```
{ $result = $firstnumber / $secondnumber; }
// other code with exceptions }
catch(zeroException $e) {
        echo $e->errorMessage();
}
catch(Exception $e) {
        Echo $e->getMessage();
}
```

The zeroException class *extends* the class Exception. The extends keyword is used to inherit all of the functionality of the Exception **class**. *Inheritance* is another key component of object-oriented programming (along with encapsulation and polymorphism). A *child* **class** (like zeroException) can inherit all the properties and methods of its *parent* class (Exception). The child class then can add methods (such as the function errorMessage) specific to the class. Since zeroException inherited Exception, it is treated the same as any other exception. The zeroException can be thrown (throw new zeroException("Zero")) and it can be caught (catch(zeroException $e)).

> *Program note—Programmer-created exception classes inherit from Exception. Thus, all the functionality of the Exception class is available from within any new exception class.*
>
> *Class zeroException extends Exception { }*
>
> *The previous code creates a valid new zeroException class with no new methods.*
>
> *catch(zeroException $e) { echo $e->getMessage(); }*
>
> *This catch block will be called by the new exception and display the exception message generated by the Exception class.*

For each exception or error class that is created and thrown, there must be a catch block to catch the exception or error. In the example, there are two catch blocks; one catches the zeroException and the other catches any other exceptions that might occur. Just like the previous example using a switch default or if else statement, you should always have the last catch blocks handle any remaining exceptions or errors. If the generic catch block is listed first, all exceptions would be caught by that block and not the specific block for the exception.

As stated, the developer should make every attempt to keep the application from crashing. Errors, however, are designed to display messages and shut down programs with an error code (what you consider to be "crashing" the program). Before PHP 7, in some cases, you could override this functionality by creating a method that will handle errors.

```
function errorHandler($severity, $message, $file, $line) {
    throw new errorException($message, 0, $severity, $file, $line); }
set_error_handler('errorHandler');
// set_error_handler() doesn't work with all fatal errors, some can't be thrown as Exceptions.
try { trigger_error( "User Error", E_USER_ERROR);
  }
catch(errorException $e)
{ echo $e->getMessage(); }
catch(Exception $e)
{ echo $e->getMessage(); }
// Code placed here would execute after an error with this handler. It would not execute if
// there was not a handler.
```