Java Core - Data types, variables, operators, arrays, loops

OOP

Java Core - Exceptions

Java Core - Collections

Java Core - Strings

Java Core - IO

Java Core - Multithreading

Java - JDBC

DB, SQL

Algorithms

Testing

Logical puzzles

**Java Core - Data types, variables, operators, arrays, loops**

* Сколько ключевых слов зарезервировано языком, что это за слова, какие из них не используются?

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Abstract | Continue | For | new | Switch |
| Assert | Default | goto\* | package | Synchronized |
| Boolean | Do | If | private | This |
| Break | Double | implements | protected | Throw |
| Byte | Else | Import | public | Throws |
| Case | Enum | Instanceof | return | Transient |
| Catch | Extends | Int | short | Try |
| Char | Final | Interface | static | Void |
| Class | Finally | Long | Strictfp | Volatile |
| const\* | Float | Native | Super | While |

\* зарезервированное слово, не используется

Ответ: 59 зарезервированных слов.

* Из каких символов может состоять имя переменной (корректный идентификатор)?

Identifiers must start with a letter, a currency character ($), or a connecting character such as the underscore ( \_ ). Identifiers cannot start with a number!

Character.isJavaIdentifierStart()

Character.isJavaIdentifierPart()  
Ответ: Знак доллара и нижнее подчеркивание.

* Что значит слово “инициализация”?

Ответ: Задание начального значения полю класса либо локальной переменной.

* На какие основные группы можно поделить типы данных?

Ответ: Целочисленные, вещественные, логические и ссылочные

* Какие примитивные типы данных Вы знаете?

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Тип** | | **Размер (бит)** | **Минимальное значение** | | | **Максимальное значение** | |
| Byte | | 8 | -128 | | | 127 | |
| Short | | 16 | -32768 | | | 32767 | |
| Int | | 32 | -2147483648 | | | 2147483647 | |
| Long | | 64 | -922372036854775808 | | | 9223720368547758f | |
| Char | | 16 | 0 | | | 65536 | |
| **Тип** | **Размер (бит)** | | | **Диапазон** | **Точность** | |
| Float | 32 | | | 3,4e-38 < |x| < 3,4e38 | 7-8 цифр | |
| Double | 64 | | | 1,7e-308 < |x| < 1,7e308 | 17 цифр | |

* Что Вы знаете о преобразованиях примитивных типов данных, есть ли потеря данных, можно ли преобразовать логический тип?

Ответ: Можно преобразовывать примитивные типы данных от byte->short->int->long->float->double. В выражениях выполняется автоматически. Нельзя преобразовывать логический тип. Потеря данных возможна от long->float и от float ->double, если не воспользоваться модификатором strictfp.

\*Модификатор strictfp, гарантирует, что результаты вычисления чисел с плавающей точкой будут одинаковы на всех платформах.

* Какими значениями инициализируются переменные по умолчанию?

Ответ: В зависимости от типа данных. boolean – false, int, sort, char, byte – 0, long – 0L, float – 0,0, double – 0,0.

* Как передается значение переменной (по ссылке/значению)?

Ответ: В Java все передается по значению, а когда передается объект, то передается значение ссылки, а не прямой адрес в памяти (как в других языках).

* Что вы знаете о функции main, какие обязательные условия ее определения?

Ответ: public static void main(String … args)

иначе будет брошено java.lang.NoSuchMethodError: main Exception in thread "main"

* Какие логические операции и операторы Вы знаете?

Ответ:

1.отрицание (NOT) ! (обозначается восклицательным знаком);

2.конъюнкция (AND) & (амперсанд);

3.дизъюнкция (OR) | (вертикальная черта);

4.исключающее ИЛИ (XOR) ^ (каре).

* В чем разница краткой и полной схемы записи логических операторов?

Ответ: При использовании краткой схемы проверяются оба операнда. При использовании полной схемы проверяется только один.

* Что такое таблица истинности?

Ответ: Это таблица, описывающая логическую функцию.

* Что такое тернарный оператор выбора?

Ответ: Это оператор, оперирующий тремя операндами и использующий знак вопроса и двоеточие. Операнд1 ? операнд2 : операнд3. Если первый thru, то выполняется второй операнд, если первый false, то выполняется третий операнд.

* Какие унарные операции Вы знаете?

Ответ: Унарный плюс, унарный минус, инкремент, декремент. Логический оператор дополнения. Побитовое дополнение.

* Какая арифметическая операция выполняется при побитовом сдвиге вправо/влево?

Ответ: Двойка, возведенная в степень.(степень двойки 32.)

* Какова роль и правила написания оператора выбора?

Ответ:

* Какие циклы Вы знаете, в чем их отличия?

Ответ: Цикл for, while и do while, for each. Отличия while от do while в том, что цикл do while сначала выполнит, потом проверит. То есть do while выполнится хотя бы один раз. for each удобней использовать при работе с коллекцией, массивами, тогда ты оперируешь итератором.

* Что такое “итерация цикла”?

Ответ: Это полный проход цикла.

* Какие параметры имеет цикл for, можно ли их не задать?

Ответ: Цикл for имеет три параметра. Первый – инициализация, второй – условие выхода из цикла, третий – итерация. Да, все три параметра можно не задавать.

* Какой оператор используется для немедленной остановки цикла?

Ответ: break.

* Какой оператор используется для перехода к следующей итерации цикла?

Ответ: continue.

* Что такое массив?

Ответ: Массив – это конечный последовательный набор разных или одинаковых элементов одного типа.

* Какие виды массивов Вы знаете?

Ответ: Все массивы в java являются динамическими.

* Что Вы знаете о классах оболочках?

Ответ:

* Что такое автоупаковка (boxing/unboxing)?

Ответ: int c = Integer a +int b(unboxing)

Integer c = int a + int b(boxing)

* Что такое сериализация? Ключевое слово transient?

Ответ:Сериализация – это процесс сохранения состояния объекта в последовательность байтов;

десериализация – это процесс восстановления объекта из этих байт. Необходимо для передачи данных между объектами.

Ключевое слово transient применяется к полям классам и означает, что данное поле не входит в персистентное состояние класса, то есть при сериализации данное поле не будет сохраняться, и при десериализации оно не будет восстанавливаться.

**OOP**

* Назовите принципы ООП и расскажите о каждом.

Ответ: Три принципа ООП:  
1) Полиморфизм – это свойство системы использовать объекты с одинаковыми интерфейсами без информации о типе данных и внутренней структуре объекта.   
Примеры полиморфизма:   
System.out.println() – параметр.  
toString – ситуация.  
Уменьшение дублирования кода.   
Перегрузка и переопределение.   
2) Наследование – это свойство системы, позволяющее описать новый класс, на основе уже существующего с частично или полностью заимствующейся функциональностью.   
3) Инкапсуляция – это сокрытие свойств объекта и его реализации. Для этого используются модификаторы доступа, такие как:   
а) privet – члены класса доступны только внутри класса.  
b) default(package) – модификатор по умолчанию, члены класса видны только внутри пакета.  
c) protected – тоже, что и privet, но можно наследоваться из других пакетов.  
d) public – члены класса доступны всем.  
Геттеры позволяют достучаться до полей класса с модификатором privet.  
Сеттеры позволяют проверить закрытые поля на валидность.  
Пакеты так же являются частью инкапсуляции.

* Дайте определение понятию “класс”.

Ответ:   
Класс – это разновидность абстрактного типа данных в ООП, характеризуемый способом своего построения.  
Класс – это описание объектов и действий, которые можно с ним выполнять.  
Класс – это тип данных, состоящий из fields - полей и method - методов.

* Что такое поле/атрибут класса?

Ответ: Поле класса – это переменная или объект, связанная с классом или объектом.

* Как правильно организовать доступ к полям класса?

Ответ: Область доступа полей класса делается закрытой(privet), то есть доступ к ним разрешается только методам того же класса.

* Дайте определения понятию “конструктор”.

Ответ: Конструктор инициализирует объект во время создания. Конструктор имеет такое же имя, как и у класса, в котором он находится, а синтаксис аналогичен синтаксису метода. Как только он определен, конструктор автоматически вызывается непосредственно после создания объекта, перед завершением выполнения команды new. У конструктора нет типа возвращаемого результата.   
С помощью this можно вызывать цепочку конструкторов.  
«Для стартовой инициализации полей класса».

* Чем отличаются конструкторы по-умолчанию, копирования и конструктор с параметрами?

Ответ: Конструктор по-умолчанию – это конструктор, который может быть вызван без аргументов. Неявно присутствует всегда.  
Конструктор копирования – это специальный конструктор, применяемый для создания нового объекта как копии уже существующего. Такой конструктор принимает как минимум один аргумент, ссылку на копируемый объект. Конструктор задан неявно для каждого класса.  
Конструктор с параметром - это конструктор, принимающий один или более аргументов.

* Какие модификаторы уровня доступа Вы знаете, расскажите про каждый из них.

Ответ: а) privet – члены класса доступны только внутри класса.  
b) default(package) – модификатор по умолчанию, члены класса видны только внутри пакета.  
c) protected – тоже, что и privet, но можно наследоваться из других пакетов.  
d) public – члены класса доступны всем.

* Расскажите об особенностях класса с единственным закрытым (private) конструктором.

Ответ: Единственный закрытый конструктор гарантирует, что объект может быть создан только вашим методом (гарантирует, что не будет проблем с конструктором по-умолчанию).

* О чем говорят ключевые слова “this”, “super”, где и как их можно использовать?

Ответ: this – это ссылка на самого себя.  
this – возвращает ссылку на собственный созданный объект этого класса.   
super – также вернет ссылку на самого себя, но приведенную к родителю.  
Разница заключается в программной реализации. this будет искать ссылку внутри класса, super же пойдет выше к классу родителя. Через this к методам обычно не обращаются, так как методы редко повторяются. this и super вызываются внутри класса.

* Дайте определение понятию “метод”.

Ответ: Метод – это именованный обособленный блок кода.

* Что такое сигнатура метода?

Ответ: Сигнатура метода – это сокращенная форма записи параметров метода и типов возвращаемого значения.

* Какие методы называются перегруженными?

Ответ: Те, которые имеют одинаковые имена и различаются лишь по параметрам.   
Переопределение – одинаковые имена и параметры.   
Необходимо для того, чтобы использовать уже существующий код, расширяя функциональность.

* Могут ли нестатические методы перегрузить статические?

Ответ: Да можно, и наоборот. Статические методы могут перегрузить нестатические.

* Расскажите о перекрытии методов.

Ответ: Перекрытие методов – это возможность, позволяющая классу ребенку обеспечивать специфическую реализацию метода, который уже был реализован в классе родителя.

* Могут ли нестатические методы перекрыть статические?

Ответ. Да, могут.

* Может ли метод принимать разное количество параметров (аргументы переменной длины)?

Ответ: Да, могут.

* Можно ли сузить уровень доступа/тип возвращаемого значения при перекрытии метода?

Ответ: Нет, нельзя.

* Как получить доступ к перекрытым методам родительского класса?

Ответ: Super

* Какие преобразования называются нисходящими и восходящими?

Ответ: Родитель к субребенку приводя – нисходящие преобразования, субребенк к родителю приводя восходящие.

* Где можно инициализировать статические/нестатические поля?

Ответ:

* Зачем нужен оператор instanceof?

Ответ: Используя оператор instanceof можно узнать, от какого класса произошел объект. Два аргумента, слева ссылка на объект, а справа имя типа.

* Зачем нужны и какие бывают блоки инициализации?

Ответ: 1) Статические блоки инициализации – это блоки, код которых выполняется при первом запуске класса.  
2) Нестатические блоки инициализации – это блоки, которые позволяют проводить инициализацию объектов вне зависимости от того, какой конструктор был вызван.

* Каков порядок вызова конструкторов и блоков инициализации двух классов: потомка и его предка?

Ответ: Сначала работают блоки инициализации родителя, потом потомка. Конструктор – родителя. Конструктор – потомка.

* Где и для чего используется модификатор abstract?

Ответ: Модификатор abstract используется для объявления абстрактных классов и методов.

* Можно ли объявить метод абстрактным и статическим одновременно?

Ответ. Нет, нельзя.

* Как влияет модификатор static на класс/метод/поле?

**Ответ: Static – средство уникальное для класса а не объектов этого класса.  
Static:   
1) Поля с модификатором static.  
Поле, имеющее модификатор static, существует в одном экземпляре для всего класса. Если поле не содержит модификатора static, то каждый объект содержит его копию. Существовать можно один экземпляр статического поля, даже если не создано ни одного объекта, статическое поле существует, и оно принадлежит классу, а не конкретному объекту.   
Статические поля класса относятся к классу, а не к экземпляру класса.**

**2) Методы с модификатором static.  
Статические методы – это методы, которые не оперируют объектами.**

**Статические методы не работают с объектами, из них невозможно получить доступ к полям экземпляра. Нестатические методы имеют доступ к статическим полям класса.  
Используются только в двух случаях, когда методу не нужен доступ к информации о состоянии объекта, поскольку все параметры задаются явно, либо когда методу нужен доступ лишь к статическим полям класса.  
3)Классы с модификатором static.  
Только вложенный класс может быть статическим. Статический вложенный класс ничем не отличается от любого другого внутреннего класса за исключением того, что его объект не содержит ссылку на создаваемый его объект внешнего класса.**

* О чем говорит ключевое слово final?

Ответ: final – это модификатор(/константа/класс/метод), который означает, что присваивать значение можно только один раз и изменять его нельзя.  
Класс, который нельзя расширить, называется финальным.  
Отдельный метод, помеченный модификатором final, не может быть переопределен в подклассы.  
Модификатором final можно так же описывать поля, являющиеся константами.  
Все методы финального класса автоматические являются финальными.   
Но поля класса в константы не превращаются.

* Дайте определение понятию “интерфейс”.

Ответ: Интерфейс – это поведение какого-то объекта. Пример: контракт.   
Все методы, приведенные в интерфейсе, должны быть реализованы.

* Какие модификаторы по умолчанию имеют поля и методы интерфейсов?

Ответ: Отталкиваясь от джавы 7, интерфейс не может содержать в себе поля. Методы в интерфейсах по умолчанию либо public, либо abstract.

* Почему нельзя объявить метод интерфейса с модификатором final или static?

Ответ:

* Что вы знаете о вложенных классах, зачем они используются?

**Ответ: Вложенный класс – это класс, определенный внутри другого класса.  
Причины применения:   
1) Объект внутреннего класса имеет доступ к данным объекта, в котором он определен.  
2) Вложенный класс можно скрыть от других классов того же пакета.  
3) Анонимный вложенный класс удобен, если нужно определить обратный вызов в процессе выполнения программы, при этом не приходится создавать большой объем дополнительного кода.**

* Каким образом из вложенного класса получить доступ к полю внешнего класса?

Ответ: Через ссылку. Например: .this

* В чем разница вложенных и внутренних классов?

Ответ.

* Какие классы называются анонимными?

Ответ: Анонимный класс – это вложенный класс, который не имеет имени. То есть при создании одного единственного объекта данного класса этому классу имя можно не присваивать.

* Как связан любой пользовательский класс с классом Object?

Ответ: Все классы наследуются от класса Object.

* Расскажите про каждый из методов класса Object.

Ответ: Методы класса Object:

* Метод equals() – метод, который позволяет сравнивать ссылки по значению.
* Метод getClass() – метод, который возвращает объект класса, который описывает класс (имя, методы, поля), от которого был порожден этот объект.
* Метод clone() – метод сначала проверяет, можно ли клонировать исходный объект, если нет, то выкидывается эксепшен. Если можно, то порождается новый объект от того же класса, от которого был создан исходный объект.
* Метод toString() – метод, который позволяет получить текстовое описание любого объекта.
* Методы wait(), notify(), notifyAll() – методы используются для поддержки многопоточности.
* Метод finalize() – метод вызывает сборщик мусора, когда нет больше ссылок на объект.
* Метод hashCode() – возвращает хеш-код объекта.
* В чем особенность работы методов hashCode и equals?

Ответ:

* Правила переопределения метода clone()?

Ответ:

* Что за метод такой Object.finalize()?

Ответ:

* Какой метод возвращает строковое представление объекта?

Ответ: Метод toString()

* Чем отличается абстрактный класс от интерфейса, в каких случаях, что Вы будете использовать?

Ответ:

* Можно ли получить доступ к private переменным классам и если да, то каким образом?

Ответ: При помощи переопределения метода get().

**Java Core - Exceptions**

* Дайте определение понятию “исключение”.

Ответ: Исключение – это аварийное состояние, которое возникает в кодовой последовательности во время выполнение. Другими словами, это ошибка времени выполнения.

* Какова иерархия исключений.

Ответ: 

* Можно/нужно ли обрабатывать ошибки jvm?

Ответ: Ошибки jvm обрабатывать можно, но не нужно.

* Какие существуют способы обработки исключений?

Ответ:

* О чем говорит ключевое слово throws?

Ответ: Ключевое слово thorws помогает явно выбрасывать исключения. Также ключевое слово throws перечисляет типы исключения, которые метод может выбрасывать. Это необходимо для всех исключений, кроме исключений типа Error, RuntimeException или любых их подклассов.

* В чем особенность блока finally?

Ответ: Блок кода, который помечен ключевым словом finally, обязательно выполнится.

* Может ли не быть ни одного блока catch при отлавливании исключений?

Ответ:

* Может ли один блок catch отлавливать несколько исключений (с одной и разных веток наследований)?

Ответ: Да, может.

* Что вы знаете об обрабатываемых и не обрабатываемых (catched/uncatched) исключениях?

Ответ:

* В чем особенность RuntimeException?

Ответ: RuntimeException – это исключение, причина которого скрыта в логике программы, и вина лежит полностью на разработчике. Такие исключениями являются uncatched, то есть не обрабатываемые.

* Как написать собственное (“пользовательское”) исключение?

Ответ:

* Какой оператор позволяет принудительно выбросить исключение?

Ответ: оператор finally.

* Есть ли дополнительные условия к методу, который потенциально может выбросить исключение?

Ответ: Ключевое слово throws, которое прописывается в сигнатуре метода, и обозначающее, что метод потенциально может выбросить исключения с указанным типом.

* Может ли метод main выбросить исключение во вне. Если да, то где будет происходить обработка данного исключения?

Ответ: Да может, обработка будет происходить на уровне jvm.

* Если оператор return содержится и в блоке catch и в finally, какой из них “главнее”?

Ответ:

* Что такое checked и unchecked Exception?

Ответ: Checked – это контролированное исключение, которые должны обрабатываться блоком catch или описываться в сигнатуре метода. Unchecked, могут не обрабатываться, и не быть описанным.  
Unchecked – это неконтролируемые исключения, обрабатывать которые можно, но не нужно. Так как нет у программиста инструмента отлавливать неконтролируемые исключения. Так же, если есть возможность, необходимо стараться избежать исключения типа unchecked, а не пытаться их отлавливать.

Все классы, которые реализуют интерфейс AutoCloseable, могут быть использованы в конструкции try с ресурсами.

**Java Core - Collections**

* Дайте определение понятию “коллекция”.

Ответ: Коллекции – это хранилище, поддерживающие различные способы накопления и упорядочения объектов с целью обеспечения возможностей эффективного доступа к ним.

* Назовите преимущества использования коллекций.

Ответ: 1. Уменьшает затраты на программирование. 2. Увеличивает скорость и качество программ. 3. Поощряет повторное использование. (API);

* Какие данные может хранить коллекции?

Ответ: Объекты.

* Какова иерархия коллекций?

Ответ:



* Что вы знаете о коллекциях типа List?

Ответ: Список – это упорядоченная коллекция. Список может содержать повторяющиеся элементы. Интерфейс List сохраняет последовательность добавления элементов и позволяет осуществлять доступ к элементу по индексу.

* Что вы знаете о коллекциях типа Set?

Ответ: Интерфейс Set определяет множество (набор). Он расширяет Collection и определяет поведение коллекций, не допускающих дублирование элемента. Интерфейс Set не определяет никаких собственных дополнительных методов.  
Интерфейс Set заботиться об уникальности хранимых объектов, уникальность определяется реализацией метода equals.

* Что вы знаете о коллекциях типа Map, в чем их принципиальное отличие?

Ответ: Коллекция типа Map представляет собой набор данных, в котором можно быстро найти существующий элемент. Однако для этого нужно иметь точную копию требуемого элемента. Обычно имеется некоторая информация(ключ), по которой и можно найти соответствующий элемент. Специально для этого предназначена структура данных, которая называется карта. Карта хранит пары ключ-значение. Каждое значение можно найти по ключу.

* Назовите основные реализации List, Set, Map.

Ответ: 1. Реализация List:   
1.ArrayList – коллекция, представляя собой обычный массив, длина которого автоматически увеличивается при добавлении новых элементов.   
2.LinkedList – двусвязный список, состоящие из узлов, каждый из которых содержит как собственные данные, так и две ссылки на следующий и предыдущий узел списка.   
2.Реализация Set:   
1.HashSet – коллекция, которая не позволяет хранить одинаковые объекты(как и любой Set). HashSet инкапсулирует в себе объекты HashMap,( то есть использует для хранения хэш-таблицу).  
 2.LinkedHashSet – поддерживает связной список элементов набора в том порядке, в котором они добавлялись.   
3.TreeSet – коллекция, которая хранит свои элементы в виде упорядоченного по значениям бинарного дерева.  
3. Реализация Map:   
Хэш-карта HashMap и карта-дерево TreeMap. В хэш-карте ключи расположены случайным образом, в карте-дереве, в строгом порядке. Хэш-функция применяется только к ключам, и сравниваются только ключи, соответствующие значение не сравниваются и не хэшируются.

* Какие реализации SortedSet Вы знаете и в чем их особенность?

Ответ: SortedSet – это интерфейс, реализуемый классом TreeSet.

* В чем отличия/сходства List и Set?

Ответ: Сходство List и Set в том, что это упорядоченные коллекции и обе эти коллекции сохраняют последовательность добавления элементов.   
Отличия Set от List в том, что Set заботиться об уникальности элементов, а List нет.

* Что разного/общего у классов ArrayList и LinkedList, когда лучше использовать ArrayList а когда LinkedList?

Ответ:ArrayList – это список , реализованный на основе массива. LinkedList – это классический двухсвязный список, основанный на объектах с ссылками между ними. Все упирается в то, что нам необходимо. Если происходит постоянная вставка/удаление элементов с середины списка или в случаях, когда необходимо гарантировать время добавление элемента в списках, используется LinkedList. Во всем остальном, LinkedList проигрывает ArrayList, и по потреблению памяти и по скорости выполнения операции.

* В каких случаях разумно использовать массив а не ArrayList?

Ответ: Массив стоить применять лишь в том случае, если вы заранее и точно знаете размер массива, и то, что использование этого массива в коде ограниченно. Во всех остальных нужно использовать ArrayList.

* Чем отличается ArrayList от Vector?

Ответ: Vector – синхронизирован, ArrayList – нет. (ArrayList – увеличивает свой размер, по заранее заданной формуле, «oldCapacity\*3/2+1», тогда как Vector, увеличиваю длину на заранее заданное количеству элементов, если эти элементы не заданы, то Vector увеличивает свою длину вдвое.)

* Что вы знаете о реализации классов HashSet и TreeSet,?

Ответ: 1.TreeSet – обеспечивает упорядоченное хранение данных в виде бинарного дерева.   
2.HashSet – использует тот же подход что и HashMap, за исключением того, что в HashSet ключом выступает сам элемент.

* Что будет, если в Map положить два значения с одинаковым ключом?

Ответ: Ключи должны быть уникальными: нельзя сохранить два значения с одинаковым ключом. Если вызвать метод put() дважды с одинаковым ключом, то второе значение заменит первое.

* Как задается порядок следования объектов в коллекции, как отсортировать коллекцию?

Ответ: Класс Collections.

* Дайте определение понятию “итератор”.

Ответ: Итератор – это интерфейс предоставляющий доступ к элементам коллекций и навигацию по ним.

* Какую функциональность предоставляет класс Collections?

Ответ: Общие интерфейсы по работе с коллекциями. Копировать, сортировать, поиск по коллекции, перемешать коллекция.

* В чем разница между HashMap и HashTable?

Ответ:1. HashMap - состоит из корзин. Корзины – это элементы массива, которые хранят ссылки на списки элемента. При добавлении новой пары ключа-значения, вычисляется хеш-код, на основании которого вычисляются номер корзины(номер ячейки в массиве), в которую попадает новый элемент. Если корзина пуста, то в нее сохраняется ссылка на вновь добавляемый элемент, если же там уже есть элемент, то происходит последовательный переход по ссылкам между элементами в цепочки, в поисках последнего, от которого и ставиться ссылка на вновь добавляемый элемент. Если был найден элемент с таким же ключом, то он заменяется. Все происходит за константное время.

2.HashTable – подобен HashMap, но он синхронизирован. Он так же сохраняет пары «ключ-значение» только ни ключи, ни значения не могут быть равны null. Используя HashTable мы указываем объект, который служит ключом, и значение, которые мы связываем с этим ключом. Ключ затем хешируется, а хеш-код используется в качестве индекса, по которому значение сохраняется в таблице.

* Какой предок у класса Hashtable?

Ответ: Dictionary – предок класса HashTable.

* Как получить не модифицируемую коллекцию?

Ответ: Для того чтобы создать не модифицируемую коллекцию необходимо переопределить метод size() и iterator().

* Какие коллекции синхронизированы? -

Ответ: Vector, HastTable.

* Как получить синхронизированную коллекцию из не синхронизированной?

Ответ: Нужно использовать методы класса Collections.

* Как получить коллекцию только для чтения?

Ответ: Необходимо использовать следующие методы:   
1. Collections.unmodifiableList(list);  
2. Collections.unmodifiableSet(set);

3. Collections.unmodifiableMap(map);

Все они принимают коллекцию в качестве параметра, и возвращают коллекцию только для чтения с теми же элементами внутри.

* Почему Map не наследуется от Collection?

Ответ: Потому что они не совместимы. В интерфейс Collection описан метод add(Object o). Map не может содержать этот метод, потому что работает с парами ключ/значение. Так же у Map есть представления keySet, valueSet, которых нет в коллекции. В связи с этими различиями, интерфейс Map не может наследовать интерфейс Collection и представляет собой отдельную ветвь.

* В чем разница между Iterator и Enumeration?

Ответ: Итераторы отличаются от перечисления по нескольким признакам:  
1. Существуют итераторы, которые позволяют удалять элементы из своей коллекции во время перебора с использованием метода remove(). Класс Iterator данного функционала не имеет. С помощью перечислителя нельзя добавлять/удалять элементы.  
2. Перечислитель присутствует во всех устаревших классах, таких как Vector/Stack, тогда как итераторы есть во всех современных классах-коллекциях.

* Как реализован цикл foreach?

Ответ:

* Почему нет метод iterator.add() чтобы добавлять элементы в коллекцию?

Ответ: Единственная задача итератора это перебор коллекций. Каждая коллекция имеет метод add(), которым мы можем воспользоваться. Нет смысла, добавлять этот метод в итератор, потому что коллекции могут быть упорядоченными и неупорядоченными, и метод add() при этом должен быть устроен по разному.

* Почему в классе итератор нет метода для получения следующего элемента без передвижения курсора?

Ответ: Потому что итератор это указатель на некоторый объект в коллекции и надо применить метод next() чтобы перейти на следующий объект.

* В чем разница между Iterator и ListIterator?

Ответ: Есть три различия:  
1. Iterator может использоваться для перебора элементов Set,List и Map. В отличии от него ListIterator может быть использован только для перебора элементов коллекции List.

2. Iterator позволяет перебирать элементы только в одном направлении при помощи метода next(). Тогда как ListIterator позволяет перебирать список в обоих направлениях, при помощи метода next() previous().

3. При помощи ListIterator мы можем модифицировать список добавляя/удаляя элементы с помощью методов add() **и remove().**Iterator не поддерживает данного функционала.

* Какие есть способы перебора всех элементов List?

Ответ: Есть четыре способа:  
1. Цикл с итератором.  
2. Цикл for.  
4. Цикл while.

* В чем разница между fail-safe и fail-fast свойствами?

Ответ: Итератор fail-safe вызывает исключение сразу как обнаружит, что структура коллекции была изменена, после начала перебора. Под изменение структуры, подразумевается добавление/удаление/изменение элемента коллекции, в то время как другой поток перебирает эту коллекцию. В противоположность итератору fail-safe, итератор fail-fast не вызывают никаких исключений при изменении структуры потому что он работает с клоном коллекции, вместо оригинала.

* В каких случаях может быть выброшено ConcurrentModificationException?

Ответ: Если мы попытаемся добавить элемент через итератор(попробовать)

* Что такое стек и очередь, расскажите в чем их отличия?

Ответ: Обычно очереди работают по принципу FIFO – первый вошел, первый вышел. Стек почти как очередь, но работает по принципу LIFO – последний вошел, первый вышел.(Стек – это магазин автомата при стрельбе. Очередь – это трубка в которую мы опускаем шарики.)

* В чем различия между интерфейсами Comparable и Comparator?

Ответ: Для того чтобы рассортировать элементы, класс должен реализовывать интерфейсы Comparator и Comparable. Именно потому, классы-обертки, такие как Integer, Double и String реализуют интерфейс Comparable. Интерфейс Comparable помогает сохранять естественную сортировку, тогда как Comparator позволяет сортировать элементы по разным шаблонам. Экземпляр компаратора обычно передается конструктору коллекции, если коллекция это поддерживает.

* Почему коллекции не наследуют интерфейсы Cloneable и Serializable?

Ответ: Первая причина, потому что нет необходимости. Функционал предоставляемый Cloneable и Serializable попросту не нужен коллекциям.  
Вторая причина – далеко не всегда необходим подкласс Cloneable потому что каждая операция клонирования потребляет слишком много памяти, и неопытные программисты могут расходовать ее сами не понимая последствий.

И последняя причина заключается в том, что клонирование и сериализация являются очень узко-специфичными операциями, и реализовывать их нужно только тогда, когда это необходимо.

* Опишите, как работают HashSet, LinkedHashSet, TreeSet и объясните между ними отличия.

Ответ: TreeSet – обеспечивает упорядоченное хранение данных в виде бинарного дерева.  
2.HashSet – использует тот же подход что и HashMap, за исключением того, что в HashSet ключом выступает сам элемент.   
3. LinkedHashSet – поддерживает связной список элементов в том порядке,, в котором они вставлялись. Это позволяет организовать упорядоченную итерацию вставки в набор.

* Каким образом можно отсортировать коллекцию?

Ответ: При помощи классов Collections.

* Как правильно удалить элемент из ArrayList или другой коллекции при поиске этого элемента в цикле?

Ответ: Либо с помощью итератор. Либо с помощью метода remove();

* Как правильно удалить элемент из коллекции при итерации в цикле?

Ответ: c помощью метода remove();

**Java Core - Strings**

* Какие “строковые” классы Вы знаете?

Ответ: String, StringBuffer и StringBilder.

* Какие основные свойства ”строковых” классов (их особенности)?

Ответ: Стринг – иммутбл. Остальные мутбл

* Можно ли наследовать строковый тип, почему?

Ответ: Нет нельзя, так как все строковые классы помечены модификатором final.

* Дайте определение понятию конкатенация строк.

Ответ: Конкатенация строк – это объедение нескольких строк, в одну.

* Как преобразовать строку в число?

Ответ: Преобразовать строк в число можно несколькими способами. Первое – с помощью конструктора. Второе – с помощью метода valueOf класса соответствующего типа данных.(возвращает обертку, использует парсИнт, но приводит к обертке) Третье – с помощью метода parse класса соответствующего типа данных.(возвращает примитивные типы данные)

* Как сравнить значение двух строк?

Ответ: При помощи метода equals, используя метод compareTo()(выводит меньше или больше ноля, или ноль)

* Как работает сравнение двух строк?

Ответ: 1. «==» - сравнивает по ссылкам, если два объекта ссылаются на один и тот же объект, то объекты равны. 2. Метод equals – сравнивает по значению ссылок. 3. Метод compareTo – определяет какая строка больше, сравнивает как и equals по значению ссылок.

* Как обрезать пробелы в конце строки?

Ответ: Нужно воспользоваться методом trim.

* Как заменить символ в строке?

Ответ: Нужно использовать метод replace() где первым пишется старые элемент, и через запятую новый.

* Как получить часть строки?

Ответ: Нужно использовать метод substring();

* Дайте определение понятию “пул строк”?

Ответ: Пул строк – это хэш строк который хранится в памяти, который кэширует строки, под которые не выделилась память, в одной области видимости.

* Какой метод позволяет выделить подстроку в строке?

Ответ: substring();

* Как разбить строку на подстроки по заданному разделителю?

Ответ: Метод split();

* Какой метод вызывается для преобразования переменной в строку?

Ответ: Необходимо использовать метод toString();

* Как узнать значение конкретного символа строки, зная его порядковый номер в строке?

Ответ: CharAt();

* Как найти необходимый символ в строке?

Ответ: CharAt();

* Можно ли синхронизировать доступ к строке?

Ответ: Можно.

* Что делает метод intern()?

Ответ: System.out.println("true" == new String("true").intern());

* Чем отличаются и что общего у классов String, StringBuffer и StringBuilder?

Ответ: String, StringBuffer, StringBuilder – это три класса оперирующие строками.   
Отличие между String и StringBuffer, StringBuilder есть в том, что стринг являются иммутбл, то есть неизменяемым, в отличие же от своих собратьев, StringBuffer и StringBuilder. Что значит иммутбл? Это означает, что вы не можете модифицировать объект класса String, но можете заменить его созданием нового экземпляра класса. StringBuffer же и StringBuilder, мутбл и это позволяет нам работать со строками без затраты памяти. В чем же тогда отличие StringBuffer и StringBuilder? Разница заключается в том, что StringBuilder куда более быстрее StringBuffer, но связано это с тем, что он не синхронизирован, а значит не потока безопасен.

* Как правильно сравнить значения строк двух различных объектов типа String и StringBuffer  
  Ответ: Привести объект StringBuffer к toSting();

**Java Core - IO**

* Какие существуют виды потоков ввода/вывода?

Ответ:

* Назовите основные предки потоков ввода вывода.

Ответ: InputStream и OutputStream.

* Что общего и Чем отличаются следующие потоки: InputStream, OutputStream, Reader, Writer?
* Что Вы знаете о RandomAccessFile?

Ответ: Класс произвольного доступа к файлу RandomAccessFile может реализовывать интерфейсы как к DataInput, так и к DataOutput. Это означает, что класс RandomAccessFile может быть использован как для ввода данных из файла, так и для вывода в файл.

* Какие есть режимы доступа к файлу?
* В каких пакетах лежат классы-потоки?

Ответ: в классах пакета **java.io**

* Что Вы знаете о классах-надстройках?
* Какой класс-надстройка позволяет читать данные из входного байтового потока в формате примитивных типов данных?
* Какой класс-надстройка позволяет ускорить чтение/запись за счет использования буфера?
* Какие классы позволяют преобразовать байтовые потоки в символьные и обратно?
* Какой класс предназначен для работы с элементами файловыми системы (ЭФС)?
* Какой символ является разделителем при указании пути к ЭФС?
* Как выбрать все ЭФС определенного каталога по критерию (например, с определенным расширением)?
* Что Вы знаете о интерфейсе FileFilter?
* Что такое сериализация?
* Какие условия “благополучной” сериализации объекта?
* Какие классы позволяют архивировать объекты?

**Java Core - Multithreading**

* Дайте определение понятию “процесс”.
* Дайте определение понятию “поток”.
* Дайте определение понятию “синхронизация потоков”.
* Как взаимодействуют программы, поцессы и потоки?
* В каких случаях целесообразно создавать несколько потоков?
* Что может произойти если два потока будут выполнять один и тот же код в программе?
* Что Вы знаете о главном потоке программы?
* Какие есть способы создания и запуска потоков?
* Какой метод запускает поток на выполнение?
* Какой метод описывает действие потока во время выполнения?
* Когда поток завершает свое выполнение?
* Как синхронизировать метод?
* Как принудительно остановить поток?
* Дайте определение понятию “поток-демон”.
* Как создать поток-демон?
* Как получить текущий поток?
* Дайте определение понятию “монитор”.
* Как приостановить выполнение потока?
* В каких состояниях может прибывать поток?
* Какие способы синхронизации есть в Java?
* Что является монитором при вызове нестатического и статического метода?
* Что является монитором при выполнении участка кода метода?
* Какие методы позволяют синхронизировать выполнение потоков?
* Какой метод переводит поток в режим ожидания?
* Какова функциональность методов notify и notifyAll?
* В чем возможный недостаток вызова метода notifyAll?
* Что позволяет сделать метод join?
* Каковы условия вызова метода вайт notify?
* Дайте определение понятию “взаимная блокировка” (dead lock).
* Чем отличаются мтеоды interrupt, interrupted, isInterrupted?
* В каком случае будет выбрашено исключение InterruptedException, какие методы могут его выбросить?
* Как работает метод Thread.yield()? Чем отличаются методы Thread.sleep() и Thread.yield()?
* Как правильно завершить работу потока? (“убить поток”)
* На каком объекте происходит синхронизация при вызове static synchronized метода?
* Для чего используется ключевое слово volatile?

**Java - JDBC**

* Этапы работы с базой данных с использованием JDBC?
* Как создать Connection?
* Чем отличается Statement от PreparedStatement?
* Как вызвать хранимую процедуру?
* Как правильно закрыть Connection?

**Web Technologies - Common**

* Что такое www?

Ответ: www – это единство информационных ресурсов, которые связаны между собой средствами телекоммуникаций и основаны на гипертекстовом представлении данных, разбросанных по всему миру.

* Что такое w3c?

Ответ:w3c – это Консорциум Всемирной паутины, организация разрабатывающая и внедряющая технологически стандарты для Всемирной паутины.

* Что такое TCP/IP?

Ответ: TCP/IP – набор сетевых протоколов передачи данных, используемых в сетях, в том числе Интернет. TCP – transport control Protocol. IP – internet protocol. В DOD модели TCP/IP расположены на сетевом уровне. В OSI модели, TCP расположен на транспортном уровне, а IP на сетевом.

* Что такое http?

Ответ: http – протокол передачи гипертекста, используется для передачи веб-страниц. Использует 80 порт.

* Что такое ftp?

Ответ: ftp – это протокол передачи файлов, используется для передачи файлов между двумя устройствами. Он использует один порт для доставки данных, и еще один для отправки управляющих сигналов. Обычно используются порты номером 20 и 21

* Чем отличаются http и https?

Ответ: https – это, в свою очередь, безопасный вариант http.

* Что такое протокол передачи данных, какие Вы знаете?

Ответ: Протокол передачи данных – это набор соглашений интерфейса логического уровня, которые определяют обмен данными между различными программами. Эти соглашения задают единообразный способ передачи данных и обработки ошибок.

* Что такое web server?

Ответ: Web server – это сервер, принимающий HTTP-запросы от клиентов, обычно веб-браузеров, и выдающий им HTTP-ответ, как правило, вместе с HTML-страницей, изображением, файлом, медиа-потоком или другими данными.

* Что такое web приложение?

Ответ: web приложение – это клиент-серверное приложение, в котором клиентом обычно выступает веб-браузер, а сервером – веб-сервер. Логика веб-приложений распределена между сервером и клиентом, хранение данных осуществляется, преимущественно, на сервере, обмен информации происходит в сети.

* Что такое application server?

Ответ: Application server – это программная платформа(фреймворк), предназначенный для эффективного исполнения процедур, которые поддерживают построение приложения. Сервер приложения действует как компонентов, доступных разработчику программного обеспечения.

* Чем отличаются web server и application server?

Ответ: Сервер приложения – сервер, исполняющий некоторые прикладные программы. Веб-сервер – это сервер, принимающий HTTP-запросы от клиентов, обычно веб-браузеров, и выдающий им HTTP-ответ

* Какие методы передачи данных Вы знаете?

Ответы: TCP и UDP – это два основных протокола, которые используются на транспортном уровне для передачи данных. TCP протокол устанавливает логическое соединение (сессию) между двумя хвостами в сети. Он устанавливает соединение, используя процедуру троекратного рукопожатия. Это гарантирует доставку всех фрагментов информации в нужном порядке. Но из-за этого существенно падает скорость такого трафика. UDP протокол, в свою очередь совершенно не заботиться даже о наличии соединения. Он просто передает данные, даже не заботясь о нумерации фрагментов и о том, получил ли конечный узел передаваемую информацию или нет. Такая особенность делает UDP очень быстрым.

* Чем отличаются методы get и post?

Ответ:

* Что такое html?

Ответ: HTML – Язык гипертекстовой разметки.

* Что такое xml?

Ответ:XML – Расширяемый язык разметки. (поведение, и доступ к их содержимому.).

* Что такое css?

Ответ: CSS – язык описания внешнего вида документа.

* Что такое MIME type?

Ответ: MIME (Multipurpose Internet Mail Extensions, многоцелевые расширения интернет-почты) — стандарт Интернет, является частью протокола HTTP. Задача MIME это идентификация типа содержимого документа по его заголовку. К примеру, текстовый файл имеет тип text/plain, а HTML-файл — text/html.

* Что такое cookies?

Ответ: Небольшой фрагмент данных, отправленных веб-сервером и хранимый на компьютере пользователя.

* Что такое сессия?

Ответ: Сессия – это логические соединения между компьютерами.

* Дайте определение понятиям “авторизация” и “аутентификация”, в чем их различия?

Ответ: Авторизация – это предоставление определенному лицу или группе лиц, прав на выполнение определенных действий, а так же процесс проверки данных прав при попытке выполнения этих действий. Аутентификация – это процедура проверки подлинности.

* Что такое SOAP веб-сервис?

Ответ:

* Что такое REST веб-сервис?

Ответ:

Протоколы:



Сети.

1.IP-адресс – это уникальный сетевой адрес узла в компьютерной сети, построенный по протоколу IP. Должен быть глобально уникален в рамках сети Интернет, если это локальная сеть, то в рамках локальной сети.  
В четвертой версии протокола IPv4,адрес имеет длину 4 байта, представляя собой 32-битовое число.  
В шестой версии протокола IPv6, IP адрес имеет длину 16 байт, представляя собой 128-битовое число.

2. Эталонная модель OSI. Имеет 7 уровней:  
7. Прикладной уровень.(application layer). – реализовано на уровне программ.

6. Представительский уровень или уровень представления(presentation layer) – на уровне программ  
5. Сеансовый уровень(session layer) – реализовано на уровне программ.  
4. Транспортный уровень(transport layer) – реализовано на уровне программ.  
3. Сетевой уровень(network layer) – реализовано на уровень программ.  
2. Канальный уровень(date link layer) – реализовано на уровне железа и программ.

1. Физический уровень(physical layer) – реализовано на уровне железа.

Прикладной уровень.(application layer) – это самый верхний уровень модели, он осуществляет связь пользовательских приложений с сетью. К примеру: HTTP – порт 80(HTTPS; порт – 443 ) – просмотр веб-страниц, SMTP – порт 25(P0P3 – порт110 ) – передача и приём почты. FTP – порт 21(TFTP; порт – ) – передача и получение файлов. Telnet(порт 23) – удаленный доступ.  
Представительский уровень или уровень представления(presentation layer) – он преобразовывает данные в соответствующий формат. От битов к соответствующему формату. Работает с протоколами изображений, кодировки, музыки и видео и так далее.

Сеансовый уровень(session layer) – уровень организации сеанса связи между двумя компьютерами. (К примеру аудио и видео связь.)

Транспортный уровень(transport layer) – это уровень обеспечивает надежность передачи данных от отправителя к получателю. Выделяется два протокола UDP(User Datagram Protocol; порт -) и TCP(Transmissions Control Protocol; порт - ). UDP протокол, передает данные без установления соединения, не подтверждает доставку данных и не делает повторы. TCP протокол, которые перед передачей данных устанавливает соединение, подтверждает доставку данных, при необходимости делает повторы, гарантирует целостность и последовательность загружаемых данных.

Сетевой уровень(network layer) – этот уровень определяет путь, по которому данные будут переданы. Протоколы: IP, ICMP, IGMP.

Канальный уровень(date link layer) – этот уровень необходим для взаимодействия сетей на физическом уровне. Существуют два подуровня LLC(Logical Link Control) – управление логическими каналами, создан для взаимодействия с верхними уровнями. И MAC (Media Access Control) – управления доступом к передающей среде, создан для взаимодействия с нижним уровнем.   
Существую такие протоколы: PPP(Point –to-Point) – протокол для связи двух компьютеров напрямую. FDDI(Fiber Distributer Date Interface) – протокол для передачи данных на расстоянии до 200 километров. CDP(Cisco Discovery Protocol) – протокол для обнаружения соседних устройств и получения информации об этих устройствах.

Физический уровень(physical layer) – самый нижний уровень, непосредственно осуществляющий передачу потока данных.

3. MAC-адрес(Media Access Control) – это уникальный модификатор, присваиваемый каждой единице активного оборудования компьютерных сетей. 48 бит. 6 байт.

4. Порты:  
20. – FTP-date, для передачи данных File Transfer Protocol.  
21. – FTP, для передачи команд File Transfer Protocol.  
22. – SSH(Secure SHell) , криптографический протокол для безопасной передачи данных.

23. – Telnet, применяется для передачи текстовых сообщений в незащищенном виде.  
25. – SMTP(Simple Mail Transfer Protocol), применяется для пересылки почтовых сообщений в виде незашифрованного текста.

43. – WHOIS(who is?), сетевой протокол прикладного уровня, применяется для получения регистрационных данных о владельцах доменных имен, IP-адресов и автономных систем.

53. – DNS(Domain Name System), система доменных имен, компьютерная распределенная система для получения информации о доменах. Чаще всего используется для получения IP-адреса по имени хоста.  
80. – HTTP(Hyper Text Transfer Protocol), протокол для передачи гипер текста.   
143. – IMAP(Internet Message Access Protocol), протокол используется для получения и синхронизации сообщений электронной почты.

443. – HTTPS(Hyper Text Transfer Protocol Secure), протокол HTTP c шифрованием SSL и TLS.

3389 – RDP(Remove Desktop Protocol) , протокол удаленного доступа.

Тестирование:   
1. SQ(Software Quality)Качество программного обеспечения – это степень в которой программное обеспечение обладает требуемой комбинацией символов.(ЧТО БЛЯТЬ?!)  
Это совокупность характеристик программного обеспечения, относящихся к его способности удовлетворять установленные и предлагаемые потребности.

2. QA(Quality Assurance) Обеспечение качества – это совокупность мероприятий, охватывающих все технологические этапы разработки, выпуска и эксплуатации программного обеспечения (ПО) информационных систем, предпринимаемых на разных стадиях жизненного цикла ПО, для обеспечения требуемого уровня качества выпускаемого продукта.

3. QC(Quality Control) Контроль качества - это совокупность действий, проводимых над продуктом в процессе разработки, для получения информации об актуальном состоянии в разрезах: «готовность продукта к выпуску», «соответствие зафиксированным требованиям», «соответствие заявленному уровню качества продукта».

4. ST(Software Testing)Тестирование программного обеспечения – это одна из техник контроля качества, включающая в себе активность по планированию работ(Test Management), проектированию тестов(Test Design), выполнение тестирования(Test Execution) и анализ полученных результатов.(Test Analysis).

5. Verification. Верификация – это процесс оценки системы или ее компонентов с целью определения удовлетворяют ли результаты текущего этапа разработки условиям, сформированным в начале этапа. Т.е. выполняются ли наши цели, сроки, задачи по разработке проекта, определенными в начале текущей фазы.

6. Validation. Валидация – это определение соответствия разрабатываемого ПО ожиданиям и потребностям пользователя, требованиям к систему.

Отличие:  
1.Тестирование. Testing – это «контроль качества».  
2. Контроль качества. Quality control – это измерение качества продукта.

3. Обеспечения качества. Quality Assurance – это измерение и управление качеством процесса, который используется для создания качества продукта(или качественного продукта).

Баг – это отклонения фактического результата, от ожидаемого результата.   
Жизненный цикл бага:  


Тестирование – это поиск багов.  
Нахождение багов:  
1. Мы узнаем (или уже знаем) ожидаемый результат.  
2. Мы узнаем (или уже знаем) фактический результат.  
3. Мы сравниваем пункт один и пункт два.

Спецификация – это детальное описание того, как должно работать ПО.

В большинстве случаем баг – это отклонение от спецификации.

1. По знанию внутренностей системы.

а.) Черный ящик (black box). Тестирование проходит без знания того, как устроен виртуальный мост. Идеи для тестирования идут от предполагаемых паттернов (образцов) поведения пользователей.

б) Белый ящик (white box). Тестирование проходит со знанием об устройстве и логике тестируемой части.

в) Серый ящик(gray/grey box). Тестирование сочетает подход черного и белого ящика,  
информативное тестирование ориентированное на пользователя.

2. По объекту тестирования.

а) Функциональное тестирование (functional testing) – это тестирование функциональности ПО  
б) Тестирование интерфейса пользователя (UI testing) – это тестирование при котором проверяются элементы интерфейса пользователя.

в) Тестирование локализации (localization testing) – тестирование, связанное с адаптацией сайта для пользователей разных стран.

г) Тестирование скорости и надежности (load/stress/performance testing) – это проверка поведения веб-сайта (или его отдельных частей) при одновременном наплыве множества пользователей.

д) Тестирование безопасности (security testing) – сущность тестирования заключается в том, чтобы усложнить условия для кражи данных, денег или информации.

е) Тестирование опыта пользователя (usability testing) – призвано объективно оценить опыт пользователя, который будет работать с разрабатываемой программой.

ё) Тестирование совместимости (compatibility testing) – это проверка того, как наш веб-сайт взаимодействует с железом и ПО(браузерами) наших пользователей.

Тест-кейс – это набор тестовых шагов для проверки определенного функционала программы.  
Баг-репортер – это документ, описывающий ситуацию или последовательность действий, приведшую к некорректной работе программы, с указанием причин и ожидаемого результата.

**ARP** ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *Address Resolution Protocol* — протокол определения адреса) — протокол в компьютерных сетях, предназначенный для определения [*MAC-адреса*](https://ru.wikipedia.org/wiki/MAC-%D0%B0%D0%B4%D1%80%D0%B5%D1%81) по известному [IP-адресу](https://ru.wikipedia.org/wiki/IP-%D0%B0%D0%B4%D1%80%D0%B5%D1%81).

Selenium – это набор инструментов для автоматизированного управления браузерами. Selenium, поддерживается производителями популярных браузеров.   
Продукты семейства Selenium:  
1. Selenium IDE – необходим для небольших сценариев и быстрого автоматизированного воспроизведения бага. Вспомогательный скрипт для выполнения отдельных рутинных действий при ручном тестировании. Selenium IDE – это расширение для браузера Firefox, которое позволяет воспроизводить и записывать действия пользователя в браузере. Главный минус, доступен только для браузера Firefox.

2. Selenium Web Driver – набор библиотек, для различных языков программирования, позволяющих управлять браузером из программы, написанном на этом языке программирования. Selenium Web Driver – это, надежный фреймворк автоматизации, способный работать с любым браузером. Большой тестовый набор, включающий в себя тесты с достаточно сложной логикой поведения и проверок.

3. Selenium Server, может принимать команды с удаленной машины, где работает сценарий автоматизации, и исполнять их в браузере. Несколько серверов Selenium могут образовывать сеть, которая называется Selenium Grid, что позволяет легко масштабировать стенд автоматизации.

4. Selenium RC – предшественник Selenium Web Driver, на данный момент имеющий статус «замороженного».

Фреймворк – это   
Maven – это инструмент для проекта на основе описания их структур в файлах.POM – один файл.

Git - Распределенная система управления версиями. Система контроля версиями.  
Основные команды Git(a):  
1. $ git init – эта команда создает в текущем каталоге новый подкаталог с именем .git содержащий все необходимые файлы репрозитория – основу Git репрозитория  
2. $ git add – указывает индексируемые файлы, а git commit – коммитит файлы(фиксирует изменения)  
3. git clone – клонирует почти все файлы с сервера.  
4. git status – показывает в каком состоянии находятся файлы.

5. git ignor – игнорирует файлы.  
6. git diff – более расширенная версия git status, позволяет посмотреть что мы изменили, но еще не проиндексировали и что мы проиндексировали и собираемся фиксировать.

7. git rm – помогает удалить файлы. (удаляет из отслеживаемого индекса.)

8. git log – просмотр истории коммитов.

9. git pull – автоматически извлекает и сливает данные из удаленные ветки, в вашу текущую ветку.  
push, merge.

Test Ng – это фреймворк для тестирования, написанный Java, он взял много чего с JUnit и NUnit. Но он не только унаследовался от существующего JUnit, a также внедрения новых инновационных, которые делают его мощным, простым в использовании.   
TestNg предназначается для:  
unit тестирования;

функционального тестирования;

CSS – язык описания внешнего вида документа.  
XPath – язык запросов к элементов XML модели. Чтобы обратиться к элементу «логотип» на странице, нужно указать путь к этому логотипу. Этот путь называется след XPath. Двигаться по кспасу мы можем как от родителя к ребенку, так и от ребенка к родителю, что позволяет гибко описать путь к элементу.

Пример: .//input[@type='password'] – поиск формы заполнение пароля.  
XML – это язык разметки похожий на HTML, был создан для описания данных, теги не определены, есть возможность самому создавать теги.

DOM – это объектная модель документа, не зависящий от платформы и языка, программный интерфейс, позволяющий программам и скриптам получить содержимое HTML, XHTML и XML документов, а также изменять содержимое и структуру этих документов.  
\*Скрипт – это язык сценариев, краткое описание действий, выполняемых системой.

Использованная литература.  
1. Видеокурс Selenium2.0 - Баранцев   
2. Видеоуроки Автоматизации - Поляруш   
3. [http://selenium2.ru/](https://vk.com/away.php?utf=1&to=http%3A%2F%2Fselenium2.ru%2F) - Баранцев   
4. Java для тестировщиков - Юрий Ткач   
5. Шылдт Java  
6. Канер «Теория тестирования».  
7. Савин «Теория тестирования»

**DB, SQL**

* Что такое нормализация.

Ответ: Процесс преобразвания отношения базы данных к виду, отвечающему нормальным формам, называется нормализацией.

* Какие есть типы связей в базе данных. Приведите пример.

Ответ:   
1. Один к одному – Любому экземпляру сущности А соответсвует только один экземпляр сущности В. И наоборот.   
Пример: У любого конкретного ученика может быть только одна характеристика, и эта характеристика относится только к единственному ученику.   
  
2. Один ко многим – Любому экземпляру сущности А соответсвует ноль, один или несколько экземпляров сущности В, но любому экземпляру сущности В соответствует только один экземпляр сущности А.   
Пример: Ученику ставят много оценок; поставленная оценка принадлежит только одному ученику.  
  
3. Многие ко одному – Любому экземпляру сущности А соответствует только один экземпляр сущности В, но любому экземпляру сущности В соответствует ноль, один или несколько экземпляров сущности А.   
Пример: Преподаватель работает только в одном кабинете, однако рабочий кабинет может быть закреплен за несколькими преподавателями.  
Разница между один ко многим и многие ко одному в том, какая из сущностей является главной.   
  
4. Многие ко многим – Любому экземпляру сущности А, соответствует ноль, один или несколько экземпляров сущности В, и любому экземпляру сущности В, соответствует ноль, один или несколько экземпляров сущности А.  
Пример: Ученик учится у нескольких преподавателей. И каждый преподаватель работает со многими учениками.

* Что такое primary key (первичный ключ)?

Ответ: Первичный ключ(primary key) – один из потенциальных ключей отношения, выбранный в качестве основного ключа.

* Что такое foreign key (внешний ключ)?

Ответ: Внешний ключ(foreign key) – понятие относящееся к ограничениям целостности баз данных. Внешний ключ представляет собой подмножество атрибутов некоторой переменной отношения, значения которых должны совпадать со значениями некоторого потенциального ключа.

* Что такое индексы в базе данных? Для чего их используют? Чем они хороши и чем плохи?

Ответ: Индекс – это объект базы данных, создаваемый с целью повышения производительности поиска данных. Используются для быстрого поике по таблицам и столбцам таблицы.   
Хороши индексы тем, что….

* Какие есть типы JOIN’ов. Кратко опишите каждый из типов.

Ответ: 1. INNER JOIN – оператор внутреннего соединения INNER JOIN соединяет две таблицы. Порядок таблицы для оператора не важен, посколько оператор является симметричным.

Пример: **SELECT** users.id, users.name, departments.name **AS** departments\_name  
**FROM** users u  
**INNER** **JOIN** departments d **ON** users.departments\_id = departmens.id

* Что такое LEFT JOIN, RIGHT JOIN? Чем они отличаются?
* Для чего используется слово HAVING?
* Задача: есть две сущности АВТОРЫ и КНИГИ, связь М-М (многие к многим). Создайте структуру таблиц для этих сущностей и напишите запрос, который выберет всех авторов, которые НЕ являются соавторами ни к одной из книг.
* Задача: Дана таблица с двумя столбцами ID (уникальное поле), Name (строка). Необходимо найти ID строк, в которых Name повторяется (то есть имеются две или более строки с одинаковым Name).
* Что такое транзакция. Опишите свойтва транзакции ACID.
* Опишите уровни изоляции транзакций.
* Отличие SQL от MySQL?

Ответ: SQL – это язык запросов, для управления системами управлением базами данных. А MySQL – это одна из подмножества SQL, на подобии такие как, Oracle MS SQL, PostgreSQL и многих других. И для любой СУБВ необходим язык запросов, и таким языком является SQL.

**Algorithms**

* Реализуйте алгоритм изменения порядка элементов в связанном списке (reversing a linked list).
* Реализуйте алгоритм обхода бинарного дерева (binary tree traversal).
* Напишите метод на вход которого передается List<Integer> list и Integer sum и который возвращает boolean, который показывает есть ли в списке 2 элемента, сумма которых равна sum. Оцените сложность алгоритма. Предложите другой способ реализации.
* Напишите метод, который на вход принимает целое число и определяет, является ли число четным. Предложите еще 2 способа реализации.

Сортировка вставками:  
**public** **static** **void** sort(**int**[] m, **int** a, **int** b) {

**int** t;

**int** i, j;

**for** (i = a; i < b; i++) {

t = m[i];

**for** (j = i - 1; j >= a && m[j] > t; j--)

m[j + 1] = m[j];

m[j + 1] = t;

}

}

**Testing**

* Чем стаб (stub) отличается от мока (mock)?

RegEx

Patterns

**Logical puzzles**

1. Про лампочки

Задача: В комнате есть 3 выключателя от лампочек, которые находятся в коридоре за закрытой дверью. За какое минимальное открытие двери можно определить, какой выключатель к какой лампочке относится?

2. Про пилюли

Задача: Каждый день больной должен принимать таблетки, чтобы не умереть. По одной таблетке из каждой баночки. Иная дозировка – смерть. Баночки идентичные, таблетки – тоже. Оставалось 4 таблетки (по две каждого сорта), когда больной их случайно перемешал. Как можно закончить курс лечения и остаться в живых?

3. Про мертвого мужчину

Задача: Среди пшеничного поля нашли мертвого мужчину, крепко сжимавшего в руках спичку. Отчего он умер?

4. Про люки

Задача: Почему люки круглые?

5. Про блендер

Задача: Вас уменьшили и бросили в блендер, который включится через 30 секунд. Что вы будете делать?

6. Про фальшивую монету

Задача: У вас есть 8 монет: 7 весят одинаково, одна весит меньше. Пользуясь весами, найдите легкую монетку менее чем за три шага.

7. Про торт

Задача: Вам подарили на день рождения торт. Как его разделить на 8 равных частей тремя разрезами?

8. Про веревки

Задача: У вас есть две веревки. Про одну известно, что она сгорает ровно за 1 час. Необходимо с помощью этих веревок определить интервал времени в 15 минут. Нужно учитывать, что веревки горят неоднородно.

9. Про цепочку

Задача: На постоялый двор приехал путешественник. Денег у него с собой не было, но была золотая цепочка из шести звеньев. Хозяин постоялого двора согласился принять в оплату комнаты за каждый день по одному колечку с этой цепочки, но так чтобы распиленных колец он получил не больше одного. Как постояльцу нужно распилить цепочку, чтобы можно было ежедневно расплачиваться на протяжении пяти дней?

10. Про настройщиков рояля

Задача: Рассчитайте, сколько в нашем городе настройщиков рояля?