A discussion of the formatted print statement

Initially we learned how to print using the print statement and the comma.

```
variable1 = “This is the story of a lovely lady who was raising tree very lovely girls.”
print (“The first line of the Brady Bunch title song is: “,variable1)
```

Using the comma for a tab never really handled the spacing well. So Python added the .format() approach.

```
variable1= “This is the story of a lovely lady who was raising tree very lovely girls.”
print(“The first line of the Brady Bunch title song is: {0}”).format(variable1)
```

which substituted the contents of variable1 into the first place holder ({0}).

In order to handle multiple variables and their spacing better, Python added the formatted print statement.

```
variable1= “This is the story of a lovely lady who was raising tree very lovely girls.”
print(f “The first line of the Brady Bunch title song is: {variable1}.”)
```

The formatted print statement allows the user to enter spaces, punctuation, and new lines at will. Consider how the this saves one from having to concatenate multiple strings in a the print statement. Rather than print(“The first line of the Brady Bunch title song is “ + variable1 + “.”) one can simply place the variable (inside of curly brackets) into the formatted print statement. The intitial “f” tells print to do the concatenation for you.