = 1, \dots, k$.

Задание к лабораторной работе по теме:

Программные реализации.

Рассмотреть фрагменты программ на языке Python. Создать аналоги на языке C++ / C#.

```

1.
from tkinter import *

class But_print:

    def __init__(self):

```

```

self.but=Button(root)

self.but['text']='Print'

self.but.bind('<Button-1>',self.printer)

self.but.pack()

def printer(self,event):

    q = 3 + 2

    print("Tutor_02!",q)

root = Tk()

obj = But_print()

root.mainloop()

```

2.

```

from tkinter import *
def button_clicked():
    print('Click!')
root = Tk()
button1=Button()
button1.pack()
button2=Button(root,bg='red',text = 'Click me!', command = button_clicked)
button2.pack()
root.mainloop()

```

3.

```

from tkinter import *
import time
def button_clicked():
    button['text'] = time.strftime('%H:%M:%S')
root = Tk()
button=Button(root)
button.configure(text=time.strftime('%H:%M:%S'), command=button_clicked)
button.pack()
root.mainloop()

```

4.

```

from tkinter import *
import time
def button_clicked():
    button['text'] = time.strftime('%H:%M:%S')
root = Tk()
button=Button(root)
button.configure(text=time.strftime('%H:%M:%S'), command=button_clicked)
button.pack()
root.mainloop()

```

5.

```
from tkinter import *
from tkinter.filedialog import *
import fileinput
def _open():
    op = askopenfilename()
    for l in fileinput.input(op):
        txt.insert(END,l)
```

6.

```
from tkinter import *
from tkinter.filedialog import *
import fileinput

def _save():
    sa = asksaveasfilename()
    letter = txt.get(1.0,END)
    f = open(sa,"w")
    f.write(letter)
    f.close()
```

7.

```
from tkinter import *
from tkinter.filedialog import *
import fileinput

root = Tk()
m = Menu(root)
root.config(menu = m)

fm = Menu(m)
m.add_cascade(label = "File",menu=fm)
fm.add_command(label = "Open...",command=_open)
fm.add_command(label = "Save as...",command=_save)
txt = Text(root, width = 40, height = 15, font = "14")
txt.pack()
root.mainloop()
```

8.

```
import tkinter
from tkinter import *
from math import sqrt

def myOutput_01():
    root = tkinter.Tk()
    text = tkinter.Text(root)
    text.pack()
    text.insert('end',u"строка сразу видна")
    root.after(2000, text.insert, 'end', u"\nчерез две")
    root.after(3210, root.destroy)
    root.mainloop()
```

9.

```

import tkinter
from tkinter import *
from math import sqrt

def solver(a,b,c):
    """ Solves quadratic equation and returns the result in formatted string """
    D = b*b - 4*a*c
    if D >= 0:
        x1 = (-b + sqrt(D)) / (2*a)
        x2 = (-b - sqrt(D)) / (2*a)
        text = "The discriminant is: %s \n X1 is: %s \n X2 is: %s \n" % (D, x1, x2)
    else:
        text = "The discriminant is: %s \n This equation has two complex-conjugate
roots" % D
    return text

```

10.

```

import tkinter
from tkinter import *
from math import sqrt

def handler():
    """ Get the content of entries and passes result to the text """
    try:
        # make sure that we entered correct values
        a_val = float(a.get())
        b_val = float(b.get())
        c_val = float(c.get())
        inserter(solver(a_val, b_val, c_val))
    except ValueError:
        inserter("Make sure you entered 3 numbers")

```

Индивидуальные задания к разделу 1.

Воспроизведите представленные фрагменты программ на языке C++ / C# (учтите отличия в технике работы с декораторами в Python и C++/C#):

1.

```

class Borg:

    # __shared_state = {"1":"2"}

    my__shared_state = {"1":"2"}

    def __init__(self):

        self.x=7

        self.__dict__=self.my__shared_state

        pass

    def get_shared(self):

```

```
return self.my__shared_state
```

```
b1,b2=Borg(),Borg()
print(b1.get_shared())
print(b2.get_shared())
b1.x=4
b2.y=9
p1 = b1 is b2
print("Borg object 'b1': ",b1)
print("Borg object 'b2': ",b2)
print("Object state 'b1' :",b1.__dict__)
print("Object state 'b2' :",b2.__dict__)
print(b1.get_shared())
print(b2.get_shared())
print(p1);
```

2.

```
from abc import ABCMeta, abstractmethod
```

```
class PizzaFactory(metaclass = ABCMeta):
```

```
    @abstractmethod
    def createVegPizza(self):
        pass

    @abstractmethod
    def createNonVegPizza(self):
        pass;
```

3.

```
class IndianPizzaFactory(PizzaFactory):
```

```
def createVegPizza(self):  
    return DeluxVeggiePizza()
```

```
def createNonVegPizza(self):  
    return ChickenPizza()
```

4.

```
class USPizzaFactory(PizzaFactory):
```

```
    def createVegPizza(self):  
        return MexicanVegPizza()
```

```
    def createNonVegPizza(self):  
        return HamPizza()
```

5.

```
class VegPizza(metaclass = ABCMeta):
```

```
    @abstractmethod
```

```
    def prepare(self, VegPizza):  
        pass
```

6.

```
class NonVegPizza(metaclass = ABCMeta):
```

```
    @abstractmethod
```

```
    def serve(self, VegPizza):  
        pass
```

7.

```
class DeluxVeggiePizza(VegPizza):
```

```
def prepare(self):  
    print("Prepare ", type(self).__name__)
```

8.

```
class ChickenPizza(NonVegPizza):  
    def serve(self, VegPizza):  
        print(type(self).__name__, " is served with Chicken on ",  
              type(VegPizza).__name__)
```

9.

```
class MexicanVegPizza(VegPizza):  
    def prepare(self):  
        print("Prepare ", type(self).__name__)
```

10.

```
def makePizzas(self):  
    for factory in [IndianPizzaFactory(),USPizzaFactory()]:  
        self.factory = factory  
        self.NonVegPizza=self.factory.createNonVegPizza()  
        self.VegPizza=self.factory.createVegPizza()  
        self.VegPizza.prepare()  
        self.NonVegPizza.serve(self.VegPizza)
```

Индивидуальные задания к разделу 2.

Воспроизведите представленные фрагменты программ на языке C++ / C#. Некоторые операции, возможные при использовании языка Python, невозможны при использовании языка C++/C#. В таких случаях следует создавать аналоги.

Класс aPerson должен быть рассмотрен всеми студентами. Недостающие классы следует создать самостоятельно.

```
class aPerson:

    def __init__(self,first,last,age):

        self.__firstname,self.__lastname,self.__age = first,last,age

    def __str__(self):

        return self.__firstname + '+' + self.__lastname + '+' + str(self.__age)
```

1.

```
from aPerson import *
```

```
class anEmployee(aPerson):
```

```
    def __init__(self,first,last,age,staffnum):

        super().__init__(first,last,age)

        self.__staffnumber = staffnum

    def __str__(self):

        return super().__str__() + ', ' + self.__staffnumber
```

2.

```
class A:
```

```
    def __init__(self):

        print('A')
```

```
class B(A):
```

```
    def __init__(self):

        print('B')
```



```
super().__init__()
```

```
class C(A):
```

```
    def __init__(self):  
        print('C')  
        super().__init__()
```

```
class D(B,C):
```

```
    def __init__(self):  
        print('D')  
        super().__init__()
```

```
d=D()
```

3.

```
class aPerson:
```

```
    def __init__(self,first,last,age):  
        self.__firstname,self.__lastname,self.__age = first,last,age  
  
    def __str__(self):  
        return self.__firstname + ' '+self.__lastname + ' '+str(self.__age)
```

```
from aPerson import *
```

```
def Start():
```

```
    St1 = aPerson('Oleg','Kotov',20)  
    print(St1)
```

4.

```
class Length:
```

```
__metric = {'mm': 0.001, 'cm': 0.01, 'm': 1, 'km': 1000,  
            'in': 0.0254, 'ft': 0.3048, 'yd': 0.9144, 'mi': 1609.344}
```

```
def __init__(self, value, unit = 'm'):  
    self.value = value  
    self.unit = unit
```

5.

```
def Converse2Metres(self):  
    return self.value * Length.__metric[self.unit]
```

```
def __add__(self, other):  
    if type(other) == int or type(other) == float:  
        l = self.Converse2Metres() + other  
    else:  
        l = self.Converse2Metres() + other.Converse2Metres()  
    return Length(l /  
                  Length.__metric[self.unit], self.unit)
```

6.

```
def __str__(self):  
    return str(self.Converse2Metres())  
  
def __repr__(self):  
    return "Length("+str(self.value)+",""+self.unit+"")"  
  
def __iadd__(self, other):  
    if type(other) == int or type(other) == float:
```

```

        l = self.Converse2Metres() + other
    else:
        l = self.Converse2Metres() + other.Converse2Metres()
    self.value = l / Length.__metric[self.unit]
    return self

```

7.

```

from person import *
class Employee(Person):
    def __init__(self,first,last,age,staffnum):
        super().__init__(self,first,last,age)
        self.__staffnumber = staffnum
    def __str__(self):
        return super().__str__() + ' ' +self.__staffnumber

```

8.

```

class S(object):
    def __init__(self,v):
        self.val = v

```

```

x=S(42)
print(x.val)
x.new = 77
print(x.new)
print(x.__dir__())

```

9.

```

def dict_01():

```

```
"""Dictionary 1"""  
  
city_population = {'Bratislava': 410000, "Brno" : 430000,  
"Warszawa' : 1500000, "Praha" : 1100000, "Kyiv" : 220000}  
  
N_Kyiv = city_population['Kyiv']  
  
print(N_Kyiv)  
  
city_population["Kharkiv"]=1400000  
  
print(city_population)  
  
N_Warszawa = city_population["Warszawa"]  
  
print(N_Warszawa)  
  
print(city_population)
```

```
10.  
def dict_02():  
  
    """Dictionary 2"""  
  
    city = { }  
  
    tc = type(city)  
  
    print(city,tc,sep=' &&& ')  
  
    city['Ufa'],tc = 1100000, type(city)  
  
    print(city,tc,sep=' @@@ ')
```

Индивидуальные задания к разделу 3.

Воспроизведите представленные фрагменты программ на языке C++ / C#. Некоторые операции, возможные при использовании языка Python, невозможны при использовании языка C++/C#. В таких случаях следует создавать аналоги.

```
1.  
class EventManager(object):  
  
    def __init__(self):  
  
        print("Event Manager:: Let me talk to the folks\n")
```

```
def arrange(self):  
    self.hotelier = Hotelier()  
    self.hotelier.bookHotel()  
  
    self.florist = Florist()  
    self.florist.setFlowerRequirements()  
  
    self.caterer = Caterer()  
    self.caterer.setCuisine()  
  
    self.musician = Musician()  
    self.musician.setMusicType()
```

2.

```
class Hotelier(object):  
    def __init__(self):  
        print("Arranging the Hotel for Marriage?  ")  
  
    def __isAvailable(self):  
        print("Is the Hotel free for the event on given day?")  
        return True  
  
    def bookHotel(self):  
        if self.__isAvailable():  
            print("Registered the Booking\n\n")
```

3.

```
class Florist(object):  
    def __init__(self):
```

```
print("Flower Decorations for the Event? ")
```

```
def setFlowerRequirements(self):  
    print("""Carnations, Roses and Lillies would be used for  
    Decorations\n\n""")
```

4.

```
class Caterer(object):  
    def __init__(self):  
        print("Food Aragements for the Event ")  
  
    def setCuisine(self):  
        print("Chinese & Continental Cuisine to be served\n\n")
```

5.

```
class Musician(object):  
    def __init__(self):  
        print("Musical Arrangement for Marriage ")  
  
    def setMusicType(self):  
        print("Jazz and Classical will be played\n\n")
```

6.

```
class You(object):  
    def __init__(self):  
        print("You:: Whoa! Marriage Arragements??!!!!")  
  
    def askEventManager(self):  
        print("Let's Contact the Event Manager\n\n")
```

```
em = EventManager()
```

```
em.arrange()
```

```
def __del__(self):
```

```
    print("You:: Thanks to Event Manager, all preparations done!")
```

7.

```
class Actor(object):
```

```
    def __init__(self):
```

```
        self.isBusy = False
```

```
    def occupied(self):
```

```
        self.isBusy = True
```

```
        print(type(self).__name__,"is occupied with current movie")
```

```
    def available(self):
```

```
        self.isBusy = False
```

```
        print(type(self).__name__,"is free for the movie")
```

```
    def getStatus(self):
```

```
        return self.isBusy
```

8.

```
class Agent(object):
```

```
    def __init__(self):
```

```
        self.principal = None
```

```
    def work(self):
```

```
        self.actor = Actor()
```

```
    if self.actor.getStatus():
        self.actor.occupied
    else:
        self.actor.available()
```

9.

```
class Payment(metaclass = ABCMeta):
```

```
    @abstractmethod
```

```
    def do_pay(self):
```

```
        pass
```

```
class Bank(Payment):
```

```
    def __init__(self):
```

```
        self.card,self.account = None,None
```

```
    def __getAccount(self):
```

```
        self.account = self.card
```

```
        return self.account
```

```
    def __hasFunds(self):
```

```
        print("Bank:: Cheking if Account",self.__getAccount(),
```

```
              "has enough funds")
```

```
        return True
```

```
    def setCard(self,card):
```

```
        self.card = card
```

```
    def do_pay(self):
```

```
        if self.__hasFunds():
```



```

        print("Bank:: Paying the merchant")
        return True
    else:
        print("Bank:: Sorry, not enough funds!")
        return False

class DebitCard(Payment):
    def __init__(self):
        self.bank = Bank()

    def do_pay(self):
        card = input("Proxy:: Punch in Card Number:  ")
        self.bank.setCard(card)
        return self.bank.do_pay()

```

10.

```

class You:
    def __init__(self):
        print("You:: Let's buy the Denim shirt")
        self.debitcard = DebitCard()
        self.isPurchased = None

    def make_payment(self):
        self.isPurchased = self.debitcard.do_pay()

    def __del__(self):

```

```
if self.isPurchased:

    print("You:: Wow! Denim shirt is Mine:  ")

else:

    print("You:: I should earn more  :(")
```

Индивидуальные задания к разделу 4.

Задания выполняются на языках C++/C# и Python.

Вариант 1. Информационная система ВУЗа

Студенты, организованные в группы, учатся на одном из факультетов ВУЗа.

В учебном процессе участвуют преподаватели кафедр, административно относящихся к одному из факультетов. Преподаватели подразделяются на следующие категории: ассистенты, преподаватели, старшие преподаватели, доценты, профессора.

Учебный процесс регламентируется учебным планом. Учебный план содержит перечень учебных дисциплин с указанием курса и семестра для студентов каждого года набора, количества часов на каждый вид занятий по дисциплине (виды занятий: лекции, семинары, лабораторные работы, консультации, курсовые работы, ИР и т.д.) и формы контроля (зачет, экзамен). Перед началом учебного семестра деканаты раздают на кафедры учебные поручения, в которых указывается, какие кафедры (не обязательно относящиеся к данному факультету) какие дисциплины и для каких групп должны вести в очередном семестре. Преподаватель может вести занятия по одной или нескольким дисциплинам для студентов как своего, так и других факультетов. Сведения о проведенных экзаменах и зачетах собираются деканатом.

Обязательные функции информационной системы:

1. Получить перечень студентов указанных групп либо указанного курса факультета (по половому признаку, году рождения, возрасту, признаку наличия детей, по признаку получения и размеру стипендии).
2. Получить список преподавателей указанных кафедр либо указанного факультета полностью либо указанных категорий (ассистенты, доценты, профессора и т.д.) по половому признаку, году рождения, возрасту, признаку наличия и количеству детей.
3. Получить перечень кафедр, проводящих занятия в указанной группе либо на указанном курсе указанного факультета в указанном семестре, либо за указанный период.
4. Получить список и общее число преподавателей, проводивших (проводящих) занятия по указанной дисциплине в указанной группе либо на указанном курсе указанного факультета.
5. Получить перечень преподавателей, проводивших (проводящих) лекционные, семинарские и другие виды занятий в указанной группе либо на указанном курсе указанного факультета в указанном семестре (за указанный период).
6. Получить список и общее число студентов указанных групп, сдавших зачет либо экзамен по указанной дисциплине с указанной оценкой.
7. Получить список и общее число студентов указанных групп или указанного курса указанного факультета, сдавших указанную сессию на отлично, без троек, без двоек.
8. Получить перечень преподавателей, принимающих (принимавших) экзамены в указанных группах, по указанным дисциплинам, в указанном семестре.

Вариант 2. Информационная система проектной организации

Проектная организация представлена следующими категориями сотрудников: конструкторы, инженеры, техники, лаборанты, прочий обслуживающий персонал.

Сотрудники разделены на отделы, руководимые начальником. Каждый сотрудник числится только в одном отделе.

Проектная организация заключает договоры с заказчиками на выполнение проектов.

По одному договору может выполняться более одного проекта, и один проект может выполняться для нескольких договоров. Суммарная стоимость договора определяется стоимостью всех проектных работ, выполняемых для этого договора. Каждый договор и проект имеет руководителя и группу сотрудников, выполняющих этот договор или проект, причем это могут быть сотрудники не только одного отдела.

Ведется учет кадров, учет выполнения договоров и проектов, стоимостной учет всех выполненных работ.

Обязательные функции информационной системы:

1. Получить данные о составе указанного отдела или всей организации (по указанной категории сотрудников, по возрастному составу).
2. Получить перечень руководителей отделов.
3. Получить перечень договоров или проектов, выполняемых в данный момент или в указанный период времени.
4. Получить информацию о том, какие проекты выполняются (выполнялись) в рамках указанного договора и какие договоры поддерживаются указанными проектами.
5. Получить данные о стоимости выполненных договоров (проектов) в течение указанного периода времени.
6. Получить сведения об участии указанного сотрудника или категории сотрудников в проектах (договорах) за определенный период времени.
7. Получить данные о численности и составе сотрудников, участвующих в указанном проекте (в целом и по отдельным категориям).
8. Получить сведения об эффективности договоров (стоимость договоров, соотношенная с затраченным временем или стоимость с учетом привлеченных людских ресурсов).
9. Получить сведения об эффективности проектов (стоимость проектов, соотношенная с затраченным временем или стоимость с учетом привлеченных людских ресурсов).

Вариант 3. Информационная система авиастроительного предприятия

Структурно предприятие разбито на цеха, которые в свою очередь подразделяются на участки. Выпускаемые изделия предприятия - самолеты (гражданские, транспортные, военные), планеры, вертолеты, дельтапланы, ракеты (артиллерийские, авиационные, военно-морские), прочие изделия. По каждой категории изделий может собираться несколько видов изделий.

Сотрудники предприятия - инженерно-технический персонал (инженеры, технологи, техники) и рабочие (сборщики, токари, слесари, сварщики и пр.).

Рабочие объединяются в бригады, которыми руководят бригадиры. Бригадиры выбираются из числа рабочих, мастера, начальники участков и цехов назначаются из числа инженерно-технического персонала.

Каждое изделие собирается в своем цехе (в цехе может собираться несколько видов изделий) и в процессе изготовления проходит определенный цикл работ, перемещаясь с одного участка на другой. Все работы по сборке конкретного изделия на определенном участке выполняет одна бригада рабочих, при этом на участке может

работать несколько бригад. Возглавляет работу на участке начальник участка, в подчинении которого находится несколько мастеров. Различные изделия могут проходить одни и те же циклы работ на одних и тех же участках цеха.

Обязательные функции информационной системы:

1. Получить перечень видов изделий, собираемых указанным цехом (в целом и отдельной категории).
2. Получить число и перечень изделий (отдельной категории и в целом), собранных указанным цехом, участком, предприятием за определенный отрезок времени.
3. Получить данные о кадровом составе цеха, предприятия в целом и по указанным категориям инженерно-технического персонала и рабочих.
4. Получить перечень участков и их начальников указанного цеха или предприятия в целом.
5. Получить перечень работ, которые проходит указанное изделие.
6. Получить состав бригад указанного участка, цеха.
7. Получить список мастеров указанного участка, цеха.
8. Получить перечень изделий отдельной категории и в целом, собираемых в настоящий момент указанным участком, цехом, предприятием.
9. Получить состав бригад, участвующих в сборке указанного изделия.
10. Получить число и перечень изделий отдельной категории и в целом, собираемых указанным цехом, участком, предприятием в целом в настоящее время.

Вариант 4. Информационная система библиотечного фонда города

Библиотечный фонд города составляют библиотеки, расположенные на территории города. Каждая библиотека включает в себя абонементы и читальные залы.

Каждый читатель, будучи зарегистрированным в одной из библиотек, имеет доступ ко всему библиотечному фонду города.

Библиотечный фонд (книги, журналы, газеты, сборники статей, сборники стихов, диссертации, рефераты, сборники докладов и тезисов докладов и пр.) размещен в залах-хранилищах различных библиотек на определенных местах хранения (номер зала, стеллажа, полки) и идентифицируется номенклатурными номерами. При этом существуют различные правила относительно тех или иных изданий: какие-то подлежат только чтению в читальных залах библиотек; для тех, что выдаются, может быть установлен различный срок выдачи и т.д. С одной стороны, библиотечный фонд может пополняться, с другой, - с течением времени происходит его списание.

Произведения авторов, составляющие библиотечный фонд, также можно разделить на различные категории: учебники, повести, романы, статьи, стихи, диссертации, рефераты, тезисы докладов и т.д.

Сотрудники библиотеки, работающие в различных залах различных библиотек, ведут учет читателей, а также учет размещения и выдачи литературы.

Обязательные функции информационной системы:

1. Получить список читателей, на руках у которых находится указанное издание (книга, журнал и т.д).
2. Получить список литературы, которая в настоящий момент выдана с определенной полки некоторой библиотеки.
3. Получить список читателей с просроченным сроком литературы.
4. Получить перечень указанной литературы, которая поступила (была списана) в течение некоторого периода.
5. Выдать список библиотекарей, работающих в указанном читальном зале некоторой библиотеки.

6. Получить перечень читателей, которые в течение указанного промежутка времени получали издание с некоторым произведением, и название этого издания.
7. Получить список читателей, не посетивших библиотеку в течение указанного времени.
8. Выдать из библиотечного фонда указанной библиотеки список инвентарных номеров и названий произведений указанного автора.
9. Получить список самых популярных произведений.

Вариант 5. Информационная система спортивных организаций города

Спортивная инфраструктура города представлена спортивными сооружениями различного типа: спортивные залы, манежи, стадионы, корты и т.д.

Спортсмены под руководством тренеров занимаются отдельными видами спорта, при этом один и тот же спортсмен может заниматься несколькими видами спорта, и в рамках одного и того же вида спорта может тренироваться у нескольких тренеров. Все спортсмены объединяются в спортивные клубы, при этом каждый из них может выступать только за один клуб.

Организаторы соревнований проводят состязания по отдельным видам спорта на спортивных сооружениях города. По результатам участия спортсменов в соревнованиях производится награждение.

Обязательные функции информационной системы:

1. Получить перечень спортивных сооружений указанного типа в целом или удовлетворяющих заданным характеристикам (например, стадионы, вмещающие не менее указанного числа зрителей).
2. Получить список спортсменов, занимающихся указанным видом спорта.
3. Получить список спортсменов, тренирующихся у заданного тренера.
4. Получить список спортсменов, занимающихся более чем одним видом спорта с указанием этих видов спорта.
5. Получить список тренеров указанного спортсмена.
6. Получить перечень соревнований, проведенных в течение заданного периода времени в целом либо указанным организатором.
7. Получить список призеров указанного соревнования.
8. Получить перечень спортивных клубов и число спортсменов этих клубов, участвовавших в спортивных соревнованиях в течение заданного интервала времени.
9. Получить список тренеров по определенному виду спорта.
10. Получить список спортсменов, не участвовавших ни в каких соревнованиях в течение определенного периода времени.
11. Получить перечень спортивных сооружений и даты проведения на них соревнований в течение определенного периода времени.

Вариант 6. Информационная система гостиничного комплекса

Гостиничный комплекс состоит из нескольких зданий-гостиниц (корпусов). Каждый корпус имеет ряд характеристик, таких, как класс отеля (двух-, пятизвездочные), количество этажей в здании, общее количество комнат, комнат на этаже, местность номеров (одно-, двух-, трехместные и т.д.), наличие служб быта: ежедневная уборка номера, прачечная, химчистка, питание (рестораны, бары) и развлечения (бассейн, сауна, бильярд и пр.). От типа корпуса и местности номера

зависит сумма оплаты за него. Химчистка, стирка, дополнительное питание, все развлечения производятся за отдельную плату.

С крупными организациями (туристические фирмы, организации, занимающиеся проведением международных симпозиумов, конгрессов, семинаров, карнавалов и т.д.) заключаются договора, позволяющие организациям бронировать номера с большими скидками на определенное время вперед не для одного человека, а для группы людей. В брони указывается класс отеля, этаж, количество комнат. Броня может быть отменена за неделю до заселения.

Новые жильцы пополняют перечень клиентов гостиницы. Ведется учет свободных номеров, дополнительных затрат постояльцев гостиницы. Ведется учет долгов постояльца гостинице за все дополнительные услуги.

Обязательные функции информационной системы:

1. Получить перечень фирм, забронировавших места в объеме, не менее указанного, за весь период сотрудничества, либо за некоторый период.
2. Получить перечень постояльцев, заселявшихся в номера с указанными характеристиками за некоторый период.
3. Получить общее количество свободных номеров на данный момент.
4. Получить сведения о количестве свободных номеров с указанными характеристиками.
5. Получить список занятых номеров, которые освобождаются к указанному сроку.
6. Получить данные об объеме бронирования номеров данной фирмой за указанный период.
7. Получить сведения о фирмах, с которыми заключены договора о брони на указанный период.
8. Получить сведения о наиболее часто посещающих гостиницу постояльцах по всем корпусам гостиниц, по определенному зданию.
9. Получить сведения о новых клиентах за указанный период.
10. Получить сведения о конкретном человеке, сколько раз он посещал гостиницу, в каких номерах и в какой период останавливался, какие счета оплачивал.
11. Получить процентное отношение всех номеров к номерам, бронируемым партнерами.

Вариант 7. Информационная система библиотеки вуза

Библиотека включает в себя абонементы, читальные залы и справочную систему каталогов и картотек. Читателями библиотеки вуза имеют право быть: студенты всех форм обучения, профессорско-преподавательский состав, аспиранты, ассистенты и другие сотрудники подразделений вуза, слушатели подготовительного отделения (ПО), факультета повышения квалификации (ФПК), стажеры, абитуриенты. Слушатели ФПК, абитуриенты, стажеры - разовые читатели - имеют право пользоваться только читальными залами.

Читатели библиотеки имеют право получать книги и другие источники информации на всех пунктах выдачи библиотеки (абонементах и читальных залах), а также получать необходимые издания по межбиблиотечному абонементу, сделав предварительно заказ.

За нарушение правил пользования библиотекой читатели лишаются права пользования всеми пунктами обслуживания библиотеки на установленные администрацией сроки (от 1 до 6 месяцев). В случае невозвращения в библиотеку книг в установленный срок читатель обязан заплатить штраф.

При поступлении новых изданий в библиотеку они должны быть внесены в картотеку с указанием их количества для каждого абонемента и читального зала. Выдача книг, сроки, штрафы и т.п. собираются и обрабатываются администрацией.

Обязательные функции информационной системы:

1. Получить перечень и общее число читателей для данного читального зала или абонента, либо по всей библиотеке.
2. Получить список и общее число всех читателей-задолжников, задолжников со сроком более 10 дней на данном абонементе либо по всей библиотеке, по категориям читателей.
3. Получить перечень и общее число книг, поступивших и утерянных за последний год, для данного читального зала, абонемена или по всей библиотеке, по указанному автору, году выпуска, году поступления в библиотеку.
4. Получить перечень и общее число книг, заказанных на межбиблиотечном абонементе за последний месяц, год.
5. Получить количество экземпляров книги для данного читального зала или абонемена, во всей библиотеке, всех изданий.
6. Получить перечень и общее число читателей, лишенных права пользования библиотекой сроком более двух месяцев, во всей библиотеке, по категориям читателей.
7. Получить перечень и общее число книг, заказанных данным читателем за последний месяц, семестр, год, список книг, которые у него на руках.
8. Определить, есть ли данная книга в наличии на абонементах, и в каком количестве.
9. Получить перечень читателей, у которых на руках некоторая книга и читателя, который раньше всех ее должен сдать.

Вариант 8. Информационная система городской телефонной сети

ГТС представляет собой разветвленную сеть локальных АТС. АТС подразделяются на городские, ведомственные и учрежденческие. У каждой АТС есть свои абоненты. У абонента может стоять телефон одного из трех типов: основной, параллельный или спаренный. За каждым абонентом (у него есть фамилия, имя, отчество, пол, возраст и т.д.) закреплен свой номер телефона. Каждому номеру телефона соответствует адрес (индекс, район, улица, дом, квартира), причем параллельные или спаренные телефоны обязательно должны находиться в одном доме.

Все телефоны городской АТС имеют выход на междугороднюю связь, но для конкретного абонента он может быть либо открыт, либо закрыт по какой-либо причине (отключен по желанию абонента, за неуплату и т.п.). Сведения о междугородних переговорах собираются и анализируются на ГТС.

Абоненты обязаны платить абонентскую плату. Плата должна вноситься каждый месяц до 20-го числа. При неуплате после письменного уведомления в течение двух суток абонент отключается. При задолженности за междугородние разговоры и неоплате после письменного уведомления производится отключение только возможности выхода на междугороднюю связь. Включение того и (или) другого производится при оплате стоимости включения, абонентской платы и пени.

В городе также существуют общественные телефоны и таксофоны, расположенные по определенным адресам.

Обязательные функции информационной системы:

1. Получить перечень и общее число абонентов указанной АТС.
2. Получить перечень и общее число должников на указанной АТС, по всей ГТС, по данному району, абонентов, которые имеют задолженность уже больше недели

- (месяца), по признаку задолженности за междугороднюю связь и (или) по абонентской плате, по размеру долга.
3. Определить АТС (любого или конкретного типа), на которой самое большое (маленькое) число должников, самая большая сумма задолженности.
 4. Получить перечень и общее число общественных телефонов и таксофонов во всем городе, принадлежащих указанной АТС, по признаку расположения в данном районе.
 5. Получить перечень и общее число абонентов указанной АТС, по всей ГТС, по данному району, по типам АТС, имеющих параллельные телефоны.
 6. Определить, есть ли по данному адресу телефон, общее количество телефонов и (или) количество телефонов с выходом на междугороднюю связь, с открытым выходом на междугороднюю связь в данном доме, на конкретной улице.
 7. Определить город, с которым происходит большее количество междугородних переговоров.
 8. Получить перечень и общее число должников на указанной АТС, по всей ГТС, по данному району, которым следует послать письменное уведомление, отключить телефон и (или) выход на междугороднюю связь.

Вариант 9. Информационная система театра

Работников театра можно разделить на актеров, музыкантов, постановщиков и служащих. Актеры, музыканты и постановщики, работающие в театре, могут уезжать на гастроли. Актеры театра могут иметь звания заслуженных и народных артистов, могут быть лауреатами конкурсов. Также актерами театра могут быть и студенты театральных училищ. Каждый актер имеет свои вокальные и внешние данные (пол, возраст, голос, рост и т.п.), которые могут подходить для каких-то ролей, а для каких-то нет (не всегда женщина может сыграть мужчину и наоборот).

Для постановки любого спектакля необходимо подобрать актеров на роли и дублеров на каждую главную роль. Естественно, что один и тот же актер не может играть более одной роли в спектакле, но может играть несколько ролей в различных спектаклях. У спектакля также имеется режисер-постановщик, художник-постановщик, дирижер-постановщик, автор. Спектакли можно подразделить по жанрам: музыкальная комедия, трагедия, оперетта и пр. В репертуаре театра указывается какие спектакли, в какие дни и в какое время будут проходить, а также даты премьер. В кассах театра можно заранее приобрести билеты на любые спектакли. Администрацией театра фиксируется количество проданных билетов на каждый спектакль.

Обязательные функции информационной системы:

1. Получить список и общее число все работников театра, актеров, музыкантов, по стажу работы в театре, по половому признаку, году рождения, возрасту, признаку наличия и количества детей, размеру заработной платы.
2. Получить перечень и общее число спектаклей, указанных в репертуаре на данный сезон, уже сыгранных спектаклей, спектаклей указанного жанра, когда-либо сыгранных в этом театре, за указанный период.
3. Получить перечень и общее число всех поставленных спектаклей, спектаклей указанного жанра, когда-либо поставленных в этом театре, поставленных за указанный период.
4. Получить список авторов поставленных спектаклей, авторов, живших в указанном веке, авторов указанной страны, авторов спектаклей указанного жанра когда-либо поставленных в этом театре, поставленных за указанный период времени.
5. Получить список актеров, подходящих по своим данным на указанную роль.

6. Получить общее число и список актеров театра, имеющих звания.
7. Получить список для указанного спектакля: актеров, их дублеров, имена режисера-постановщика, художника-постановщика, дирижера-постановщика, авторов, дату премьеры.
8. Получить перечень и общее число ролей, сыгранных указанным актером всего, за некоторый период времени, в спектаклях определенного жанра, в спектаклях указанного режисера-постановщика.
9. Получить сведения о числе проданных билетов на все спектакли, на конкретный спектакль, на премьеры, за указанный период.
10. Получить общую сумму вырученных денег за указанный спектакль, за некоторый период времени.
11. Получить перечень и общее число свободных мест на все спектакли, на конкретный спектакль, на премьеры.

Вариант 10. Информационная система аэропорта

Работников аэропорта можно подразделить на пилотов, диспетчеров, техников, кассиров, работников службы безопасности, справочной службы и других, которые административно относятся каждый к своему отделу. В отделах существует разбиение работников на бригады. Отделы возглавляются начальниками, которые представляют собой администрацию аэропорта.

За каждым самолетом закрепляется бригада пилотов, техников и обслуживающего персонала. Пилоты обязаны проходить каждый год медосмотр, не прошедших медосмотр необходимо перевести на другую работу. Самолет должен своевременно осматриваться техниками и при необходимости ремонтироваться. Подготовка к рейсу включает в себя техническую часть (техосмотр, заправка необходимого количества топлива) и обслуживающую часть (уборка салона, запас продуктов питания и т.п.).

В расписании указывается тип самолета, рейс, дни вылета, время вылета и прилета, маршрут (начальный и конечный пункты назначения, пункт пересадки), стоимость билета. Билеты на авиарейсы можно приобрести заранее или забронировать в авиакассах. До отправления рейса, если в этом есть необходимость, билет можно вернуть. Авиарейсы могут быть отменены, если не продано меньше установленного минимума билетов.

Авиарейсы можно разделить на следующие категории: внутренние, международные, чартерные, грузоперевозки, специальные рейсы. Пассажир при посадке в самолет должен предъявить билет, паспорт, а для международного рейса обязан также предъявить заграничный паспорт и пройти таможенный досмотр. Пассажиры могут сдавать свои вещи в багажное отделение. На рейсы грузоперевозок и специальные рейсы билеты не продаются. Для спец. рейсов не существует расписания. Билеты на чартерные рейсы распространяет то агенство, которое его организовало.

Обязательные функции информационной системы:

1. Получить список и общее число всех работников аэропорта, начальников отделов, работников указанного отдела, по стажу работы в аэропорту, половому признаку, возрасту, признаку наличия и количеству детей.
2. Получить перечень и общее число работников в бригаде, по всем отделам, в указанном отделе, обслуживающих конкретный рейс.
3. Получить перечень и общее число пилотов, прошедших медосмотр либо не прошедших его в указанный год.
4. Получить перечень и общее число самолетов, приписанных к аэропорту.

5. Получить перечень и общее число самолетов, прошедших техосмотр за определенный период времени, отправленных в ремонт в указанное время, отремонтированных заданное число раз, по количеству совершенных рейсов до ремонта, по возрасту самолета.
6. Получить перечень и общее число рейсов по указанному маршруту, по длительности перелета.
7. Получить перечень и общее число отмененных рейсов полностью, в указанном направлении, по указанному маршруту, по количеству невостребованных мест.
8. Получить перечень и общее число рейсов, по которым летают самолеты заданного типа и среднее количество проданных билетов на определенные маршруты.
9. Получить перечень и общее число авиарейсов указанной категории, в определенном направлении, с указанным типом самолета.
10. Получить перечень и общее число пассажиров на данном рейсе, улетевших в указанный день, по признаку сдачи вещей в багажное отделение, по половому признаку, по возрасту.
11. Получить перечень и общее число свободных и забронированных мест на указанном рейсе, на определенный день, по указанному маршруту, по цене, по времени вылета.

Приложение 2. Средства для проведения промежуточной аттестации.

Код индикатора	Индикатор достижения компетенции	Оценочные средства
ПК-1: Способность к анализу проблемной ситуации, разработке требований к системе, постановке целей создания, разработке концепции и технического задания на создание Web-приложения, представления концепции, технического задания на Web-приложение и изменений в них заинтересованным лицам		
ПК-1.1	Анализирует требования к разработке Web-приложений и базам данных	<p><i>Дано задание на разработку WEB- приложения для оперативного информирования заказчиков продукции о состоянии их заказов. Набор состояний: заказ ожидает очереди, заказ выполняется (необходимо показать степень выполнения), заказ выполнен, заказ отправлен заказчику.</i></p> <ol style="list-style-type: none"> 1. Оценить полноту требований к приложению. 1. Разработать объектную схему пользовательского интерфейса. 2. Разработать объектную схему программы. 3. Предложить иерархию классов. 4. Определить отношения между классами. 5. Определить отношения между объектами (реализациями классов) <p><i>Реализовать программно иерархическую систему из 3 классов, связанных отношениями наследования. Выполняем в языке PHP (допустимо – в Django Python). Оценить характеристики разрабатываемого приложения и их соответствие требованиям.</i></p>
ПК-1.2	Оценивает качество разработки технических спецификаций на Web-приложения	<p><i>Для всех элементов разрабатываемого приложения предложить спецификации: набор входных параметров с указанием имени и типа параметра, набор выходных параметров с указанием имени и типа параметра.</i></p> <p><i>Для визуальных элементов определить характеристики: размеры, цветовое решение. Определить условия состояния видимости, невидимости, активности,</i></p>

Код индикатора	Индикатор достижения компетенции	Оценочные средства
		<i>неактивности. Для многоязычного интерфейса определить для каждого момента актуальный язык текста. Оценить качество каждой из спецификаций и их качество в совокупности.</i>
ПК-1.3	Оценивает качество проекта на разработку Web-приложения и баз данных	<i>Для разрабатываемого приложения оценить взаимодействие элементов. Соответствие времени обработки событий, транзакций в базе данных, взаимодействия с удаленными базами данных. Оценить затраты времени на обслуживание ресурса. Оценить реакцию ресурса на сбой оборудования. Оценить реакцию ресурса на некорректные данные. Оценить реакцию ресурса на некорректные действия пользователя.</i>
ПК-5: Способность к формализации и алгоритмизации поставленных задач, к написанию программного кода с использованием языков программирования, определения и манипулирования данными и оформлению программного кода в соответствии установленными требованиями		
ПК-5.1	Оценивает качество математической модели при формализации задачи предметной области	<i>Дано задание на разработку системы моделирования заявок на ремонт оборудования. Оценить математические модели. 1. Время поступления заявок на ремонт и его длительность заранее регламентированы. 2. Время поступления заявок на ремонт и длительность ремонта являются случайными величинами (рассмотреть несколько вариантов закона распределения данных случайных величин). 3. Рассмотреть процесс поступления заявок на ремонт, как вид марковского случайного процесса. Реализовать программно поток поступления заявок с использованием генерации псевдослучайных последовательностей. Реализовать программно проверку условий принадлежности случайного процесса марковскому типу. Программные реализации должны быть выполнены на двух языках программирования.</i>
ПК-5.2	Оценивает качество разработанных алгоритмов для последующего кодирования	<i>Дано задание на разработку системы моделирования заявок на ремонт оборудования. Оценить алгоритмы моделирования. 1. Параметры случайного процесса получаются моделированием случайных величин. 2. В алгоритме используются параметры теоретических законов распределения случайных величин. 3. В алгоритме используются непараметрические статистики. Выполнить программную реализацию предложенных процессов не менее, чем на двух языках программирования. Сопоставить полученные результаты по производительности.</i>
ПК-5.3	Оценивает выбор программных средств для программирования и манипулирования данными в соответствии установленными требованиями	<i>Создать пользовательский интерфейс для системы моделирования потока заявок на ремонт средствами одного из средств визуального моделирования во взаимодействии с PHP (Django Python)</i>

