1) Какие виды permission бывают ?

Ответ : declare permission / runtime access permission (dengerous / not dengerous) runtime с **6 андроида**

2) Какие важные атрибуты указываються в теге application ?

Ответ : **theme, label, roundIcon, name (для кастом класса наследника application (singelton)), targetApi — либо его же в gradle**

3) Что значит android:exported="true" / "false" ?

Ответ: Так мы можем открыть либо зарыть доступ других приложений к нашему

3) Сервис работает в main потоке как сделать так что бы он работал в другом ?

Ответ: Мы можем при создании сервиса наследываться от интент сервиса(Deprecated) Job Intent Servise и тогда он будет работать в другом потоки либо мы можем создавать поток в ручную

4) Как остановить сервис ?

Ответ: Мы можем остановить сервис из самого сервиса вызвав функуию StopSelf(), либо из Activity вызвать метод stopService(), передавая ему объект Intent, определяющий нужный сервис.

5) Как получить все приложения установленные на телефоне ?

Ответ: при помощи PackageManager

6) Как расшифровуеться APK ?

Ответ: Android Package Kit

7) что такое DEX?

Ответ: Одна из самых замечательных особенностей `Dalvik Virtual Machine` (виртуальная машина в Android) заключается в том, что она не использует байт-код Java. Вместо этого был введен собственный формат, называемый DEX, и даже инструкции байт-кода не совпадают с инструкциями байт-кода Java. Dex лучше сжимают информацию о коде чем стандартный jar

8) Зачем нужна виртуальная машина

Ответ : она нужна для того что бы добавить ещё один уровень абстракции для выполнения кода кроссплатформенно

9) Как происходит создание APK из kotlin кода ?

Ответ: Gradle превращает kotlin/java → код собираеться в DEX →все xml ресурсы генерируют R класс → формируеться APK. Ресурсы отдельно, манифест отдельно, вся логика в DEX файлы. У DEX файла не может быть болше чем 65 000 методов. К примеру если мы подключим RXJava там много классов но из них юзаеться не много. Для борьбы с эти используем **minifyEnabled true** и это даёт возможност что бы в АПК файлы ложились только те файлы которые используються

10) AAB (Android boundle) в чём разница ?

Ответ : Он адаптирует размер в котором будут только те компоненты которые нужны по рамеру екрана, языка и т. д.

11) Как запросить что бы система выдела больше ОЗУ на приложение ?

Ответ : Нужно в манифесте у тега апликейшен написать атрибут largeHeap = true

12) В чём разница между serializable parsializable ?

Ответ : serializable работает на рефлексии определяя как обьект превратить в строку. Для того что бы применить serializable можно просто имплиментировать интерфейс serializable, parsializable это интерфейс в котором есть 2 метода на запись и читение и там нужно определить как обьект будет записываться или читаться, за счёт этого parsializable работает быстрее чем serializable

13) Что происходит в OnCreateView/OnDestroyView ЖЦ фрагмента ?

Ответ : В этих колбеках парситься xml фрагмента а в OnDestroyView освобождаеться память

14) Как добавить синхронно/асинхронно фрагмент ?

Ответ : По дефолту после FragmentManager.add() → FragmentManager.commit() фрагмент добавляеться синхронно, что бы сделать это асинхронно применяем FragmentManager.commitNow()

15) Что происходит когда вызываеться новое Activity освобождаеться ли память от старого ?

Ответ : Активити не удаляеться из памяти а добавляеться в стек и что бы удалить Активити из стека нужно вызвать функцию Activity.`finish()` который автоматичиски вызываеться из onDestroy

16) Как бороться с тем что из-за deeplink активити может открыться много раз ?

Ответ : Для того что бы такого не происходило можно использывать атрибут у Активити в манифесте launch mode = «**singleInstance**»

17) Что такое Task и где это используеться ?

Ответ : Task – группа из нескольких Activity, с помощью которых пользователь выполняет определенную операцию. Обычно стартовая позиция для создания. Activity они находяться в некотором Task внутри которого можно открыть активность из другого приложения и после вернуться к предидущей активности так как он находяться в одном стеке одного Taska

17) Какое событие вызываеться при нажати юзера по экрану ?

Ответ : ActionDown — когда нажимаем на экран , ActionUp — когда убираем палец из экрана

18) Dp это ?

Ответ : это независимый от плотности пиксилей размер пикселя. Он равен одному физическому пикселю на экране с разрешением 160 dpi (dot per inch)

19) Как вызвать перересовку кастом view ?

Ответ : view.invalidate()

20) Какой класс нужно использовать что бы отрисовывать в background thread view?

Ответ : SurfaceVIew

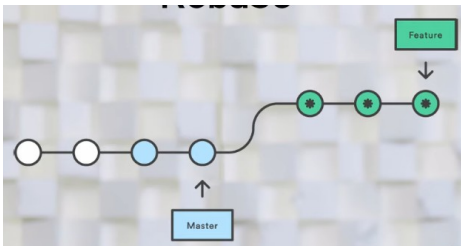
21) Зачем нужен DiffUtil ?

Ответ : Дженерик класс в котором нужно переопределить две дженерик функции : areItemsTheSame и areContentsTheSame что бы понимать как необходимо сравнивать обьекты. Процесс нахождения самого лучшего по времни алгаритма происходит змейками по алгоритму Майерса. Это ускоряет процесс обновление обьектов в RecyclerView. Он помогет не писать нам каждый раз adapter.notifyitemchanged(position)

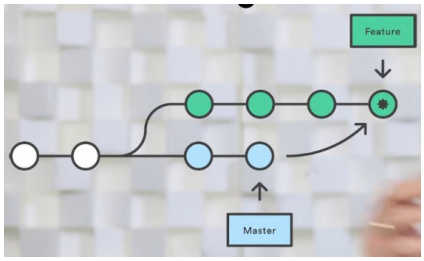
22) Что такое git rebase и чем он отличаеться от merge ?

Ответ : rebase как и merge может перенести коммиты из одной ветки в другую (сливая их). merge : после применения будет добавленный новый коммит (мерж коммит). rebase он в отличие от мержа не добовляет новй комит в ветку в торую мы хоти слить другую а пробрассывает во всю нашу истрию ветки текущее состоянии ветки с которой мы хоти создать слитие

rebase



commit



23) В чём отличие андроид 4 и 5 ?

Ответ : В андроид 5 появися матириал

24) В чём отличие андроид 5 и 6 ?

Ответ : В андроид 6 появились рантайм пермишены

25) В чём отличие андроид 6 и 7 ?

Ответ : В 7 андроид появилась функия doze , и появился split screen

25) В чём отличие андроид 7 и 8 ?

Ответ : В 8 андроид появилась Picture-in-Picture mode(в малепньком екране отображение),добавили каналы для уведомлений

26) В чём отличие андроид 8 и 9 ?

Ответ : Теперь андроид уничтножает сервисы которые работают в бекграунде но не являються foreground

27) В чём отличие андроид 9 и 10 ?

Ответ : Добавили потдержку бекапов

28) В чём отличие андроид 10 и 11 ?

Ответ : добавили One-time permissions

29) В чём отличие андроид 11 и 12 ?

Ответ : добавили Game Mode, так же добавили approximate location для того что бы пользователь мож не делиться конкретным местоположением

30) В чём отличие Dalvik Virtual Machine и ART(Android Run Time) ?

Ответ : Виртуальная машина Dalvik или DVM — это виртуальная машина на основе регистров, оно хранит код в особом виде а потом прибразовывает его байт код при запуске приложения

Android Run Time - это приложения, которые полностью скомпилированы, когда они установлены на устройстве. Следовательно, более высокая производительность, поскольку нет необходимости преобразовывать код в байт-код, а затем компилировать. Но недостатком является то, что вам нужно больше места для хранения и немного больше времени для установки

31) В чём отличие dagger 1 / 2 ?

Ответ : В первом даггере нету код генерации и всё работает на рефликсии — рантайм хуже компиляция быстрее, и наоборот в даггер 2 всё работает на код генерации, но в тестовом приложении можно включить что бы он работал на рефлексии и быстрее компилировался

32) В чём отличие АПК с подписью и без подписи ?

Ответ : с подписью идёт на гугл плей маркет, а без подписи будет считаться подозрительным

33) Что такое JIT ?

Ответ : JIT-компиляция - динамическая компиляция — технология увеличения производительности программных систем, использующих байт-код, путём компиляции байт-кода в машинный код или в другой формат непосредственно во время работы программы. Таким образом достигается высокая скорость выполнения по сравнению с интерпретируемым байт-кодом

34) Разница межу pull/fetch ?

Ответ : Команда `pull` автоматически сливает коммиты, не давая вам сначала просмотреть их. Если вы не пристально следите за ветками, выполнение этой команды может привести к частым конфликтам.

При использовании `fetch`, git собирает все коммиты из целевой ветки, которых нет в текущей ветке, и сохраняет их в локальном репозитории. Однако он **не сливает их** после их можно слить при помощи merge

35) Как расшифровуеться xml ?

Ответ : Extensible Markup Language - Расширяемый Язык Разметки

36) Что значит модификатор native у метода класса ?

Ответ : Это означает что реализация метода написаниа на чистом c++

37) Перечислите методы класса Object/Any ?

Ответ : equals(), hashCode(), clone(), wait(), notify(),notifyAll(), toString(), finalize(), getClass()

38) Перечислите маркеры интерфейсы ?

Ответ : Seriazible, Clonnable

39) Как сравниваються обьекты в java ?

Ответ : Если для данного ссылочного обьекта не переопределены equals / hashCode тогда сравниваються ссылки. Обычные ссылочные обьекты сравниваються только через метод equals который сравнивает все поля обьекта. Если говорить про **хеш мап** то там помимо метода equals используют метод hashCode для ускорения сравнивания. Если оба этих метода переопределены то сначала они сравниваються через более быстрый метод hashCode и если хеш коды равны то вызываеться более долгий метод equals который уже сравнивает медленно каждый обьект тем самым мы увеличиваем скорость сравнивания в разы так как сразу отбрасываем обьекты у которых разные хеш кода

40) Что делает метод Object.finalize() ?

Ответ : Вызываеться перед смертью обьетка из-за удаления обьектоиз памяти гарбедж коллектором (типа деструктор) но он не обязательно будет вызван у обьекта и луче его не использовать и и спользовать только с фантом референсом

41) Что такое try(BufferReader){ … }catch{ … } ?

Ответ : Мы можем вызвать блок трай кетч с ресурсами и в конце того как блок отработает ресурсы которые имплементят интерфейс closiable будут закрыты

42) К чему можно применить модификатор final ?

Ответ : Мы можем применить можификатор final к классу полям класса и методам класса

43) зачем нужна аннотация @CallSupper ?

Ответ : Требует вызова метода суперкласса в рнаследниках

44) К чему можно применить модификатор static ?

Ответ : Можификатор можно применить к методу, полю, блоку кода(инит блоку), внутреннему классу

45) Перегрузка и переопределение метода в чём разница ?

Ответ : Перегрузка метода это когда мы пишем в ожном классе один метод с разными сигнатурами, а переопределение это когда мы в классе наследнике переопределяем повидение метода не изменяя сигнатуру

46) Какой модификатор применяеться к методу если он не прописан явно ?

Ответ : package visible — будут доступны в своём пакете

47) Что будет выводиться HasMap.get() если equal/HashCode не переопределены ?

Ответ : null

48) Можна ли переопределить static метод в классе наследнике ?

Ответ : нет

49) Можна ли переопределить конструкторы в java?

Ответ : да, мы можем иметь много разных сигнеатур конструктора

50) Опишите как лучше всего использывать модификаторы доступа в Java?

Ответ : private — статичиские поля, либо обычные поля (не методы), protected — методы у австрактного класса, public — методы (полей не должно быть)

51) В каком порядке вызываеться конструктор, конструктор супер класса, статичискитй инит блок ?

Ответ : static init block → costructor → super costructor

52) Как называеться метод в котором будет описана работа сервиса ?

Ответ : onStartCommand(intent: Intent, flags: Int, startId: Int): Int — возвращает константу : к примеру **START_REDELIVER_INTENT —** если мы укажем эту константу Если система преждевременно завершила работу сервиса, он запустится повторно, но только когда будет сделан явный запрос на запуск или если процесс завершился до вызова метода **stopSelf()**

53) Foreground сервис чем отличаеться от обычного ?

Ответ : Такой сервис обязан от ображать уведомление о том что он работает. Для того что бы сделать сервис таким необходимо в onStartCommand создать notification и передать его в функцию : startForeground(1, notification)

54) Как называеться action для интент фильтра после перезагрузки телефона ?

Ответ : BOOT_COMPLETED / QUICKBOOT_POWERON

55) В чём разница между foregraund / backgraund сервисами ?

Ответ : бекграуд сервис оградничен с 8 андроида и нужно использовать либо foregraund servise либо work manager

56) Какие виды reciaver вы знвеете ?

Ответ : локальный и системный

57) Какие виды библеотек вы знаеете и вчём их различия?

Ответ : AAR — содержит помомо обычного кода андроид классы (контент провайдеры и сервисы)JAR — содержит исключительно чистый java код

58) От какого класса наследуються Application?

Ответ:Context

59) Зачем нужен контекст, назовите его основные функции ?

Ответ : Он предостовляет доступ к окружению нашего приложения. GetResourses, getPackageManager, getColor, getDarwable, getFilesDir

getAssets — даёт доступ ко всем файлам с неординарным расширением которые находяься в папке Assets

getResousrses — даёт доступ ко всем Ресурсам

PackageManager — Даёт доступ у всем приложением установленных на телефоне

MainLooper — класс который контролирует выполнение задач в главном потоке

60) В чём отличие контекста Application и Activity?

Ответ : Application одно а активити может быть много и нельзя хранить контекст активити в других классах у которых ЖЦ отличный от Активити — утечкак памяти. А вот контекст Application множно хранити и не будет утечек

61) Библиотека для нахождения утечек памяти ?

Ответ:LeackCanary

62) Что такое ProGuard?

Ответ : утилита, которая удаляет из готового кода неиспользуемые фрагменты и изменяет имена переменных и методов для усложнения реверс-инжиниринга приложения. Также позволяет уменьшить размер загружаемых на устройство файлов.

63) Что такое рефлексия ?

Ответ:это инструмент с помощью которого можно узнать всю информацию о классе: имена методов, параметры, поля, аннотации и т.д.
Эти возможности во всю используются во различных сериализаторах json/xml

64) Опишите инкапсуляцию, наследывание,полиморфизм, абстракцию ?

Ответ:инкапсуляция — концепция объектно-ориентированного программирования, согласно которой в одном обьекте содержиться все методы и поля которые описываю этот обьект, то есть они как бы инкапсулированы внутри класса и скрыты от внешего наблюдателя

Наследование - концепция объектно-ориентированного программирования, согласно которой абстрактный тип данных может наследовать данные и функциональность некоторого существующего типа, способствуя повторному использованию компонентов программного обеспечения.

Полиморфизм — это концепция ООП которая имеет два вида :

Параметрический полиморфизм — в котором подразумиваеться что мы для одгог класса можем писать функции с одинаковыми менами но разными сигнатурами и возможно содержанием

Полиморфизм подтипов — в котором подразумиваеться возможность классам наследникам переопределять поведение функций

Абстрациия — это концепция ООП по которой при проектировании класса будут выделены только те поведения и те поля которые не избыточно, а лишь допустимым и необходимым образом описывают модулируемый обьект

65) В чём различия наследования, композиции, агрегации ?

Ответ : Наследование — это когда класс-наследник имеет все поля и методы родительского класса, и, как правило, добавляет какой-то новый функционал.

Композиция – это когда класс являеться полем другого класса и не существует отдельно от этого класса. Он создается при создании класса в котором он являеться полем и полностью управляется этм классом.

Агрегация – это когда экземпляр класса создается где-то в другом месте кода, и передается в конструктор класса который его использует в качестве параметра. В даггере реализуеться агрегация.

66) Почему запрещенно множественное наследывание и как этот запрет обойти ?

Ответ:Оно запрещено потому что из-за него возникает проблемма ромба, когда мы сначала наследуем 2 класса от одного, а потом наследуем от этих двух классов один и тогда получеться что мы наследуем два раза однои то же поле или метод родительского класса, что бы обойти это используем интерфейсы либо Комозицию/ Агрегацию/ наследование

67) В чём различие между интерфейсом и абстрактным классом ?

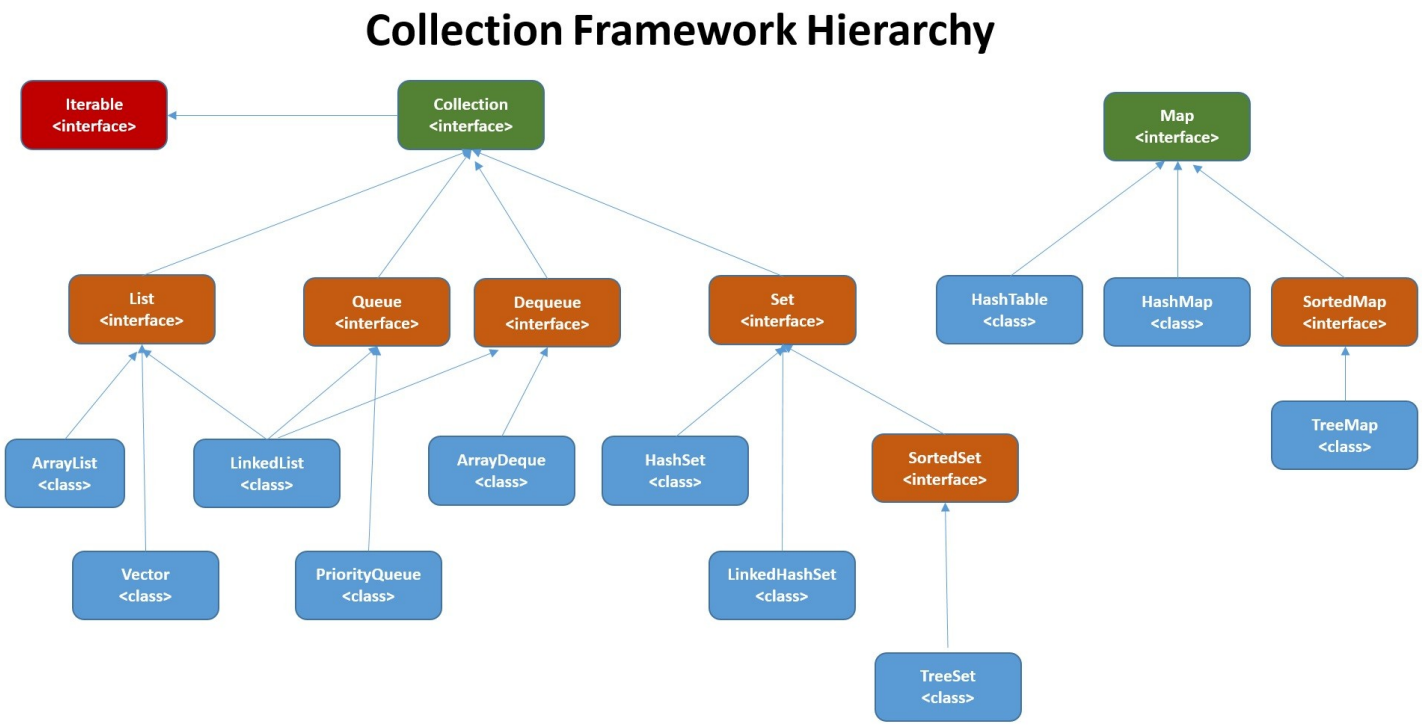
Ответ:В Java 7 интерфейсы это просто методы у котороых нету реализации,а в Java 8 мы можем писать дефолтную реализацию для методов, а в абстрактом классе кроме самих методов есть ещё и поля, есть конструкторы и методы тоже не обязательно что бы были абстрактные а могут быть с реализацией, ну и главное отличе что наследывать абстрактный класс можно только один, а интерфейсов сколько угодно.

68) Как програмно запретить создание обьекта класса ?

Ответ:сделать конструктор приватным либо класс сделать абстрактным

69) Расскажите иерархию колекций в java ?

Ответ : По интерфейсу Iterable можно пройтись итератором а по Map нельзя



69) Разница HashSet, LinkedHashSet, TreeSet ?

Ответ :

HashSet — реализован на основании хеш-таблице, елементы храняться не по порядку, сложность добавления, удаления, и поска константное

TreeSet — реализован на основании красно-чёрного дерева, елементы упорядочены но скорость поиска удаления, добавления O(nlog(n))

LinkedHashSet — реализован на основании хеш таблицы с помощью связного списка за счёт черего достигаеться упорядоченость елементов и высокая скорость выполнения основных методов

69) Разница PriorityQueue, Deque ?

Ответ :

PrioirtyQueue — это обычная очередь которая работает по принципу FIFO (первый вошёл первый вышел) только помимо этого все елементы сорируються в естевственном порядке (если не передан особыц компоратор *Comparator()* в конрструктор)

Deque — это подинтерфейс of *Queue*, предоставляющий методы для вставки элемента в начало или конец, а также методы для доступа или удаления его первого или последнего элемента. Так же эта структура данных позоляет ей дествовать как обычный стек

69) Разница ArrayList, LinkedList, Vector ?

Ответ :

ArrayList — Это как расширяемый массив, под капотом он содержит массив который изначально инициализируеться 10 ячейками если другое не указаоно в конструкторе, и после при добавлении и переполнении создаёться новый массив с большим размером (+50% от предидущего) и переписывает туда все даные. Скорость поиска елемента по индексе дв жтой структуре данных константная, а вот удаление и добавление приводит к большим потерям в пямяти и производительности. Когда мы приблезительно знаем в каких приделах будут размеры массива необходимо задавать эти размеры в конструкторе для улучшения призводительности .

LinkedList — Это двухсвязный список в который состоит из Узлов каждый из которых содержит ссылки на предидущий и последующий. Первый и последний узел так же связаны ссылками друг на дрга что ускоряет поиск в этом листе если елемент находиться сзади листа. Скорость поиска по индексу в этой структуре данных линейная, но удаление и добавление не требует дополнительных затрат памяти в слевствии чере сели мы часто изменяем список , то необходимо искользовать эту структуру данных

Vector — это структ. Даных как ArrayList но являеться потокобезопастно (synchronized) и может работать одновременно с несколькими потоками (он deprecated)

70) Разница Iteration, Enumiration ?

Ответ : Enumiration не имеет метода удалить обьект, и соотвецтвенно не возникает проблеммы что при прохождению по массиву Iteration вы хотите что-то удалить в выкидываеться ошибка : ConcurrentModificationExeption. Enumiration считатеться устаревшим и для решения этой проблеммы необходимо исспользовать ConcurrentHashMap/ConcurrentList

71) Какие методы имеет класс Iteration ?

Ответ : hasnext, Next — для перемещения по коллекции и так же remove — который приводит к ConcurrentModificationExeption

72) Какие виды дженериков вы знаете ?

Ответ : <T> - обычный типизированый класс, <T extends B> - ограничиваем то что наш жденерик должен буть наследником, <T super B> - наш дженерик должен быть покласом суперкласса

74) В чём оссобенность String класса являеться ли он иммутабельным?

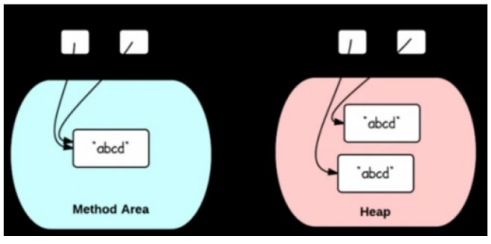
Ответ : Стринг являеться обёрткой над массивом char [], Модифицировать строку не возможно. Все строки финальные обьекты, при конкатанации масив char[] переписываеться полностью в новый с новой длиной. Для того что бы бороться с этим есть классы StringBuffer и StringBuilder. **StringBuffer потокобезопасен, а StringBuilder — нет (ВАЖНО).**

75) Почему мы не создаём String через конструктор new String ()?

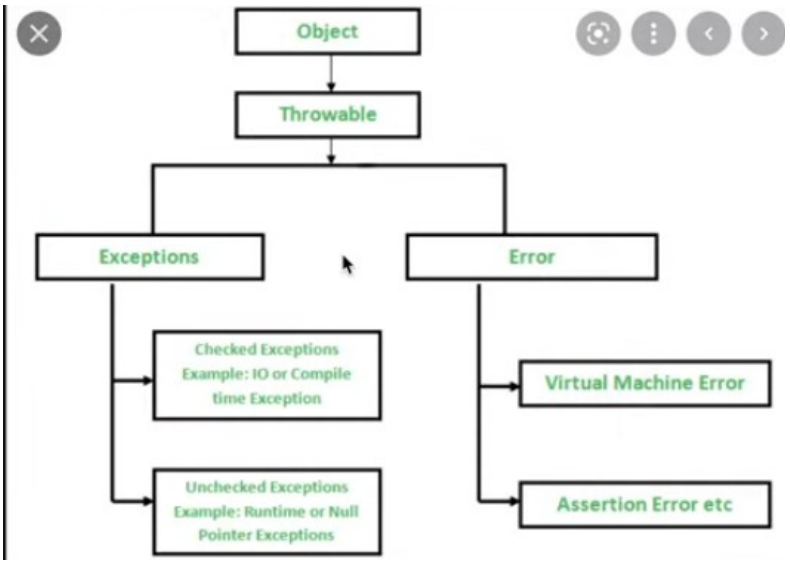
Ответ : В целом мы можем создать строку через конструктор, но тогда обьет будет добавлен в кучу. Обычно стринг, как и другие литералы(иммутабельные классы) хранится в стеке.

76) Что такое String pool?

Ответ : Все строковые литералы храняться в стринг пулах. И соотвецтвенно если мы создаём ещё один обьект стринг, который имеет то же значение то мы не создаём новый обьект а переиспользуем сувществующий (Начиная с java7 они храняться в heap)



78) Назавите иерархию исключений в Java и в чём их отличия ?



Два основных типа исключений это

checked — При привыполнении некоторых задач мы точно знаем что может возникнуть exeption, к примеру при читении файла. Такие опасные функции сама среда разработки потребует обернуть в try{}catch{} **ClassnotFoud Exeption, IOExeption , FileNotFoundExeption, OutOfMemoryError (heap)**

unchecked — Эти ошибки происходят в блоках в которых в целом они не должны присходить и там они не прогозируемы, по этому нам не подсказывает среда разработки что ту часть кода необходимо обернуть в try{}catch{}. **ArrayIndexOutOfBouds ArifmethikExeption NullPointerExeption IllegalArgumentExeption StackowerflowExeption**

77) Все наследники какого класса unchecked ?

Ответ : Все наследники Error uncheckd (VmError, AssertationError)

77) Отличие exeption Kotlin/Java ?

Ответ : В котлин все exeption **unchecked**

78) Integer a =127; int b = 127 ; Integer a1 =128; int b1 = 128 a==b a1 == b1?

Ответ : До 128 инежер (обёртка над примитивом будет сравинать примитивы после 127 будут сравниваться обьекты)

79) Почему Integer.MAX_VALUE + 1 == Integer.MIN_VALUE?

Ответ : После перполнения инта (32 байта) он переносит значение в найменьшее

80) Что такое инстанциация и инициализация ?

Ответ : Истанциация — это когда мы обьявляем что у нас есть какя-то переменая Class a; но не инициализируем её, после эту переменную можно проинициализировать a = new Class(1)

81) Работает ли Switch() с double/flowable ?

Ответ : нет, по тому что есть проблемма со сравнением чисел с плавающей запятой не могут быть представленны точно. Это связано с несоразмерностью десятичной и двоичной систем счисления. Поэтому десятичные числа, в основном, могут быть представлены в двоичном виде

82) Что такое ключивое слово transient ?

Ответ : При сериализации обьект мы можем добавить этот модификатор к полю что бы данное поле не было перобразовано в стрингу.

83) Что такое Анонимный класс ?

Ответ : Это такой вид inner класса поведение которого мы описываем при создании екземпляра класса

84) В чём отличие вложенного статичиского от не статичиского класса ?

Ответ : Вложенный не статичиский класс обьявняеться внутри другого класса он связан с эти классом и что бы получить внутренний класс нам сначала необходимо создать вснений класс. Вложенный статичиский класс не имеет доступа к полям внешнего класса

85) Разница между throw и throws ?

Ответ : throw new Exeption — что бы выбросить ексепшен throws — иы указываем у сигнатуры метода если мы не хотим обрабатывать этот ексепшен в функции а хотим делигирывать эту ответственность на того кто эту функцию вызывает

86) Какие виды ссылок есть в Java и в чём их отличия ?

Ответ : Ссылки расположены от самых сильных к самым слабым

Strong references - Любой объект что имеет strong ссылку запрещен для удаления сборщиком мусора до тех пор пока он не ссылаеться на null, это самые обычные ссылки когда мы создаём обьект new Class()

SoftReference - Такая ссылка будет удалена как только системе потребуеться больше памяти для работы он удалит эту ссылку но не моментально а отложенно

Weak Reference — Такая ссылка будет удалена как только системе потребуеться больше памяти для работы. New Weak Reference<String>

Phantom Reference — Эта ссылка может быть удалена в любой момент, но не может ыть удалена пока она находиться в `ReferenceQueue`, `из важного у` Phantom Reference ВСЕГДА вызываеться finalize()

87) Как работает GarbageCollector ?

Ответ : Heap Делится на две области:

- New (Yang) Generation - объекты, кот. тут считаются короткоживущими.
- Old Generation (Tenured) - обекты считаются долгоживущими.

Алгоритм GC исходит из того предположения, что большинство java-объектов живут недолго. Быстро становятся мусором. От них необходимо довольно оперативно избавляться. Что и происходит в New Generation. Там сбор мусора гораздо чаще, чем в Old Generation, где хранятся долгоживущие объекты. После создания объект попадает в New Generation и имеет шанс попасть в Old Generation по прошествии некоторого времени (циклов GC).

Для каждой из этих обласей есть ещё дополнительное разделение состоит из:

- Eden — Сюда помечаються обьекты из Heap. Если нет места запускается GC.
- Survivor - точнее их два, S1 и S2, и они меняются ролями. Хранятся объекты, которые признаются живыми во время GC.

88) Все способы как создать поток ?

Ответ : Мы можем создать свой класс унаследывавшись от класса Thread и в переопределённом меторе Run можем определить логику которая будет выполняться в этом новом потоке, либо мы можем создать имплементацию интерфейса Runable в котором в методе Run() добавить новую логику и после передвть в конструктор Thread() эту имплементацию .

89) Как сделать метод потокобезопастным (пока исполняеться в одном потоке не может быть вызван в другом) ?

Ответ : sychronized - к функции либо к части кода

90) Что значит voletile переменная ?

Ответ : этот модификатор мы применяем к переменной что бы все обращения к ней были атомарными и не возникало проблем с тем что с одного потока мы читаем данные которые другой поток уже считал и уже редактирует.

91) Можно ли наследоваться от дата классов ?

Ответ : Нет, это запрещено. Но мы можем дата класс наследывать от другого класса

92) Отличие sealed и enum классов ?

Ответ : enum константа может принимать единственное значение, а sealed могут иметь несколько екземпляров со своими значениями. Что самое главное в эих классах мы можем динамичиски задавать значения.

93) Что такое `object` / companion object ?

Ответ : `object` - это анонимный класс

```
val helloWorld = object {

    val hello = "Hello"
    val world = "World"
    // тип анонимных объектов - Any, поэтому `override` необходим в `toString()`
```

```
    override fun toString() = "$hello $world"
}
```

companion object — это альтернатива статичисим методам либо обьектам класса

95) Swich в котлине ?

Ответ : when () { s - > }

96) Расскажите про null safe операторы в котлин ?

Ответ : ?. → оператор вызываемый у nullable класса который при обращении к null обьету вместо того что бы вызвать nullPointerExeption вернёт null как результат вызова оператора

?: - елвис оператор если то что находиться слева от оператора не равно нул то он верёт это значение в ином случает вернёт то что находиться справа

```
val l = b?.length ?: -1
```

!! - мы утерждаем что обьект к которому мы обращаеться он не нул и тогда при обратом вылетет nullPointerExeption

97) Какая главная проблемма by lazy {} инициализации и что это значит ?

Ответ : by lazy — это делегат которому мы передаём обязанность проинициализировать екземпляр класса в первые когда к нему обратяться и сделать это единожды. Главная проблемма заключаеться в том что это может вызвать утечки памяти

98) Какие функции определены сразу у дата классов ?

Ответ : equals(), hashCode(), toString(), copy(), и так же можно получит поля как компоненты дата клсасса

```
data class Music(val title: String, val author: String)

val music = Music("Kalinka", "Abba")
```

```
val title = music.component1()
val author = music.component2()
```

99) Как работают extention функции ?

Ответ : Они приобразовуються в класс врапер с статик функцией которую мы добавляем

100) Что значит слово reified ?

Ответ : Оно пишиться перед дженерик классом и даёт возможность узнать изнутри класса какой именно класс был передан в Дженерик

```
inline fun <reified T> genericsExample(value: T) {
    println(value)
    println("Type of T: ${T::class.java}")
}
```

101) В чём разница между launch и await ?

Ответ : launch отдаёт Job, a await выдаёт Deffer<Type>

102) Как синхронизировать корутины ?

Ответ : при помощи Mutex().widthLock{ fun1(); fun2() } - мьютекс должен быть одинаковым для обеих функций

103) Разница между lateinit / by lazy(){} ?

Ответ : lateinit всегда var когда by lazy может быть использован только val и это гарантирует что он не может быть инициализирован повторно, для lateinit тип не может быть примитивным.

104) Как понять что мы в первый раз открыли активити ?

Ответ : Boundle SavedInstaceState = null

105) В чём разница между action и categhory у intent ?

Ответ : **Action** - строка, указывающая общее действие для выполнения (например, просмотр или выбор). ACTION_VIEW - Используйте это действие в намерении с startActivity(), когда у вас есть некоторая информация, которую действие может показать пользователю, ACTION_SEND — Используем когда у вас есть некоторые данные, которыми пользователь может поделиться через другое приложение, ACTION_DIAL

Category - Строка, содержащая дополнительную информацию о типе компонента, который должен обрабатывать намерение. В намерение можно поместить любое количество описаний категорий, но для большинства намерений категория не требуется. CATEGORY_BROWSABLE - Целевое действие позволяет веб-браузеру запускать себя для отображения данных, CATEGORY_LAUNCHER - Действие является начальным действием задачи и указано в средстве запуска системных приложений.

106) Зачем при поаороте екрана пересоздаёться activity ?

Ответ : По тому что при повороте екрана нужно занаво пропарсить алтернативный лейаут (ландскейп ориентации)

107) Являеться ли LayoutInflater сингелтоном и почему ?

Ответ : Да, он сингелтон по тому что он очень много весит и что бы не пересоздовать его мы всегда обращяемся к одному и тому же, он получаеться из BaceContext

108) Какую функцию необходимо переопределить что бы ложить информацию в бандл , и как называетьсят тот в котором мы получаем информацию себе назад ?

Ответ : OnSavedInstanceState , onRestoreInstanceState

109) Можно ли сделать так что бы активити не пересоздавалась при повороте екрана ?

Ответ : Да, для этого в манифесте у активности необходимо прописать атрибут configChanges = «orientation »

110) Как изменить что бы в портретной ориентации была одна колонка recycler view а при смени ориентации на ландскейп было 2 или более рядка ?

Ответ : Можно поменять в onConfigurationChanged колбеке у адаптера recycler view ставить лейаут менеджер с другими колонками.

111) Как вызвать onDestroy не проходя через onPause, onStop?

Ответ : если в onCreate() вызвать метод finish()

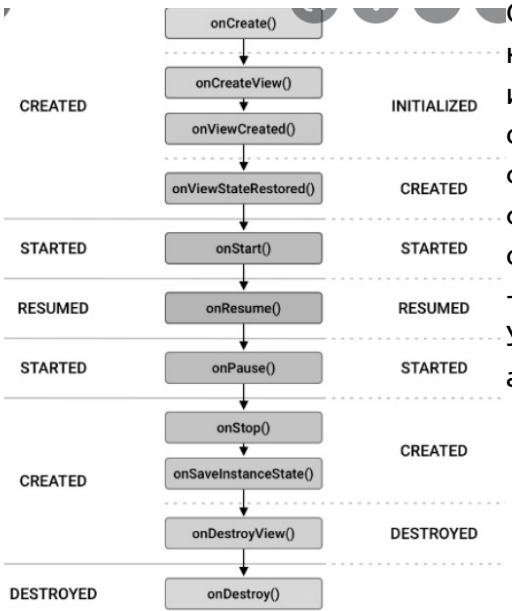
112) Расскажите ЖЦ activity ?

Ответ : onCreate() → onRestart() → onResume() → onPause() → onStop() → onDestroy()

113) Расскажите приоритеты при убивании процессов ?

Ответ : Сначала идут самые не убиваемые. ForegroundActivity() → VisebleActivity(onPause) → BackgroundActivity(onStop) → emptyProcess(Servise/BroadcastReciaver)

114) Расскажите ЖЦ fragment ?



Ответ : onAttach() прикрипляем фрагмент к активности → onCreate() создаём инстанс фрагмента → onCreateView() создаём view в которой будет находиться фрагмент→ onViewCreated() Говорит фрагменту что активити уже прошла стадию onCreate→ onStart() → onResume → onPause() → onStop() → onDestroyView() Убивает view которая была приатачина к активити → onDestroy() → onDetach()

114) Разница replace / add у фрагмент менеджера ?

Ответ : add - добавляет один фрагмент поверх другого, replace — заменяет фрагменты

115) Перечислите все возможности передачи данных между фрагментами ?

Ответ : SharedViewModel — вю моделька на активити а не на фрагмент и отсюда мы можем получть данные, либо можно создать локальный броадкаст ресивер через который передать данные.

116) Как из одного активити передать данные в другое активити ?

Ответ : это можно сделать через интент добавив данные в extra, либо более приятный способ это вызвать startActivityForresult(intent, variable), setResult()

117) Расскажите все типы данных в Java/Kotlin ?

Ответ :

java :

Data Type	Size	Description
`byte`	1 byte	Stores whole numbers from -128 to 127
`short`	2 bytes	Stores whole numbers from -32,768 to 32,767
`int`	4 bytes	Stores whole numbers from -2,147,483,648 to 2,147,483,647
`long`	8 bytes	Stores whole numbers from -9,223,372,036,854,775,808 to 9,223,372,036,854,775,807
`float`	4 bytes	Stores fractional numbers. Sufficient for storing 6 to 7 decimal digits
`double`	8 bytes	Stores fractional numbers. Sufficient for storing 15 decimal digits
`boolean`	1 bit	Stores true or false values
`char`	2 bytes	Stores a single character/letter or ASCII values

kotlin : Помоимо того что есть в Java есть ещё Scientific numbers E / e добовляеться для того что бы отображать степень 10 `val myNum1: Float = 35E3F`

118) В чём разница между FragmentManager / Fragment trasition ?

Ответ : Manager — руководит добвалениями и удаленимя фрагментов а транзакшен позваляет редоктрировать процесс переключения

119) Какие launchMode вы занете ?

Ответ : SingleInstance — не будут создавтья более одного инстанса активити SingleTop - Activity можно создать столько, сколько мы захотим. Единственное отличие состоит в том, что если уже есть экземпляр Activity с таким же типом наверху стека в вызывающей задаче, не будет создано никакого нового Activity, вместо этого Intent будет отправлен существующему экземпляру Activity через метод `onNewIntent`, Standart - будет всегда создавать новую Activity

120) В чём разница между Activity / AppCompatActivty ?

Ответ : AppCompatActivity - это то же активти но но потдерживает новые функции для старых устройств

121) Что значит `windowSoftInputMode` ?

Ответ : Это аттрибут активити который отвечает за взаиможействие активности с клавиатурой. Если значение равно **adjustResize —** активность изменяеться что бы оставить место для клавиатуры , если значение равно **adjustPan —** активность не изменяет размеры что бы дать место для клавиатуры

122) Сколько конструкторов у кастомного view должно быть ?

Ответ : Необходимо переопределить 3 коструктора :

123) Расскажите ЖЦ view ?

Ответ : **onMeashure()** → Установить сколько в пиксилях будет занимать view,

onLayout() → Вычисляеться позиция этой view

onDraw() → Рисует view

124) Как добавить кастомные атрибуты у кастомного view ?

Ответ : Необходимо в xml описать в xml

<declare-satlable>

<attr name = «atrribute format=»boolean»>

И после в layout подключаем xmlns — пространство имён из файла с тегом <declare-satlable>

125) Что такое canvas ?

Ответ : Это холст на котором можно рисовть при помощи класса paint

126) Как сделать так что бы в Layout обьекты располагались по кругу ?

Ответ : Написать свой LayoutManager в котором оперделить такое расположение расположение

127) Как сделать так что бы в Layout обьекты располагались по кругу ?

Ответ : Написать свой LayoutManager в котором оперделить такое расположение расположение

128) Как сделать так что бы вьюшка маштабирывалась в зависимости от того какое она занимает пространство (flexable layouts)?

Ответ : для этого в xml мы прописываем avtosizetMin; autosivedMax, step в атрибутах при обьявлении и размер view будет меняться в зависимости от колличества места которое она может занять

129) Что такое denscitie либо плотность пикселей ?

Ответ : это колличество пикселей которое вмещаеться в едениу размера к приемру в один дюйм

130) Как мы создаём свой RecyclerViewAdapter ?

Ответ : В самом простом случае если у нас обычный ViewHolder, что-то вроде TextView/ImageView, тогда наследуемся от класса RecyclerView.Adapter<ViewHolder>, если у нас в качестве layout для itemов листа используеться свой view то мы должны создать класс который наследуеться от ViewHolder и в конструктор которого котрого передать binding нашего layouta, создать метод bind в котором мы из нашего дата класса вставляем данные в binding, и создать статисикую функцию в from в которой из данных нашего дата класса возвращать экземпляр ViewHolder

```
class MyAdapter : RecyclerView.Adapter<MyAdapter.MyviewHolder >{

class RepoViewHolder private constructor(val binding: RepoItemBinding) : ViewHolder(binding.root) {

        fun bind(item: Pair<Repo, Boolean>) {}

        companion object {
                fun from(parent: ViewGroup): RepoViewHolder {}}

}
```

131) Какие функции необходимо переопределить для наследника RecyclerView ?

Ответ : **`getItemCount()`** возвращает общее количество элементов списка. Значения списка передаются с помощью конструктора. **`onCreateViewHolder()`** создает новый объект ViewHolder всякий раз, когда RecyclerView нуждается в этом. **`onBindViewHolder()`** принимает объект ViewHolder и устанавливает необходимые данные для соответствующей строки во `view`-компоненте

132) Как решить вопрос с разными view для RecyclerView ?

Ответ : Если мы имеем разные ViewHolder для разных обьектов для этого необходимо создать sealed class в котроый добавить эти 2 класса, далее в onBindViewHolder проверять какиой из типов нашего холдера к нам пришёл с пошощью when() далее вызываем bind у этого холдера с данными которые пришли.

133) в чём суть патерна ViewHolder ?

Ответ : Суть паттерна заключается в том, что для каждого элемента списка создаётся объект, хранящий ссылки на отдельные вьюхи внутри элемента. Таким образом, приходится выполнять достаточно дорогой вызов findViewById(int) только один раз при создании элемента.

134) Какой класс можно использывать что бы ловить разные жесты ?

Ответ : GestureDetector

135) Что такое патерн adapter ?

Ответ : Когда приходят одни данные и они преобразовуються в сложный view

136) Что такое диалог и в чём его отличие от диалог фрагмента ?

Ответ : Диалог это просто окно а диалог фрагмент это всё таки фрагмент который сохраняеться в стеке и к нему применимы все вещи которыеп

137) В чём отличие sneack bar от toast ?

Ответ : Они отличаються визуально а так же тоаст будет видно даже если приложение свёрнуто а снеакбар только пока оно видно

138) Что такое Penging Intent ?

Ответ : Это интент которому мы передаём ответственность совершить интент в будущем когда будет на это время. За частую мы используем такие интенты когда нам в будущем нужно изменить интент

139) в каком методе мы вополняем работу Intentservise?

Ответ : `onHandleIntent`

140) Как в Ретрофите в гет методе поставить атрибут в путь в контретом месте ?

Ответ :

```
@GET("users/{name}/commits")
Call<List<Commit>> getCommitsByName(@Path("name") String name);
```

141) Как в Ретрофите в гет методе поставить атрибут в КОНЕЦ пути ?

Ответ : Для этого просто в параметре функции добавляем аннотацию @Query(«id»)

142) Как в Ретрофите в пост методе добавить значение ?

Ответ : Для этого просто в параметре функции добавляем аннотацию @Body

143) Как в Ретрофите передать значение url как пораметр функции ?

Ответ : Для этого просто в параметре функции добавляем аннотацию @URL

144) Как в Ретрофите изменять все запросы ?

Ответ : Для этого необходимо создать класс наследник класса Interceptor, переопределить функцию intercept и далее мы можем редактировать chain запрос и добовлять к примеру хедер при помомощи функции .addHead(«api-key»,12312312321)

145) Как в Ретрофите добавить возможность возвращять типы Rxjava ?

Ответ : При создании ретрофит инстанса AddCallAdapterFactory(RxJavaAdapterFactory.create())

146) Как создать реторфит ?

Ответ : Для начала нам нужно создать интерфейс в котором обьявить методы в которых описать методы которые должны выполняться потом создать инстанс ретрофита передав в функцию create наш интерфейс

147) Какие основные составляющие базы данных Room?

Ответ : Интерфейс с аннотацией @Dao в котором описаны все методы которые будут применяться для работы с базой данных, data class с аннтоацтией @Entity(tablename=»Name») который будет представлять таблицу из БД , и абстрактный клсаа который наследуються от RoomDatabace и аннотрованый аннтацией @DataBase(entities), который будет содержать абстрактное поле с нашим Dao интерфейсом и будет обеденять все наши Entities.

148) Какие аннотации для методов Dao вы знаете ?

Ответ : @Delete, @Insert,@Update,Query(«»)

149) Какие особенности инжекта в поле при помощи Dagger ?

Ответ : Во первых нельзя инжектить в private поля, во вторых для того что бы заинжектить в поле необходимо в компоненте создать метож в который аргемнтом буде т фрагмент или активность в которую мы инжектим поле и вызвать эту функцию у инстанса дагера в функции onStart()

150) Как в дагере заинжектить интерфейс ?

Ответ : Что бы связать интерфейс с реализацией В даггер 2 необходимо создать метода который необходимо аннотировать при помощи аннотации @Bind а как параметр передать реализацию возвращаемного метода

151) Как передать контекст в дагер ?

Ответ : Для этого необходимо создать интерфейс с аннотацие @Component.Factory внутри которого передать как параметр контекста котороый необходимо заанатировать @BindsInstance

152) Что такое subcomponent в дагер ?

Ответ : Это компонет который наследует дерево обьектов которые инжекстяться родительским компонентом. Компонент родитель должен содержать методы которые возвращяют Factory для каждого из subcomponent, так же они одожны быть подключены через аннотацию `@Module( subcomponents = [ RegistrationComponent::class,])`. По этому необходимо определить фактор метод в сабкомпоненте

```
@Subcomponent.Factory
interface Factory {
   fun create(): RegistrationComponent
}
```

Создать инстанс сабкомпонента можно только из инстанса родительского компонента, вызвав у него соотвецтвующий метод

153) Что такое Component dependencies дагер ?

Ответ : Это способ связывания двух или более компонентов. Важно два компонента связанные таким образом не могут името одинаковый scope. Родительский компонент в своем интерфейсе должен явно задавать объекты, которыми могут пользоваться зависимые компоненты. У класса который должен наследовать какие- то зависимости мы указываем в теге @Component(dependencies = AppComponent::class.java). И при инициализации зависимого компонента необходимо передать инстанс компонента родителся.

154) Что такое LifecycleOwner ?

Ответ : Это интерфейс с единственным методом getLifecycle() который должен быть переопределён любым наследником который имеет жизненный цикл.

155) View Model vs OnSavedInstanceState ?

Ответ : При удалении процесса Данные будут сохранины в SavedInstancestate а VM

156) Paging 2 Какие методы должен переопределять класс наследник PagingSourse?

Ответ : loadInitial(LoadInitial) — вызаёт то колличество данных которым должны изначально проинициализировать лист, loadRange определяет в каком ренже подгружаються дальше данные

157) Paging 3 Какие методы должен переопределять класс наследник PagingSourse?

Ответ : loadInitial(LoadParams) — загружае данні у відповідності до LoadParams де вказано і початкову кількість елементів котру загружаєм і то якими самтками підгружати. Также необходимо переопределить функцию `getRefreshKey` в который вызываеться когда происходят изменения в списке и что бы не возвращяться к началу списко перерисовуеться по мере надобности тот обьект который изменился.

158) Как создать свой WorkManager?

Ответ : Нужно унаследоваться от класса WorkManager и переопределить метод doWork в котором определить свою логику. После необходимо обернуть класс в WorkRequest и далее передать этот reuest в метод enque() у ворк менеджера

159) Какие виды WorkRequest вы знаете и какие критерии к ним можна применить ?

Ответ : OneTimeRequest, PreoidicTimeRequest, критерии setRequiresBatteryNotLow - Критерий: уровень батареи не ниже критического. SetRequiredNetworkType - Критерий: наличие интернет, setReuestDeviceisIdle, setRequiresCharging

160) Как передать и как получить данные из WorkManager?

Ответ : Что бы передать необходимо вызвать метода srtInputData, и в doWork getInputData, для того что бы получить данные из workManager можно вызвать метод observe при создании WorkManager и получить данные

161) Как создать notification ?

Ответ : Для этого необходимо начиная с 8 андроида создать notificationchanel(для того что бы легче контролировать уведомления) после при помощи notification buildera создать инстас В Notification и далее при помощи функции notify() отобразить повещение

162) Какие виды анимаций вы знаете ?

Ответ : ProperetyAnimation — мы изменяем характеристики вьюх с некоторым дюрейшеном при помощи класса ObjectAnimator и далее применяем её к вьюхи , AnimatedDrawable это когда мы описываем анимацию в xml файле в теге <animation list> <item></>, Физичиские анимации Fling/Spring animation которые делают анимацию с трением либо анимацию пружины. Tansition animation — в этой анимации мы создаём 2 сцены и связываем их одним обектом с тегом tansitionName=»» далее мы описываем вид анимации и его длительность в отдельном xml файле и (виды анимаций ChangeBounds , Fade, TransitionSet) и далее у TransitionManager вызваем функцию go() с двумя сценами. Так же есть ещё такой вид constaraintLayout MoutionLayout в котором мы анимацию описываем в отдаешьном xml файле и есть очень удобное окто демострации анимации прямо в предпросмотре.

163) Какую функцию необходисо переопределить у ViewPager ?

Ответ : getItem которая соотвецтвующему индексу сопостовляет соотвецтвующий фрагмент

164) Почему необходимо использовать @Binds вместо @Provide ?

Ответ : Используя Binds мы уменьшаем колличество кода которое будет генерировать даггер так как для таких провайдов дагер не генерирует wrapp Factory классов

165) Перечислите два сопособа передаать обекты в Даггер при создании его истанса ?

Ответ : @Component.Factory , @Component.Builder

166) Что такое Scope в Dagger и как создать кастом скоуп?

Ответ : По умолчанию все компоненты и методы имеют один ии тот же скоуп сингелтон(Относительно истанса компонента). Скоуп являеться аннотацией которую мы можем ставеть перед компонентом и функцией. Скоуп аннотация отвечает за жизненный цикл компонента либо инстанса который мы инжектим. Для того что бы изменить скоуп мы можем создать свою annotation класс который необходимо заннаторывать @Scope @Rtention() и после этого наш компонент может иметь отличный жизненный цикл.

167) Что такое multibinding в Даггер ?

Ответ : Это способ в случае если мы байндим одному интерфейсу множество реализаций то их можно не указывать каждый по отдельности а передавать их в сете заанотировав функцию аннотацией @IntoSet

168) Что необходимо использовать что бы заанимировать добавление/удаление itemos в recyclerView ?

Ответ : ItemAnimator

169) Как создать Dagger Hilt ?

Ответ : Для этого необходимо класс наследник Application зааннотировать аннотацией @HiltAndroidApp

170) Как заинжектить в поле в Dagger Hilt ?

Ответ : Для этого не надо в компоненте прописывать функцию которая параметром принимет класса в который мы инжектим и вызывать эту функцию у компонет класса. Для этого достаточно аннотироыать этот класс аннотацией @AndroidEntryPoint

171) Какие готовые компонеты с готовым скоупом есть в Dagger Hilt и как молуль добавить к этой компоненте ?

Ответ : Есть готовые компоненты : SingletonComponent → ApplicationScope

ViewModelComponent → ViewModelScope / ActivityComponent → ActivityScope

FragmentComponent → FragmentScope / ViewComponent → ViewScope / ServiceComponent → SecvsScope

Что бы приязать модуль к любому из готовых компонентов достатоно аннотировать его **InstallIn(Singlecomponent::class)**

172) Как добавтить контекст в метод из Dagger Hilt ?

Ответ : Для этого необходимо заанотровать в функции аргумент контекста @ApplicationContext

173) Как провайдить по разному созданные инстансы одного и того же класса в Dagger Hilt ?

Ответ : Для этого необходимо использовать созданные собой @Qulifier annotation class

174) Как заинжектить VievModel Dagger ?

Ответ : Для мы будем инжектить ViewModelFactory как поле

175) Как заинжектить VievModel Dagger Hilt?

Ответ : Для этого можно просто делегировать инициализацию by ViewModel

private val exampleViewModel: ExampleViewModel by viewModels()

176) Какие способы создания observable вы знаете?

Ответ : . create () - и в нури написать логику имитера , `fromCallable(), fromIterable(), fromArray(), just(Item), Range(1,1000)`

177) Какие способы создания observable вы знаете?

Ответ : . create () - и в нури написать логику имитера , `fromCallable(), fromIterable(), fromArray(), just(Item), Range(1,1000)`

178) В чём отличие Callable и Runable?

Ответ : Callable возвращяет значение а не просто запускает task, Callable → FutureTask → Thread

179) В чём отличие map и flatMap?

Ответ : map всегда должен возвращать елемент который мы отредактируем каким-то образом. Fletmap может вернуть 0 1 или более елементов так же flatmap принимает на вход данные, излучаемые одним `Observable`, и возвращает данные, излучаемые другим `Observable`, подменяя таким образом один `Observable` на друго

180) В чём отличие map и flatMap?

Ответ : map всегда должен возвращать елемент который мы отредактируем каким-то образом. Fletmap — трансформирует данные получаемые от обсервабла в Другие Обсерваблы и сглажевает (flatten) емиты от всех этих обсерваблов в один результирующий обсервабл

181) Какие ещё виды map вы знвете в RxJava?

Ответ : condactMap — Работает так же как и FlatMap но сохраняет порядок емитов , switchMap — тоже что и FlatMap но при получении нового елемента из цепочки предидуще потоки удаляються и активным остаётся только последний Observable

182) Как изменить поток в котором будут выполняться операции над излученными данными и в котором будут излучаться данные

Ответ : ObserveOn — поток для операции с данными которые уже излучины (можно применять много раз изменяя поток в котором будут происходить операции с данными)SubscrubeOn — поток где происходит излучение данных (в create либо fromIterable) **применяеться только последний**

183) Назвоите все операторы Комбинирования потоков в RxJava ?

Ответ : Observable.zip() → Соденяет емиты один к одному и возвращяет пары (1,2), merge — просто обьеденяет два емита идущих один за другим, comcat — сначала заемитит всё первый Observable потом второй и потом последующщий

184) Как решить проблемму утечек памяти в RxJava ?

Ответ : Собирать все подписки в CompositeDisposioble и отписываться в методах clear()/onDestroy()

185) Как решить проблемму утечек памяти в RxJava ?

Ответ : Собирать все подписки в CompositeDisposioble и отписываться в методах clear()/onDestroy()

186) Как бороться с проблемой backpreashure RxJava ?

Ответ : для этого мы и спользуем Flowable которому передаём стратегию по взамиодействию с такими проблеммами : ERR0R. В случае backpressure бросается исключение `MissingBackpressureException`. BUFFER. Все элементы добавляются в буфер. DR0P. Все новые элементы, которые консьюмер не успевает обработать, удаляются и не доставляются консьюмеру. LATEST. Стратегия противоположная Drop. Консьюмер получает только самый последний элемент. Так же для этого можно использовать оператор window который собирает данные за какойто промежуток времерни и потом после этого окна испускает значения.

187) Какой самый экономичный тип графичиского файла ?

Ответ : WebP → Png(transperency) → Jpg (god copressing algorithm)

188) Что такое Exequtor ?

Ответ : Это способ дешовый работы с многопоточностью при помощи того что при пуске программы мы создаём thread pool а после уже работаем с ним передавая ему обёкты Runnable / Callable

189) Что такое Hendler ?

Ответ : Это класс, который используется для работы с очередью сообщений, связанной с потоком. Хэндлер позволяет отправлять сообщения в другие потоки с задержкой или без, а также обрабатывать полученные сообщения. При создании Hendler мы указываем Лупер связанный с ним и при помощи функции post{} помещаем лямду на выполнение

190) Что такое Looper ?

Ответ : Это класс который запускает цикл обработки сообщений, связанный с потоком. Поток работает, пока связанный с ним лупер не будет остановлен.

191) Опишите модель взоимодействия MVC ?

Ответ : В этой моделе есть 3 основных слоя : model — бизнес логика и дата, источники данных, реппозитории классы view — xml файлы, controller — активити либо фрагмент который отвечает за логику отображение

192) Опишите модель взоимодействия MVVM ?

Ответ : В этой моделе есть 3 основных слоя : model — бизнес логика и дата классы, репозитории, источники данных view — Activity/Fragment тоторые отображают данные из бищнес логики, viewModel — мостик между model и view взаимодействие между которомыми происходит реактивно — view ничего не знает о model и наоборот

193) В чём разница между функциями коллекций assosiateWidth(), assosiateBy() ?

Ответ : assosiateWidth создаёт Map сиз начений коллекции и значений получиных в лябде переданой в эту функцию, assosiateBy — то же что и assosiateWidth но если есть повторяющиеся значения то он и будут удалины и остаться только последние из низ

194) Что делает функция коллекций partition() ?

Ответ : оставляет в листе знаения которые продят по придекату переданому в функции, остальные отделяет в отдельный лист

195) Как называеться лейаут в котором обьекты могут наслаиваться друг на друга ?

Ответ : Coordinator layout

196) Как преобразовать горячий поток в холодный в RxJava и наоборот ?

Ответ : Что бы гарячий поток сделать холодным необходимо применить функцию cache(), что бы преобразовать холодный в горячий достаточно у Обсервабла вызвать функцию publish

197) Как как избавиться от 4 конструкторов в castomView ?

Ответ : @JvmOverloads

198) Расскажите про модули в чистой архетектуре ?

Ответ : В чистой архететуре выбеляют 2 основынх модуля core / app.

Core — это самый независимый модуль, он должен содержать минимум зависимостей от классов с андроидом, это необходимо что бы в будущем можно было достаточно легко белать unit тесты для этого модуля. Этот модуль содержит 3 основных слоя : useCaselayer — в нём осдержаться классы которые должны выполнять только одну задачу для взаимодействия между presentation и data слоями. DataLayer — в нём содержаться интерфейся репозитория и источников данных. Domain слой который содержит модели и бизнес логику.

App — это модул в котором происходит взаимодействие с андроид. Он состоит из 2 слоёв : PresentationLayer — содержит в себе классы которые отображают данные Fragment/ ViewModel, FrameworkLayer — Содержит имплиментации data sourse и repository, а так же свои модели .

199) Какие функции должен переопределять класс на который делегируют что-то ?

Ответ : getValue(properety) — вызываеться когда мы обращаемся к переменной setValue(propereteу) — когда присваеваем значения переменной

200) Какие scope функции kotlin вы знаете ?

Ответ : let — применяеться к обьекту, принимает лябде все вызовы в которой действуют в контексте обьекта (context objct — it), apply — тоже что и let только обьект в лямбде это this и можно вызвать функции неявно, width(object) — обьект в чём контексте мы хотим выполнять дейсвтия передаёться как параметр

201) Как работает jetpackCompose ?

Ответ : Вся графика опсываеться в функции которая обозначаеться аннотацтией @Compose → Для того что бы изменить что-то в лейауте мы вызваем механизм recomposition — заново вызваем функцию composiable → значения между двумя состоянимя мы передаём пр помощи помещания их в slotTable при помощи функции remember{ var i =1 }. Так же для облегчения работы с состояниями есть класс MuatbleState — при изменении значения екземпляра класса который нахордитьс я внутри mutable state вызываеться автомотичиски recomposition и перерисовуеться UI

202) Как в jetpackCompose создать recyclerView?

Ответ : LazyColumn