Exceptions

This lesson gives a brief introduction to exceptions.

WE'LL COVER THE FOLLOWING

- Exceptions

Exceptions

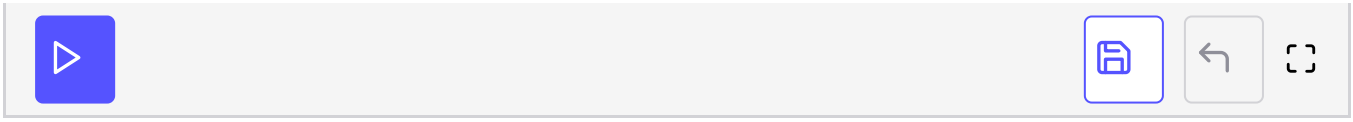
Unexpected situations are part of programs: user mistakes, programming errors, changes in the program environment, etc. Programs must be written in ways to avoid producing incorrect results when faced with exceptional conditions.

Some of these conditions may be severe enough to stop the execution of the program. For example, a required piece of information may be missing or invalid or a device may not be functioning correctly. The exception handling mechanism of D helps with stopping program execution when necessary and recover from unexpected situations when possible.

As an example of a severe condition, we can think of passing an unknown operator to a function that knows only the four arithmetic operators.

```
import std.stdio;
import std.format;

void main() {

    int first = 16;
    int second = 8;

    string operator = "^";
    switch (operator) {

    case "+":
        writeln(first + second);
        break;

    case "-":
```

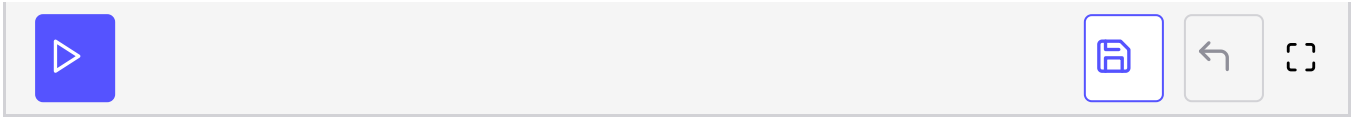
```
        writeln(first - second);
        break;

    case "x":
        writeln(first * second);
        break;

    case "/":
        writeln(first / second);
        break;

    default:
        throw new Exception(format("Invalid operator: %s", operator));
    }
}
```



The `switch` statement above does not know what to do with operators that are not listed on the `case` statements, so it *throws* an *exception*.

There are many examples of thrown exceptions in Phobos. For example, `to!int`, which can be used to convert a `string` representation of an integer to an `int` value throws an exception when that representation is not valid:

```
import std.conv;

void main() {
    const int value = to!int("hello");
}
```



Exception thrown while using to!int

The program terminates with an exception that is thrown by `to!int`. `std.conv.ConvException` at the beginning of the message is the type of the thrown exception object. We can tell from the name that the type is `ConvException`, which is defined in the `std.conv` module.

In the next lesson, we will explore the `throw` statement.