

**Федеральное государственное автономное образовательное учреждение
высшего образования
«Санкт-Петербургский политехнический университет Петра Великого»**

УТВЕРЖДАЮ
Директор ИКНК
_____ Д.П. Зегжда
«17» июня 2025 г.

РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ (МОДУЛЯ)

«Объектно-ориентированное программирование»

Разработчик	Высшая школа программной инженерии
Направление (специальность) подготовки	09.03.04 Программная инженерия
Наименование ООП	09.03.04_01 Технология разработки и сопровождения качественного программного продукта
Квалификация (степень) выпускника	бакалавр
Образовательный стандарт	СУОС
Форма обучения	Очная

СОГЛАСОВАНО	Соответствует СУОС
Руководитель ОП	Утверждена протоколом заседания
_____ А.В. Петров	высшей школы "ВШПИ" от «21» мая 2024 г. № 1

РПД разработали:

Специалист по учебно-методической работе 1 категории Т.А. Вишневская
Старший преподаватель А.П. Маслаков

1. Цели и планируемые результаты изучения дисциплины

Цели освоения дисциплины

Целью изучения данного раздела является формирование у будущих специалистов представления о принципах разработки программ, понимания особенностей обработки информации в ЭВМ, освоение структурного программирования, принципов декомпозиции поставленных задач с использованием объектно-ориентированного программирования на одном из языков высокого уровня (Java). Данный курс предназначен для изучения преимущества использования объектно-ориентированного языка, овладения технологиями разработки программного обеспечения на высокоуровневом языке (Java), изучения принципов управления параллельными вычислениями, овладения навыками разработки enterprise-приложений с использованием современных библиотек и фреймворков. Курс способствует умению формализовать вычислительные процессы, координировать их работу и распределять ресурсы между ними в рамках моделей асинхронных процессов. Получение опыта разработки программ с использованием объектно-ориентированного программирования и отладки приложений с использованием возможностей интегрированных сред разработки

Результаты обучения выпускника

Код	Результат обучения (компетенция) выпускника ООП
ОПК-6	Способен разрабатывать алгоритмы и программы, пригодные для практического использования, применять основы информатики и программирования к проектированию, конструированию и тестированию программных продуктов
ИД-5 ОПК-6	Разрабатывает программное обеспечение на основе шаблонов и методов ООП
ОПК-8	Способен осуществлять поиск, хранение, обработку и анализ информации из различных источников и баз данных, представлять ее в требуемом формате с использованием информационных, компьютерных и сетевых технологий
ИД-2 ОПК-8	Разрабатывает программное обеспечение для обработки и анализа информации с применением принципов ООП

Планируемые результаты изучения дисциплины

знания:

- Знает основы объектно-ориентированной парадигмы
- Знает основы объектно-ориентированной парадигмы

умения:

- Умеет применять шаблоны и методы ООП при разработке программного обеспечения
- Умеет создавать информационные системы с использованием методов ООП

навыки:

- Владеет навыками разработки программного обеспечения на одном из объектно-ориентированных языков программирования
- Владеет современным инструментарием для разработки программного обеспечения

2. Место дисциплины в структуре ООП

В учебном плане дисциплина «Объектно-ориентированное программирование» относится к модулю «Модуль цифровых компетенций» / «Программирование».

Изучение дисциплины базируется на результатах освоения следующих дисциплин:

- Алгоритмизация и программирование

3. Распределение трудоёмкости освоения дисциплины по видам учебной работы и формы текущего контроля и промежуточной аттестации

3.1. Виды учебной работы

Виды учебной работы	Трудоемкость по семестрам
	Очная форма
Лекционные занятия	28
Практические занятия	18
Самостоятельная работа	74
Промежуточная аттестация (зачет)	4
Промежуточная аттестация (зачет с оценкой)	6
Курсовое проектирование	14
Общая трудоемкость освоения дисциплины	144, ач
	4, зет

3.2. Формы текущего контроля и промежуточной аттестации

Формы текущего контроля и промежуточной аттестации	Количество по семестрам
	Очная форма
Текущий контроль	
Курсовые работы, шт.	1
Промежуточная аттестация	
Зачеты, шт.	1
Зачеты с оценкой, шт.	1

4. Содержание и результаты обучения

4.1 Разделы дисциплины и виды учебной работы

№ раздела	Разделы дисциплины, мероприятия текущего контроля	Очная форма		
		Лек, ач	Пр, ач	СР, ач

1.	Основные языковые конструкции	2	1	4
2.	Классы и объекты. Принципы ООП.	1	2	4
3.	Интерфейсы.	1	1	4
4.	Аннотации.	2	1	3
5.	Обработка ошибок	2	1	4
6.	Обобщения	1	1	6
7.	Коллекции	1	1	6
8.	Рефлексия	1	1	6
9.	Лямбда-выражения	1	1	3
10.	Stream-API	2	2	6
11.	Многопоточность	8	2	16
12.	JVM и сборка мусора	2	0	4
13.	Разработка графических пользовательских интерфейсов	2	2	2
14.	Сборка проектов	2	2	6
Итого по видам учебной работы:		28	18	74
Зачеты, ач				0
Зачеты с оценкой, ач				0
Часы на контроль, ач				0
Курсовое проектирование				14
Промежуточная аттестация (зачет)				4
Промежуточная аттестация (зачет с оценкой)				6
Общая трудоёмкость освоения: ач / зет				144 / 4

4.2. Содержание разделов и результаты изучения дисциплины

Раздел дисциплины	Содержание
1. Основные языковые конструкции	История языка. JRE и JDK. Правила именования. Примитивные типы данных и операции над ними, преобразование типов, классы-обёртки. Операторы цикла и ветвлений. Одномерные и многомерные массивы.
2. Классы и объекты. Принципы ООП.	Структура класса, модификаторы членов класса. Поля, методы, конструкторы, вложенные классы. Абстрактные методы и классы. Класс String. Потоки ввода-вывода. Принципы ООП. Методы класса Object.
3. Интерфейсы.	Общее описание и назначение. Проектирование интерфейсов. Модификаторы и модификаторы по-умолчанию. Реализация интерфейсов в классах. Свойства интерфейсов. Эволюция интерфейсов по версиям языка.
4. Аннотации.	Общее описание и назначение. Синтаксис и примеры стандартных аннотаций. Создание собственных аннотаций, аннотации с параметром.
5. Обработка ошибок	Понятие об исключениях в Java. Throwable и Exception. Проверяемые и не проверяемые исключения. Ключевые слова, используемые при обработке исключений. Логирование. Утверждения.
6. Обобщения	Общее описание и назначение. Синтаксис дженериков. Ограниченные типы. Wildcards и ограниченный Wildcards
7. Коллекции	Структура и иерархия collection framework. Интерфейс Collection. Итераторы. Реализации коллекций и словарей.
8. Рефлексия	Определение и назначение. Взаимодействие с кодом в время работы программы. Доступ к приватным членам классов.
9. Лямбда-выражения	Синтаксис. Функциональные интерфейсы. Свойства лямбда-выражений. Встроенные функциональные интерфейсы.
10. Stream-API	Назначение, составные части выражения, особенности работы. Потоки для работы с примитивами. Параллельные потоки данных. Класс Optional.
11. Многопоточность	Процессы и потоки. Создание и запуск потоков. Состояния потоков и синхронизация. Библиотека Java Concurrency.

12. JVM и сборка мусора	Принцип работы виртуальной java-машины. Типы сборщиков мусора в стандартной поставке jdk. Сборка мусора по поколениям и без.
13. Разработка графических пользовательских интерфейсов	Библиотеки разработки графических пользовательских интерфейсов - AWT, SWING, JavaFX, SWT. Подключение к проекту, преимущества и недостатки.
14. Сборка проектов	История сборки проектов на java. Модульность в языке Java. Сборщики проектов ANT, Maven, Gradle - подключение, репозитории, особенности работы.

5. Образовательные технологии

Успешное изучение курса требует посещения лекций, активной работы на лабораторных работах, выполнения всех учебных заданий преподавателя, ознакомления с основной и дополнительной литературой.

6. Лабораторный практикум

Не предусмотрено

7. Практические занятия

№ раздела	Наименование практических занятий (семинаров)	Трудоемкость, ач
		Очная форма
1.	В компьютерной игре герой (класс Hero) может перемещаться между двумя точками (метод move) различными способами: идти пешком, ехать на лошади, лететь и т. п. Реализовать классы, позволяющие выбирать и менять в ходе выполнения программы способ перемещения героя, используя паттерн “стратегия” (strategy). Продемонстрировать работу реализованных классов	2
2.	Написать консольное приложение, которое: а. Считывает из текстового файла размерность матрицы $N \times N$. б. Создаёт и заполняет матрицу случайными числами от $-N$ до N . в. Последовательно поворачивает матрицу на 90, 180 и 270 градусов против часовой стрелки и делит каждый элемент на сумму соседних. д. Каждую из трёх получившихся матриц вывести в общий файл Требования к обработке исключительных ситуаций: а. контролировать состояние потоков ввода/вывода (отсутствие записи в файле, недопустимые значения, etc); б. генерировать и обрабатывать исключение при некорректных математических операциях; с. выбрасывать исключение при нехватке памяти; д. реализовать собственные классы исключений для случаев • деление на 0 • файл не существует • $N > 1\,000\,000$	2

3.	<p>Создайте иерархию животного царства на свое усмотрение - выбираем один тип (простейшие/губки/хордовые и т. д.), а далее наследуйте классы, отряды, семейства, роды и виды) - должна получиться иерархия в 6 уровней, выбор животных на ваш вкус, особо много создавать не нужно. Затем, для получившейся иерархии, выполняем следующее: а. Создаем обобщенный класс Queue, представляющий из себя очередь фиксированного размера со стандартными методами очереди - add и get b. Создаем методы produce и consume: первый метод должен возвращать upper bound generic очереди (например, наследники позвоночных) из n животных, которая будет генерироваться на ваше усмотрение и подаваться во второй метод. Consume будет их распределять в 2 или более lower bound очереди - например, родители кошек и родители змей, выбор типов также остаётся за вами. с. Демонстрируем работу всех методов на конкретных собственных кейса</p>		2
4.	<p>Написать аннотацию с целочисленным параметром. Создать класс, содержащий только приватные методы (3–4 шт.), аннотировать любые из них. Вызвать из другого класса все аннотированные методы столько раз, сколько указано в параметре аннотации.</p>		4
5.	<p>С использованием StreamAPI реализовать следующие методы:</p> <ul style="list-style-type: none"> ● метод, возвращающий среднее значение списка целых чисел; ● метод, приводящий все строки в списке в верхний регистр и добавляющий к ним префикс «_new_»; ● метод, возвращающий список квадратов всех встречающихся только один раз элементов списка; ● метод, принимающий на вход коллекцию строк и возвращающий все строки, начинающиеся с заданной буквы, отсортированные по алфавиту; ● метод, принимающий на вход коллекцию и возвращающий её последний элемент или кидаящий исключение, если коллекция пуста; ● метод, принимающий на вход массив целых чисел, возвращающий сумму чётных чисел или 0, если чётных чисел нет; ● метод, преобразовывающий все строки в списке в Map, где первый символ – ключ, оставшиеся – значение; 		2

6.	<p>Создать супервизор (управляющую программу), которая контролирует исполнение абстрактной программы. Абстрактная программа работает в отдельном потоке и является классом с полем перечисляемого типа, который отражает ее состояние (UNKNOWN, STOPPING, RUNNING, FATAL ERROR) и имеет поток- демон случайного состояния, который в заданном интервале меняет её состояние на случайное. У супервизора должны быть методы остановки и запуска абстрактной программы, которые меняют ее состояние. Супервизор является потоком, который циклически опрашивает абстрактную программу, и если ее состояние UNKNOWN или STOPPING, то перезапускает ее. Если состояние FATAL ERROR, то работа абстрактной программы завершается супервизором. Все изменения состояний должны сопровождаться соответствующими сообщениями в консоли. Использовать конструкции с wait/notify</p>	3
7.	<p>Создать очередь сообщений, в которую пишут N потоков (количество потоков задается через args), и читают N потоков, использовать только пакет java.util.concurrent. Имена потоков должны быть осмыслены, использовать конструкции с wait/notify запрещено.</p>	3
Итого часов		18

8. Организация и учебно-методическое обеспечение самостоятельной работы

Примерное распределение времени самостоятельной работы студентов

Вид самостоятельной работы	Примерная трудоемкость, ач
	Очная форма
Текущая СР	
работа с лекционным материалом, с учебной литературой	10
опережающая самостоятельная работа (изучение нового материала до его изложения на занятиях)	4
самостоятельное изучение разделов дисциплины	6
выполнение домашних заданий, домашних контрольных работ	8
подготовка к лабораторным работам, к практическим и семинарским занятиям	10
подготовка к контрольным работам, коллоквиумам	0
Итого текущей СР:	38
Творческая проблемно-ориентированная СР	
выполнение расчётно-графических работ	0
выполнение курсового проекта или курсовой работы	20
поиск, изучение и презентация информации по заданной проблеме, анализ научных публикаций по заданной теме	10
работа над междисциплинарным проектом	0
исследовательская работа, участие в конференциях, семинарах, олимпиадах	0
анализ данных по заданной теме, выполнение расчётов, составление схем и моделей на основе собранных данных	6
Итого творческой СР:	36
Общая трудоемкость СР:	74

9. Учебно-методическое обеспечение дисциплины

9.1. Адрес сайта курса

<https://dl.spbstu.ru/course/view.php?id=1858>

9.2. Рекомендуемая литература

Основная литература

№	Автор, название, место издания, издательство, год (годы) издания	Год изд.	Источник
1	Давыдов В.Г. Язык и технологии Java, 2012. URL: http://elib.spbstu.ru/dl/pwd/2441.pdf	2012	ЭБ СПбПУ

Ресурсы Интернета

1. Java-хаб: <https://habr.com/ru/hub/java/>
2. Документация Spring Framework: <https://docs.spring.io/spring/docs/current/spring-framework-reference/>

9.3. Технические средства обеспечения дисциплины

Среда разработки Eclipse или IntelliJ idea

open_jdk версии 11 или выше

СУБД H2

Система контроля версий git

JavaFX sdk

10. Материально-техническое обеспечение дисциплины

Для успешного проведения лабораторных занятий необходимо использование компьютерного класса, имеющего не менее 10 компьютеров, оснащенных необходимым программным обеспечением

11. Критерии оценивания и оценочные средства

11.1. Критерии оценивания

Для дисциплины «Объектно-ориентированное программирование» предусмотрены следующие формы аттестации: зачёт, зачёт с оценкой. Дисциплина реализуется с применением системы индивидуальных достижений.

Текущий контроль успеваемости

Максимальное значение персонального суммарного результата обучения (ПСРО) по приведенной шкале - 100 баллов

Максимальное количество баллов приведенной шкалы по результатам прохождения двух точек контроля - 80 баллов.

Подробное описание правил проведения текущего контроля с указанием баллов по каждому контрольному мероприятию и критериев выставления оценки размещается в СДО в навигационном курсе дисциплины.

Промежуточная аттестация по дисциплине

Максимальное количество баллов по результатам проведения аттестационного испытания в период промежуточной аттестации – 20 баллов приведенной шкалы.

Промежуточная аттестация по дисциплине проводится в соответствии с расписанием.

Соответствие знаний и навыков программирования поставленным целям и задачам курса.

В качестве критерия оценивания знаний и умений по дисциплине производится оценка качества выполненных лабораторных работ. Зачет ставится в случае удовлетворительной сдачи не менее 7 программ, принимаемых на лабораторных занятиях

Результаты промежуточной аттестации, определяются на основе баллов, набранных в рамках применения, СИД

Баллы по приведенной шкале в рамках применения СИД (ПСРО+ ПА)	Оценка по результатам промежуточной аттестации
	Экзамен/диф.зачет/зачет
0 - 60 баллов	Неудовлетворительно/не зачтено
61 - 75 баллов	Удовлетворительно/зачтено

Баллы по приведенной шкале в рамках применения СИД (ПСРО+ ПА)	Оценка по результатам промежуточной аттестации
	Экзамен/диф.зачет/зачет
76 - 89 баллов	Хорошо/зачтено
90 и более	Отлично/зачтено

11.2. Оценочные средства

Оценочные средства по дисциплине представлены в фонде оценочных средств, который является неотъемлемой частью основной образовательной программы и размещается в электронной информационно-образовательной среде СПбПУ на портале etk.spbstu.ru

12. Методические рекомендации по организации изучения дисциплины

Для лучшего усвоения навыков процедурного и объектно-ориентированного программирования целесообразно увеличить число часов лабораторных занятий за счет практикума по программированию (факультатив).

Для лучшего усвоения навыков объектно-ориентированного программирования и получения знаний по последним достижениям в средствах разработки, отладки программ целесообразно предоставлять свободный доступ к Internet ресурсам на практических занятиях.

Для лучшего усвоения навыков объектно-ориентированного программирования и получения знаний по последним достижениям в средствах разработки, отладки программ целесообразно предоставлять свободный доступ к ресурсу github.com на практических занятиях.

Успешное изучение курса требует посещения лекций, активной работы на лабораторных работах, выполнения всех учебных заданий преподавателя, ознакомления с основной и дополнительной литературой.

Во время лекции студент должен вести краткий конспект. Если какие-то материалы конспекта вызывают затруднения, необходимо постараться найти ответы на затруднительные вопросы, используя рекомендуемую литературу. Если студенту самостоятельно не удалось разобраться в материале, необходимо сформулировать вопросы и обратится за помощью к преподавателю на консультации или ближайшей лекции. Обучающемуся необходимо регулярно отводить время для повторения пройденного материала, проверяя свои знания, умения и навыки по контрольным вопросам.

Лабораторные работы составляют важную часть профессиональной подготовки студентов. Они направлены на подтверждение теоретических положений и формирование учебных и профессиональных практических умений.

При подготовке к экзамену в дополнение к изучению конспектов лекций и учебных пособий, необходимо пользоваться учебной литературой, рекомендованной в настоящей программе. При подготовке к экзамену нужно изучить теорию: определения всех понятий и подходы к оцениванию до состояния понимания материала и самостоятельно решить по нескольку типовых задач из каждой темы. При решении задач всегда необходимо уметь качественно интерпретировать итог решения.

13. Адаптация рабочей программы для лиц с ОВЗ

Адаптированная программа разрабатывается при наличии заявления со стороны обучающегося (родителей, законных представителей) и медицинских показаний (рекомендациями психолого-медико-педагогической комиссии). Для инвалидов адаптированная образовательная программа разрабатывается в соответствии с индивидуальной программой реабилитации.