

**Федеральное государственное автономное образовательное  
учреждение высшего образования  
«Московский физико-технический институт  
(национальный исследовательский университет)»**

**УТВЕРЖДЕНО**

**Директор по цифровизации  
образования**

**Д.И. Гриц**

	<b>Рабочая программа дисциплины (модуля)</b>
<b>по дисциплине:</b>	Java для Data Science
<b>по направлению:</b>	Бизнес-информатика
<b>профиль подготовки:</b>	Финансовые технологии и аналитика центр дополнительного, дополнительного профессионального и онлайн-образования "Пуск" центр дополнительного, дополнительного профессионального и онлайн-образования "Пуск"
<b>курс:</b>	1
<b>квалификация:</b>	магистр

Семестр, формы промежуточной аттестации: 1 (осенний) - Экзамен

Аудиторных часов: 36 всего, в том числе:

лекции: 18 час.

семинары: 18 час.

лабораторные занятия: 0 час.

Самостоятельная работа: 69 час.

Подготовка к экзамену: 30 час.

Всего часов: 135, всего зач. ед.: 3

Программу составили:

Е.А. Савицкая, начальник отдела

О.А. Культепина, методист

Программа обсуждена на заседании центра дополнительного, дополнительного профессионального и онлайн-образования "Пуск" 13.06.2022

## Аннотация

В рамках дисциплины «Java для Data Science» обучающиеся познакомятся со средствами объектно-ориентированного программирования, изучат язык и платформу Java, научатся создавать собственные продукты для управления данными.

### 1. Цели и задачи

#### Цель дисциплины

- овладеть практическими навыками использования Java-технологий для подготовки и анализа больших данных.

#### Задачи дисциплины

- освоение языка программирования и платформы Java;
- изучение базовых понятий и принципов объектно-ориентированного программирования;
- изучение особенностей объектно-ориентированного программирования в Java;
- развитие навыков применения языка программирования Java для работы с большими данными.

### 2. Перечень формируемых компетенций

Освоение дисциплины направлено на формирование следующих компетенций:

Код и наименование компетенции	Индикаторы достижения компетенции
ПК-17 Способен применять методы системного анализа и моделирования для анализа, совершенствования и проектирования архитектуры предприятия	ПК-17.1 Понимает и использует математические методы для информационно-аналитической поддержки принятия решений
	ПК-17.2 Умеет применить программный инструментарий для изменения архитектуры предприятия

### 3. Перечень планируемых результатов обучения по дисциплине (модулю)

В результате освоения дисциплины обучающиеся должны

знать:

- объектно-ориентированное программирование в Java;
- структуру программы на Java;
- инструменты и методы командной работы на платформе GitHub;
- шаблоны проектирования в Java;
- многопоточное программирование;
- Data Science с позиции Java разработчика.

уметь:

- объектно-ориентированное программирование в Java;
- структуру программы на Java;
- инструменты и методы командной работы на платформе GitHub;
- шаблоны проектирования в Java;
- многопоточное программирование;
- Data Science с позиции Java разработчика.

владеть:

- навыками работы с файлами в Java;
- навыками применения инструментов Java в Data Science.

### 4. Содержание дисциплины (модуля), структурированное по темам (разделам) с указанием отведенного на них количества академических часов и видов учебных занятий

#### 4.1. Разделы дисциплины (модуля) и трудоемкости по видам учебных занятий

	Трудоемкость по видам учебных занятий, включая самостоятельную работу, час.
--	-----------------------------------------------------------------------------

№	Тема (раздел) дисциплины	Лекции	Семинары	Лаборат. работы	Самост. работа
1	Основы Java разработки	4	4		14
2	Java Core	4	4		14
3	Система контроля версий «Git»	4	4		14
4	Шаблоны проектирования в Java	4	4		14
5	Многопоточное и функциональное программирование в Java	2	2		13
Итого часов		18	18		69
Подготовка к экзамену		30 час.			
Общая трудоёмкость		135 час., 3 зач.ед.			

#### 4.2. Содержание дисциплины (модуля), структурированное по темам (разделам)

Семестр: 1 (Осенний)

##### 1. Основы Java разработки

Основной понятийный аппарат для Java-программиста. Введение в программирование на языке Java. Структура программы на Java. Условные операторы и циклы в Java. Типы данных: примитивы и объекты. Одномерные и многомерные массивы. Объектно-ориентированное программирование в Java. Структура класса в Java. Модификаторы доступа и наследование в Java. Полиморфизм в Java. Абстракции и интерфейсы в Java. Исключения, Stacktrace. Generics в коллекциях и методах. Коллекции List. Коллекции Queue. Коллекции HashMap и HashSet. Коллекции TreeMap и TreeSet.

##### 2. Java Core

Лямбда-выражения и функциональные интерфейсы в Java. Работа со структурами данных с помощью инструмента Stream API. Методы работы, содержание и анализ потоков, повторных вызовов. Потоки ввода-вывода. Работа с файлами в Java. Процессы перевода структур в Java в последовательности (Сериализация). Сборка проектов и фреймворки Maven и Gradle. Работа с файлами CSV, XML, JSON. Тестирование кода и Unit-тесты. Инструмент Mockito. Мокирование вызовов, Mock. Основы работы с сетью: модели OSI, TCP, UDP. Протокол HTTP и вызов удалённых серверов. Виртуальная машина JVM. Организация памяти, сборщики мусора, VisualVM

##### 3. Система контроля версий «Git»

Внедрение системы контроля версий разрабатываемого приложения с помощью использования сервиса GitHub. История работы разработчика в сервисе и структура сервиса. Инструменты и методы командной работы на платформе GitHub. Формирование портфолио на базе системы GitHub.

##### 4. Шаблоны проектирования в Java

Внедрение системы контроля версий разрабатываемого приложения с помощью использования сервиса GitHub. История работы разработчика в сервисе и структура сервиса. Инструменты и методы командной работы на платформе GitHub. Формирование портфолио на базе системы GitHub.

##### 5. Многопоточное и функциональное программирование в Java

Определение многопоточного программирования. Многопоточное (параллельное) программирование. Создание и запуск потоков. Содержание работы с синхронизацией. Переменные многопоточной программы. Коллекции для параллельной (конкурирующей) работы. Клиент-серверное взаимодействие. Пакеты Blocking и Non-Blocking IO. Функциональное программирование в Java среде. Понятия TDD и DDD. Методология разработки TDD и DDD.

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

Занятия по учебной дисциплине проводятся с использованием дистанционных образовательных технологий. Каждый обучающийся обеспечен доступом к образовательной платформе <https://netology.ru/>.

## **6.Перечень рекомендуемой литературы**

Основная литература

1. Программирование на Java [Текст] : курс лекций для студентов вузов / Н. А. Вязовик .— М. : Интернет-Ун-т Информ. технологий, 2003 .— 592 с.

Дополнительная литература

1. Библиотека профессионала. Java 2 [Текст]/К. С. Хорстманн, Г. Корнелл , -М., Вильямс, 2008
2. Структуры данных и алгоритмы Java [Текст] : [учеб. пособие для вузов] / Р. Лафоре ; [пер. с англ. Е. Матвеева] .— 2-е изд. — СПб. : Питер, 2011 .— 701 с.

## **7. Перечень ресурсов информационно-телекоммуникационной сети "Интернет", необходимых для освоения дисциплины (модуля)**

- 1) <http://docs.oracle.com/javase/specs/> Java language specification
- 2) <http://docs.oracle.com/javase/tutorial/> Oracle java tutorials

## **8. Перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине (модулю), включая перечень необходимого программного обеспечения и информационных справочных систем (при необходимости)**

Scientific bibliographic databases in the field of Data Science, available on the Internet in free mode - Science Citation Index (Web of Science), Medline (PubMed), Scientific Electronic Library (NEB).

<https://www.java.com/ru/>

<https://stackoverflow.com/>

## **9. Методические указания для обучающихся по освоению дисциплины (модуля)**

Студент, изучающий дисциплину, должен с одной стороны, овладеть общим понятийным аппаратом, а с другой стороны, должен научиться применять теоретические знания на практике. В результате изучения дисциплины студент должен знать основные определения дисциплины, уметь применять полученные знания для решения различных задач.

Успешное освоение курса требует:

- посещения всех занятий, предусмотренных учебным планом по дисциплине;
- ведения конспекта занятий;
- напряжённой самостоятельной работы студента.

Самостоятельная работа включает в себя:

- чтение рекомендованной литературы;
- проработку учебного материала, подготовку ответов на вопросы, предназначенных для самостоятельного изучения;
- решение задач, предлагаемых студентам на занятиях;

– подготовку к выполнению заданий текущей и промежуточной аттестации.

Показателем владения материалом служит умение без конспекта отвечать на вопросы по темам дисциплины.

Важно добиться понимания изучаемого материала, а не механического его запоминания. При затруднении изучения отдельных тем, вопросов, следует обращаться за консультациями к преподавателю.

Возможен промежуточный контроль знаний студентов в виде решения задач в соответствии с тематикой занятий.

## ПРИЛОЖЕНИЕ

### ОЦЕНОЧНЫЕ МАТЕРИАЛЫ ПО ДИСЦИПЛИНЕ (МОДУЛЮ)

по направлению:	Бизнес-информатика		
профиль подготовки:	Финансовые технологии и аналитика онлайн-образования "Пуск"	▲	▲
	онлайн-образования "Пуск"	▲	▲
курс:	1		
квалификация:	магистр		

Семестр, формы промежуточной аттестации: 1 (осенний) - Экзамен

#### Разработчики:

Е.А. Савицкая, начальник отдела

О.А. Культепина, методист

## 1. Компетенции, формируемые в процессе изучения дисциплины

Код и наименование компетенции	Индикаторы достижения компетенции
ПК-17 Способен применять методы системного анализа и моделирования для анализа, совершенствования и проектирования архитектуры предприятия	ПК-17.1 Понимает и использует математические методы для информационно-аналитической поддержки принятия решений
	ПК-17.2 Умеет применить программный инструментарий для изменения архитектуры предприятия

## 2. Показатели оценивания компетенций

В результате изучения дисциплины «Java для Data Science» обучающийся должен:

### знать:

- объектно-ориентированное программирование в Java;
- структуру программы на Java;
- инструменты и методы командной работы на платформе GitHub;
- шаблоны проектирования в Java;
- многопоточное программирование;
- Data Science с позиции Java разработчика.

### уметь:

- объектно-ориентированное программирование в Java;
- структуру программы на Java;
- инструменты и методы командной работы на платформе GitHub;
- шаблоны проектирования в Java;
- многопоточное программирование;
- Data Science с позиции Java разработчика.

### владеть:

- навыками работы с файлами в Java;
- навыками применения инструментов Java в Data Science.

## 3. Перечень типовых (примерных) вопросов, заданий, тем для подготовки к текущему контролю

Во время текущего контроля студент должен уметь ответить на следующие вопросы:

1. Основной понятийный аппарат для Java-программиста.
2. Введение в программирование на языке Java.
3. Структура программы на Java.
4. Условные операторы и циклы в Java.
5. Типы данных: примитивы и объекты.
6. Одномерные и многомерные массивы.
7. Объектно-ориентированное программирование в Java.
8. Структура класса в Java.
9. Модификаторы доступа и наследование в Java.
10. Полиморфизм в Java. Абстракции и интерфейсы в Java.
11. Исключения, Stacktrace. Generics в коллекциях и методах.
12. Коллекции List.
13. Коллекции Queue.
14. Коллекции HashMap и HashSet.
15. Коллекции TreeMap и TreeSet.
16. Лямбда-выражения и функциональные интерфейсы в Java.
17. Работа со структурами данных с помощью инструмента Stream API.
18. Методы работы, содержание и анализ потоков, повторных вызовов.
19. Потоки ввода-вывода. Работа с файлами в Java.
20. Процессы перевода структур в Java в последовательности (сериализация).

21. Сборка проектов и фреймворки Maven и Gradle.
22. Работа с файлами CSV, XML, JSON.
23. Тестирование кода и Unit-тесты.
24. Инструмент Mockito.
25. Мокирование вызовов.
26. Основы работы с сетью: модели OSI, TCP, UDP.
27. Протокол HTTP и вызов удалённых серверов.
28. Виртуальная машина JVM.
29. Организация памяти, сборщики мусора, VisualVM.
30. Внедрение системы контроля версий с помощью использования сервиса GitHub.
31. История работы разработчика в сервисе и структура сервиса.
32. Инструменты и методы командной работы на платформе GitHub.
33. Понятие порождающих, структурных и поведенческих шаблонов.
34. Порождающие шаблоны: Builder, Singleton, Factory Method, Abstract Factory, Prototype.
35. Структурные шаблоны: Proxy, Decorator, Adapter.
36. Поведенческие шаблоны: Command, Iterator, Observer, Chain of Responsibility.
37. Свойства качественного кода и принцип SOLID.
38. Многопоточное (параллельное) программирование.
39. Создание и запуск потоков.
40. Содержание работы с синхронизацией.
41. Переменные многопоточной программы.
42. Клиент-серверное взаимодействие.
43. Пакеты Blocking и Non-Blocking IO.
44. Функциональное программирование в Java среде.

Во время занятий могут проходить интерактивные обсуждения в чатах курса, что будет являться домашним заданием. Успешное выполнение всех заданий по курсу и выполнение контрольных срезов знаний дает преимущество на экзамене.

#### **4. Перечень типовых (примерных) вопросов и тем для проведения промежуточной аттестации обучающихся**

Примеры билетов для экзамена:

Билет 1.

Требуется написать функцию `getArrayParams(arr)`, которая получает на вход массив чисел от -100 до 100 и возвращает минимальное, максимальное и среднее арифметическое значений массива.

1. Создайте функцию, которая принимает на ввод массив.
2. Внутри функции задайте 3 переменных `min`, `max`, `sum`, присвоив им некоторые первоначальные значения.
3. Пройдите по массиву циклом `for` либо `while` и для каждого элемента определите:
  - если элемент больше предыдущего максимума, то максимум становится равным элементу,
  - если элемент меньше предыдущего минимума, то минимум становится равным элементу.
4. Добавляем элемент к сумме `sum` для последующего вычисления среднего.
5. После прохождения цикла функция должна возвращать объект вида: `{max: ..., min: ..., avg: ...}`, то есть минимальное, максимальное и средние значения. Чтобы вычислить среднее, надо сумму элементов поделить на их количество.
6. Среднее надо округлить до 2-х знаков после запятой.

Билет 2.



В репозитории сервис отправки сообщений находится код приложения отправки локализованных сообщений, в котором на основе ip-адреса, переданного в заголовке, определяется язык отправляемого сообщения. ip-адрес начинающийся со "172." относится к российскому сегменту, а с "96." - к американскому. Для российских адресов отправляется текст на русском, а для американских адресов и всех остальных - на английском. Наша задача написать/добавить unit-тесты с использованием библиотеки mockito для проверки корректности работы функционала.

Что нужно сделать:

1. Написать тесты для проверки языка отправляемого сообщения (класс `MessageSender`) с использованием `Mockito`
  2. Проверить, что `MessageSenderImpl` всегда отправляет только русский текст, если ip относится к российскому сегменту адресов.
  3. Проверить, что `MessageSenderImpl` всегда отправляет только английский текст, если ip относится к американскому сегменту адресов.
  4. Написать тесты для проверки определения локации по ip (класс `GeoServiceImpl`).
- Проверить работу метода `public Location byIp(String ip)`.
5. Написать тесты для проверки возвращаемого текста (класс `LocalizationServiceImpl`).
  6. Проверить работу метода `public String locale (Country country)`.

Реализация

1. Склонируйте удаленный репозиторий сервиса <https://github.com/neee/geo-service> или сделайте его fork (предпочтительно) или скачайте к себе в виде архива.
2. Подключите к maven-проекту зависимости junit и mockito (их нужно добавить в файл `pom.xml`).
3. Создайте класс для тестов в папке `src/test/java` (можете также создать подпапки в соответствии с package'ом класса, который вы будете тестировать).
4. Создайте тесты в соответствии с задачей (для сервиса `MessageSenderImpl`, нужно обязательно создать заглушки (mock) в виде `GeoServiceImpl` и `LocalizationServiceImpl`) минимум 4 unit теста. Отправьте задачу на проверку.

Билет 3.

1. Произведите Fork репозитория с задачами (fork необходимо делать перед выполнением каждой домашней работы).
2. Перейдите в папку задания. `cd ./8.decorators`.
3. Откройте файл `task.js` в вашем редакторе кода и выполните задание.
4. Откройте файл `index.html` в вашем браузере с помощью консоли DevTools и убедитесь в правильности выводимых результатов.
5. Откройте файл `test-runer.html` в вашем браузере и убедитесь, что все тесты выполняются (на вкладке Spec List можно видеть какие тесты выполнились, а какие нет).
6. Добавьте файл `task.js` в индекс git с помощью команды `git add %file-path%`, где `%file-path%` - путь до целевого файла `git add task.js`.
7. Сделайте коммит, используя команду `git commit -m '%comment%'`, где `%comment%` - произвольный комментарий к вашему коммиту. `git commit -m 'first commit variables'`.
8. Опубликуйте код в репозиторий `homeworks` с помощью команды `git push -u origin main`.
9. Пришлите ссылку на репозиторий.

Билет 4.

Напишите усовершенствованный кэширующий декоратор `cachingDecoratorNew`, аналогичный рассмотренному на лекции, таким образом, чтобы он кэшировал только последние 5 различных вызовов функции. То есть чтобы кэш мог хранить только 5 значений.

Для того, чтобы тесты выполнялись, функция должна возвращать следующие строки(!) "Вычисляем: 10" для первого вызова (10 для примера) и "Из кэша: 10" для повторного.

Рекомендуется параллельно выводить результаты в консоль, чтобы было удобнее отлаживать.

Оценка отлично (10 баллов) - выставляется обучающемуся, показавшему всесторонние, систематизированные, глубокие знания учебной программы дисциплины, проявляющему интерес к данной предметной области, продемонстрировавшему умение уверенно и творчески применять их на практике при решении конкретных задач, свободное и правильное обоснование принятых решений.

Оценка отлично (9 баллов) - выставляется студенту, показавшему всесторонние, систематизированные, глубокие знания учебной программы дисциплины и умение уверенно применять их на практике при решении конкретных задач, свободное и правильное обоснование принятых решений.

Оценка отлично (8 баллов) - выставляется студенту, показавшему всесторонние, систематизированные, глубокие знания учебной программы дисциплины и умение уверенно применять их на практике при решении конкретных задач, правильное обоснование принятых решений, с некоторыми недочетами.

Оценка хорошо (7 баллов) - выставляется студенту, если он твердо знает материал, грамотно и по существу излагает его, умеет применять полученные знания на практике, но недостаточно грамотно обосновывает полученные результаты.

Оценка хорошо (6 баллов) - выставляется студенту, если он твердо знает материал, грамотно и по существу излагает его, умеет применять полученные знания на практике, но допускает в ответе или в решении задач некоторые неточности.

Оценка хорошо (5 баллов) - выставляется студенту, если он в основном знает материал, грамотно и по существу излагает его, умеет применять полученные знания на практике, но допускает в ответе или в решении задач достаточно большое количество неточностей.

Оценка удовлетворительно (4 балла) - выставляется студенту, показавшему фрагментарный, разрозненный характер знаний, недостаточно правильные формулировки базовых понятий, нарушения логической последовательности в изложении программного материала, но при этом он освоил основные разделы учебной программы, необходимые для дальнейшего обучения, и может применять полученные знания по образцу в стандартной ситуации.

Оценка удовлетворительно (3 балла) - выставляется студенту, показавшему фрагментарный, разрозненный характер знаний, допускающему ошибки в формулировках базовых понятий, нарушения логической последовательности в изложении программного материала, слабо владеет основными разделами учебной программы, необходимыми для дальнейшего обучения и с трудом применяет полученные знания даже в стандартной ситуации.

Оценка неудовлетворительно (2 балла) - выставляется студенту, который не знает большей части основного содержания учебной программы дисциплины, допускает грубые ошибки в формулировках основных принципов и не умеет использовать полученные знания при решении типовых задач.

Оценка неудовлетворительно (1 балл) - выставляется студенту, который не знает основного содержания учебной программы дисциплины, допускает грубейшие ошибки в формулировках базовых понятий дисциплины и вообще не имеет навыков решения типовых практических задач.

## **5. Методические материалы, определяющие процедуры оценивания знаний, умений, навыков и (или) опыта деятельности**

Экзамен по дисциплине проводится в форме выполнения итогового задания.