# Chapter Review Questions

Name: Aamir Khan

Date: 7-12-2022

Chapter 11 Review Questions (10 points):

1. List and describe five common examples of exceptions in Java.

**1. Memory Exhaustion**

**2. Division by Zero**

**3. Array Index out of Bounds**

**4. Arithmetic overflow**

**5. Invalid Method Parameters**

1. Explain what is a “resource leak” and how we can prevent them as a programmer.

**A resource leak occurs when the program fails to properly release a resource that’s no longer needed.**

1. In the event that no exceptions are thrown in a try block; describe the program control steps proceeding after the try block completes its execution.

**If no exceptions are thrown, all catch blocks will be skipped and control will resume from after the last catch block. If a finally block is present, that will execute first.**

1. Describe what happens in the even that no catch handler matches the type of a thrown object.

**If no catch handler matches the type of the thrown object, this causes the search for the exception to continue in the next enclosing try statement. If there are no enclosing try statements with catch blocks and the exceptions aren’t checked, a stack trace is printed and the block will terminate. However, if the exceptions are checked but aren’t caught, a compilation error occurs.**

1. Explain when it would be useful for a programmer specify a superclass type as the type in a catch block.

**When we need to deal with related types in the same way, we can specify a superclass type. It allows us to deal with related exception types in a uniform way.**

1. Explain why exception handling is an effective means for dealing with constructor failure.

**By using exception handling in a constructor, we can handle exceptions when we create objects. This allows us to effectively deal with constructor failures when exceptions are encountered in the object’s initialization.**

1. Describe a key reason for using finally blocks.

**If there’s some code that you need to execute regardless of whether an exception was thrown, a finally block is useful for this.**

1. Describe the order of execution of a program that has nested try/catch blocks in the event an exception occurs in the inner-most try/catch block.

**For each nested try block, the corresponding catch block is checked. If there’s no catch block for the nested try block, then the parent block’s catch block is checked. Each try/catch block is analyzed on its own before returning control to the statement that contains the try/catch block. If there are no corresponding catch blocks, the catch blocks of the parent block are checked one by one for a match.**

1. Explain the advantage of using catch(Exception exceptionName).

**This allows us to catch all types of exceptions.**

1. Explain why exceptions are appropriate for dealing with errors produced by methods of classes in the Java API libraries.

**We have numerous exception types that are specifically useful in dealing with errors produced by class methods in the Java API libraries. This is why it’s best to use exception handling to handle these errors, as we already have ways to handle these exceptions.**