Session: Exceptions in Java

.....

Bianca Knülle, 43373/ Despoina Sfantou, 43368/ Julia Fellmeth, 43391/ Kethella Oliveira, 43385

1. What is wrong with this code?

```
public void Foo ()  }
        try {
          lock() // lock some resource
          // open some resource
          // try to change some things
          // fool around a bit

        }
        catch (Exception e)   {
        }
        catch (ConcurrentModificationException e) {

        System.out.println(„ bad stuff going on today!“)
        }
        finally {
                return;
        }
}
```

Answer: The catch(Exception e) is an empty block and thus unnecessary. Furthermore, the ‘finally’ block returns a ‘return’.

2. What will be the output of the program?

```
public class TestException
{
   public static void badCall()
   {
      System.out.print("throwing it ");
      throw new RuntimeException();
   }
   public static void main(String [] args)
   {
      try
      {
         System.out.print("hello ");
         badCall();
      }
      catch (Exception re )
      {
```

```
        System.out.print("caught ");
    }
    finally
    {
        System.out.print("finally ");
    }
    System.out.println("after ");
  }
```

Output: hello throwing it caught finally after

3. Output?

```
public class TestException1
{
        public static void main(String [] args)
        {
            try
            {
                badMethod();
                System.out.print("A");
            }
            catch (Exception ex)
            {
                System.out.print("B");
            }
            finally
            {
                System.out.print("C");
            }
            System.out.print("D");
        }
        public static void badMethod()
        {
            throw new Error();
        }
}
```

The output of the code is that the badMethod() is called and thus a new Error is thrown.

4. Make it compile!

```
public class TestException2
        {
            class TestException extends Exception {}  // inner class
                    public void runTest() throws TestException {}

                    public void test()
                        {
                            runTest();
                        }
        }
```

Answer:

```
public class TestException2
        {
           class TestException extends RuntimeException {}  // inner class
                public void runTest() throws TestException {}

                public void test()
                   {
                      runTest();
                   }
        }
```

OR

```
public class TestException2
        {
           class TestException extends Exception {}  // inner class
                public void runTest() throws TestException {}

                public void test()
                   {
                           try{
                             runTest();
                   } catch (TestException e){
                           e.printStackTrace();
                }
        }
  }
```

5. When should you re-throw a caught exception?

If a catch block cannot handle the particular expression it has caught, you can rethrow the exception. The regrow exception causes the originally thrown object to be then rethrown.

6. A banking software detects, that a certain customer ID is not in the database.
Is this a) a system exception, b) a custom exception, c) no exception.

In most cases it has to do with a custom exception, because one, most likely, will handle it specifically,.

7. Was ist der Vorteil von Exceptions gegenüber dem Auswerten von Fehlerwerten im Return?

The main advantage of exceptions compared to the evaluation of error values in return is the fact that one can write much cleaner code. In addition, no one needs to handle an error unless one wants to. But, all in all, the main aspect of exceptions is clean code.