**OOP**

1. Hash code() и equals() (связь);
2. Abstract class vs Interface;
3. Final, finally, finalize;
4. Классы-оболочки;
5. Collections;
6. int a = 1;

Integer b = 2;

**int a+b - ?**

**а если Integer b = null ?**

1. Thread-safe and non-thread safe collections (examples);
2. Static и все что с ним связано. Как использовать где применять. Особенности доступа к статическим переменным/методам;
3. Метод для преобразования потоконебезопасной коллекции в потокобезопасную;
4. try {

a = 5;

throw new Exception();

} catch (Exception e) {

a = 10;

Throw new Exception();

} finally {

a = 15;

throw new Exception();

}

**Какая ошибка вылетит и какой будет а**

1. try {

a = 5;

return a;

throw new Exception();

} catch (Exception e) {

a = 10;

return a;

Throw new Exception();

} finally {

a = 15;

return a;

throw new Exception();

}

**Какая ошибка вылетит и какой будет а**

1. Иерархия ошибок;
2. Наследование;
3. Classloader (иерархия);
4. Написать код (метод) в котором проверяется HashMap, если в нём есть какое-то значение, то оно должно возвращаться, если нет, надо засетать пустую строку и её вернуть. Написать код, чтобы он был как можно более эффективным (меньше затратных действий).
5. Какие потокобезопасные коллекции более «быстрые» – обычные (типа Vector или HashTable) или из пакета concurrent?
6. Как устроен ArrayList и LinkedList?
7. В каких областях памяти хранятся значения и объекты, массив?
8. Если Interf – это интерфейс, а Klass – это класс, который имплементит интерфейс и у него есть метод (которого нет в Interf) например method1(). Можно ли написать так:

Interf a = new Klass();

a.method1();

То же самое, если Interf – не интерфейс, а абстрактный класс.

1. Бывают ли интерфейсы без методов. Для чего?
2. Есть List. Удалить из него элемент;
3. Garbage Collector – если ли ещё сборщики мусора кроме него и как он определяет, будет ли использоваться ссылка на объект или нет?
4. Есть массив из 10-ти элементов. Создать 10 потоков, которые принимают по числу из массива, обрабатывают и возвращают обратно. Собрать все обработанные числа обратно в массив.
5. Если в коллекцию часто добавлять элементы, и удалять, какую лучше использовать? Почему (как они устроены)?
6. Как отсортировать коллекцию?
7. А если я хочу одну коллекцию отсортировать одним способом, а вторую другим способом?
8. А можно ли с помощью equals?
9. Методы класса Object;
10. XML – парсеры, что лучше использовать в каких случаях, well-formed, valid – в чем отличие dtd и xsd, какая схема написана в формате xml? (xsd)
11. Зачем нужен метод notifyAll? А если потоков нет, зачем каждому объекту этот метод?
12. Синхронизация. Synchronized. Volatile
13. По каким объектам синхранизирются статические и нестатические методы.
14. А если я хочу сохранить объект не в байт-код, а в xml-файл? Можно?
15. Final vs finally vs finalize
16. Сериализация. Transient
17. Поток. Создание и запуск потока. Состояния потока.
18. Многопоточность. Для чего. Как реализовывать. Как обезопасить используемые ресурсы.
19. Wait vs notify vs notifyAll.
20. Отличие HashMap от TreeMap
21. Как происходит вставка элемента в HashMap.
22. Назовите основные отличия Java От С++
23. Типы ссылок в Java
24. Какие типы данных вы знаете? назвать размеры всех примитивов.
25. Фигурирует ли где-нибудь в java понятия signed/unsigned ?
26. Расскажите, пожалуйста, что такое Comparator & Comparable , для чего применяются в стандартных коллекциях?
27. Что такое полиморфизм? Приведите примеры.
28. Можно ли написать try и finally без catch, когда это может понадобиться.
29. Всегда ли выполнится finally? В каких ситуациях finally может не выполниться?
30. Что такое enum и как с ним работать. Для чего он создан в java?
31. Как сделать singleton?
32. Для чего существуют классы обёртки(Напр. Integer для int) ?

**Strings**

1. В чём состоит особенность строк в java?
2. Как сделать строки изменяемыми?
3. Что такое пул литералов и в чём его смысл, для чего он существует?
4. Как перевести строку в пул литералов?

Database

1. Зачем нужно View и какие поля там будут;
2. Joins;
3. Indexes;
4. Последовательность действий при запросе в бд (типа загрузка драйвера, открытие коннекшена, и т.д.)
5. Statement и остальные, что из них быстрее, в чем ещё отличие statement и prepareStatement
6. Есть таблицы Customer и Order. Вывести всех кастомеров, у которых суммарный заказ > 10000.
7. Что такое агрегирующая функция, примеры;
8. Результат запроса *Select \* from Table1, Table2*;
9. Есть 2 таблицы

|  |  |  |
| --- | --- | --- |
| IDDepartment | Department |  |
|  | | |
| IDEmployer | Salary | IDDepartment |

- Написать запросы на выборку Department, salary если сумма salary>100;

- Написать запрос на ту же выборку через join;

1. Запрос к БД через jdbc
2. Типы драйверов для соединения с БД.
3. Индексы в sql. Вторичный первичный ключи.
4. Различия между JPA, Hibernate, JDBC . Преимущества недостатки.
5. Для чего применяется ключевое слово having?
6. Есть таблицы Student и Mark. Вывести фамилию студента и его среднюю оценку
7. Как ускорить чтение из таблицы, если часто читаем?
8. Какая структура у индексов и где они хранятся?
9. Придумать и написать sql injection (представь, что ты злодей и хочешь сломать систему). Как бороться с sql injection?

**Web**

1. Что такое HTTP;
2. Cохраняет ли http протокол своё состояние.
3. Session and cookies; Как можно достать, куки, а если куки удались, можно ли достать сессию
4. Какие бывают scopes;
5. Как сервер понимает, что для пользователя создана сессия и не нужно её создавать?
6. Отличие авторизации от аутентификации;
7. Сервер приложений. Веб-сервер. Отличия
8. По самым подробным этапом рассказать, что делается, когда пользователь кликает на кнопку. (Типо с формы поля сетаются в request, потом вызывается контейнер сервлетов, потом он как-то по request понимает, куда нужно идти дальше (в дескриптор развертывания, а их может быть несколько, нужно как-то понимать в какой), после этого (если использовать struts нужно заходить в конфигурационный файл, там находится action, ну и так далее);
9. Контейнер сервлета. Сервлет. Методы. Жизненный цикл;
10. Какие есть методы у сервлета кроме GET и POST, когда лучше использовать GET.
11. POST vs GET методы. Отличия;
12. forward vs redirect. Отличия;
13. jsp:include vs include
14. Где в запросе хранятся передаваемые через POST параметры?
15. Javascript: В чем отличие оператора “==” от “===”?
16. jQUery: приведите пример цикла forEach
17. Кто контролирует длину url и какие ситуации могут произойти, если url слишком длинный?
18. Расскажите подробно, как произойдёт redirect в http ?
19. Расскажите про Async Servlet.

Что входит в jee?

Technology

*Struts*

1. Рассказать про Struts;

*Spring*

1. Рассказать про Spring;
2. Spring inversion of control.  Что такое. Как используется и для чего.
3. Spring dependency injection.  Что такое. Как используется и для чего.
4. Какие скоупы есть у бинов в Spring. Какой дефолтный?

*Hibernate*

1. Уровни кеша;
2. Способы запросов;
3. Как из select \* from a,b достать только объекты b;
4. Entities состояния;
5. Сессия в хибернейте;
6. sessionFactory.getCurrentSession() vs entityManagerFactory.createEntityManager()
7. У какого объекта (если использовать Hibernate) вызываются методы для работы с БД;
8. Если в хибернейте сделать:

Object o = transaction.get(id, table);

o.setName(name);

o.commit();

Сохранится ли name?

*JMS*

1. Что такое JMS

*Ant $ Maven*

1. Различия Ant & Maven;
2. Команды в консоли для JVM, Maven, Ant;
3. Какой командой билдится проект?
4. Рассказать Maven lifecycle.
5. Для чего предназначена секция dependencyManagement ?

*Testing*

1.Какой версией JUnit пользовалась?

1. Для чего писать тесты?
2. Что бы вы протестировали в калькуляторе?

*Git*

1. Что именно хранит git, каким образом он отслеживает изменения?
2. Ты сделал изменения какие-то на своей локальной машине и как их залить в удалённый репозиторий.
3. Обратная операция, как выкачать новую версия проекта из удалённого репозитория к себе на локальную машину.

*Web services*