1 Theory

(a) In Java, an exception is an object that wraps an error event that occurred within a method and contains information on the state of the program when the error occurred and the origins of it.

(b) Errors are really big problems that can't be expected or caught. They are very disruptive and the programmer doesn't have to take care of them.
Exceptions are problems that can be expected and the programmer must take care of them.

(c) RuntimeException happens during the execution of the program and isn't necesarilly handled while CheckedException happens during the compiling of the program and must be handled in order to execute the program. RuntimeException are :

1. ArithmeticException
2. ArrayStoreException
3. ClassCastException

CheckedException are :

1. InstantiationException
2. NoSuchFieldException
3. NoSuchMethodException

(d) We can use a try-catch block which deals with the exception directly where it happens or we can use throws which simply sends the exception to another method where it might be handled or thrown again.

(e) We use try where the compiler could raise an error and then catch to tell the compiler what he must do in case it can't compile the try block.

(f) We can have multiple exception types in a catch clause separated by —.

(g) finally is used to tell the compiler what do to do whether an exception is raised or not. It is mostly used to free resources.

(h) Throw keyword is used in the method body to throw an exception, while throws is used in method signature to declare the exceptions that can occur in the statements present in the method. Throws can also declare multiple exceptions while throw handles only one.

```
void divide(int x, int y){
        if(y == 0)
           throw new ArithmeticException("An integer
           should not be divided by zero!!");
        else
           return (x/y);
   }
```

```
void divide(int x, int y) throws ArithmeticException{
    return (x/y);
    }
```

(i) We should try to catch specific exception types that could happen, and not every single type possible.

(j)

(k)

(l)

(m) We need to implement AutoCloseable. This way the program can automatically free the resources, which a programmer tends to forget and it also makes the code uglier.

(n) When we create classes we could also want to create exception classes to handle errors that could happen in the respective classes.

3 Debugging

(b)
```
Error:(16, 9) java: unreported exception java.io.IOException; must be
caught or declared to be thrown
```

(c) IllegalStateException is a RuntimeException so we can compile without handling the error, but IOException is a