"инкапсуляция" - это, в первую очередь, объединение поведения с данными в логический закрытый модуль, реализующий свой бизнес-функционал через открытые интерфейсы, а не просто "закрытие данных и методов через геттеры и сеттеры". Здесь принципиальным моментом является не ограничение доступа к полям и методам, а разбитие функциональности программы на логические блоки по принципу Single responsibility, а потом уже сокрытие реализации этих модулей и выставление открытых интерфейсов наружу.

Адекватный ответ на вопрос - почему нельзя сравнивать строки по == заключается в том, что часть строк созданы не кодом вашего приложения, а являются входынми данными (тот же Scanner, к примеру) и даже при том, что вы, может, и не использовали явным образом new String() ни в одной части вашего приложения, результат по == выдаст false там, где по значению строки равны. А на вопрос - почему Мапа не в коллекции - мапа не реализуетIterable

Свернуть

0:59

перейти к нашему кандидатам [музыка]

1:05

еда друзья я заявил о том что у нас будет крутой кандидат многие подумают что сейчас воруются мерило будет

1:10

рассказывать про Джава мемори модул канкаренси на её пакет разнообразная реактивное программирование во всяких

1:16

формах его проявлениях Но нет сегодня у нас просто крутой гость и его зовут Виктория Да друзья это девочка Впервые

1:24

на нашем канале в рамках тестовых собеседований пришла к нам девочка это супер круто Я прошу вас поддержать

1:30

Викторию в комментариях потому что действительно Это для неё необходимо немного предыстории перед тем как пройти

1:37

этот тестовый техническое собеседование Виктория уже имела забронированную дату для прохождения реального технического

1:42

собеседования в компанию где она потенциально может получить работу пообщавшись на эту тему мы с ним вместе

1:48

решили а почему бы нам не провести тестовое техническое собеседование и выявить проблемные места которые можно

1:54

устранить до прохождения реального собеседования и знаете что это действительно Вики помогло поскольку мы

2:00

смогли выявить проблемные места заблаговременно их устранить заблаговременно отполировав Как

2:06

говорится и после прохождения технического собеседования Вика получила положительный отзыв как Junior Java

2:12

Developer что согласитесь крутое достижение поэтому поддержим еще раз Вику в комментариях и отдельно хочется

2:18

отметить Вика волнуется на собеседовании тестовом ни тестам как и любой другой разработчик поэтому хочется сразу

2:24

сказать всем диван на штурмовым критиком АйТи и программирования Я хотел бы лично

2:30

посмотреть как вы будете проходить свое первое техническое собеседование что ж друзья погнали Да узнаем как Вика его

2:36

прошла Какие проблемы мы выявили и какой план для устранения этих проблем Мы

2:41

обозначили погнали [музыка]

Вопросы

2:46

Хорошо давай тогда начнем и традиционно Первые вопросы связанные с

2:55

архитектурой и дизайном Что такое

3:01

методология программирования Когда у нас все является взаимодействующими объектами

3:09

хорошо

3:19

Окей А сможешь вот ты говоришь Все крутится вокруг объектов

3:25

А в чем тогда разница вот Есть еще такая штука Как допустим процедурное программирование всем это знаем оно

3:32

знакомо допустим там с того же там JavaScript может быть PHP на многих других языков программирования включая

3:39

таких как грудь допустим тоже на жилье по сути мы можем использовать

3:45

так называемые вот в чем тогда вот основная своей точки зрения различия между процедурным программированием и

3:52

объект Ну там все строятся на процедурах

3:58

процедуры это какие-то исполняемые сценарии я понимаю объект

4:05

Это объекты То есть это вижу ближе к какой-то бизнес логике и

4:12

предметному представлению окей а как думаешь можно вот процедурно

4:18

программирование представить что-то в виде объектов я не очень знакома с процедурным

4:26

программированием но возможно в каком-то виде можно на самом деле действительно можно в

4:32

каком-то виде но там нету ни классов ни интерфейсов но и многих вещей которые свойственны ВВП поэтому А тут ты

4:38

действительно права что процедурно программирование оно связано с выполнением каких-то определенных операций где все опять же крутится

4:45

вокруг данных но данные сконцентрированы в этих операциях И этим

4:50

все сказано в оп мы создаем специализированные объекты которые обладают поведением состоянием и уже

4:56

работаем от этого то чего не хватает хорошо по поводу столпов Давай вернемся к ним с

5:04

твоей точки зрения в чем смысл популяции Смысл в том что мы скрываем

5:13

данные внутри класса и запрещаем доступ к ним Извне И изменять

5:21

данные класса могут только его методы Окей получается все классы все данные

5:31

должны быть закрыты при помощи другие исключения когда допустим нам

5:39

приватная модификатор доступа не нужно обеспечить на каком-то другом уровне

5:44

предположим [музыка] Не совсем понимаю вопрос то есть смотри

5:52

мы если мы хотим запретить прямой доступ к данным мы можем их скрыть но оставить

5:58

доступ к ним через методы То есть через гетторе Да это здесь Вопросов нет

6:06

правильно но представим себе ситуацию что если я хочу все-таки

6:12

обращаться к данным Я не хочу иметь гетры и сеттеры но при этом я хочу

6:17

организовать инкапсуляцию предположим на уровне пакета возможно ли это чтобы

6:23

внутри пакета все было доступно А вот снаружи пакета допустим есть класс у нее

6:29

свойство снаружи пакета это доступно модификатор доступа

6:39

А как он пишется

6:44

это правда

6:51

Привет Сегодня друзья наша интеграционочка и нет я вам не буду рассказывать про крутые курсы пайтон

6:57

который так всем нужен Однако не так уже просто найти работу в этом Поэтому вместо этого я вам расскажу про крутую

7:03

новость которая очень сильно хочу вам поделиться и сто процентов она будет вам интересно при поддержке моих коллег и

7:08

партнеров мне удалось основать тренировочную платформу i-proje ты можешь просто называть её тренинг центр

7:14

Но давай-ка лучше я тебе расскажу чем мы там на ней занимаемся и какие проблемы решаем

7:22

а фишка является очень лакомным кусочком и все стремятся в неё попасть и в этом ничего плохого нет в этом удивительного

7:28

тоже ничего нет Однако времена Пионеров прошли ведь современный клиент он же

7:33

работодатель четко знает чего он хочет Какие технологии применять И как это все должно быть реализовано соответственно

7:40

это существенно понижает цену вновь испеченных новичков которые закончили

7:45

ВУЗы прошли курсы Нет не подумайте я не буду здесь рассказывать том что ВУЗы

7:50

плохие или курсы плохие это всё круто и курсы нужны и ВУЗы нужны Однако речивается о том чтобы получить работу

7:56

для современного новичка необходимо обзавестись опытом работы и выходит здесь у нас Замкнутый цикл чтобы

8:02

получить работу Мне нужно получить работу чтобы получить работу опять не нужно получить работу а где взять работу

8:07

если для этого нужна работа кажется что какая-то безвыходная ситуация ведь получить первую работу не так уж и

8:13

просто Либо ты должен быть каким-то реально выдающимся вундеркингом что тебя и без опыта работы возьмут на работу но

8:19

решение есть для обхода щепетильной ситуации и давай-ка я тебе расскажу как мы решаем эту проблему при помощи

8:26

тренировочной платформы iprose решение базируется на многолетнем опыте преподавания работы в различных

8:31

it-компаниях и реализации современного программного обеспечения Все это позволяет осознавать проблематику

8:37

современного новичка it ведь Как показывает опыт время лишь активная практика в кругу специалистов позволяет

8:42

достичь крутых результатов Короткие сроки тренировочная платформа iprody позволяет погрузиться в разработку

8:48

реальных клиентских проектов для получения столь желаемого и необходимого коммерческого опыта Весь процесс

8:54

построен таким образом что вы становитесь членом команды из 5-6 разработчиков под наблюдением высококвалифицированного Тим Лида Все

9:00

это позволяет принять участие в разработке реальных клиентских проектов вы решайте задачи среднего и высокого

9:06

уровня сложности принимаете участие в обсуждении этих самых задач то есть в их анализе и дизайне понимаете участие в

9:13

разнообразных командных митингах И разумеется в код других вы можете получить реально свежую обратную связь о

9:19

том как вы что-то реализовали И разумеется ис имеет свое положение до того как его кто-то заметит другой и

9:25

конечно же принимаете активное участие в код ревью что позволяет вам получить быструю обратную связь что ведет свою

9:32

очередь к улучшению ваших навыков и повышению вашего уровня квалификации все

9:37

это в конечном итоге ведет к повышению уровня вашей экспертизы при реализации клиентских проектов что соответственно

9:43

влияет на уровень вашей привлекательности ведь таким образом вы автоматически становитесь лакомным кусочком в глазах работодателя и конечно

9:49

же HR рекрутеры теперь не хотят обходить вас стороной поскольку Они видят что вы

9:55

потенциально новый сотрудник в компанию который уже с первых дней работы в этой компании сможет приступить к реализации

10:00

задач которых так ждет работодатель или его клиенты А все это потому что теперь ты не просто джун ты джун с большой

10:07

буквы ты джун плюс что я вижу тебя Это заинтересовало хочешь присоединиться к

10:12

команде современных разработчиков решать крутые задачи и получать таски посложнее так еще и побольше никакой проблемы в

10:19

этом нет это очень просто переходи по ссылке в описании озна комьтесь с деталями программы повышения квалификации заполни контактную форму

10:26

нажми кнопку отправить и в кратчайшие сроки мы с тобой свяжемся помни что путь мечте начинается с первых шагов Так

10:33

давай же реализовывать вместе на тренировочной платформе Ай против

10:42

хорошо здорово Скажи пожалуйста в чем смысл

10:47

наследования наследование А мы можем создавать новые

10:53

классы на основе уже существующих и Наследники получают

11:00

могут использовать их реализованные методы добавлять свои

11:05

методы и точно также с данными могут они переопределять эти методы

11:12

допустим Мне нравится поведение там Родительское Да конечно можем переопределять можем добавлять свои

11:19

А если я не хочу переопределять я могу Я хочу запретить определение допустим если

11:25

такая возможность

11:31

[музыка]

11:40

Хорошо как он работает

11:48

я не Я не уверен что мы можем делать методы

11:54

файл можно Ну хорошо

12:01

запрещаем наследование Да абсолютно на самом деле здесь все просто То есть я

12:09

нет сомнений что ты знаешь как модификатор работает на переменных это понятно из-за менять На классах ты права

12:15

их нельзя наследовать а если мы хотим именно запретить переопределение определенных методов то мы можем

12:21

подцепить его на метод файл Но на самом деле я бы так сказал это далеко не самый

12:28

распространенный кейсы то что допустим есть такая возможность но вот лично я и свой опыт использовал эту возможность

12:35

один или два раза за все время сколько должны быть какие-то реально веские основания чтобы запрещать конкретному

12:41

методу соотношении с другими этого же класса

12:47

определению это очень кейс серьезные с

12:53

другой стороны я видел много таких случаев и в большей степени это как ошибка дизайн так сказать кто-то просто

12:59

это сделал Потому что так решил хотя на самом деле можно было бы и разрешить Но это все очень субъективно здорово Хорошо

13:06

давай тогда продвинемся в дизайн Какие знаешь шаблоны проектирования который

13:13

доводилось использовать или Вероятно это просто

13:24

[музыка]

13:32

очень много на самом деле Давай поговорим про адаптер В чем смысл

13:37

адаптера адаптер позволяет работать несовместимым

13:45

интерфейсом вместе то есть допустим если у нас есть какой-то класс который работает с джейсонами А у нас появились

13:52

новые водные нам нужно работать еще из xml Мы можем написать это для того чтобы

13:58

он просто конвертировал Джейсон и все отлично просто лучше не

14:04

придумать супер пример Давай тогда Вернемся более простой

14:11

всемирно известный синглтон заезженный но здесь вопрос будет с подвохом

14:17

Понятное дело что синглтон для создания одного экземпляра и поддерживаем одного экземпляра Но вопрос другой А что если

14:25

наш синглтон будет выполняться в многопоточной среде что нам необходимо

14:30

сделать для того чтобы обеспечить создание экземпляра но при этом всем

14:37

потока безопасности чтобы не было такого что два потока одновременно врываются и создают два экземпляра

14:44

возможно имеет смысл сделать метод Get instance синхронизированным

14:50

почему бы нет А что ты выберешь синхронизированный метод или синхронизированный блок скорее

14:57

синхронизированный блок а Чем это выражено Расскажи Ну в данном конкретном случае объяснить

15:05

я наверное не смогу Но обычно используется синхронизированные блоки Потому что если у нас есть несколько

15:11

синхронизированных методов в классе то если синхронизируется один То есть если

15:18

становится один метод то закрывается второй потому что они синхронизируются на объекте то есть

15:27

соответственно с блоками внутри методов мы можем избежать

15:32

какие-то объекты на которых они будут синхронизироваться если

15:39

то синхронизированный метод он всегда привязан кзысь но синхронизированном блоке мы можем

15:45

привязать что-нибудь другое хорошо

15:52

здорово А знаешь ли какие-то другие способы обеспечения здесь синхронизации

15:59

Но не прибегая к методу что-нибудь более такое свежее

16:05

[музыка] Нет наверное не скажу Может быть слышал

16:11

что-то про Локи допустим я слышала да я знаю что они есть но

16:19

никогда не использовала То есть это по моему из пакета конкаент у нас идет

16:24

Ну хорошо Еще вопрос про синглтон В чем разница

16:31

игр и лези синглтон жадный ленивый

16:39

создается в момент первого обращения насколько я помню Игорь соответственно

16:45

сразу как думаешь почему Нам нужен игр и

16:51

почему существует чем это связано Возможно мы не уверены что он будет

16:59

когда-нибудь использован резонно А еще

17:05

продолжать мысль мы не уверены что это используется но предположим

17:10

скажем А что если мы не уверены в этом но его создание нам ничего не стоит

17:17

простую переменную но что если нам нужно пронизировать какой-то сложный процесс

17:23

чем здесь преимущество Lazy собственно преимущество в том что мы не будем

17:31

задействовать ресурсы если они нам не нужны Хорошо давай сдвинемся в вопросы про

17:40

стринг и еже с ним что такое стринг пол

17:47

это механизм который работает со строками когда мы

17:53

создаем новые строки он их сохраняясь там И если мы создали один стринг и

18:00

дальше создаем такой же то он будет ссылаться на этот же объект в

18:08

памятиле что и первый То есть это сделано для экономии ресурсов

18:15

хорошо Почему нельзя сравнивать строки через двойное равно у нас есть оператор

18:22

двойного равно оператор эквивалентности Почему нельзя сравнивать Ну это будет

18:27

сравнение просто ссылок

18:35

мы можем это делать но

18:41

какой смысл Не ну хорошо допустим может быть смысла

18:47

и нет но тут тогда вопрос такой больше в теоретическом практически какие есть виды сравнения наверняка

18:55

кто-нибудь задулся когда-либо что вот есть сравнение там ссылочная и сравнение

19:01

не только ссылочные да то есть вот если мы говорим про двойное равно и метод

19:08

iqos конкретно в разрезе про стринг вот метод иквул что он внутри себя

19:15

сравнивает сравнение с двойным равно сравнивает содержимое

19:22

А почему так важно это делать в сравнении с просто равно потому что у

19:30

нас могут быть разные ссылки но одинаковые объекты по equels

19:37

хорошо Да Действительно это так и есть это как раз вот то что говорил есть ссылочное сравнение есть так называемая

19:44

структура если конечно правильно произношу и помню Смысл в том что иногда нужно проверить

19:52

ссылку этого достаточно а иногда нам нужно проверить эквивалентность объектов поля к полю чтобы все его значения были

19:58

действительно одинаковые Хорошо если возвращаясь к метод iqos Мы уже обсудили

20:04

что мы сравниваем ссылку Мы также сравниваем поля Но что

20:11

мы еще должны сравнить [музыка] это применимо к стрингу или в целом

20:18

сейчас предусмотрено безусловно потому что это правило вообще но в целом мы если так

20:25

сказать в целом Если в целом то не переопределенный equals он сравнивает

20:31

объекты как единую область памяти если при определенные то соответственно мы

20:38

можем сравнивать любые поля которые мы захотим да Но помимо полей там еще что-то есть Ну то есть я могу привести

20:45

пример представим себе У нас есть Класс А у него есть поле и просто велик и вот

20:52

есть класс B у него есть тоже такое же поле intoula value и у них значение

20:57

одинаковые Ну то есть класс А который хранит значение 20 и класс B который хранит значение 20 если мы будем

21:05

сравнивать по полям с большой долей вероятности они будут эквивалентны но

21:10

здесь есть какой-то различие ты имеешь в виду то что нам нужно проверить

21:16

нал и общность классов абсолютно

21:25

Хорошо давай тогда продвинемся чуточку дальше в тренде есть такая такое понятие как

21:34

конкатанация строк мы знаем что это такое Это склеивание строк между собой Однако есть другая проблема

21:40

она говорит о том что в цикле когда мы пробегаемся по строке и делаем

21:45

множественного конкатацию это ведет к Under performing потребляется большое

21:50

количество памяти у меня здесь вопрос задачи общеизвестная как ее решить Мне

21:55

надо все-таки в цикле делать строки но как решить так чтобы не было даунгрейд и поппер

22:02

в данном случае нам нужно просто использовать string Builder потому что он позволяет

22:08

изменять строки и не создают каждый раз То есть он изменяемый в отличие от

22:14

стринга OK Здорово Но если мы откатимся

22:20

в то что стринг все-таки он им Ньютон Да он неизменный каждый раз когда мы его

22:25

изменяем создается реально новый экземпляр старый остается но все-таки string Builder он все-таки билды строку

22:32

не задумывалась ли ты как тогда под капотом он устроен что же там такое что избавляет нас от создания строки Но

22:38

делает это все-таки в конечном итоге возможно Я подозреваю что там просто массив

22:44

символов Абсолютно верно Ну или просто если свестись массив байтиков Да абсолютно А

22:52

что если нам необходимо использовать многопоточной среде если какая-то для этого возможность безопасно это делать

22:59

Да просто нужно использовать который является синхронизованным

23:08

Думаю здесь достаточно Давай поговорим с тобой про Collection Framework в javi

23:16

сразу вопрос не важно Почему коллекция интерфейс ее называют

23:25

коллекцией но коллекция потому что она не принадлежит к этой

23:32

иерархии и собственно идет она отдельно и это все-таки словарь больше

23:38

А как ты можешь а вот с твоей точки зрения охарактеризовать разницу между

23:43

типичной коллекцией неважно какой и вот словарем в чем там существенное различие

23:50

существенное различие в том что листы у нас хранят просто объекты а

23:56

здесь это ключ значение Давай тогда

24:03

поговорим про и линки для В чем разница между

24:10

разница в реализации реалист имеет под капотом массив

24:17

линки-твист это связанный список супер супер

24:23

чем между ними различия с точки зрения их

24:30

с точки зрения добавления значения когда своей точки зрения эффективно применять

24:35

релиз А когда Линкин естественно ты хочешь добавлять в этот список значений

24:43

применять если у нас предполагается большое количество изменений в списке В

24:50

том числе ставки и удаление объектов из середины списка

24:58

применять Когда у нас просто сохранение чтения без особых изменений

25:05

если мы говорим про линки лист Какая скорость доступа к значению

25:16

что если хотим получить первое значение или последнее какая будет в этом случае

25:26

от единиц до тогда спрошу

25:36

ты права от единицы Но почему допустим с первым значением мы можем сказать Окей

25:41

Она же 1 да то есть Нам нужно получить первое значение всегда стоит первое значит если мы пробегаемся по нему мы

25:49

сразу получаем То есть это явно от единицы если мы ищем Нам повезло оно

25:54

Первое это от единицы Но если все-таки я хочу получить именно последнее значение какая Вот здесь будет скорость доступа

26:02

точнее чем обеспечен доступ хранит ссылки на первый и последний свой

26:09

объект А можешь рассказать почему Ну смысла чем

26:15

линьки благодарен или кому он благодарен за такое

26:20

нет но у него есть внутренний класс нот и собственно он хранит эти ноды Ну

26:26

ссылки на них и при добавлении следующего меняется эти

26:32

ссылки Да но надо устроена таким образом что нода сама по себе это последовательность

26:39

каждого надо хранить ссылку впереди стоящую и позади стоящее но тем не менее условно говоря знает там какую-то ноду

26:48

первая нода не знает что существует последняя условно говоря или насколько то что там находится после неё следующая

26:55

нода насколько она последняя или все-таки посередине где-нибудь но ты действительно правильно сказал что он

27:00

хранит отдельно в своих состояний ссылку на первый последний элемент и получить

27:08

доступ можно через специализированные методы Пласт этому он обеспечен вернее

27:14

этому он благодарен интерфейсу двойная очередь

27:20

мало кто задумывается но в основе линки глиста в том числе лежит двойная очередь

27:25

если мы запястьем линки лист к этому интерфейсу или изначально присвоим к

27:32

интерфейсу Дека объект linkedly что у тебя будут методы соответствующие Они

27:38

как раз обеспечивают от единички Хорошо давай поговорим про такие вот интересные

27:43

вещи опять же про коллекция Что такое Fail Fast и fail

27:50

это связано с оперированием не скажу

27:56

тогда выражение может быть незнакомое

28:01

перефразирую что произойдет если я начну и тренироваться по коллекции

28:08

рассматривать ее значение и тут же эти значения удалять мы не можем этого делать насколько я

28:15

помню

28:21

мы если нам это нужно мы можем создать копию листа и изменять его

28:28

время итерации мы не можем изменять

28:33

список по которым мы идем Да ты права вылетит Уже на второй попытке вылетит

28:39

исключение кто называется ансапорта poporation Exception Окей Ну ты сказала создать

28:46

копию листа с одной стороны ответ хороший с другой стороны хочется копнуть смотри если мы

28:54

создаем копию листа предположим один к одному просто это будет новая ссылка опять же когда мы тренируемся по этой

29:01

коллекции мы попадаем ровно в Точно такую же ситуацию которую ты описала что по ней ирируются невозможно и удалять

29:09

одновременно тогда как это копирование можно правильно осуществить чтобы все

29:15

действительно работало или может быть есть уже какие-то конкретные реализации листов тех же самых которые внутри себя

29:24

поддерживают это внутреннее копирование с возможностью во время интегрирования

29:30

нет про коллекции такие точно не знаю по поводу копирования возможно есть в

29:40

collections какие-то статические методы для копирования коллекций я безусловно есть Но это

29:48

много чести здесь не даст поскольку мы обратно сливаемся проблем того что как

29:54

как итерирование реализовано потому что в коллекциях есть понятие итератор это

30:02

отдельный интерфейсы если мы возьмем допустим релиз скопируем его это будет

30:07

новая ссылка скопированные значения начнем по нему оперироваться и удалять

30:13

это будет та же самая проблема поскольку экземпляр класса который не поддерживают

30:20

такие операции время италирования то есть это уже заложено в самом коде чтобы

30:26

правильно самое интересное ты реально брала необходимо в Европе но это копия

30:31

должна поддерживаться внутри то есть есть специализированные коллекции которые во время интеллирования делают

30:38

копию самого себя ты этого не видишь разумеется это бьет по памяти ниже пояса

30:43

Но тогда ты можешь Благодаря этой копии иперироваться и удалять значение потом

30:49

Разумеется эти значения обновляются допустим пример хорошей копии он райт

30:54

собственно его название так и это сложно это из пакета канкары

31:02

синхронизированный Но каждый раз когда мы делаем значение

31:08

изменяем какую-то операцию добавление удаление он делает копию И разумеется мы

31:13

можем безопасно Хорошо давай тогда здесь двинемся немножечко

31:21

дальше и поговорим с тобой про рефлексию и аннотации можно рассказать в

31:30

чем смысл reflection Pi в контексте языка Java

31:35

Смысл в том что мы можем обращаться с классами как с объектами то есть

31:41

обращаться с классами с полями

31:48

конструкторами Окей Можем ли мы получить доступ к

31:55

приватному методу и выполнить его да можем через рефлексию можем Окей

32:03

А допустим изменить приватное значение по-моему тоже можем

32:09

Здорово Что для этого необходимо Но предположим я не буду вдаваться в

32:14

подробности как получить метод это все сахар что главное мне необходимо сделать

32:20

перед тем вот я нашел тот самый метод свойства Я знаю точно на приватная

32:25

сомнений нет Мне необходимо его выполнить или изменить что перед

32:31

выполнением операции изменения выполнения обязательно

32:40

не скажу Я знаю ты знаешь но представим себе как

32:45

только я попытаюсь выполнить операцию по изменению приватного поля Он

32:51

не скажет нет фронтальная во время выполнения этой программы вылетит ошибка что

32:58

пропорции Это я из головы придумал это выражение мы не можем не можем выполнить эту

33:07

операцию поскольку данное свойство но приватное Что необходимо сделать чтобы

33:12

вскрыть вот эту приватную шкатулку

33:19

нет идей есть специальное свойство Set аксесси

33:25

был Нет не слышала Ну хорошо На самом деле мне кажется

33:30

просто забыл там я ничего страшного Это абсолютно нормальная штука reflection API это

33:39

темные такие места но на самом деле это правда что перед тем как выполнить

33:45

что-то приватно нужно просто для этого метода или свойствами Специальный центр называется Access был по моему или акции

33:51

неважно ставишь просто свойства значение

33:57

и потом выполняешь потом когда происходит завершение операции закрываешь его обратно чтобы она не

34:03

оставался открытым Хорошо давай поговорим про аннотации Зачем нам нужны

34:09

аннотации некоторые механизм метаданных которые

34:17

можно добавлять класса полям методом конструктором

34:22

задавать некоторую функциональность на них

34:28

Окей предположим себе ситуацию Я хочу создать

34:34

аннотацию чтобы она была применима только на

34:40

уровне классов Я между на свойства не надо на методы Не надо Что для этого необходимо сделать

34:47

там уже вешается момент создания нотации вешаем на неё

34:53

аннотацию и указываем Какая именно я честно говоря не помню я сам порой

35:01

путаюсь на самом деле забываю когда много Не используют на самом деле Таргет

35:07

что если я хочу чтобы у меня аннотация дошла до пережила компиляцию то есть мы

35:16

знаем что у нас наш код проходит несколько стадий то есть компиляция Затем он уже превращается

35:23

то что мы можем выполнять но мы понимаем что не всегда аннотации

35:30

доступны во время выполнения программы собственно Что необходимо сделать чтобы конкретная аннотация была доступной во

35:38

время выполнения программы Если не ошибаюсь точно также есть какая-то нотация где можно это указать

35:44

при создании да помнишь какие есть уровни Ну то есть это аннотация

35:50

называется ретеншн если я не ошибаюсь сколько будет жить помнишь Из каких уровней сколько

35:57

вот этих ретеншено сколько видов есть Нет не помню

36:07

не волнуйся на самом деле есть три ей богу я сейчас сам не помню как

36:13

называется как они называются но могу описать первое будет доступно время компиляции но после компиляции

36:20

отбросится потом последнее самое мощное она будет

36:25

доступна в в Инстаграме и вторая которая также есть

36:31

которая будет после компиляции на остается Но в рам-тайме то есть применяется Но в рантами ее уже не будет

36:38

я сам на самом деле криво описал как не пасть потому что довольно таки часто приходится писать

36:44

код собственно с аннотациями но в целом их там три Хорошо

36:51

давай тогда с тобой поговорим про кон Карен здесь мы с тобой уже много раз

36:59

затронули этот вопрос У меня самое главное вопрос про синхронизацию то есть

37:07

есть методы блоки синхронайст Знаешь в чем проблема ключевого слова

37:14

что там такого нехорошего в нем есть и почему вот появился новый пакет

37:24

ну одна проблема которую я писала новый пакет по-моему там есть класс который

37:32

облегчает все [музыка]

37:39

Окей окей они задумывался ли ты почему Ну появился облегчение понятно но там

37:49

есть какая-то техническая подковырка то есть Каким образом это синхронизация обеспечивается допустим у нас есть

37:56

множество потоков и они выполняют иногда очень редко операцию записи

38:03

но в большей степени выполняется в основном операция чтения У нас есть 10 потоков какая проблема Если мы сделаем

38:11

класс синхронизированными методами при помощи ключевого слова какая проблема здесь возникает 10 потоков

38:18

начинает выполнять метод что как только один поток его

38:26

захватывает все остальные Его ждут абсолютно то есть никакой

38:32

синхронизация Но многопоточность работает линейно Именно поэтому и появился новый пакет

38:43

и там это проблема очень хорошо решена

38:49

Здорово можешь описать такие принципы как

38:57

консистентность и атомарность атомарность это неделимость то есть

39:07

грань когда несколько операций могут выполняться как единое целое то есть

39:15

допустим в том же concorent сейчас есть классы

39:20

как Atomic integer по-моему которые могут

39:27

которые предоставляют атомарные операции инструмента

39:34

то есть и они неотделимые то есть другие потоки не могут вмешаться в их выполнение

40:00

это какая-то гарантия того что у нас будет ожидаемый результат

40:08

на самом деле

40:19

что это обработка нескольких операций как

40:24

единого целого допустим сейчас Джаве пятнадцатый появился такое понятие как

40:30

экстракт канкаренси это произошло джеб 428 и там идея того что можно запускать

40:36

множество асок для выполнения потоков несколько

40:41

для выполнения но обрабатывать их как единый Объем работы символ юнитов вор

40:47

так называемые действительно про томарность Но это протоман атамар нас на другом уровне и ты классно брала пример

40:53

с инкрементом потому что оригинальный крем Это несколько операций то есть операции изменения операция присвоении

41:00

второй момент консистентность это обеспечение того что

41:08

все данные после работы над ними будут находиться в согласованном состоянии то есть не должна быть такого что один

41:14

поток изменить данные другой поток изменяет тоже данные эти же то есть так называемые

41:19

и в конечном итоге у нас получается битое значение

41:25

хорошо Твой ответ защитный очень понравилось Расскажи

41:34

пожалуйста возвращаясь к коллекции Как создать синхронизированную коллекцию мы можем

41:39

использовать коллекции которые есть

41:45

[музыка] хешмак копии

41:51

арелист Если не ошибаюсь карент linked Key

41:59

достаточно А знаешь какие то способы помимо пакета

42:05

дедовские варианты насколько я знаю в

42:12

классе collections есть статический метод какой-то для того чтобы делать

42:18

коллекции синхронизованной Абсолютно верно Есть

42:25

карнайзер хорошо идет вопрос с подвохом есть такая

42:32

коллекция которую все называют называется

42:40

Почему не рекомендуется использовать

42:45

честно говоря не знаю

42:51

а можно сказать Ну то есть немножко в этом Ну по-моему это какой-то вариант

42:58

маплы собственно какая-то реализация Почему ее не рекомендуют использовать

43:04

сказать не могу но тоже на самом деле сказала она синхронизирована

43:10

действительно Это первый вариант и был это одно и то же практически за

43:17

некоторыми исключениями и как раз ее не рекомендуется она полностью

43:23

синхронизирована как мы с тобой уже определили синхронизация она не дается бесплатно это просадка по

43:30

производительности и как допустим нам рекомендует

43:35

книгах эффективная Java как вариант если вам не нужна синхронизация ничего не

43:41

надо синхронизировать синхронизирует только когда в этом действительно есть необходимость и соответственно опять же

43:49

возвращаясь обратно ты ответил что есть Стрим стринг баффер это идеальный пример того что Джош блох пытался описать что

43:56

если вам нужно просто управлять созданием строки пожалуйста Stream

44:01

Builder если вам нужно этим управлять в многопоточной среде пожалуйста Стрим баффер хэштей был в тот момент когда он

44:08

был создан он был единственным в этом плане Разумеется синхронизация или не синхронизация она все равно там была вне

44:15

зависимости от того какой среди Мы работаем хорошо здесь здесь все отлично Давай поговорим про исключения В чем

44:24

различие между ключевыми словами фроу и

44:29

мы используем когда хотим выбросить исключение

44:36

Передаем исключение то что мы будем пробрасывать его в чем

44:42

разница между так называемый человек исключение это исключение которые мы

44:51

должны обрабатывать в блоках

44:57

соответственно то что мы не должны обрабатывать

45:06

а если я не хочу использовать но у меня чек исключение То есть мне надо

45:12

что-нибудь сделать мне просто компилятор не позволит но я не хочу использовать блок какой у меня есть вариант мы можем

45:21

просто выбрать его наверх а как это будет выглядеть в Козе Ну то есть Понятное дело что мы выбросим

45:27

наверное представим себе ситуацию А я пишу код У меня есть метод простой ничего не делать там внутри просто одна

45:34

строка Exception Exception сам себе является проверяемым исключением и вот

45:40

сто процентов у меня такой комбинации будет ошибка компиляции скажет что

45:48

человек то есть Я не побеспокоился об обработке

45:54

проверяемого исключения при этом всем блокчей у меня не задействован я не хочу

46:00

его задействовать и выброс все-таки происходит но компилятор

46:05

такой не примет как это можно решить предполагаю что можем написать какой-то

46:10

свой хендер не исключительно силами Java

46:24

нет идей нет А как насчет ключевого слова фроу

46:33

так не совсем понимаю но у нас же есть ключевое слово которое мы пишем в сигнатуре и метода как ты

46:41

сказала которая сигнализирует о том что Да так мы пробрасываем его наверх о чем

46:47

было сказано А ты имела ввиду ну да то есть мы не хотим его обрабатывать можем

46:53

пробросить наверх и это уже будет не нашей проблемой а проблемы другого метода Ну то есть вот это пробрасывание

47:00

наверх то есть мы говорим о том что ты просто всегда туре указываешь ключевое

47:05

слово это будет защита и тип исключения Все я хорошо хорошо Ну

47:13

конечно не совсем пробрасывание если говорить технических деталях но я понял согласен

47:21

Давай тогда поговорим про ключевое слово про блок

47:27

может блок быть без

47:36

Ну то есть в целом Да но как бы какой в нем смысл

47:42

тогда в этом случае Не ну как бы логически Да но если мы

47:47

говорим технически это возможно или нет технически нас может быть блок Try и

47:53

файлы Family по моему окей да и в таком случае мы

48:01

упускаем А может ли у нас быть несколько Catch но предложим там тратить

48:10

Да может но тогда нужно соблюдать иерархию исключений Потому что если у

48:15

нас идут какие-нибудь Exception и Exception

48:21

вторым блоком то второй блок никогда не будет срабатывать потому что все будет перехватываться первым отлично

48:31

Что такое ресурсы мы при объявлении при открытии

48:41

блока траи можем поместить туда какие-то ресурсы которые будут автоматически закрыты После выполнения кода либо при

48:50

выбрасывании исключений не нужно соответственно делать это

48:56

вручную я могу Туда положить любой ресурс или эти ресурсы должны быть

49:01

чем-то обусловлено они должны быть

49:13

хорошо вопрос Вот как раз так лозу был и очень

49:20

похоже это отступление от исключений представим себе ситуацию У нас есть

49:26

так называемый улучшенный цикл for где мы пишем с правой стороны тип коллекции

49:33

предположим или массива с левой стороны его конкретный элемент который содержит и по сути это доступ к конкретному

49:40

элементу в процессе это есть Могу ли я создать свой собственный класс

49:50

и поставить его в блок for each при этом

49:55

всем этот класс не является коллекцией Если не ошибаюсь он должен быть

50:06

Абсолютно верно вопрос защиты Да действительно можешь рассказать об этом про это было то опять немножко в

50:14

коллекции но все-таки это очень сильно похожи с тем как работает

50:21

по сути нам нужно создать какой-то свой список по которым мы можем дальше

50:27

италироваться либо нам нужно создать итератор и предоставить его

50:36

обязательно список должен быть это даже не должна быть коллекция в целом речи

50:41

там то что мы можем переключать я допустим Один раз работал с документом

50:48

Excel и вот как раз API которая была его элементы блоки

50:54

ячейки они были представлены виде того самого не удивлен что я могу и между ними

51:01

переключаться Хотя по сути и намека на это никакого не было поэтому Да действительно в блок Tribes resources и

51:09

в for H мы можем поставлять свои собственные реализации и они вовсе не обязаны быть ресурсами или коллекцией

51:16

в этом отношении Хорошо тогда давай переключимся с тобой

51:23

долгожданную тему работа с базами всегда такая животрепещущая штука однако

51:30

через это тоже стоит пройти Давай поговорим про таблицы ты говоришь о том

51:36

что управление таблицами правда

51:42

если так на поверхности Мне необходимо создавать таблицу Я как правило хочу

51:47

таблицам добавлять ограничения Какие ограничения

51:52

существуют они называются концентрики Какие ты знаешь

52:01

всякие у них

52:06

Окей Какие в себя праймерики первичный ключ

52:13

включает другие ограничения он по сути составной

52:22

или перефразирую какие значения недопустимы для Прайма реки

52:33

не скажу но допустим Можем ли мы вот у

52:38

нас есть таблица и в ней есть поле которое отмечено

52:44

праймерики Могу ли я вставить на качество возможно один раз

52:51

Скорее всего нет нет нельзя а могу ли вставить дубликат Допустим меня

52:58

правильно реки повторяются в одной записи

53:04

состоит из

53:09

Иначе как мы будем искать это все для

53:15

чего Нам нужен для связывания таблиц и создание

53:22

взаимоотношений между таблицами но смотри если мы говорим про Вот именно

53:29

связывание Ведь мы можем связать по первичным ключам допустим у нас есть первичный ключ в одной таблице и во

53:37

второй таблице тоже есть какое-то поле которое отвечает которое соответствует этому первичному ключу предположим

53:43

первая таблица users у него есть свой средний свойства у него есть атрибут ID и вторая допустим User

53:51

details и там есть User ID технически без фрейм и мы можем их

53:59

связать Ведь мы можем выполнить операцию Join зная что одно поле соответствует должно соответствовать потенциально

54:05

другому тогда здесь вопрос Если мы не обязательно должны для этого случая

54:10

применять формки Зачем же он тогда все-таки нужен То есть связывание понятно То есть связыва ние мы можем

54:18

обеспечить и без

54:25

Что произойдет если я попытаюсь удалить

54:33

User detail который связан Или точнее что может произойти если я попытаюсь

54:38

удалить юзер дитейл который связан с юзером или наоборот удалить юзер который

54:45

связан с юзер но зависимости от того как мы это

54:51

настроили дело то есть если у нас Каскадный

55:00

при удалении юзера будут удалены и все данные его юзер

55:06

если этого нет то соответственно будет удален только юзер и

55:11

останется вот это как раз есть достижение вот этого

55:17

самого фрейм Потому что когда мы устанавливаем фрейм нас есть возможность установки этой каскадной операции то

55:25

есть опять же знаешь просто звучит Это для тебя вероятно в

55:30

новой форме так как я это трактую хорошо здесь мне нравится все давай двинемся

55:38

чуть дальше Расскажи В чем разница между операцией Union и операции

55:48

Join это объединение таблиц в одну сводную таблицу

55:55

Union честно говоря не помню

56:03

давай так если поговорим про Join попытаемся выцепить одну из другого

56:09

что-то подразумеваешь под объединением Ну то есть потому что если технический то объединение это Union переводе но ты

56:17

говоришь что Join он объединяет несколько таблиц между собой в одной из результирующую таблицу как это будет

56:23

конечном итоге выбить То есть это буквальном смысле склеивания двух таблиц

56:28

или это будет выборка определенных данных из нескольких таблиц как

56:37

у нас есть несколько типов джойнов того какой мы используем соответственно будет

56:45

определенная редактирующая выборка если мы используем то у нас будут все данные из первой и

56:53

второй таблицы но соответственно где-то будут если нет составления может быть когда мы

57:02

получаем а пересекающиеся поля

57:09

пересекающиеся строки скажем так соответственно может быть Left или Raid

57:16

Join когда мы получаем полностью данный из одной

57:21

таблицы и пересекающиеся с ней данный из второй Окей

57:27

а что будет вот допустим возьмем такой типовой Лев Джон Что будет в случае если

57:36

данных каких-то не хватает в правой таблице то как результирующая таблица

57:42

будет выглядеть там денег хватает данных второй поля будут просто нал и все в

57:49

любом случае мы получим все данные из первой таблицы

57:54

хорошо такой вопрос опять же

58:00

Обязательно ли соответствие структур таблиц при выполнении операции то есть

58:07

представим себе есть таблица юзер он состоит и черный и есть таблица

58:16

User details там есть тоже ID там есть юзер ID и еще какая-то другая информация

58:22

когда Мы выполняем операцию Join Необходимо ли Вот эта структуральное

58:28

соответствие по структуральным соответствиям я подразумеваю что таблица правая таблица левая имеет одинаковое

58:35

количество атрибутов эти атрибуты соответствуют друг другу то есть по сути структура одной таблицы соответствует

58:42

структуре другой таблицы Нет не обязательно нам главное что

58:48

Наша задача указать По какому полю они будут соединяться то есть

58:54

мы можем взять юзарей

59:00

Наша задача точнее Главное чтобы они были равны

59:07

да на самом деле Да в общем вот мы ответили на вопрос В чем разница Union

59:13

также говорит Ты все знаешь и между джоем Join то принципе соединение данных не

59:19

столько про таблицу сколько присоединение данных и я бы так выразился немножко кто-то поправит

59:25

но в целом это так и есть Мы берем одни данные мы сопоставляем с другими данными

59:30

ищем пересечение либо отсутствие этого пересечения в зависимости как это рассматривать под каким углом в случае с

59:37

Union в буквальном смысле Соединяем две таблицы по принципу одна таблица соответствует структура на другой И

59:44

разумеется разница только в том Будут ли повторяться данные которые дублируются вправо и в левой таблице или нет потому

59:52

что есть Юнион All и просто Union подразумевает что там будут только уникальное значение Union All с учётом

59:59

дубликатов вот собственно А и всё различие между ними Давай тогда

1:00:04

поговорим про транзакции мы знаем что в базах данных

1:00:12

нормальная ситуация когда выполняется параллельные транзакции и вот Какие

1:00:18

возникают аномалии при выполнении

1:00:27

Ну вообще нам должны обеспечивать атомарность

1:00:33

операции и откат в случае неудачных выполнений соответственно

1:00:40

проблемы могут быть наверное те когда там где транзакции пытаются как-то

1:00:46

получить доступ к одним и тем же данным или изменить хорошо начало ты заговорила за

1:00:54

атомарность мне тут сразу приходит вопрос есть принципы

1:01:00

что это за принцип фундаментальный набор принципов которые гарантирует

1:01:07

собственно там барный консистентность

1:01:19

и еще атомарность

1:01:27

Это последняя скорее всего и есть буква

1:01:38

а задам вопрос наводящий если две транзакции

1:01:45

работают над определенной над определенными данными неважно это выборка данных это конкретная запись

1:01:52

конкретная таблица в этом разрезе неважно может ли одна транзакция врываться в процесс изменения данными

1:02:00

Да молодец что ты знаешь что ты знаешь А если зафиксировать это Да действительно

1:02:06

атомарность что весь набор операции в рамках одной транзакции как ты уже

1:02:11

неоднократно описывал он воспринимается как один сингл юнитов на английском

1:02:17

языке то есть как единое целое у нас консистентность что мы говорим что

1:02:22

данный согласовано после того как все изменено все другие должны Разумеется транзакции понимаете о том что это есть

1:02:28

затем Произошло это изменение изоляция что одна транзакция не врывается в другую и не мешает ей

1:02:35

разумеется именно то что все данные которые после транзакции были зафиксированы или все

1:02:42

состояние которое было зафиксировано оно должно отражаться в нашей таблице Или

1:02:48

допустим Если был railback какой-то он тоже должен отражаться ты сказала Кстати

1:02:57

как вообще выражается роллбек Ну то есть можешь мне описать процесс условно

1:03:04

написание кода У тебя есть операция ты делаешь изменение данных в Джаве допустим

1:03:14

идти пишешь простой код тебе Необходимо взять табличку и изменить в ней какую-то запись вот с этой точки зрения как этот

1:03:21

код будет выглядеть Я не спрашиваю как это точно будет но что в этом коде

1:03:27

неизменно ты должна обеспечить чтобы транзакция хорошо работал и в случае

1:03:32

каких-то проблем мы могли откатить состояние Ну то есть мы должны открыть соединение

1:03:39

вероятно из-за комитить в результате все это дело и закрыть

1:03:45

Окей камень хорошее слово камин А что

1:03:50

произойдет если какая-то проблема Ну то есть мы же не знаем наверняка

1:03:57

выполнится эта операция или не выполнится Камиль же по сути это

1:04:02

фиксация состояния то есть мы коммитием только в том случае когда все хорошо Ну а если не все хорошо чего здесь

1:04:13

Ну допустим во время выполнения операции произошла какая-то проблема что нам и

1:04:19

при этом все опишу более широкий кейс У нас есть

1:04:25

несколько таблиц наша операция сложная мы говорим про Тамара Нам необходимо выполнить запись в несколько таблиц

1:04:33

соответственно во время записи во вторую таблицу произошла ошибка первый таблица

1:04:38

уже зафиксированы данные они там уже как-то видоизменены во второй таблице

1:04:44

при попытке что-то изменить произошла ошибка А до 3 мы такие не добрались мы

1:04:50

можем в этом случае делать или нет Нет я думаю что нет должны быть

1:04:58

откатанные все изменения которые не были завершены

1:05:04

да то есть тот кейс который ты описывал на просто не хватает ролбэка Ну то есть ты сказал открыть соединение

1:05:10

побеспокоиться чтобы закрыть его выполнить операцию все-таки закомитесь и в конце просто нужно добавить роллбэк

1:05:17

если мы говорим про джаву там условно низком уровне если какое-то исключение произошло во

1:05:25

время выполнения наших операций Разумеется отловить и сделать и будет нам счастье смотреть все Ты

1:05:31

знаешь пришло время решать задачи

Решение задачи

1:05:38

скинул ссылку собственно

1:05:43

Подключайся пока расскажу про задачу Нам необходимо реализовать функцию faceboost известная задача нам передается значение

1:05:52

в виде Инта и нам необходимо собственно привести код

1:06:01

следующее состояние если число кратное трем то массив массив заносится если кратно 5 бас если

1:06:10

кратно одновременно 5 то заносим физбассем случае выбрасываем

1:06:17

исключение или аргументы Exception с любым сообщением сообщения здесь не важно что необходимо реализовать

1:06:26

и следующее покрыть тестами то есть специальный

1:06:32

класс уже подготовлен необходимо использовать просто gunit Framework 5 не

1:06:38

надо исполнить J просто базовый набор тестов уже все зависимости в памятники

1:06:44

подключены за это беспокоиться не надо вот на эту задачу выделяется 20 минут

1:06:52

времени всего-то сессия 30 Если есть какие-то вопросы по этой задачи

1:06:59

необходимо или хочется прояснить Сейчас самое время их

1:07:07

Ну так Ну пока вроде Вопросов нет хорошо

1:07:14

не супер сложно это больше про логику о том как кот формировать Давай тогда

1:07:20

начнем У тебя собственность 20 минут на решение этой задачи и исключительно покрытие тестов

1:07:26

мой совет сконцентрируйся на решение значит не пытается сделать супер крутой код для начала реализую чтобы он

1:07:34

собирался и потом Выполни собственно напиши тесты

1:07:39

сценарий сценарий уже перечислены то есть на случай база случае физ базы

1:07:45

случай аргументы никакие другие кейсы предусматривать надо 4 тест сценария

1:07:53

который выполняется успешно проходят готовы приступать

1:08:00

время пошло Если будут какие-то вопросы спрашивают

1:08:07

[музыка]

1:08:22

[музыка]

1:09:02

[музыка]

1:09:18

[музыка]

1:09:36

[музыка]

1:09:53

[музыка]

1:10:09

[музыка]

1:10:48

уверена что не хочешь взять asertions Gun

1:10:56

говоря не помню не понимаю о чем речь но ты пытаешься использовать asert как

1:11:02

ключевое слово Java Но это Но это не фреймвор это же

1:11:11

тебе необходимо использовать

1:11:17

да [музыка]

1:11:42

[музыка]

1:12:19

[музыка]

1:12:44

[музыка] Здесь вроде все норм Давай попробуем его

1:12:53

выполнить Все норм искал но его необходимо еще

1:12:59

выполнить что нам не хватает

1:13:09

Как как

1:13:17

запущу посмотрим что же происходит

1:13:26

это время

1:13:39

побежал и один тест провалился причем мы не знаем какой на самом деле у

1:13:47

тебя четыре сценария Я даже понять не имея какой конечно тыкнул скажет первый тест провалился тут у меня

1:13:55

сразу к тебе вопрос Тебе не кажется что должно быть четыре

1:14:00

сценария а не один сценарий который содержит 4 вложенных сценария

1:14:07

Давай исправим для начала Давай тогда Сначала это исправим

1:14:26

[музыка]

1:14:32

Окей давай попробуем это все удалить

1:14:40

смысле не удалить выполнить запущу еще разочек и посмотрим

1:14:47

выполняться выполнится ли все тесты либо отвалился только один

1:14:59

и выполнился только один тест хорошо исключением Судя по всему по номеру

1:15:06

но у нас не выполнился тест 2 и тест номер один самое интересное если

1:15:14

посмотреть в эти тесты то содержимое массива Вроде один к одному

1:15:20

в чем проблема

1:15:25

но видно что это разные Шкоду что массивы разные содержимое

1:15:32

одинаково сравнивает ссылки получается да как можем исправить эту проблему через

1:15:40

для этого есть какие не обязательно что ты точно наверняка

1:15:46

знаешь ответ на какие твои идеи или как Какие

1:15:51

твои действия скорее

1:15:58

если я не ошибаюсь У него был метод сорт и клосс правильно Да да он так называть

1:16:09

просто удали все выражение и напиши слово Он тебе подскажет не просто лучше

1:16:23

поменять где это необходимо и Давай попробуем выполнить

1:16:31

Давай я запускаю

1:16:42

[музыка]

1:16:55

и тесто выполнились не выполнился только последний Он говорит о том что массив отличается

1:17:02

Третий да то есть ты написала Фис бас 15

1:17:08

если мы перейдем в класс основной здесь что-то не то просто буквы

1:17:19

Давай еще раз перезапустим

1:17:27

кстати почти 20 минут как раз 20 минут уже подходит концу

1:17:36

и тесты выполнить супер Поздравляю к тестам Никаких претензий нет Я не говорю

1:17:42

о том что имена именование неправильно какие-то вещи с моей точки зрения может быть не Perfect

1:17:48

вообще здесь об этом речь не идет ты написал неплохие сценарии ты прошла путь

1:17:56

класть все сценарии в одну пачку потом ты их разбила заменила дефекты сценарий

1:18:02

сравнить массив действительно сравнивать его посылки по ссылке а не по значению

1:18:10

соответственно внутри которой он хранит и это все решила но у меня есть как бы

1:18:15

такой вопрос на улучшение У нас есть еще чуть-чуть времени минут пять может быть

1:18:20

чуть больше но все-таки смотрю вот этот код на реализацию метода Если

1:18:27

возможность улучшить этот код чтобы избавить нас от такого количества If и

1:18:32

допустим избавиться от одного их при этом всем Чтобы у нас реализации покрывались кратно 3 кратно 5 и кратным

1:18:40

временем и 5 Как ты думаешь Как можно было бы этот

1:18:47

код улучшить так чтобы он действительно продолжил выполняться тесты проходили но

1:18:52

при этом все убрать одно условное выражение то есть один их и еще один

1:18:58

есть какие-то мысли [музыка]

1:19:11

Нет пока ничего не могу сказать А что если прибегнуть к стринг билдеру

1:19:20

не улавливаю честно говоря но тут даже не про string Builder это

1:19:28

это все удобный апм что если мы скажем Подумаем в ту сторону что

1:19:35

значение может быть кратно 3 если накратно 3 мы добавляем

1:19:42

значение может быть также

1:19:48

но если мы сказали что пункт 1 и 2 уже

1:19:54

выполнили Разве мы не прошли пункт 3 когда и трем и 5 оно кратно одновременно

1:20:00

и соответственно возникает на мысль что этот код можно улучшить чтобы избавиться от верхнего стоит самый большой и

1:20:08

который по сути повторяется может быстренько изменить этот код не прибегая экстрим билдеру Можешь

1:20:14

использовать если хочешь использовать так или иначе результат должен быть

1:20:21

такой же обновить код чтобы избавиться от одного условного выражения при этом тесты все были пройдены

1:20:56

[музыка]

1:21:04

нравится не самая просит

1:21:12

не самое правильное решение

1:21:17

[музыка]

1:21:35

Я думаю мы можем попробовать исправить этот смысле выполнить этот код еще раз в

1:21:41

наших тестовых сценариях Давай нет вернуть нужно

1:21:47

в любом случае Нам получается нужно Здесь проверять третьим If есть ли у нас

1:21:54

[музыка] не пустой ли у нас значение стринга Окей хорошо соответственно

1:22:02

третий и возвращается но мы говорим о ефе который связан в

1:22:10

данном случае с нашими калькуляции

1:22:17

Но это другая это другой это другое

1:22:33

[музыка]

1:22:49

Хорошо хорошо мы действительно создали другой If но

1:22:55

это уже другой действительно и в которой предусматривает исключение допустим Если бы мы не возвращали

1:23:00

не выбрасывали исключение а возвращали бы пустой массив предположим то этого бы

1:23:06

действительно здесь избежать Нам действительно как-то нужно идентифицировать с другой стороны мы убрали ненужную калькуляцию Это куда

1:23:13

важнее потому что это абсолютно другое уже уровень Давай попробуем выполнить этот код

1:23:19

и посмотрим работает ли он

1:23:32

Ладушки получилось поздравляю молодец и закончили эту задачу

Отзыв о собеседовании

1:23:41

[музыка] и в целом мы подошли к концу Давай немножко поговорим пару минут с тобой

1:23:48

как ты вообще себя ощущаешь лично оцениваешь после прохождения этого интервью

1:23:54

Ну понятно что есть что учить что доучивать что зарабатывать и в первую

1:24:01

очередь теорию по базам данных наверное самое такое больное получилось А так-то ну не

1:24:09

ужас ужас Я полностью с тобой согласен я полностью с тобой согласен я хочу от

1:24:15

себя лично добавить что мне кажется ты просто себя недооцениваешь У тебя кажется ты не знаешь на самом деле

1:24:22

разбираться то все ты знаешь люди у нас не вылетают как роботов сразу мгновенно

1:24:29

ответа и быстро Поэтому иногда нужно подумать иногда мы что-то не слышали как оно звучит но знаем если

1:24:37

пообщаться чуть-чуть на парочку Катков глубже то сразу становится

1:24:43

это просто жадная не хочешь никому ничего рассказывать ни с кем делиться своими знаниями а вот с точки зрения

1:24:51

психологического фактора то есть давление стресс который ты испытываешь на интервью

1:24:57

осталось ли у тебя преодолел ли станет тебе легче в

1:25:03

следующих интервью поверил ты себя больше что не так это и страшно на самом деле что попробовал один раз избавилась

1:25:09

от этих страхов и дальше просто уже течешь вперед Ну я думаю определенно будет легче тут

1:25:17

вопрос стресса наверное А тот который мы сами на себя нагоняем это самое сложное

1:25:24

и страшные в общем-то я думаю да будет в любом случае легче

1:25:31

потому что первый раз самый страшный особенно когда

1:25:40

я понимаю Понимаю не переживай на самом деле Действительно это все чем чаще хостим легче это просто обычная

1:25:47

тренировка для этого собственно мы делаем такие интервью и я думаю что и

1:25:52

надеюсь что тебе оно очень сильно поможет и в будущем потому что первое ты преодолел какие-то определенные барьеры

1:25:57

которые вероятно Тебе может быть казались может действительно они были Ты знаешь Теперь что тебе необходимо

1:26:03

реально подтягивать Какие знаки что Не боги горшке обжигают мы все в чем-то

1:26:10

хромаем Я не исключение тоже порой забывать старые вещи допустим когда мы с тобой обсуждали рефлексия ей богу не

1:26:17

помню как называются аннотации отвечающие за решению полюсе в них я

1:26:23

знаю как они работают понятно с технической точки зрения могу объяснить но я не помню как называется

1:26:29

немножко начинаешь уже разбираться О'кей всё-таки знаешь что это такое поэтому

1:26:35

важно понимать что во время интервью не столько важно как зубов отскакивает

1:26:41

термины сколько Действительно ты понимаешь специфику как она работает Мне

1:26:47

кажется у тебя как раз все хорошо тебе просто нужно наловчиться А скажи вот

1:26:53

следующим Что ты будешь делать Вот что ты сразу первым делом будешь подтягиваться тоже сказал что базу

1:27:00

данных но вероятно но вероятно может быть какая-то конкретная тема которую ты точно все первым делом тот пойдешь уже

1:27:07

через 15 минут с наше интервью и начнешь

1:27:13

а Наверное это вот вся теоретическая часть История это вопрос транзакции в

1:27:20

том числе Да я думаю Хорошо хорошо начало потому

1:27:25

что база такие широкие Мне нравится книга есть такой Джош Джордж Если не

1:27:31

ошибаюсь это очень старый дед не заживлен еще книга семидесятого начала

1:27:37

80-х годов книга он описывает теорию революционных баз данных толстая книга

1:27:43

на тысячу страниц Вот такая супер крутая если тебе будет интересно заглянем потому что там очень много ответов на

1:27:49

вопрос [музыка] мотивационные вопросы почему что-либо существует в революционных базах данных

1:27:57

как оно работает главное как это воспринимает все время мне эта книга перед собеседованием давным-давно очень

1:28:03

сильно помогал Я оттуда вытаскивал гораздо больше важный информации потом смог на ее основе строить свои ответы

1:28:10

чем какие-то топики в интернете это вопрос кому что больше нравится читать

1:28:17

какие-то книги для задротов Или допустим какие-то короткие топики которые наводят

1:28:23

на мысль Ну что ж книгами для задротов перед

1:28:29

собеседованием что их много и они огромные это невозможно Все засунуть в язык коротких промежуток

1:28:36

времени это это правда Это действительно правда но вот мне как-то дизла я

1:28:42

почему-то интуитивно всегда читал именно те топики которые мне задавали на собеседование

1:28:52

это знаю другой ничего не знаю мне кажется прихожу меня именно об этом спрашивает мне повезло

1:28:58

доля везением интервью тоже особенно старте карьеры тоже играет немаловажную

1:29:04

роль Я думаю можем закончить нашу сессию хочется тебе отдельно сказать спасибо за

1:29:12

свое время за твои желания поучаствовать в интервью главное выполнить задачи проверить себя

1:29:18

первую очередь это исключительно тебя важно потому что на этом ты сможешь строить свои дальнейшие шаги и конечно

1:29:26

же прожила тебе удачи грядущих технических интервью поиска работает чтобы быстрее появилась И ты

1:29:33

начала перформить развиваться как it-специалист Спасибо тоже Огромное спасибо было супер

1:29:41

полезно Хорошо тогда будем прощаться до новых встреч пока пока

Выводы

1:29:50

[музыка] вот так вот друзья и прошло тестовое техническое собеседование отдельно

1:29:56

хочется отметить что это видео уже выходит после того как Вика прошла реально технической собеседование в

1:30:02

компанию где получила положительный отзыв на позицию Junior Java Developer что хочется пожелать и всем но знайте за

1:30:10

такими успехами всегда лежит кропотливая работа Что могу сказать про век мы вместе с ней реализовывали проект Да я

1:30:17

был себе таким ментором А Вика была такая себе кто реализовывает проект под моим руководством и речь не о том что я

1:30:24

индивидуально защищаю поскольку она обучалась со мной нет я хочу просто еще раз отметить за любыми успехами всегда

1:30:30

стоит кропотливая работа Многие могут жаловаться что кому-то везет кому-то не

1:30:35

везет кто-то прошел собеседование на ша она в конечном итоге друзья как подсказывает мой индивидуальный опыт

1:30:41

лишь пот и труд все перетрут поэтому трудитесь учитесь Работайте над ошибками

1:30:46

регулярно и все получится у каждого ведь АйТи Это большой океан возможностей Куда

1:30:52

может реально залететь каждый А на сегодня это все вы были на техническом интервью на канале Джет балб С вами был

1:30:58

Макс Добрынин Подписывайтесь Наш канал ставьте колокольчик не пропускайте все самое интересное что происходит нашем

1:31:05

канале также ставьте комментарии Это позволяет нам развиваться на этих просторах Ютуба хочется отдельно

1:31:10

отметить У нас есть крутой телеграм-канал и там уже дофига подписчиков которые читают наши новости

1:31:16

наши призы участвуют в этих квизах индивидуальные посты от меня считают Ну и в целом всякую интересную всячину

1:31:22

включая технические рассуждения этих самых участников между собой До новых встреч друзья пока-пока

Русский (создано автоматически)