МИНИСТЕРСТВО ТРАНСПОРТА РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«РОССИЙСКИЙ УНИВЕРСИТЕТ ТРАНСПОРТА»
(РУТ (МИИТ)

Рабочая программа дисциплины (модуля),
как компонент образовательной программы
высшего образования - программы бакалавриата
по направлению подготовки
09.03.01 Информатика и вычислительная техника,
утвержденной первым проректором РУТ (МИИТ)
Тимониным В.С.

РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ (МОДУЛЯ)

Объектно-ориентированное программирование

Направление подготовки: 09.03.01 Информатика и вычислительная техника

Направленность (профиль): IT-сервисы и технологии обработки данных на транспорте

Форма обучения: Очная

1. Общие сведения о дисциплине (модуле).

Целью освоения дисциплины (модуля) является - обучение студентов основам объектно-ориентированного проектирования и программирования в современных средах разработки программного обеспечения.

Задача освоения дисциплины (модуля) - получение знаний и практических навыков в области проектирования и разработки объектно-ориентированных программ.

2. Планируемые результаты обучения по дисциплине (модулю).

Перечень формируемых результатов освоения образовательной программы (компетенций) в результате обучения по дисциплине (модулю):

ОПК-9 - Способен осваивать методики использования программных средств для решения практических задач;

ПК-1 - Способен анализировать большие данные с использованием существующей в организации методологической и технологической инфраструктуры;

ПК-2 - Способен осуществлять концептуальное, функциональное и логическое проектирование систем среднего и крупного масштаба и сложности ;

ПК-3 - Способен осуществлять разработку требований и проектирование программного обеспечения;

ПК-7 - Способен к организации процессов разработки программного обеспечения .

Обучение по дисциплине (модулю) предполагает, что по его результатам обучающийся будет:

Знать:

- представление о функциональных возможностях языка,
- Методы разработки алгоритмов в парадигме объектно-ориентированного программирования,
- принципы объектно-ориентированного анализа и проектирования, основы объектно-ориентированного подхода к программированию,
- форматы и интерфейсы обмена данными между информационными системами,
- основы конфигурационного управления,
- виды архитектур информационных систем.

Уметь:

- эффективно использовать инструментарий высокоуровневых языков программирования для анализа больших данных,

- разрабатывать алгоритмы и компьютерные программы, пригодные для практического применения,

- Описывать задачу с точки зрения объектно-ориентированного подхода и подбирать соответствующие структуры для разработки алгоритма и программного кода,

- проводить объектную декомпозицию информационной системы, вырабатывать и обосновывать архитектурное решение,

- писать в современных средах разработки объектно-ориентированные программы.

Владеть:

- владеет навыками программирования, отладки и тестирования прототипов,

- основными приемами объектно-ориентированного программирования на языке Python,

- навыками использования библиотек Python, работой с классами и шаблонизаторами,

- системами контроля версий и поддержки конфигурационного управления, отслеживания ошибок,

- навыками работы в современных средах разработки и тестирований приложений, системах контроля версий и поддержки конфигурационного управления, отслеживания ошибок

3. Объем дисциплины (модуля).

3.1. Общая трудоемкость дисциплины (модуля).

Общая трудоемкость дисциплины (модуля) составляет 3 з.е. (108 академических часа(ов).

3.2. Объем дисциплины (модуля) в форме контактной работы обучающихся с педагогическими работниками и (или) лицами, привлекаемыми к реализации образовательной программы на иных условиях, при проведении учебных занятий:

Тип учебных занятий	Количество часов	
	Всего	Семестр №6
Контактная работа при проведении учебных занятий (всего):	64	64
В том числе:		
Занятия лекционного типа	16	16
Занятия семинарского типа	48	48

3.3. Объем дисциплины (модуля) в форме самостоятельной работы обучающихся, а также в форме контактной работы обучающихся с педагогическими работниками и (или) лицами, привлекаемыми к реализации образовательной программы на иных условиях, при проведении промежуточной аттестации составляет 44 академических часа (ов).

3.4. При обучении по индивидуальному учебному плану, в том числе при ускоренном обучении, объем дисциплины (модуля) может быть реализован полностью в форме самостоятельной работы обучающихся, а также в форме контактной работы обучающихся с педагогическими работниками и (или) лицами, привлекаемыми к реализации образовательной программы на иных условиях, при проведении промежуточной аттестации.

4. Содержание дисциплины (модуля).

4.1. Занятия лекционного типа.

№ п/п	Тематика лекционных занятий / краткое содержание
1	Тема 1. Парадигмы программирования, понятия классов и объектов. - Элементы объектно-ориентированного программирования (далее ООП). - История, принципы и преимущества ООП. - Классы и экземпляры классов, объекты. - Атрибуты и методы классов. Поиск в иерархии классов. - Деревья классов. Создание классов в программе.
2	Тема 2. Основные операторы создания класса ООП. - Оператор class, связи с деревьями классов. - Экземпляры, атрибуты, поля классов. - Параметр self в создании классов, принципы работы параметра self. Инициализатор __init__ в описании классов. Конструктор объекта __new__. Деструктор.
3	Тема 3. Методы классов в ООП. - Оператор функции def. Аргументы функции. - Определение методов класса Правила создания методов класса. - Вызовы методов класса в объектах. - Уровни методов класса. Параметр cls. Понятие «магического метода».
4	Тема 4. Уровни доступа к атрибутам класса. - Реализация режимов доступа к классам Python. - Публичный доступ к классу. Наследованный доступ к классу из дочерних классов. - Закрытие класса от доступа извне. - Расширенные функции доступа к классу.Метод __slots__.
5	Тема 5. Наследование в ООП Python. Основные правила наследования. Композиция классов.

№ п/п	Тематика лекционных занятий / краткое содержание
	- Роль наследования в ООП. - Родительские и дочерние классы. Суперклассы. - Пространство имен классов. Виды наследования. - Отличие композиции класса от наследования.
6	Тема 6. Наследование в ООП Python. Делегирование и доступ к классам при наследовании. -Функция super.() для применения при наследовании классов. - Делегирование атрибутов классов. - Наследование от встроенных типов type и от object.
7	Тема 7. Свойства классов в ООП Python. - Доступ к привычным методам наследованных классов. - Реализация свойств (property) классов для чтения и записи данных (геттеры и сеттеры). - Оптимизация программирования свойств. - Функция-декоратор @property.
8	Тема 8. Полиморфизм в ООП Python. - Изменение вычислительного результата простых арифметических операций. - Виды полиморфизма. Полиморфизм методов класса. - Переопределение методов суперклассов при наследовании. - Абстрактные классы и полиморфизм. - Создание абстрактных классов и применение их полиморфизма в объектах.
9	Тема 9. Множественное наследование в ООП Python. Миксины. - Задача множественного наследования. - Проблема «ромбов» наследования. Алгоритм C3 MRO. - Особенности применения функции super.(). - Дополнение функциональности класса применяя “mixins” без переопределения атрибутов при множественном наследовании.
10	Тема 10. Перегрузка операций в ООП. - Распространенные методы перегрузки операций. - Итерируемые объекты класса. - Методы __next__, __iter__. __getitem__, __setitem__, __delitem. Методы __getattribute__, __getattr__, __setattr__. Методы __str__, __repr__.
11	Тема 11. Декораторы в ООП. - Виды декораторов. - Преимущества и недостатки декораторов, область применения. - Правила программирования декораторов. - Объекты-декораторы и метод __call__ в Python. - Декораторы методов класса. Декораторы классов.
12	Тема 12. Метаклассы. Вложенные классы в ООП. - Метакласс и объект type. Связь между экземпляром, классом и метаклассом. - Правила создания метакласса. Области для применения метаклассов. - Преимущества вложенных классов в класс (без наследования). - Правила работы в коде с вложенными классами.

4.2. Занятия семинарского типа.

Практические занятия

№ п/п	Тематика практических занятий/краткое содержание
1	Тема 1. Правила формирования класса для программирования в IDE PyCharm. Отработка навыков создания простых классов и объектов класса. - Решение задач на понимание парадигмы классов, экземпляра класса, атрибутов класса и атрибутов объекта (экземпляра класса). - Создание простых классов и объектов класса включает определение класса с помощью ключевого слова class, определение атрибутов класса с помощью переменных экземпляра, используя синтаксис точки для доступа к атрибутам и методам экземпляра.
2	Тема 2. Правила использования стандартного первого аргумента self для методов объекта, методов инициализации и конструирования - Решение задач для понимания инициализации объекта с помощью метода __init__, - конструирования методом new, применение деструкторов класса.
3	Тема 3. Программирование композитных классов. - Создание экземпляров классов - объединение их поведения с помощью методов (композиция) в различных вычислительных задачах.
4	Тема 4. Программирование классов с различными уровнями доступа к полям и методам классов. Статические методы класса. - Правила использования синтаксиса программирования Python для указания уровня видимости атрибутов класса. - Решение задач с закрытыми полями и методами в классе, статическими атрибутами @staticmethod и @classmethod.
5	Тема 5. Программирование классов с одиночным наследованием. - Разбор различных задач наследования в классах от одного класса-родителя. - Создание новых классов на основе существующих классов, наследуя их атрибуты и методы. - Дочерний класс (подкласс или производный класс). - Наследование атрибутов и методов родительского класса (суперкласса или базового класса).
6	Тема 6. Программирование делегирования и доступ к классам при наследовании. - Использование функции super() для доступа к атрибутам и методам родительского класса из дочернего класса. - Определение атрибутов в родительском классе и их использование в дочернем классе. - Базовое поведение класса при делегировании атрибутов родительского класса. - Расширение поведения программы дочерними классами. - Наследование от встроенных типов, таких как int, str и list. Наследование от type и object.
7	Тема 7. Реализация свойств классов (property), чтения и записи данных (геттеры и сеттеры). - Программирование свойств классов с использованием декоратора property и геттеров/сеттеров, а также применение операторов Python, таких как is, not, and, or, not in, in, ==, !=, >, <, >=, <=, контроль доступ к атрибутам класса, обработка операции чтения и записи данных в/из свойств. - Программирование дескрипторов свойств класса.

№ п/п	Тематика практических занятий/краткое содержание
8	Тема 8. Реализация полиморфизма в классах. - Техника использования одного и того-же кода для разных типов данных, изменение вычислительного результата наследуемых методов. - Программирование структурного, функционального и параметрического полиморфизма. - Абстрактные классы - базовый класс с абстрактными методами, их реализация в производных классах.
9	Тема 9. Программирование множественного наследования в классах. Техника создания кода типа «mixin». - Наследование от нескольких суперклассов для создания более сложных и гибких программных систем. - Понимание работы алгоритма C3 MRO и его применение для эффективного использования множественного наследования. - Программирование расширения функциональности класса без переопределения его атрибутов при множественном наследовании. - Создание новой функциональности класса не изменяя его код и не влияя на поведение других классов, которые наследуются от него.
10	Тема 10. Программирование методов перегрузки операций в классах. - Создание собственных итерируемых классов методом __next__, - Создание собственных итерируемых классов методом __iter__.
11	Тема 11. Программирование методов перегрузки операций в классах. - Обработка вызовов свойств методами перегрузки операций __getattribute__, __getattr__, __setattr__. - Перегрузка операций вывода результатов, приведение к строке методами__str__ и __repr__.
12	Тема 12. Создание декораторов для методов и классов. - Рассмотрение различных способов создания объектов-декораторов, включая анонимные функции, функции-обертки и классы-обертки. - Практика использования метода _call_для вызова объекта-декоратора как функции.
13	Тема 13. Разработка простых метаклассов. Создание метакласса, записывающий в файл данные об его использовании другими классами. - Создание метакласса, который проверяет класс на запрет использования цифр в именах атрибутов и методов. - Создание функции, которая создает класс, на основе переданных ей названия, атрибутов и методов. - Создание классов ORM для работы с базой данных на основе метакласса.
14	Тема 14. Моделирование движения транспортного средства технологией объектно-ориентированного программирования. Разработка кода ООП по формулам расчета для моделирования ускорения автомобилей различных видов в наследуемых классах.

4.3. Самостоятельная работа обучающихся.

№ п/п	Вид самостоятельной работы
1	Работа с учебной литературой
2	Участие в онлайн-конференциях и мастер-классах

№ п/п	Вид самостоятельной работы
3	Поиск алгоритмов обработки данных в открытых источниках
4	Подготовка к промежуточной аттестации
5	Подготовка к текущему контролю.
6	Подготовка к промежуточной аттестации.
7	Подготовка к текущему контролю.

5. Перечень изданий, которые рекомендуется использовать при освоении дисциплины (модуля).

№ п/п	Библиографическое описание	Место доступа
1	Юре, Л. Анализ больших наборов данных / Л. Юре, Р. Ананд, Д. У. Джеффри ; перевод с английского А. А. Слинкин. — Москва : ДМК Пресс, 2016. — 498 с. — ISBN 978-5-97060-190-7	https://e.lanbook.com/book/93571
2	Кожомбердиева, Г. И. Программирование на языке Java: многопоточные приложения : учебное пособие / Г. И. Кожомбердиева. — Санкт-Петербург : ПГУПС, 2012. — 44 с. — ISBN 978-7641-0401-0	https://e.lanbook.com/book/64399
3	Риз, Р. Обработка естественного языка на Java : учебное пособие / Р. Риз ; перевод с английского А. В. Снастина. — Москва : ДМК Пресс, 2016. — 264 с. — ISBN 978-5-97060-331-4	https://e.lanbook.com/book/93272

6. Перечень современных профессиональных баз данных и информационных справочных систем, которые могут использоваться при освоении дисциплины (модуля).

https://habr.com/ru - база знаний в виде статей, обзоров

https://journal.tinkoff.ru/short/ai-for-all/ - база данных нейронных сетей

https://vc.ru/services/916617-luchshie-neyroseti-bolshaya-podborka-iz-top-200-ii-generatorov-po-kategoriyam - база данных нейронных сетей

https://github.com/abalmumcu/bert-rest-api - профессиональная платформа для командой работы над проектов (нейронная сеть bert)

http://library.miit.ru/ - электронно-библиотечная система Научно-технической библиотеки МИИТ

https://proglib.io/p/raspoznavanie-obektov-s-pomoshchyu-yolo-v3-na-tensorflow-2-0-2020-11-08 - профессиональная библиотека программистов

https://yandex.cloud/ru/blog/posts/2022/12/andrey-berger-and-yandex-cloud?utm_referrer=https%3A%2F%2Fyandex.ru%2F – библиотека профессиональных статей разработчиков Яндекс

https://yandex.cloud/ru/blog - библиотека профессиональных статей разработчиков Яндекс

https://tproger.ru/translations/opencv-python-guide - библиотека основных команд OpenCV

7. Перечень лицензионного и свободно распространяемого программного обеспечения, в том числе отечественного производства, необходимого для освоения дисциплины (модуля).

Java EE 7 SD
Microsoft Office 2007
VS code

8. Описание материально-технической базы, необходимой для осуществления образовательного процесса по дисциплине (модулю).

Компьютер преподавателя
Компьютеры студентов
экран для проектора, маркерная доска,
Проектор

9. Форма промежуточной аттестации:

Экзамен в 6 семестре.

10. Оценочные материалы.

Оценочные материалы, применяемые при проведении промежуточной аттестации, разрабатываются в соответствии с локальным нормативным актом РУТ (МИИТ).

Авторы:

доцент, к.н. Академии "Высшая инженерная школа" Б.В. Игольников

Согласовано:

Заместитель директора академии Д.В. Паринов

Председатель учебно-методической комиссии Д.В. Паринов