writeln and write

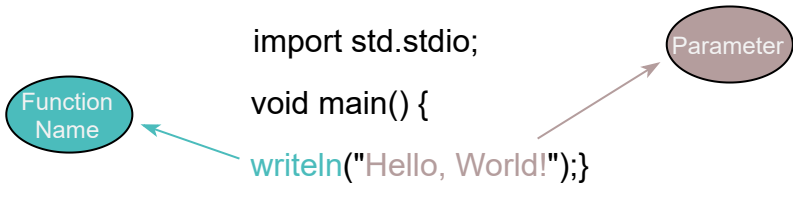
This lesson explores the difference between writeln and write with the help of a coding example.

WE'LL COVER THE FOLLOWING

Functions and parameters

In the previous lesson, we learned that `writeln` takes a string within parentheses and prints the string. Before we move onto an example code to understand the difference between working of `writeln` and `write`, let's get a little insight into some programming terms.

- **Functions:** The parts of a program that perform a task are called **functions**.
- **Parameters:** A parameter is information provided to functions in order for the functions to complete their work.
  The act of giving information to functions is called *passing parameter values* to them. Parameters are passed to functions within parentheses, separated by commas as illustrated below.



Function and parameters

Note: The word parameter describes the information that is passed to a

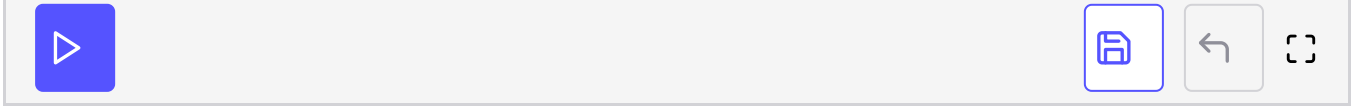
function at the conceptual level. The concrete information that is actually passed during the execution of the program is called an **argument**. Although not technically the same, these terms are sometimes used interchangeably in the software industry.

Difference between `writeln` and `write`

Now that we are familiar with the terms *function* and *parameter*, let's see how `writeln` is different from `write`.
`writeln` can take more than one argument. It prints them one after the other on the same line:

```
import std.stdio;

void main(){
    writeln("Hello, World!", "Hello, fish!");
}
```



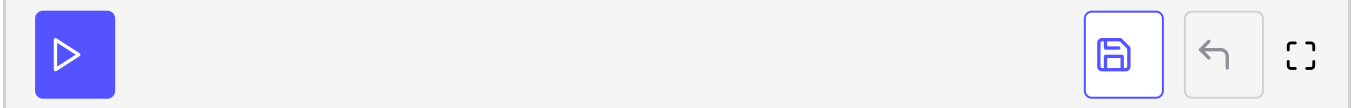
writeln code example

Sometimes, not all of the information that is to be printed on the same line may be readily available to be passed to `writeln`. In such cases, the first parts of the line may be printed by `write`, and the last part of the line may be printed by `writeln`.

`writeln` takes the cursor to the next line; `write` stays on the same line:

```
import std.stdio;

void main(){
    // Let's first print what we have available:
    write("Hello,");
    // ... let's assume more operations at this point ...
    write("World!");
    // ... and finally to print a blank like:
    writeln();
}
```



write code example

Calling `writeln` without any parameter merely completes the current line, or if nothing has been written, it outputs a blank line.

Comment lines

Lines that start with `//` are called **comment lines**, or **comments** for short. A comment is not a part of the program code in the sense that it doesn't affect the behavior of the program. Its only purpose is to explain what the code does in that particular section of the program. The audience of a comment is anybody who may be reading the program code later, including the programmer who wrote the comment in the first place.

In the next lesson, you will find a simple challenge to test your understanding of the concepts covered so far.