Exception handling

Catching and throwing exception objects
predefined exception classes

Catching and throwing exception objects

Principle: do not handle the exceptions directly inside the code , but beside the code block in question

- inside the code block, we generate (throw) an exception object
- aside from the code block, we handle (catch) this exception object in a dedicated block
- the exception object is passed as an argument to this block:

```
try
{
   code which may throw some exceptions
}
catch (exception_class1 e1)  {  handle exception e1  }
. . . . . .
catch (exception_classn en)  {  handle exception en  }
```

if not caught immediately, an exception may "propagate"
to another enclosing **try**
or to another calling method until it is caught
or eventually up to the **main** method if it is not caught at all
→ printing of an error message and exit from the program

exception objects:

Exception is the generic class of all the exceptions

→ constructor **Exception()** or **Exception(error message string)**

method **e.getMessage()** returns the error message string

several predefined exceptions, all subclasses of **Exception**,
are thrown automatically by the language

examples:

```
try
{
   int d = 0;
   int a = 100 / d;
}
catch (ArithmeticException e)
{
   System.out.println(e.getMessage);
   System.exit(0);
}
```

```
double[] a = new double[10];
int index;
index = 10;
try
{
   a[index] = 1.0;
}
catch (IndexOutOfBoundsException e)
{
   System.out.println("index value " + index + " out of bounds");
   System.exit(0);
}
```

or we may define and explicitly throw ourselves our own exceptions

defining and throwing exceptions:

subclass the **Exception** class
in general, just define constructor(s) in this subclass

```
class MyException extends Exception
{
   public MyException() { super (); }

   public MyException(String s)  {  super (String s);  }
}
```

then, wherever we want to throw an instance of the **MyException** class,
we insert within the **try** block of code:

```
throw new MyException();
```

or

```
throw new MyException("error message");
```

a method may be called within the **try** block of code
and this method may throw an exception susceptible
to be caught at the end of the **try** bloc

→ **throws exception** in header of method required
(except for certain very current predefined exceptions):

```
modif return_type method_name (args) throws exception
{ ……. }
```

or, if several exceptions susceptible to be thrown:

```
modif return_type method_name (args) throws exception1 , … , exceptionn
{ ……. }
```

examples:

```
void f() throws MyException
{
   throw new MyException("MyException in f");
}

try
{
   f();
}
catch (MyException e)
{ System.out.println(e.getMessage()); }
```

→ MyException in f

```
void f() throws MyException
{
   g();
   throw new MyException("MyException in f");
}

void g() throws MyException
{
   throw new MyException("MyException in g");
}

try
{
   f();
}
catch (MyException e)
{ System.out.println(e.getMessage()); }
```

→ MyException in g

predefined exception classes

"**throws exception**" in header of method is required only
for the exception classes marked with "(throws)"

java.lang

(throws) Exception : String getMessage()
ArithmeticException
NullPointerException → attempt to access a member of a **null** object
IndexOutOfBoundsException
ArrayIndexOutOfBoundsException
StringIndexOutOfBoundsException
ArrayStoreException → attempt to store the wrong type of object in an array
NegativeArraySizeException
ClassCastException → invalid cast of an object to a type of which it is not an instance
IllegalArgumentException → illegal argument to a method
SecurityException → operation not permitted for security reasons
(throws) ClassNotFoundException → a class to be loaded could not be found
(throws) NoSuchFieldException
(throws) NoSuchMethodException
(throws) CloneNotSupportedException → **clone()** called for an object that does not support it
(throws) IllegalThreadStateException → a thread is not in the appropriate state for an attempted operation
(throws) InterruptedException → the thread has been interrupted
(throws) NumberFormatException

java.io

(throws) IOException
(throws) EOFException
(throws) FileNotFoundException
(throws) InterruptedIOException → **e.bytesTransferred** number(int) of bytes read or written before the interruption

java.awt

(throws) AWTException
(throws) IllegalComponentStateException → for example, component not added to a container yet or currently hidden