**Программа учебной дисциплины**

**«Конструирование программного обеспечения»**

Утверждена

Академическим советом ООП

Протокол № от «\_\_»\_\_\_\_\_20\_\_ г.

|  |  |
| --- | --- |
| Автор | Профессор, к.т.н. Гринкруг Е.М. |
| Число кредитов | 8 |
| Контактная работа (час.) | 144 |
| Самостоятельная работа (час.) | 164 |
| Курс | Второй курс бакалавриата |
| Формат изучения дисциплины | Лекции и практические семинарские занятия  без использования online курса |

1. **ЦЕЛЬ, РЕЗУЛЬТАТЫ ОСВОЕНИЯ ДИСЦИПЛИНЫ И ПРЕРЕКВИЗИТЫ**

Целью освоения дисцилины «Конструирование программного обеспечения» является овладение студентами основными концепциями, методами и технологиями создания современного платформо-независимого программного обеспечения. Изучение дисциплины базируется на использовании Java-платформы, которая является наиболее широко используемой в мире основой разработки и функционирования современного программного обеспечения.

В результате освоения дисциплины студент должен:

**знать:**

- основные принципы и методологии разработки программного обеспечения с использованием современных подходов к его созданию и их реализации в Java-платформе;

- язык программирования Java, его поддержку в виртуальной машине (JVM) и в современных инструментальных средствах разработки, а также основные библиотеки современной Java-платформы в пределах, соответствующих Java Platform Standard Edition;

- современные среды разработки - Java Integrated Developer Environments (IDE), их возможности и практическое использование при конструировании программного обеспечения.

**уметь:**

- разрабатывать программные приложения с использованием современных професиональных средств программирования;

- отлаживать, тестировать и интегрировать разработанное программное обеспечение с использованием современных инстументов и технологий;

- использовать существующие программные библиотеки для различных программных технологий и областей применения;

- проектировать и реализовывать программные приложения на базе Java Platform Standard Edition;

- находить, изучать и использовать новые технологии и решения для выполнения требований разработки и конструирования программного обеспечения.

**владеть:**

- практическими навыками конструирования программного обеспечения на Java-платформе с применением соответствующих инструментальных средств;

- практическими навыками использования современных программных библиотек и технологий при конструировании программного обеспечения;

- приемами и практическими методами конструирования программного обеспечения с применением современных парадигм программирования.

Изучение дисциплины «Конструирование программного обеспечения» базируется на знаниях студентов в областях математики, основ информатики, теории алгоритмов и на навыках программирования и владения компьютером, приобретенных при изучении следующих дисцилин на первом году обечения в бакалавриате:

- дискретная математика;

- программирование.

Для освоения учебной дисциплины студенты должны владеть следующими знаниями и компетенциями:

- знать основные принципы работы компьютера;

- знать простейшие способы и средства разработки и отладки программ;

- обладать практическими навыками работы с компьютером на уровне пользователя.

Основные положения дисциплины могут быть использованы в дальнейшем при изучении ряда последующих дисциплин, связанных с конструированием программного обеспечения для различных областей применений, а также являются основой для написании ежегодных курсовых работ, исследовательских и дипломных проектов, которые выполняются студентами по завершении изучения данной дисциплины.

1. **СОДЕРЖАНИЕ УЧЕБНОЙ ДИСЦИПЛИНЫ**

**Тема 1. Введение: проектирование программного обеспечения и основные парадигмы программирования**

Обзор методологий программирования. Объектно-ориентированное программирование и его исторический аспект. Сопоставление class-based и prototype-based программирования объектов. Основные черты class-based объектно-ориентированного языка программирования. История и эволюция Java-программирования. Инструментальные средства Java-программирования. Обзор основных Java-технологий. Java Development Kit(JDK): эволюция и структура от версии 1.0 до 8.

**Тема 2. Введение в программирование на Java. Основные программные конструкции**

Объекты и типы. Типы и классы. Примитивные типы. Массивы. Создание объектов. Хранение объектов. Время жизни объектов и их уничтожение. Создание новых типов объектов. Лексические основы языка. Исходные файлы и их содержимое. Правила кодирования и документирования в исходных файлах. Комментарии. Пакеты и их именование. Поля и правила их инициализации. Литералы. Методы, параметры и возвращаемые значения. Операторы в языке программирования Java. Операторы управления (if-else, циклы, переключатели).

**Тема 3. Объекты и классы**

Инициализация и финализация объектов. Конструкторы. Mutable и Immutable объекты. Перегрузка методов. Конструкторы умолчания. Ключевое слово this. Вызов конструктора из конструктора. Ключевоке слово super. Классы как объекты. Ключевой слово static. Подробности жизненного цикла объекта. Финализация объектов и сборка мусора. Последовательность инициализации и финализации объекта. Статические и нестатические инициализаторы. Инициализация массивов..

**Тема 4. Повторное использование классов**

Композиция и наследование. Синтиаксис наследования. Иниуиализация базового класса. Конструкторы с аргументами. Делегирование. Сочетание композиции и наследования. Обеспечение корректной финализации. Сокрытие имен. Сравнение композиции и наследования. Семантика ключевого слова protected. Полиморфизм. Polymorphism. Восходящее и нисходящее приведение типа. Раннее и позднее связывание. Использование приватных полей и методов. Конструкторы и полиморфизм.

**Тема 5. Интерфейсы и абстрактные классы**

Абстрактные методы и абстрактные классы. Интерфейсы и их отличие от абстрактных классов. Интерфейсы и множественное наследование в Java (включая средства Java 8). Адаптация с помощью интерфейсов. Поля в интерфейсах и их инициализация. Вложенные интерфейсы. Интерфейсы и фабрики. Тегирующие интерфейсы. Клонирование и его реализация в Java. Определение static и default методов в интерфейсах (Java 8). Перечисления (enums).

**Тема 6. Внутренние и вложенные классы**

Связь между вложенным и охватывающим классом. Использование ключевых слов this и new. Вложенные классы и восходящее приведение типа. Именование вложенных классов. Локальные и анонимные классы. Фабрики и вложенные классы. Статические вложенные классы. Классы в интерфейсах. Правила доступа для вложенных классов. Преимущества и недостатки использования вложенных классов. Вложенные классы и событийно-управляемые вычисления. Наследование вложенных классов. Вложенные классы и средства в Java 8.

**Тема 7. Runtime информация о типах**

Объекты класса Class. Загрузка классов. Принципы и основы реализации загрузчика классов (ClassLoader). Получение ссылки на класс. Класс-литералы. Возможности статического и динамического контроля типов. Рефлекция и методы рефлекции. Получение информации о классе и динамические вызовы методов. Рефлекция и вопросы контроля доступа..

**Тема 8. Обработка исключительных ситуаций**

Объекты – исключения. Создание объектов-исключений. Ловушки исключений и обработка исключений. Спецификация исключений. Трассировка стека. Перевозбуждение исключений. Цепочки исключений. Стандартные исключения и их классификация. Выполнение finally-блока. Рекомендации по работе с исключениями, ограничения и применения.

**Тема 9. Параметризованные типы (Generics)**

История и цели применения параметризованных типов в языке Java (Java Generics). Простые параметризованные типы. Параметризованные интерфейсы. Параметризованные методы. Параметризация и вложенные классы. Ограничения параметризации типов в Java. Совместное использование параметризованных и непараметризованных типов. Generics и массивы. Wildcards. еализация параметризованных интерфейсов. Типы преобразований и предупреждения при использовании параметризованных типов.

**Тема 10. Контейнеры и коллекции**

Параметризованные и типизированные контейнеры. Основные понятия и интерфейсы (List, Iterator, Set, Map). Обзор принципов проектирования библиотек коллекций в Java. Foreach и итераторы. Хэширолвание и его использование. Правила использования методов hashCode() и equals(). Массивы, коллекции и параметризация. Массивы и параметризация типов. Вспомогательные классы для работы с массивами и коллекциями.

**Тема 11. Параллельное выполнение. Основные понятия**

Интерфейс Runnable. Потоки управления и класс Thread. Обзор методов класса Thread. Приоритеты потоков управления. Примитивы ожидания. Разделение ресурсов. Базовые средства синхронизации потоков управления в Java. Атомарные операции. Ключевой слово volatile. Критические секции.

**Тема 12. Многопоточные вычисления и синхронизация**

Понятие thread-safety. Влиянние требований потокобезопасности на реализацию контейнеров (thread-safe containers). Задачи организации взаимодействия и синхронихации потоков и их решения. Базовые средства поддержки параллельных вычислений в Java с использованием методов wait() / notify() класса Object.

**Тема 13. Расширенные средства параллельных вычислений**

Обзор средств организации параллельных вычислений в Java и их эволюция. Обзор библиотеки для организации параллелизма (пакет java.util.concurrency) с примерами и демонстрациями.

**Тема 14. Программирование ввода и вывода. Основы**

Класс File и его применение. Поддержка архитектуры ввода-вывода в Java. Входные и выходные потоки данных (InputStream и OutputStream) и их реализации. Понятия Reader и Writer. Использование потоков ввода-вывода. Стандартные потоки ввода-вывода.

**Тема 15. Расширенные средства программирования ввода-вывода.**

Исторический обзор развития средств поддержки ввода-вывода в Java. Библиотека ввода-вывода паккета nio. Средства и использование класса ByteBuffer. Вопросы производительности ввода-вывода. Сжатие данных, JAR-файлы, сериализация, десериализация и различные варианты их реализаций.

**Тема 16. Программирование графических пользовательсикх интерфейсов (GUI)**

Развитие средств порграммирования GUI в Java (исторический обзор). Пакеты и библиотеки JDK для программирования GUI (awt, java2D, swing and others). Базовая поддержка графики и обработки событий.

**Тема 17. Библиотеки для программирования GUI**

Обзор библиотеки Swing. Компоненты Swing и их использование. Отображение списков, таблиц, деревьев с использованием компонент swing. Контейнеры swing-компонент. Использование MVC-design pattern в библиотеке Swing.

**Тема 18. Компонентная модель JavaBeans**

Компонентно-ориентированная методология программирования. Компонентно-ориентированное программирование в Java. Bean Development Kit (BDK). События и их адаптеры. Интроспекция. Properties, Property Change Event(s), XMLEncoder/Decoder. Обзор пакета java.beans.

**Тема 19. Библиотека JavaFX**

Архитектура и принципы работы JavaFX framework. Понятия Scene и Scene graph в JavaFX. Графическая подсистема JavaFX. Модель многопоточности в JavaFX. Обзор новых возможностей программирования GUI в JavaFX. Java 3D-программирование и поддержка 3D-программирования в JavaFX. Визуальные эффекты в JavaFX.

**Тема 20. Сетевое программирование. Основы**

Платформа Java и сетевое программирование. Модель Open System Interconnection (OSI) и ее уровни. Internet-протоколы. Протоколы с установлением и без установления соединения. Протоколы TCP и UDP. Порты.

**Тема 21. Сетевое программирование. Sockets**

Internet-адресация. Распределенные вычисления и межпроцессное взаимодействие. Unicast vs. Multicast vs. Broadcast взаимодействия. Socket Application Programming интерфейс. Типы Socket’ов. Stream-mode sockets. Программирование сервера и клиента с помощью socket’ов. Параллельный сервер. Проблемы при использовании socket’ов. Прерываемые socket’ы.

**Тема 22. Сетевое программирование Высокоуровневые средства**

Понятия URL и URLConnection, Uniform Resource Locator (URL), URL и URI. Взаимодействия клиентов и сервера. Взаимодействие по http. Простой http-сервер. Обзор java-технологий для сетевого программирования.

**Тема 23. Удаленный вызов методов (RMI)**

Работа с распределенными объектами. Роли клиента и сервера. Удаленные вызовы методов и программная модель для их реализации - Remote Method Invocation (RMI). RMI Registry. Реализация удаленного интерфейса. Передача параметров и возвращаемых значений. Примеры приложений.

**Тема 24. Взамодействие в базами данных (JDBC)**

Основы архитектуры JDBC. Базовые понятия SQL. Конфигурация JDBC. Испполнение операторов SQL. Выполнение запросов. Транзакции. Храннимые процедуры. Метаданные. Derby DBMS, поставляемая вместе с JDK.

**Тема 25. Взаимодействие Java- и С-кода**

Обсуждение причин и следствий организации взаимодействия Java-платформы и иных средств программирования. Вызов C-программы из Java-программы. Передача параметров и возвращаемых значений. Передача строк. Доступ к полям. Кодировка. Вызов Java-методов из C-кода. Досткп к элементам массивов. Обработка ошибок. Использование API.

**Тема 26. Использование XML при программировании**

Общий обзор XML и его применений. Использование XML в Java-программах. Структура XML-документа. Парсинг XML. DOM-Parsers. DTD и XML Schema. Поиск информации с использованием XPath. Пространства имен. SAX-Parsers. Push vs. Pull Parsers.

**Тема 27. Работа с аннотациями, скриптование, компиляция**

Скриптование для Java-платформы. API компилятора. Использование аннотаций. Синтаксис аннотаций. Стандартные аннотации. Обработка аннотаций на уровне исходного кода и других. Генерация байткода и соответствующие библиотеки.

**Тема 28. Логгирование, тестирование и отладка**

Цели и задачи логгирования. Обзор стандартных Java-библиотек для логгирования. Unit-тестирование. Покрытие кода тестами. Примеры использования JUnit. JUnit: версии, аннотации, assertions. Создание JUnit-тестов. Использование JUnit-тестов в среде IDE(s). Введение в Test-driven development (TDD).

**Тема 29. Обзор технологий сборки программных проектов**

Сравнительный обзор технологий сборки проектов (Ant, Maven, Gradle). Типовые шаги сборки. Использование сборщиков совместно с IDE(s) и репозиториями.

**Тема 30. Обзор шаблонов проетирования**

Что такое «шаблон проектирования»? Преимущества и недостатки использования шаблонов проектирования. Классификация шаблонов: creational, structural, behavioral и другие (problem specific: concurrency design patterns, component design patterns). Обсуждение основных шаблонов проектирования и их Java-реализаций с примерами.

**Тема 31. Развитие Java-платформы и обзор нововведений в последних версиях**

Основные этапы эволюции JDK - от JDK 1.0 до JDK 8. Основные цели и усовершенствования в каждой версии JDK. Сочетание объектно-ориентированного и функционального программирования. Влияние прогресса аппаратных средств на развитие Java-платформы. Многоядерные процессоры и обработка потоков данных в Java. Передача кода в качестве параметра. Параллелизм данных и параллелизм управления. Функции, методы и Lambda в Java 8.

**Тема 32. Функциональное программирование в Java**

Добавление средств функционального программирования к объектно-ориентированному языку: цели и реализация. Lambda- выражения в Java 8. Синтаксис. Функциональные интерфейсы. Ссылки на методы, Ссылки на конструкторы. Переменные в Lambda-выражениях. Effectively final variables. Примеры использования.

**Тема 33. Библиотека Streams для обработки потоков данных**

Использовние средств параллелизма при обработке коллекций. Обзор Stream API (пакет java.util.stream). Потоки данных как высокоуровневые средства обработки коллекций. Сравнение внешней и внутренней итераций. Преимущества многоядерных архитектур при параллельной обработке коллекций.

**Тема 34. Новые средства JDK и базовых библиотек**

Как и при каких обстоятельствах надо использовать новые средства Java-платформы? Влияние новых средств на ранее разработанные программные продукты. Особенности работы с примитивными типами данных в новых Java-платформах. Оценка преимуществ от использования новых средств. Развитие Java-платформы с позиций развития параллельных архитектур.

**Тема 35. Направления применений пройденного материала. Заключение**

Области применения платформо-независимого программного обеспечения. Исторический аспект. Эволюция. Направления развития. Новые технологии и инструменты. Завершение курса.

1. **ОЦЕНИВАНИЕ**

Типы и формы контроля успеваемости по курсу приведены в нижеследующей таблице.

| Тип контроля | Форма контроля | Модули | | | | Дополнительная информация |
| --- | --- | --- | --- | --- | --- | --- |
| **1** | **2** | **3** | **4** |
| Текущий | Контрольная работа | \* |  |  |  | Тест на копьютере |
| Домашнее задание |  |  | \* |  | Разработка программы |
| Промежуточный | Экзамен |  | \* |  |  | Тест на компьютере |
| Итоговый | Экзамен |  |  |  | \* | Тест на компьютере |

**Текущий контроль:** одна контрольная работа в конце 1-го модуля и одно контрольное домашнее задание в конце 3-го модуля.

**Промежуточный контроль:** экзамен в конце 1 семестра (в конце 2-го модуля).

**Итоговый контроль:** экзамен в конце курса (в конце 4 модуля).

Оценки по всем формам контроля выставляются по 10-ти балльной шкале.

**Текущий контроль** предусматривает одну контрольную работу, выполняемую в конце 1-го модуля, одно контрольное домашнее задание, выполняемое в конце 3-го модуля, и текущий контроль самостоятельоной работы студентов в течение всего курса.

**Контрольная работа** представляет собой тест на копьютере с вопросами по пройденному материалу. Задания теста аналогичны заданиям, используемым при сертификации компанией Oracle (OCJA, OCJP), вопросы и ответы теста предполагают знание английского языка на уровне понимания профессиональных текстов.

**Контрольное домашнее задание** включает разработку, кодирование, тестирование и отладку программ реализации задания. По домашнему заданию оформляется отчет в электронном виде.

Контрольное домашнее задание размещается в системе Moodle в соответствующем разделе. В установленный срок студент загружает архив, содержащий полностью оформленный отчет и программу решения контрольного домашнего задания. Оценка за домашнее задание выставляется с учетом полноты выполнения задания и оформления результатов.

**Самостоятельная работа студентов** предполагает выполнение домашних работ (заданий к практическим занятиям). Задания к практическим занятиям ставятся преподавателями, проводящими практические занятия в подгруппах, и размещаются в системе Moodle. Выполненные задания могут загружаться студентами в Moodle (все технические вопросы определяются преподавателями практических занятий) и/или оцениваются преподавателем на практических занятиях в результате собеседования – по усмотрению преподавателя.

В случае несвоевременной (без уважительной причины) сдачи домашних работ и контрольного домашнего задания, сроки выполнения которых определяются преподавателем практических занятиий, оценка по ним снижается вдвое. При отсутствии выполненного задания в указанные преподавателем сроки (без уважительной причины) выставляется оценка 0.

**Промежуточный контроль** предусматривает экзамен в конце 2-го модуля по всем темам, пройденным в первом семестре. Экзамен проводится в виде компьютерного теста, аналогичного тесту в конце 1-го модуля, но охватывает больший объем пройденного материала (весь материал первого семестра).

**Итоговый контроль:** экзамен в конце 4-го модуля. Проводится в виде компьютерного теста по всему пройденному материалу.

**Порядок формирования оценок по дисциплине**

Преподавателями, ведущими практические занятиия, оценивается самостоятельная работа студентов с учетом правильности и полноты выполнения домашних работ по темам практических занятий. Оценки за самостоятельную работу студента преподаватель, ведущий практические занятия, выставляет в рабочую ведомость за каждое оцениваемое задание и усредняет (с обычными арифметическими округлениями) для оценки самостоятельной работы студента в конце каждого (i-го, i = 1, 2, 3, 4 ) модуля - СР(i).

Оценка за первый модуль (М1) вычисляется как взвешенная сумма (с обычным арифметическим округлением до целого) оценки за контрольную работу в конце 1-го модуля КР(1) и оценки самостоятедльной работы студента в первом модуле СР(1):

М1 = 0.75 \* КР(1) + 0.25 \* СР(1).

Оценка за второй модуль (М2) вычисляетмся как взвешенная сумма (с обычным арифметическим округлением до целого) оценки за промежуточный экзамен (Э2) и оценки самостоятельоной работы студента во втором модуле СР(2):

М2 = 0.75 \* Э2 + 0.25 \* СР(2).

Оценка за третий модуль вычисляется как взвешенная сумма (с обычным арифметическим округлением) оценки за контрольное домашнее задание (КДЗ) и оценки самостоятельной работы в третьем модуле СР(3):

М3 = 0.75 \* КДЗ + 0.25 \* СР(3).

Оценка за четвертый модуль равна оценке самостоятельной работы студента в четвертом модуле:

М4 = СР(4).

Накопленная оценка НО вычисляется как средее арифмметическое оценок за все 4 модуля (с обычным арифметическим округлением):

НО = (М1 + М2 + М3 + М4) / 4.

Итоговая оценка ИО за дисциплину (курс) вычисляется как среднее арифметическое накомпленной оценки НО и оценки за заключительный экзамен ЗЭ в конце 4 модуля. При этом округления (при необходимости) производятся в сторону оценки за заключительный экзамен (экзамен имеет больший приоритет). Например, если НО = 7 и ЗЭ = 6, то ИО = 6.

ИО = finalGrade (HO, ЗЭ), где для вычисления используется следующая функция:

**public static int** finalGrade (**int** accumulated, **int** exam) {  
 **int** sum = accumulated + exam;  
 **int** average = sum / 2;  
 **int** rest = sum % 2;  
 **if** (exam > accumulated) {  
 **return** average + rest;  
 } **else** {  
 **return** average;  
 }  
 }

Перевод в пятибалльную оценку осуществляется в соответствии со следующей таблицей.

|  |  |
| --- | --- |
| **Десятибалльная шкала** | **Пятибалльная шкала** |
| 1 – неудовлетворительно  2 – очень плохо  3 – плохо | неудовлетворительно – 2 |
| 4 – удовлетворительно  5 – весьма удовлетворительно | удовлетворительно – 3 |
| 6 – хорошо  7 – очень хорошо | хорошо – 4 |
| 8 – почти отлично  9 – отлично  10 – блестяще | отлично – 5 |

1. **ПРИМЕРЫ ОЦЕНОЧНЫХ СРЕДСТВ**

**Тематика для контрольных работ**

Контрольные работы (включая промежуточный и итоговый экзамены) проводятся в виде компьютерных тестов по материалам, аналогичным публиковавшимся материалам для подготовки к сертификациям компании Oracle (OCJA и OCJP). Все тестовые материалы предлагаются на английском языке. Предполагается, что студенты, успешно завершившие курс, имеют возможность успешного прохождения такой сертификации.

Наборы и содержание тестов соответствуют материалам курса.

**Тематика домашних заданий**

Домашние задания задаются преподавателем, ведущим практические занятия, по согласованию с преподавателем, читающим лекции. Домашние задания предполагают использование соответствующих материалов курса для их успешного выполнения и формируют навыки практического использования полученных знаний, современных инструментальных средств и технологий, включая инструменты для проектирования, кодирования, отладки, тестирования, сборки и документирования программ, составляющих содержание домашнего задания.

**Примеры вопросов для оценки качества освоения дисциплины**

1. В чем различие между class-based и prototype-based языками программирования (приведите пример)?

2. Какие программные инструменты нужны для разработки и выполнения Java-программы?

3. В чем разница между Java Runtime Environment (JRE) и Java Developer Kit (JDK)?

4. Как запускается на исполнение Java-приложение?

5. Перечислите основные отличия версий Java, начиная с версии Java 1.0

6. Что такое Java-applet и в чем отличие между Java-applet и Java-application?

7. Какие виды типов данных определены и используются в языке Java?

8. Перечислите все примитивные типы язвка Java.

9. Что такое ссылочный тип, и какие ссылочные типы есть в языке Java?

10. Что тако wrapper? Приведите примеры его использования

11. Как загружаются классы?

12. Какова цель понятия package?

13. Что такое tagging interface? Приведите примеры.

14. Есть ли какие-то Java-интерфейсы, которые реализует любой Java-массив? Как написать несколько строк кода, дающего ответ на этот вопрос?

15. В чем различие между абстрактным классом и интерфейсом?

16. Что такое singleton?

17. Можем ли мы клонировать любой объект, раз в базовом классе java.lang.Object имеется метод clone()?

18. В чем разница между композицией и наследованием?

19. Можно ли инстанциировать не public клас, имеющий public конструктор?

20. Что такое instance initializer? Что такое static initializer? Что вызывается раньше: конструктор или instance initializer?

21. Когда может быть вызван метод finalize()? С какой целью?

22. Можем ли мы изменить значения поля, декларированного в интерфейсе?

23. Перечислите все разновидности вложенных классов в Java.

24. Что такое default-constructor? Может ли такой конструктор быть у статического вложенного класса? А у нестатического вложенного класса? Сколько параметров он тогда может иметь?

25. Сколько конструкторов может иметь анонимный класс?

26. Что такое checked exception? Какие бывают виды исключений (перечислить)?

27. Что такое finally-block в коде? Когда он выполняется?

28. Что такое immutable объект?

29. Как узнать, из какого класса наследован данный класс? Как узнать, какие интерфейсы реализует класс заданного объекта?

30. Есть ли общий суперкласс у всех перечислений (enum(s)) в Java? В какой версии они были впервые внедрены в Java?

31. В каком пакете содержатся классы коллекций? Каковы основные интерфейсы и классы для работы с коллекциями?

32. В чем разница между LinkedList, ArrayList и Vector?

33. В чем разница между HashMap и Hashtable?

34. Что такое хэширование? Что будет, если мы переопределим метод hashCode() в некотором классе так, что он будет всегда возвращать значение 7?

35. Определите параметризованный тип (класс) Variable <T> (“a variable with a value of type T”), имеющий setter- и getter-методы (т.е., методы T setValue(T newValue) и T getValue()). Каким будет тип property “value” в PropertyDescriptor из BeanInfo-объекта для этого класса, который выдается с использованием класса java.beans.Introspector?

36. Привелите как можно больше способов получения копии массива.

37. Что такое thread? Как его запустить и остановить?

38. Что означеет понятие thread-safe?

39. Как реализовать thread-safe singleton?

40. В чем разница между synchronized-методом и synchronized-блоком?

41. Что такое serialization? Что такое deserialization?

42. Как можно использовать Reader? Для чего?

43. Чем отличается Swing от AWT?

44. Что такое Java bean? Каковы минимальные требования к классу, чтобы он был Java bean?

45. В чем состоит цель работы класса java.beans.Introspector?

46. Какие методы надо иметь в Java-классе для определения того, что его инстансы могут выдавать события определенного типа?

47. Чем интересны bound properties?

48. Какие инструменты для работы с XML Вы знаете? Что они делают?

49. Какие механизмы защиты данных поддерживает Java VM?

50. Что такое SecurityManager в Java?

1. **РЕСУРСЫ**

Вся необходимая литература и другие образовательныне ресурсы, включая компьютерные программы, доступны студентам на специально предназначенном для этого сервере Moodle ( <https://moodle.cs.hse.ru/> ), который используется для поддержки изучения данной дисциплины. С первого дня изучения дисцилины все студенты регистрируются на этом сервере и получают доступ к используемым ресурсам.

Все компьютерные тесты (как промежуточного, так и итогового контроля) выполняются с использованием подсистемы тестирования этого сервера в сочетании с соответствующим SafeExam-браузером, исключающим использование студентами посторонних средств при прохождении тестирования. Результаты тестирования обрабатываются и формируются сервером автоматически.

Все домашние задания по дисциплине задаются и принимаются для проверки с использованием соответствующих средств сервера, контролирующих предоставление выполненных домашних заданий.

Основыными разделами для сопровождения дисцилины на сервере являются следующие:

- общая информация и форум по дисциплине;

- презентации лекций по темам дисциплины;

- материалы ко всем пректическим семинарским занятиям по дисциплине;

- раздел, предназначенный для проведения тестирований;

- раздел домашних заданий и приема результатов их выполнения;

- образовательные ресурсы, используемые в дисциплине, включая литературу.

**Основная литература**

Вся имеющаяся на сервере литература по дисцилине предоставлена в электронном виде (в виде pdf - файлов). Если литература сопровождается архивами с кодами программ, эти архивы также имеются на сервере дисциплины.

1. Cay S. Horstmann, Gary Cornell. Core Java 2, vol.1 Fundamentals, 10th Edition, 2016.

Имеется русский перевод более ранней редакции:

Хорстманн, Кей С., Корнелл Гари. Java 2. Библиотека профессионала. Том 1. Основы. 9-е изд.: Пер. с англ.- М.: Издательский дом «Вильямс», 2014.

1. Cay S. Horstmann, Gary Cornell. Core Java 2, vol.2 Advanced Features. 10th Edition, 2016.

Имеется русский перевод более ранней редакции:

Хорстманн, Кей С., Корнелл Гари. Java 2. Библиотека профессионала. Том 2. Тонкости программирования. 9-е изд.: Пер. с агл.- М.: Издательский дом «Вильямс», 2014.

1. Herbert Schildt, Java 8. The Complete Reference. 9th Edition, Oracle press, 2014.

Имеется русский перевод:

Герберт Шилдт, Java 8. Полное руководство. 9-е изд.:, Издательский дом «Вильямс»,2015.

1. Java Developers Kit Documentation. Электронный ресурс – Режим доступа: <http://www.oracle.com/technetwork/java/javase/documentation/index.html>

**Дополнительная литература**

1. Joshua Bloch. Effective Java. Second Edition. Sun Microsystems, 2008. (Имеется русский перевод).
2. Robert Englanger. Developing Java Beans, O’Reilly, 2001.
3. Richard Warburton. Java 8 Lambdas. O’Reilly Media Inc., 2014.
4. Venkat Subramaniam. Functional Programming in Java, The Pragmatic Programmers, 2014.
5. Elliot Rusty Harold, Java Network Programming, 4th Edition, O’Reilly, 2014.
6. B.Christiansson et al, GoF Design Patterns with Examples using Java and UML2, 2008.
7. E.R.Harold, Java Network Programming, 4th Edition, O’Reilly, 2014.

**Программное обеспечение**

|  |  |  |
| --- | --- | --- |
| **№ п/п** | **Наименование** | **Условия доступа** |
| 1. | Microsoft Windows 7 Professional RUS | *Из внутренней сети университета (договор)* |
| 2. | Microsoft Office Professional Plus 2010 | *Из внутренней сети университета (договор)* |
| 3. | IDE IntellyJ IDEA | *Учебная лицензия JetBrains (договор)* |
| 4. | Java Development Kit (JDK) | *Open source* |
| 5. | Safe Exam Browser | *Open source* |

**Профессиональные базы данных, информационные справочные системы,   
интернет-ресурсы (электронные образовательные ресурсы)**

|  |  |  |
| --- | --- | --- |
| **№ п/п** | **Наименование** | **Условия доступа** |
|  | ***Профессиональные базы данных, информационно-справочные системы*** | |
| 1. | Консультант Плюс | *Из внутренней сети университета (договор)* |
| 2. | Электронно-библиотечная система Юрайт | URL: https://biblio-online.ru/ |
|  | ***Интернет-ресурсы (электронные образовательные ресурсы)*** | |
| 3. | Открытое образование | URL: https://openedu.ru/ |

**Материально-техническое обеспечение дисциплины**

Учебные аудитории для лекционных занятий по дисциплине обеспечивают использование и демонстрацию тематических иллюстраций, соответствующих программе дисциплины в составе:

- ПЭВМ с доступом в Интернет (операционная система, офисные программы, антивирусные программы);

- мультимедийный проектор с дистанционным управлением.

Учебные аудитории для лабораторных и самостоятельных занятий по дисциплине оснащены ­­JDK версии 8, JetBrains IntellyJ IDEA IDE, SafeExam-браузером для проведения тестирования­­­­­­­­­­­­­­­­­­­­­­, с возможностью подключения к сети Интернет и доступом к электронной информационно-образовательной среде НИУ ВШЭ.