Outputting Values in Programming

One of the most vital (and simple) tools in any programming language is being able to output a value to the user and even to yourself (the programmer).

Why Do We Need to Output Values?

Outputting a value to the user is vital especially if the program you are coding is a game and you want some user interactions, and more so if you want to keep the user engaged!

For example, an output could be graphics or simply a response to a user answer. Imagine we were making a quiz: we show the user a question, they respond, and we need to tell them if they are right or wrong.

Both the question and the result (right or wrong) need to be output to the user.

Outputting a value to yourself may seem more strange…however, a key part of programming is testing your own code. It would be no good if you coded an entire game, only for it to be full of errors.

For that reason, most programming environments let you write code and view how it runs side by side. And outputting certain variables at points within the program let you see what their values are and you can see if this is what you are expecting or not.

How Do We Output Values?

In general programming terms, we often refer to a line of code that outputs a value as a ‘print statement’.

The easiest language to write a print statement in is Python. They make writing print statements very easy to remember…here’s an example:

```
print("Hello, World!")
>> Hello, World!
```

As you can see, Python has an inbuilt function called ‘print()’, and you just need to put whatever you want to output within the brackets.

You may notice that the quotation marks (“”) aren’t outputted to the user. This is because the quotation marks are simply used to indicate that the message is a string (a collection of characters).

We can output a number as well:

```
print(123)
>> 123
```

Note: a number doesn’t require quotation marks.

We can also output the value of a variable:

```
myName = "Olivia"
print(myName)
>> Olivia
```

As you can see, the value myName doesn’t require quotation marks in the print statement, as otherwise the text “myName” would actually be output, rather than the value of the variable (“Olivia”).

On the other hand, in JavaScript, there isn’t really a print statement that can be used for both testing and interacting with the user.

Instead, for testing, we can ‘log’ a value to the ‘console’. The console is a simple environment that cannot be seen by the user. You can see an example here, jsconsole.com Therefore, this makes it very useful for testing purposes.

Here are the same examples from above in JavaScript:

```
console.log("Hello, World!")
>> Hello, World!

console.log(123)
>> 123

const myName = "Olivia"
console.log(myName)
>> Olivia
```

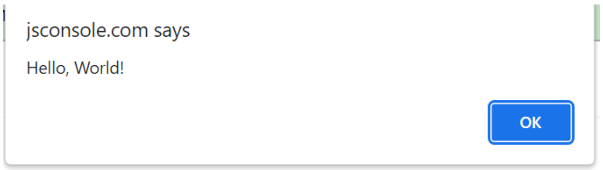
The function ‘console.log()’ works in exactly the same way as Python’s ‘print()’ function.

If we want to interact with the user in JavaScript, we can use the ‘alert()’ function. This is a pretty cool function that you should test in jsconsole.com

It works exactly in the same way as ‘print()’ and ‘console.log()’, in that you just put the values to be output in the brackets:

```
alert("Hello, World!")
```

However, instead of just returning the value, we get a pop up box:



jsconsole.com says

Hello, World!

And this box will pop up for the user only!

Now you can write a very small piece of code! If you want to do some more experimenting with print statements and outputting values, you can check out [resource].