# Hands-on Lab: Static Code Analysis

Estimated time needed: **30** minutes

## Objectives

After completing this lab you will be able to:

- Install pylint package
- Run Static Code Analysis on a python program
- Check the compliance score of a python program.
- Fix common mistakes and improve the compliance score.

# Install the pylint package

1. Open a new terminal.

```
pip3 install pylint==2.11.1
```

3. This should install the pylint package.

# Create a sample python file for static code analysis

Create a new file named **sample1.py**

Copy and paste the below code into **sample1.py**

```
# Define a function named 'add' that takes two arguments, 'number1' and 'number2'.
def add(number1, number2):
    # The function returns the sum of 'number1' and 'number2'.
    return number1 + number2
# Initialize the variable 'num1' with the value 4.
num1 = 4
# Initialize the variable 'num2' with the value 5.
num2 = 5
# Call the 'add' function with 'num1' and 'num2' as arguments and store the result in 'total'.
total = add(num1, num2)
# Print the result of adding 'num1' and 'num2' using the 'format' method to insert the values into the string.
print("The sum of {} and {} is {}".format(num1, num2, total))
```

Save the file **sample1.py**

# Run pylint

- Open a terminal
- Run the below command

```
pylint sample1.py
```

- Pylint goes through every line of code and gives you a list all the non-compliant lines.
- Pylint gives you a compliance score (10 being maximum).

# Correct the mistakes identified by pylint.

- Based on the report given by pylint changes were made to this code to address the following issues.
    - Exactly one space required after comma
    - Exactly one space required around assignment
- Create a new file named **sample2.py**
- Copy and paste the below code into **sample2.py**

```
# Define a function named 'add' that takes two arguments, 'number1' and 'number2'.
# The purpose of this function is to add the two numbers and return the result.
def add(number1, number2):
    # Return the sum of 'number1' and 'number2'.
    # This line computes the addition of the two input numbers and outputs the result.
    return number1 + number2
# Initialize the constant variable 'NUM1' with the value 4.
# Constants are usually written in uppercase letters to indicate that they should not be changed.
NUM1 = 4
# Initialize the variable 'num2' with the value 5.
# This variable will be used as the second input to the 'add' function.
num2 = 5
# Call the 'add' function with 'NUM1' and 'num2' as arguments.
# The result of this addition operation is stored in the variable 'total'.
total = add(NUM1, num2)
# Print a formatted string that displays the sum of 'NUM1' and 'num2'.
# The 'format' method is used to insert the values of 'NUM1', 'num2', and 'total' into the string.
print("The sum of {} and {} is {}".format(NUM1, num2, total))
```

Save the file **sample2.py**

# Run pylint

- Open a terminal
- Run the below command

```
pylint sample2.py
```

- This will give you the compliance score.
- This time you should see the score improve.

# Your task

Improve the score in sample2.py to a perfect 10 by correcting all the issues pointed out by pylint. If cant figure out how to solve some issues it is helpful to google the pylint message.

# Congratulations!

You now know how to perform static code analysis.

## Author(s)

Ramesh Sannareddy

### Other Contributors

Rav Ahuja

**© IBM Corporation. All rights reserved.**