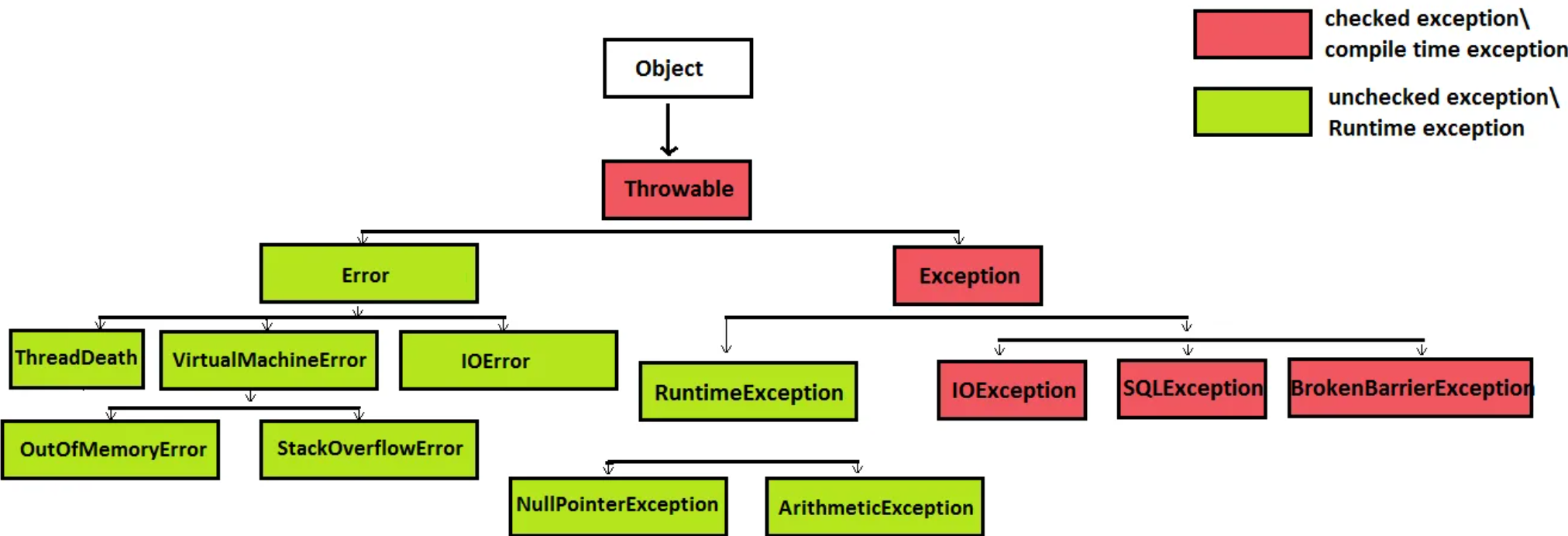
Вопросы для подготовки:

- Какова иерархия исключений.



- Можно/нужно ли обрабатывать ошибки jvm?

Можно, но разработчику не предоставлены инструменты для обработки ошибок системы и jvm

- Какие существуют способы обработки исключений?

Try catch finally throw throws

try — данное ключевое слово используется для отметки начала блока кода, который потенциально может привести к ошибке.

catch — ключевое слово для отметки начала блока кода, предназначенного для перехвата и обработки исключений.

finally — ключевое слово для отметки начала блока кода, которой является дополнительным. Этот блок помещается после последнего блока ‘catch’. Управление обычно передаётся в блок ‘finally’ в любом случае.

throw — служит для генерации исключений.

throws — ключевое слово, которое прописывается в сигнатуре метода, и обозначающее что метод потенциально может выбросить исключение с указанным типом.

- О чем говорит ключевое слово throws?

используется в сигнатуре методов для предупреждения, о том что метод может выбросить исключение

- В чем особенность блока finally? Всегда ли он исполняется?

finally – определяет блок кода, который является необязательным, но при его наличии выполняется в любом случае независимо от результатов выполнения блока try.

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

Может, если есть try и finally.

- Могли бы вы придумать ситуацию, когда блок finally не будет выполнен?

Если он недостижим, допустим если программа закрывается раньше чем он вызывается

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

Да, с java 7. Catch (IOException | SQLException)

- Что вы знаете об обрабатываемых и не обрабатываемых (checked/unchecked) исключениях?

Все исключительные ситуации делятся на «проверяемые» (checked) и «непроверяемые» (unchecked) (смотрите картинку в начале статьи). Это свойство присуще «корневищу» (Throwable, Error, Exception, RuntimeException) и передается по наследству. Никак не видимо в исходном коде класса исключения.

Throwable и Exception и все их наследники (за исключением наследников Error-а и RuntimeException-а) — checked

— Error и RuntimeException и все их наследники — unchecked

checked exception = проверяемое исключение, проверяемое компилятором.

1. Checked исключения, это те, которые должны обрабатываться блоком catch или описываться в сигнатуре метода. Unchecked могут не обрабатываться и не быть описанными.

2. Unchecked исключения в Java — наследованные от RuntimeException, checked — от Exception (не включая unchecked).

Checked исключения отличаются от Unchecked исключения в Java, тем что:

1)Наличие\обработка Checked исключения проверяются на этапе компиляции. Наличие\обработка Unchecked исключения происходит на этапе выполнения.

- В чем особенность RuntimeException?

базовый класс для ошибок во время выполнения. Относится к необрабатываемым исключениям (uncatched\unchecked). Как сказано в описании класса — это суперкласс, исключения которого могут быть выброшены во время нормальной работы JVM.

- Как написать собственное (“пользовательское”) исключение?

Создать класс который наследуется от Exception, RuntimeException…

- Какой оператор позволяет принудительно выбросить исключение?

throw

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

Нужно в сигнатуру метода добавить throws (если исключение проверяемое)

- Может ли метод main выбросить исключение во вне и если да, то где будет происходить обработка данного исключения?

Может, в JVM

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

Finally

- Что вы знаете о OutOfMemoryError?

OutOfMemoryError выбрасывается, когда виртуальная машина Java не может выделить (разместить) объект из-за нехватки памяти, а сборщик мусора не может высвободить ещё.

Область памяти, занимаемая java процессом, состоит из нескольких частей. Тип OutOfMemoryError зависит от того, в какой из них не хватило места.

java.lang.OutOfMemoryError: Java heap space -> нет места в куче (обычно из за утечки памяти)

java.lang.OutOfMemoryError: PermGen space

Данная ошибка возникает при нехватке места в Permanent области, размер которой задается параметрами -XX:PermSize и -XX:MaxPermSize.

. java.lang.OutOfMemoryError: GC overhead limit exceeded

Данная ошибка может возникнуть как при переполнении первой, так и второй областей. Связана она с тем, что памяти осталось мало и GC постоянно работает, пытаясь высвободить немного места. Данную ошибку можно отключить с помощью параметра -XX:-UseGCOverheadLimit, но, конечно же, её надо не отключать, а либо решать проблему утечки памяти, либо выделять больше объема, либо менять настройки GC.

4. java.lang.OutOfMemoryError: unable to create new native thread

Выбрасывается, когда нет возможности создать еще потоки.

- Что вы знаете о SQLException? К какому типу checked или unchecked оно относится, почему?

SQLException предоставляет информацию об ошибках доступа к базе данных или других ошибках связанных с работой с базами данных.

SQLException относится к checked исключениям, а значит проверяется на этапе компиляции. Споры об этом типе исключения идут о том, что разработчику приходится постоянно обрабатывать это исключение в коде, хотя большая часть ошибок возникает во время выполнения программы, т.е., по мнению многих, лучше бы отнести его к unchecked runtime исключениям.

- Что такое Error? В каком случае используется Error. Приведите пример Error’а

Ошибки (Errors) представляют собой более серьёзные проблемы, которые, согласно спецификации Java, не следует пытаться обрабатывать в собственной программе, поскольку они связаны с проблемами уровня JVM. Например, исключения такого рода возникают, если закончилась память, доступная виртуальной машине.

Как пример — OutOfMemoryError.

- Какая конструкция используется в Java для обработки исключений?

Try-catch-finally, try-with-resources

- Предположим, есть блок try-finally. В блоке try возникло исключение и выполнение переместилось в блок finally. В блоке finally тоже возникло исключение. Какое из двух исключений “выпадет” из блока try-finally? Что случится со вторым исключением?

Будет обработано в finally блоке. Если было выброшено два исключения — одно в try, второе в finally, то исключение в finally «проглотит» исключение выше (см. пример). Если до блока finally исключение было обработано, то мы можем получить информацию об исключении в блоке try и тем самым не потерять исключение, которое впоследствии может быть перезаписано в finally другим исключением.

- Предположим, есть метод, который может выбросить IOException и FileNotFoundException в какой последовательности должны идти блоки catch? Сколько блоков catch будет выполнено?

Общее правило — обрабатывать исключения нужно от «младшего» к старшему. Т.е. нельзя поставить в первый блок catch(Exception e) {}, иначе все дальнейшие блоки catch() уже ничего не смогут обработать, т.к. любое исключение будет попадать под ExceptionName extends Exception.

Таким образом сначала нужно обработать public class FileNotFoundException extends IOException, а затем уже IOException.