**1. MESSAGE BROKERS**

- что лучше amqp или http

- Apache Kafka(transaction log, партиции, гарантии доставки, zookeeper)

- гарантия доставки, как реализуются

- transaction log

- ActiveMQ vs RabbitMQ vs Kafka, основные протоколы, принципы работы, как хранят данные (журнал коммитов, на диске, напр. KahaDB), гарантии доставки

- как положить сообщение в очередь ActiveMQ?

**2. GIT**

- Git vs other Version Control Systems (Subversion, Mercurial, CVS) differences

- ответы на основные вопросы на собеседовании,

- Что делает git merge.

- merge vs rebase,

- как откатиться до нужного коммита (git reset --hard <идентификатор\_коммита>)

- как удалить коммит (git rebase -i < идентификатор\_коммита > -> drop)

- Как объединить коммиты в один при мерже (git rebase -i < идентификатор\_коммита > -> squash)

- stash, как ансташить (git stash apply "stash@{n}")

- reset(--hard, soft, mixed, keep) vs revert,

- cherypick

**3. DOCKER**

- Docker vs. виртуализация

- Что такое гипервизор?

- Какие сети доступны по умолчанию в Docker? устройство сети

- Add vs. Copy

- CMD vs. ENTRYPOINT в Dockerfile

- EXPOSE

- снятие дампа

- основные команды и вопросы на собесах

- Что за команда docker ps, какие параметры у нее есть?

- Dockerfile

- docker-compose

- volume, bind mounts,

- writable layer

- снятие дампа

**4. METHODOLOGY**

- Agile, Scrum, Canban

- Что такое Agile? Каковы особенности Scrum? Kanban? Waterfall? расскажите про их достоинства и недостатки.

- Расскажите про артефакты Scrum (sprint, retro, stand-up, planning, demo, burn-down diagram)

- Расскажите про артефакты Kanban (backlog, kanban-board, lean)

- agile, чем отличается agile от waterflall, роли в scrum команде (SCRUM Master, Product Owner, Development Team, Stakeholders)

**5. CI/CD**

- принципы, различие

- Continious Integration (Jenkins/hudson, CruiseControl, GitlabCI etc)

- Continious Delivery (Schef, puppet, etc)

**6.** **TEST**

- Stub/mock

- Виды тестов. unit, integration, TDD, BDD

**7. SQL**

- Sql vs NoSql, виды NoSql

- Какие типы баз данных вы знаете? (реляционные/нереляционные). В чем их основное отличие? для чего лучше подходит SQL, noSQL БД? Был ли опыт работы с NoSQL (какой из них)?

- Какие виды join вы знаете?

- Что такое внешний ключ?

- Есть 2 таблицы, в первой 2 записи, во второй 3, 2 совпадения по FK. Сколько кортежей вернут разные джоины: inner, left, right, full, cross?

- Если выполнить «select \* from a, b», что мы получим?

- «select \* from a, b where a.id=b.a\_id» или «select \* from a inner join b on a.id=b.a\_id» что будет выполнено быстрее?

- Для чего используется group by? Для чего используется having?

- Какие виды подзапросов вы знаете?

- Что такое план запроса? Для чего он нужен?

- Что такое индекс? Для чего их используют? Кластерный, некластерный

- Нормализация бд, какие нормальные формы знает

- Транзакция, Какими свойствами должна обладать транзакция? (ACID), проблемы транзакций, уровни изоляции транзакций

- Планировщик запросов (план запросов, парсить, кеш)

- Представление

- Для чего нужны операторы UNION, UNION ALL, INTERSECT, EXCEPT?

- full join vs. cross join

- Если вы собираетесь соединить несколько таблиц в запросе (например, n таблиц), сколько условий соединения вам нужно использовать?

- Партиционирование, Репликация, Шардирование

- MongoDB, особенности, когда использовать, транзакции, read/write concern, блокировки replica set, WireTiger

**8. ORM/JPA/HIBERNATE**

- В чем отличия JPA от Hibernate? Какие еще реализации JPA вы можете назвать?

- Какие требования JPA | Hibernate к Entity классам вы можете перечислить?

- Какие типы данных допустимы в атрибутах Entity класса (полях или свойствах)?

- Какие типы связей (relationship) между Entity вы знаете (перечислите восемь типов, либо укажите четыре типа связей, каждую из которых можно разделить ещё на два вида)?

- Какие типы данных можно использовать в атрибутах, входящих в первичный ключ Entity класса (составной или простой), чтобы полученный первичный ключ мог использоваться для любой базы данных? А в случае автогенерируемого первичного ключа (generated primary keys)?

- Какие два типа fetch стратегии в JPA вы знаете?

- Какие четыре статуса жизненного цикла Entity объекта (Entity Instance’s Life Cycle) вы можите перечислить?

- Какой аннотацей можно исключить поля и свойства Entity из маппинга (property or field is not persistent)?

- Какие шесть видов блокировок (lock) описаны в спецификации JPA (или какие есть значения у enum LockModeType в JPA)?

- Какие преимущества от использования Hibernate?

- Почему объекты в Hibernate не могут быть final

- В каких статусах могут находится объекты в Hibernate,

- Entitygraph

- FlushModes

- проблема n+1

- fetch size vs. batch size

- Уровни кэширования Hibernate

- как работает кеш 1 уровня

- hibernate session vs entitymanager

- Как можно настроить работу Hibernate с кэшированием запросов, два сервиса, redis про внутреннюю очередь, смотря какие условия

- оптимистическая и пессимистическая блокировки. В чем между ними разница?

- @Cacheable, распределенный и локальный кеш, инвалидация, алгоритмы вытеснения, распр. Лок

- как не меняя респонса добавить новое поле (resultset, кастомный тип данных)

- какая то аннотация forUpdate

- Что делать если запрос медленно работает? Что делать если запрос медленно работает и индексы не помогают? Что делать если запрос медленно работает и индексы не помогают, и данные часто меняются т.е. кэш не поможет

- фантомное чтение, когда возникает, как с ним бороться

- для каких типов полей неэффективно строить индекс

**9. HTTP/SERVLETS/JSP**

- В чем главная особенность протокола HTTP?

- Какой метод сервлета всегда вызывается при обработке запроса?

- Как сервлет контейнер узнает, какой сервлет должен обрабатывать запрос?

- В чем отличия redirect и forward? В каких случаях нужно применять redirect? Почему?

- Предположим мы объявили поле в классе сервлета, в чем будет особенность доступа к этому полю?

- Может ли сервлет/jsp вернуть бинарные данные (что-либо отличное от текста)? Как это реализовать?

- Может ли сервлет обработать PUT/DELETE/... запрос?

- Отличия сервлета и jsp страницы.

- Назовите стандартные переменные, которые можно использовать в сервлете/jsp?

- Что такое http сессия? Для чего она нужна? Каким образом сессия «привязана» к пользователю? Как очистить сессию? Как еще можно сохранить-передавать информацию в пределах нескольких запросов?

**10. SPRING**

- Что такое Spring? Из каких частей состоит Spring Framework?

- Объясните суть паттерна DI или IoC.

- Какие еще IoC контейнеры вы знаете?

- Какие существуют виды DI? Приведите примеры.

- Как добавить поддержку Spring в приложение/web-приложение?

- Spring IOc, Ioc - контейнер, DI

- Spring обеспечивает потокобезопасность?

- виды конфигураций

- В чем роль ApplicationContext в Spring?

- Как подымается приложение (beandefinition, ApplicationContext и все остальное)

- что такое spring bean? Как объявить бин? Какие способы настроить spring bean?

- bean scopes

- Жизненный цикл бина (@PostConstruct, init method,и все остальное)

- чем отличается жизненный цикл prototype и singleton

- @PostConstruct @Predestroy

- singleton не потокобезопасен

- prototype - в каких случаях используется, можно ли у него вызвать destroy

- как лучше внедрять бины и почему

- как внедрить prototype в singleton

- self injection

- как создать бины с одинаковыми типами

- a зависит от в, в зависит от а, как решить циклическую зависимость?

- Autowired vs Inject vs Resource

- транзакции в Spring, про аннотацию @Transactional для чего она нужна

- какие паттерны проектирования, используются в Spring Framework

- open-session-in-view

+ как написать свой starter и зачем, отличия от библиотеки @Conditional

+ spring data rest - недостатки, когда не стоит использовать

- @Async

- где хранится RequestScope (какаято мапа)

- реализации beanFactoryPostProcessor и beanPostProcessor

- Какие есть проблемы у Pageable?

- про webflux

- как работает injection в Spring, максиамально подробно

- приходилось ли работать со скоупами отличными от singleton

- как Spring понимает generic-типы при создании бинов

**11. SECURITY**

- Назовите основные Security Risk, которые возникают при разработке web приложений (особенно актуально для кандидатов с опытов в финансовой сфере)? OWASP Top 10 Vulnerabilities: (XSS, CSRF, Injections [SQL, OS, LDAP, cache], Broken Authentication and Session Management; Security Misconfiguration, Missing Function Level Access Control, Using Components with Known Vulnerabilities). Наименования могут быть другие. что каждый из них обозначает. Сталкивались ли вы в своей работе с методами борьбы с этими Security Risks? Если да, расскажите как вы их решали?

- OAuth2.0 (components, flow, types of grants)

- JWT

- OpenId

- ADFS,

- Единая точка входа SSO

- шифрование симметричное и ассиметричное base64, алгоритмы шифрования sha512…

- saml

**12. PATTERNS**

- DRY, KISS, YAGNI, SOLID

- шаблон проектирования Circuit Breaker vs. с шаблоном Bulkhead

- Паттерны микросервисов: kappa, CQRS, SAGA, CircuitBreaker, Service Discovery, Registry, API Gateway…

**13. OTHER**

- разница между SOAP и REST

- Назовите библиотеки, которые Вы использовали для работы с REST/SOAP?

- Когда по вашему мнению лучше использовать REST, а когда SOAP?

- Предположим вы создали сервис. Как вы предоставите-опишете “потребителям” этого сервиса ваши методы? В случае REST/SOAP

- SoftwareDevelopmentLifeCycle

- EJB основные идеи, виды бинов

- Виды XML-парсеров. Как задать схему для XML? Что такое XSLT?

- Что такое iBatis/MyBatis?

- CAP теорема

- Стек технологий Spring Cloud(ConfigServer, Eureka, Ribbon, Feign, Hystrix, ZuulProxy, sleuth)

- Monolit vs microservices

- что такое Apache Camel

- Rundeck

- Wiremock, serenity-bdd

- AWS, основные сервисы

- Prometheus, Grafana

- Elacticsearch, Logstash, Kibana

- зачем необходима оркестрация? Основы Mezos, Marafone, Kubernetes

- Инернет в Британии (как устроен, монополисты, провайдеры)

- Просмотреть какие-нибудь исходники сервисов ShellEnergy

- основные протоколы: HTTP (S, 2), модель OSI, TCP/IP

- Системы поддержки код-ревью (fisheye, stash etc)

- Системы онлайн-документирования (MediaWiki, TWiki, Redmine-Wiki, Confluence)

- Основные Linux команды (sudo, ifconfig, ssh, scp, top, ps, kill, grep, cat, less, head, tail, bash, iptables, tcpdump, ifconfig, gedit)

- как добавить прав на чтение группе пользователей linux (shmode)

- как узнать чем занят порт (netstat)

- ansible

**14. JAVA** (лучше разбить по темам, потому что повторять неудобно)

- Нововведения в Java 8, 11

- позднее связывание

- exceptions в сигнатуре метода правила переопределения

- Что такое интерфейс в java?

- Что допускается объявить в интерфейсе?

- Какие модификаторы доступа по умолчанию имеют методы интерфейса? А поля? Почему?

- Для чего еще можно использовать интерфейсы?

- Допускается ли наследование нескольких интерфейсов с методами одинаковой сигнатуры? (+ Что входит в сигнатуру метода?)

- Могут ли интерфейсы наследовать другие интерфейсы? Только один или сколько угодно?

- Сколько классов/интерфейсов допускается наследовать/реализовывать в java?

- Предположим есть 2 интерфейса с одинаковыми методами (имя, параметры), но возвращающими разное значение, можно ли реализовать оба этих интерфейса? Почему?

- Можно ли изменить модификаторы доступа при наследовании? Каким образом? А почему именно так? Что еще можно изменить в сигнатуре метода при наследовании?

- Перегрузка и переопределение. В чем разница?

- Сколько аргументов может быть у метода? Как объявить метод с переменным количеством аргументов?

- Как обратиться к объекту родителя, например, вызвать его метод? А если нужно вызвать метод родителя родителя?

- Что такое конструктор по умолчанию? Для чего он может понадобиться? А если его не будет, можно ли создать объект через конструктор без параметров?

- Как вызвать конструктор родителя при создании объекта?

- Какими способами можно инициализировать поля класса? А если не задать начальное значение?

- У каких переменных нету значения по умолчанию? Почему обязательно нужно инициализировать локальную переменную?

- Можно ли объявить final static метод в классе? Почему?

- Какие модификаторы доступа может иметь класс?

- Сколько классов можно объявить в одном файле? Какие условия при этом должны быть выполнены?

- Если необходимо использовать 2+ класса с одинаковыми именами (например java.sql.Date/java.util.Date) в одном файле, как это реализовать?

- Что такое абстрактный класс? Может ли абстрактный класс не иметь абстрактных методов?

- Как запретить создание объектов класса?

- В базовом классе есть public static метод. Можно ли в классе наследнике объявить статический метод с такой же сигнатурой? Почему? Какой из методов будет вызван, если мы работаем с объектом наследником через ссылку на родительский класс?

- Можем ли мы использовать this/нестатическое поле класса внутри статического метода?

- Пусть нам нужно инкапсулировать логику 2х классов в одном. Каким образом можно сэмулировать множественное наследование в таком случае?

- Что такое java Enum? Какие операции вы можете совершать с Enum?

- Можно ли объявить класс внутри метода?

- Что такое анонимные классы? Для чего их удобно использовать?

- Как обратиться к методу внутреннего класса через ссылку на внешний? А наоборот?

- Класс String. В чем его особенности? Почему нельзя унаследоваться от класса String? Почему его сделали immutable и что это дает?

- Как сконструировать JavaBean, содержащий произвольные поля, с неизменяемым состоянием (immutable)? - Какие преимущества будет иметь такой объект?

- Какие основные группы типов данных в Java вы знаете? - > В чем их отличия?

- Класс Object. Какие его методы Вы знаете/использовали?

- Объекты типа Class. Что за они? Для чего нужны? Что с их помощью можно сделать?

- Как правильно сравнивать объекты в Java? Почему для String в некоторых случаях equals и == вернут true? В каких случаях equals и == дадут верный результат сравнения объектов?

- Интерфейсы Comparable/Comparator, правила переопределения?

- Как определить, что объект определенного типа/наследует интерфейс-класс?

- Что будет, если разделить на ноль : в случае делитель int, double? Как определяется результирующий тип данных?

- В чем отличие i++, ++i?

- Классы обертки. Autoboxing/Unboxing.

- Для чего можно использовать модификатор final?

- Как запретить наследование от класса? Перегрузку метода?

- Что такое JavaBean? Каким требованиям должен удовлетворять такой класс?

- Если нужно ускорить работу со строками, какие классы вы будете использовать?

- Какие основные коллекции есть в Java?

- Каковы основные отличия List от Set?

- Каким образом определяется «одинаковость» объектов при помещении в Set?

- Как правильно переопределить equals & hashCode?

- С какими имплементациями List/Set/Map вы работали? Расскажите про...

- Как можно синхронизировать доступ например к ArrayList?

- Каким требованиям должен удовлетворять ключ Map?

- Можно ли использовать array (int[]) в качестве ключа? Почему? А если понадобится это сделать, как решить проблему?

- Что такое коллизия? Что произойдет при вставке в Set/Map в случае коллизии?

- Можно ли вставить null в List/Set/Map (ключ)? Что произойдет при вставке null значения в List/Set/ Map (ключ)?

- Для чего нужен Iterator? Какие его методы вы использовали?

- Как корректно удалить некоторые значения из коллекции в цикле?

- Iterator и ListIterator чем отличаются?

- Какие дополнительные операции позволяет выполнить ListIterator?

- ArrayList и LinkedList? Отличия? Какую реализацию предпочтете использовать при частых модификациях коллекции? Почему?

- Какой алгоритм сортировки используется для массивов примитивов? Какой для массивов объектов? Для листа объектов?

- Какова иерархия исключений в Java? Какие виды исключений вы знаете?

- Каким образом могут быть обработаны исключения в Java?

- Пусть метод может выбросить несколько разных исключений. Имеет ли значение, в каком порядке их обрабатывать, т.е. порядок блоков catch?

- Можно ли использовать конструкцию только try-finally? Для чего она может понадобиться?

- Всегда ли будет выполнен блок finally? А если в конце блока try у нас return?

- Всегда ли мы можем «поймать»/обработать исключение?

- Как создать поток в java? Почему 2мя способами?

- Пусть у нас объявлена volatile **int** переменная с начальным значением = 0. Если одновременно запустить 3 потока, каждый из которых делает в цикле инкремент этой переменной 3 раза. Какое значение будет иметь эта переменная, после выполнения всех потоков? А если убрать volatile?

- Как «защитить» объект, с которым работают несколько потоков?

- Какие способы синхронизации вы знаете?

- Пусть в объекте 2 синхронизированных метода (не статич) и один из них вызван в отдельном потоке. Что будет при вызове второго метода? А если он будет несинхронизирован?

- Что является объектом блокировки, при вызове синхронизированного статического метода?

- Как корректно завершить выполнение потока?

- Atomic?

- Что такое сериализация?

- Каким образом можно сделать объект сериализуемым? Что будет, если сериализовать объект, который не унаследовал Serializable (Externalizable) интерфейс?

- Как исключить поле объекта из сериализации?

- Можно ли переопределить механизм сериализации на свой?

- Пусть родительский класс наследует Serializable. Можем ли мы сериализовать класс наследник? Почему?

- Предположим мы передаем сериализованнй класс на другую ВМ. Какие условия должны быть выполнены для успешной десериализации?

- Любой ли java класс может быть сериализован-десериализован?

- Пусть класс наследует Serializable и содержит внутренние/анонимные классы, может ли такой класс быть сериализован-десериализован? Поясните?

- Statement и PreparedStatement. Есть одинаковый запрос, с использованием чего он будет выполняться быстрее? Почему? Какие еще преимущества у PreparedStatement’а?

- Пусть из БД в ResultSet вычитано int значение. Как определить, что это значение содержит NULL?

- Каким образом в JDBC реализована поддержка транзакций?

- Какие уровни изоляции транзакций вы можете назвать? Кратко поясните.

- java classloader

- области памяти

- Назвать разные реализации gc, механизм работы, tuning

- HashMap как потерять ключ

- list add put

- streamAPI flatMap

- Statement vs PreparedStatement (когда что использовать)

- volatile

- invoke run instead of start in thread

- квант времени

- yield

- Какие способы синхронизации потоков в java вам известны?

- ExecutorService completablefuture

- CompletableFuture get() vs. join()

- повторить java.util.concurrent(ForkJoinPool, ExecutorService…), паттерны и HashMap, garbage collector-ы

- реализация кэша

- Big O, сложность операций в Коллекциях, внутр. устройство HashMap, HashSet, ArrayList, LinkedList, Queue

- Sorting (алгоритмы сортировки)

- Binary tree

- Generics (ко\контр\ин-вариантность, PECS, type erasure, wildcard, пример generic типа)

- Многопоточность (volatile, процесс vs поток, способы создания потоков, deadlock, racecondition, conc. collections)

- heap, stack, -Xms –Xmx, happens before

- reflection dynamic proxies, dynamic class loading

- profilers

- Design patterns

- JDBC API

- что такое лямбда и чем она отличается от анонимного класса?

- стримы

- Можно ли переопределить статический метод? А перегрузить?

- Отличия equals() и ==.

- Удалить первый элемент в ArrayList и LinkedList, что быстрее и почему?

- Отличия HashSet и TreeSet.

- Что такое Deadlock, когда возникает, как избежать?

- Нужно выгрузить большую по объему таблицу в файл, как это выполнить при сильном ограничении памяти?

- зачем появились функциональные интерфейсы

- разница между orElse и orElseGet в Optional

- если в лямбде checked-exception бросается мы не имеем возможности пробросить его из метода в котором лямбда описана, почему так происходит

- какие ограничения накладываются на переменные, которые мы используем внутри лямбды (final, effective final)

- есть List<Passengers>, нужно подсчитать общий вес мужчин, как это сделать через стрим (reduce)

- tuning GC

- как определить сколько памяти нужно микросервису

- OutOfMemoryError - что делать, как разбирать причину

- Какие есть классы DateTime Api? Offset Zoned – разница

- класс Instance

## 15. Вопросы по опыту кандидата

- Расскажите о Вашем последнем проекте: краткое описание проекта, стек технологий, какие задачи на проекте выполняли, структура команды, коммуникация внутри команды и с заказчиком, как было организовано тестирование, … + уточняющие вопросы при необходимости.