Lab: Printing

Tutorial Lab 1: Printing

1. In the text editor to your left, you will see the code below:

```
string my_var = "I am learning"; //step 1
cout << my_var; //step 2
my_var = " to program"; //step 3
cout << my_var; //step 4
my_var = " in C++."; //step 5
cout << my_var << endl; //step 6
my_var = "Hooray!"; //step 7
cout << my_var; //step 8
```

2. Click `TRY IT` to see what the code outputs.

3. Click on the ++underlined++ text below to highlight some of the important points in the code:

   - Step 1 - Declare the variable `my_var` and initialize it to the value `I am learning`
   - Step 2 - Print without a new line character by not including `<< endl;`
   - Step 3 - Add a space when starting the string to avoid printing `learningto`
   - Step 6 - A newline character is added using `<< endl;`
   - Step 8 - `Hooray!` is on its own line since the `<< endl;` command was used in step 6

4. To remove the highlighting, click here: Remove Highlighting

Lab: Variables

Tutorial Lab 2: Variables

1. Use the text editor to the left.
2. Copy the code below.

```
int one = 1;
int two = 2;
int three = 3;
int four = 4;

two = one;
three = two;
four = three;

cout << four;
```

3. `TRY IT` to see the output. Click on the ++Code Visualizer++ link below to go through the program step by step.

Code Visualizer

output : 1

Lab: Challenge

Tutorial Lab 3: Fundamentals Challenge

In the code to the left, we see that there are a series of declared and initialized variables. Use these variables along with the `cout <<` and `<< endl;` commands to print out a custom message to customers who open a chat client.

Your output should look something like this:

```
Hello! Today is Wednesday, May 4.
The current wait time is 4 minutes.
```

The pattern is as follows. The `*` indicates variables:

```
*greeting* Today is *dayOfWeek*, *month* *day*.
The current wait time is *currentWaitMinutes* minutes.
```

To test the code, first click on the `COMPILE` button. This will compile your code and turn it into a program. If your program compiled successfully, you will see the message `Command was successfully executed.` Then you can run your program by clicking on the `TEST` buttons. You will see the output of a few different test cases:

Take a look at the test outputs above. Do they look like the expected outputs below? If not, your code may need some revision.

```
Hello! Today is Monday, July 4.
The current wait time is 9 minutes.
```

```
Howdy! Today is Tuesday, December 15.
The current wait time is 2 minutes.
```

```
Greetings! Today is Friday, March 13.
The current wait time is 39 minutes.
```