# Lab: Inheritance

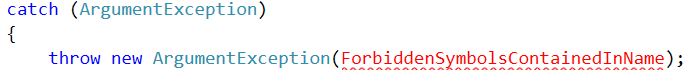
This document defines the lab overview for the ["C# OOP Basics" course @ Software University](https://softuni.bg/c-sharp-basics-oop). Please submit your solutions (source code) of all below described problems at the end of the course at [softuni.bg](https://softuni.bg/trainings/1375/java-basics-oop-june-2016).

# Introduction

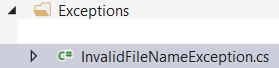
After we replaced "**display exception**" with **throw new Exception** everywhere in the **Encapsulation** lab, now it's time to make our own **Exceptions** and throw them instead of the ones we have by default. This way we can set the messages of these Exceptions in the body of the class.

# Making Exceptions for the IOManager, Tester and DownloadManager

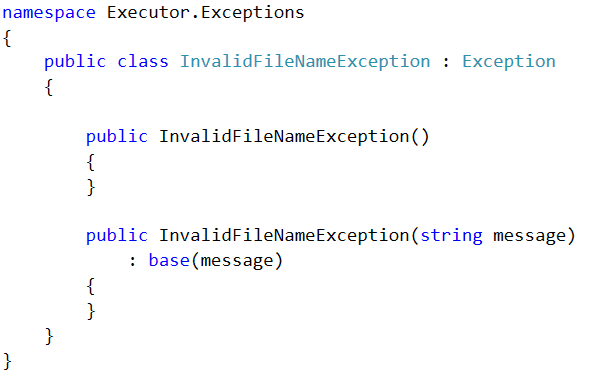
Instead of catching an ArgumentException and re-throwing it, with the corresponding message in the **CreateDirectoryInCurrentFolder** method we can make our own.



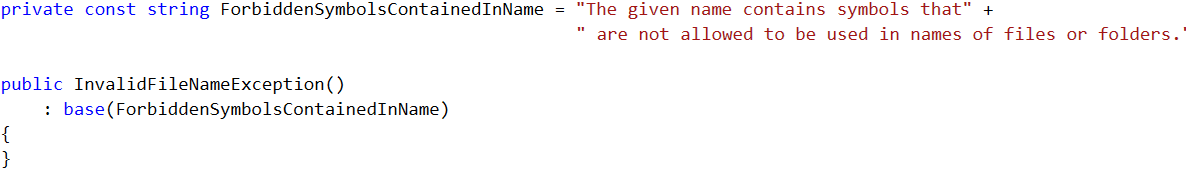
First create a folder "**Exceptions"** and a class in it called "**InvalidFileNameException"**:



Make the class **inherits the class** **Exception** and create two constructors - one with no parameters and the other with one String parameter:

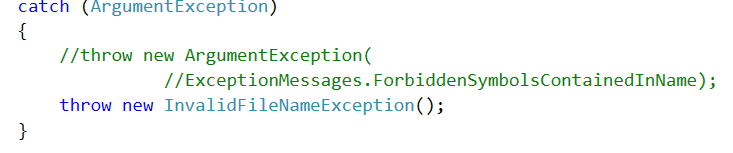


The first one should call the base class' constructor and pass it a default message. Take for that message the one from the **ExceptionMessages** class. You can also move the constant field to this class so it is decoupled (independent from other classes).



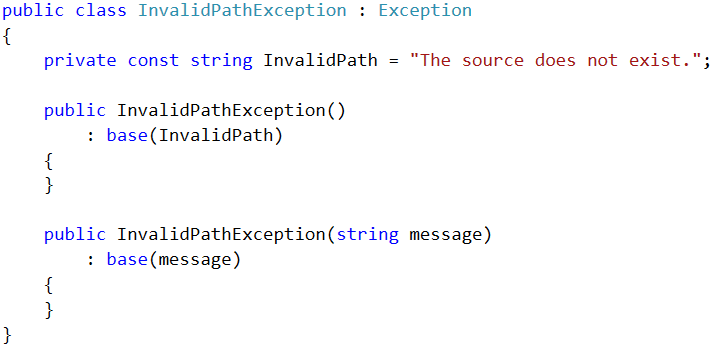
Leave the second constructor as it is. It can be used if we want to pass a different message when creating a new **InvalidFileNameException.**

Now replace the throwing of ArgumentExceptionwith our own more specific InvalidFileNameException**.**



As you can see there is no need to pass a message to the constructor because we know the default message we set is just what we need.

For the **ChangeCurrentDirectoryRelative** and **ChangeCurrentDirectoryAbsolute** methods - we can make one common exception - **InvalidPathException**. First delete the **InvalidDestination** constant from the **ExceptionMessages** class. You will see only one compile time error in the **IOManager**. We will fix it by creating a common exception.

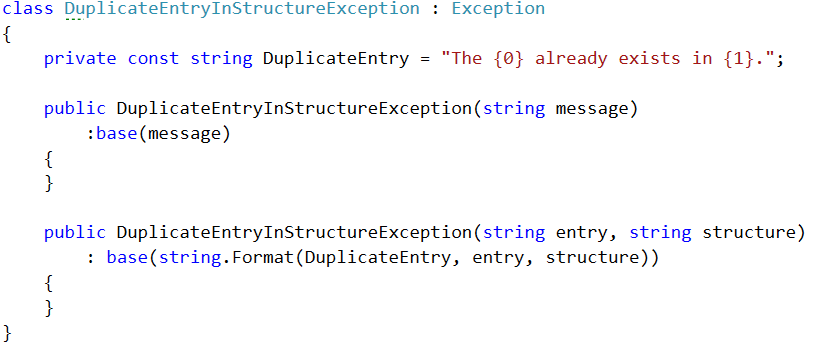


Now throw this new exception in the two methods mentioned above and everywhere else we used to throw an Exception with the **InvalidPath** or **InvalidDestination** messages.

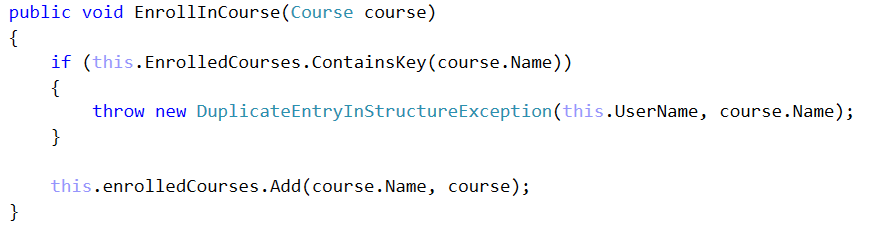
# Making Exceptions for the Models package

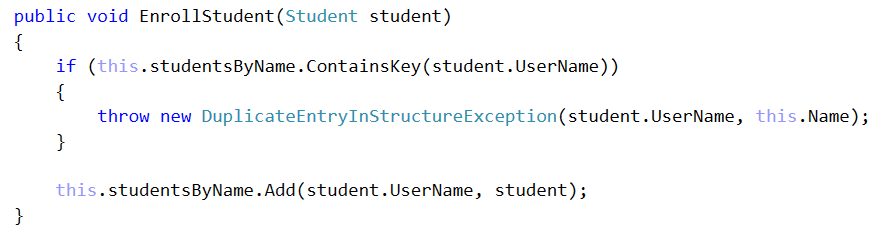
First go in the **Course** class - here we can make a more specific exception for the field **name's setter.** We can call it **InvalidStringException**. So do just that like we did in Problem 1 and put as default message the **NullOrEmptyValue**. We can use this exception for the **Student** class' **name setter**.

Next we can make an exception for the **enrollStudent** and **enrollInCourse** respectively in the **Course** and **Student** classes. This will be a more complex exception because it will accept 2 parameters - **entryName** and **structureName,** thus we will call it **DuplicateEntryInStructureException.** For the message use the same **StudentAlreadyEnrolledInGivenCourse,** but rename the constant to **DuplicateEntry**:



Now just throw the exception in both of the methods but be careful for the order of the parameters.





The last method we need to make an exception for this package is the **setMarkOnCourse.** Make a **KeyNotFoundException** following the same pattern and then throw it in the correct place.

# Making Exceptions for the Repository package

In this package the exceptions are mostly okay but you can make some of your own if you want.

Congratulations you are done with the lab exercise for **Inheritance**.