EXCEPTION HANDLING

Exercise 9

Akademia Finansów i Biznesu Vistula
Studia globalnych możliwości

mgr inż. Dominik Bielecki
email: d.bielecki@vistula.edu.pl

TASK 1.

Please create a method that will generate a **NullPointerException**.

The method should be called in a separate class in the **main**() method, and captured with a **try** and **catch** block.

In the **catch** block, please write some message for the user and call the **printStackTrace**() and **toString**() methods on the exception object.

TASK 2.

Please create a method (in the class) that depends on what value it gets as an argument throws an **ArithmeticException** or **ArrayIndexOutBoundsException** or it will not throw any exception.

Each of the above exceptions is to be handled by a separate clause **catch**, we additionally put a **finally** clause in the code that informs user that it is finally block happend.

Everything should be run in a separate class with the **main**() method.

TASK 3.

Please create an **Account** class with fields: **owner**, **balance** (**int** type) and **accountNumber**.

This class also implements: a constructor consisting of all parameters, getters and setters and a method simulating the execution of a transfer with the possibility of throwing an exception if the transfer is greater than the value of funds on the account (defined when creating the object).

The exception is a type that you create yourself, for example named **NotEnoughMoneyException**.

In the other class with the **main**() method, you will create an **Account** class object and simulate throwing an exception by executing a method simulating a transfer with a value greater than the value of the available funds. Additionally, please add a **catch** that will catch all other exceptions and a **finally** clause that will return the current account balance.

TASK 4.

Please create three new types of exceptions.

Write a class with a method which takes some value in the parameter and depending on this value throws one of those exceptions.

In the **main**() method, call this method in a **try** block, with one **catch** clause that will be able to catch any of the created exceptions (use OR statement).

TASK 5. (1)

In the next example, please take advantage of the fact that exceptions can be logged using the mechanism from the java.util.logging library (it is necessary to add an import).

The static method **Logger.getLogger**() returns the Logger object associated with the arguments of the String type (usually it represents the name of the package and class that the logging relates to) that prints the information on the standard diagnostic output(System.err).

The easiest way to write to the log is by calling it that corresponds to the level of message logging, we will use the **severe**() method for this. In the message, please include the call stack that led to the exception. The problem is that **printStackTrace**() does not return a **String** object. To get such an object, we need to use the overloaded version of **printStackTrace**() taking an object argument class **java.io.PrintWriter** (more about this will be in the next lecture). If we fill the constructor of the **PrintWriter** with the argument **java.io.StringWriter**, we get the value from which the method can be extracted the **String** object.

to be continued on the next slide ...

TASK 5. (2)

Moving on to the task, please create two exception classes using the description above, my first class looks like this:

```
6   class MyException1 extends Exception {
7       private static Logger logger = Logger.getLogger("LoggingException");
8       public MyException1() {
9           StringWriter trace = new StringWriter();
10          printStackTrace(new PrintWriter(trace));
11          logger.severe(trace.toString());
12      }
13  }
```

In a separate class create two static methods, the first one declares that it can throw our first and also second exception, but throws only the first one, and the second one declares and throws the second one.

In the **main**() method, we declare two **try** blocks for each of the methods and one **catch** that catches all exceptions.

THANK YOU

Więcej na:

www.vistula.edu.pl

Akademia Finansów i Biznesu Vistula
ul. Stokłosy 3
02-787 Warszawa
(obok stacji metro Stokłosy)