Dhirubhai Ambani
Institute of Information and Communication Technology

Software Engineering
(IT314)

Lab-5

Name	Student ID
Nikunj Adhiya	202001277

Static Analysis Tool:- pylint

S.No	Message Object	Expansion	Explanation
1.	C	Convention	It is displayed when the program is not following the standard rules.
2.	R	Refactor	It is displayed for bad code smell
3.	W	Warning	It is displayed for python specific problems
4.	E	Error	It is displayed when that particular line execution results some error
5.	F	Fatal	It is displayed when pylint has no access to further process that line.

Understanding of Errors:

1. **C0114/C0ll6:** Missing module docstring/function docstring
   This tells that the docstring is not specified for the modules or functions. The docstring is like a comment or some text description that is ignored by the compiler but helps other people to understand the purpose of the module/function docstring. It is a good practice to write docstrings.
2. **C0103:** Variable does not conform to snake_case naming style This error tells that the declared variable is not according to snake_case naming convention.
3. **W0621:** Redefining name from outerscope
   This error tells that the local variable name is the same as that of one of the global variables. This makes the local scope use the local version of the variable while hiding the global version of the variable.
4. **C0321:** More than one statement on a single line
   This tells that multiple statements are written in the same line. This is not a good practice as the code will be more readable when a single statement is there in a line.
5. **R1705:** Unnecessary else after return statement
   This is caused when an else statement is written after a "return" statement which is not executed anyhow.
6. **W0611:** Unused imports This is caused when there are any modules or packages that are imported but not used.

7. **C0303:** Trailing whitespace at the end of a line This is caused when there is empty space at the end of line and before a new line.This space is unnecessary.

1.REPO LINK

```
"""
Just to check
"""


def add(a: float, b: float) -> float:
    """
    >>> add(2, 2)
    4
    >>> add(2, -2)
    0
    """
    return a + b


if __name__ == "__main__":
    a = 5
    b = 6
```

Output

```
C:\Users\student\Desktop\202001277>py -m pylint add.py
************* Module add
add.py:18:0: C0304: Final newline missing (missing-final-newline)
add.py:6:8: C0103: Argument name "a" doesn't conform to snake_case naming style (invalid-name)
add.py:6:18: C0103: Argument name "b" doesn't conform to snake_case naming style (invalid-name)
add.py:6:8: W0621: Redefining name 'a' from outer scope (line 17) (redefined-outer-name)
add.py:6:18: W0621: Redefining name 'b' from outer scope (line 18) (redefined-outer-name)
add.py:17:4: C0103: Constant name "a" doesn't conform to UPPER_CASE naming style (invalid-name)
add.py:18:4: C0103: Constant name "b" doesn't conform to UPPER_CASE naming style (invalid-name)

-------------------------------------------------------------------
Your code has been rated at 0.00/10 (previous run: 0.00/10, +0.00)
```

Analysis:

All errors are true

Error types:

- **Missing final newline.**

  We have to add a new line when our code is complete.

- **Missing module docstring**

  At the start of the code we add a string(comment) which indicates the use of our programme.

- **Missing function docstring**

  At the start of the function we add a string(comment) which indicates the use of our function.

- **Snake_case naming style**

  It shows that we have to name our variable in proper format.

- **Redefined outer name**
  It shows that we give different variables the same name.
- **UPPER_CASE naming style**
  It shows that we have to name our variable in proper format.

After Improvements

code

```
"Programme for add two numbers"
def add(num1: float, num2: float) -> float:
    "Function for add two numbers"
    return num1 + num2
if __name__ == "__main__":
    N_1 = 5
    N_2 = 6
    print(f"The sum of {N_1} + {N_2} is {add(N_1, N_2)}")
```

output

```
C:\Users\student\Desktop\202001277>py -m pylint add.py

--------------------------------------------------------------------
Your code has been rated at 10.00/10 (previous run: 0.00/10, +10.00)
```

2.REPO LINK

CODE:

```
from math import pi

def arc_length(angle: int, radius: int) -> float:

    """

    >>> arc_length(45, 5)

    3.9269908169872414

    >>> arc_length(120, 15)

    31.415926535897928

    """

    return 2 * pi * radius * (angle / 360)

if __name__ == "__main__":

    print(arc_length(90, 10))
```

OUTPUT

```
C:\Users\student\Desktop\202001277>py -m pylint test.py
************* Module test
test.py:1:0: C0114: Missing module docstring (missing-module-docstring)

-----------------------------------
Your code has been rated at 8.00/10
```

Analysis:

- All errors are true

Error types:

- **Missing module docstring**

  At the start of the code we add a string(comment) which indicates the use of our programme.

3.REPO LINK

CODE:

```python
# Uses python3

def calc_fib(n):

    if n == 0:

        return 0

    if n == 1:

        return 1

    a = [0, 1]

    for i in range(2, n+1):

        a.append(a[i-1]+a[i-2])

    return a[-1]

n = int(input())

print(calc_fib(n))
```

OUTPUT

```
C:\Users\student\Desktop\202001277>py -m pylint test.py
************* Module test
test.py:13:0: C0304: Final newline missing (missing-final-newline)
test.py:1:0: C0114: Missing module docstring (missing-module-docstring)
test.py:2:0: C0116: Missing function or method docstring (missing-function-docstring)
test.py:2:13: C0103: Argument name "n" doesn't conform to snake_case naming style (invalid-name)
test.py:2:13: W0621: Redefining name 'n' from outer scope (line 12) (redefined-outer-name)
test.py:7:4: C0103: Variable name "a" doesn't conform to snake_case naming style (invalid-name)

-------------------------------------------------------------------
Your code has been rated at 4.55/10 (previous run: 4.55/10, +0.00)
```

Analysis:

- All errors are true

Error types:

- **Missing final newline.**

  We have to add a new line when our code is complete.

- **Missing module docstring.**

  At the start of the code we add a string(comment) which indicates the use of our programme.

- **Missing function docstring**
  At the start of the function we add a string(comment) which indicates the use of our function.
- **Snake_case naming style**
  It shows that we have to name our variable in proper format.
- **Redefined outer name**
  It shows that we give different variables the same name.

4.REPO LINK

CODE

```
# python3



def max_pairwise_product(numbers):

    n = len(numbers)

    numbers.sort()

    return numbers[n-1]*numbers[n-2]




if __name__ == '__main__':

    input_n = int(input())

    input_numbers = [int(x) for x in input().split()]

    print(max_pairwise_product(input_numbers))
```

OUTPUT

```
C:\Users\student\Desktop\202001277>py -m pylint test.py
************* Module test
test.py:12:0: C0304: Final newline missing (missing-final-newline)
test.py:1:0: C0114: Missing module docstring (missing-module-docstring)
test.py:3:0: C0116: Missing function or method docstring (missing-function-docstring)
test.py:4:4: C0103: Variable name "n" doesn't conform to snake_case naming style (invalid-name)

--------------------------------------------------------------------
Your code has been rated at 5.00/10 (previous run: 4.55/10, +0.45)
```

Analysis:

- All errors are true

Error types:

- **Missing final newline.**

  We have to add a new line when our code is complete.
- **Missing module docstring.**

  At the start of the code we add a string(comment) which indicates the use of our programme.

- **Missing function docstring**
  At the start of the function we add a string(comment) which indicates the use of our function.
- **Snake_case naming style**
  It shows that we have to name our variable in proper format.

5.REPO LINK

CODE

```python
def bin_exp_mod(a, n, b):
    """
    >>> bin_exp_mod(3, 4, 5)
    1
    >>> bin_exp_mod(7, 13, 10)
    7
    """
    # mod b
    assert b != 0, "This cannot accept modulo that is == 0"
    if n == 0:
        return 1

    if n % 2 == 1:
        return (bin_exp_mod(a, n - 1, b) * a) % b

    r = bin_exp_mod(a, n / 2, b)
    return (r * r) % b


if __name__ == "__main__":
    try:
        BASE = int(input("Enter Base : ").strip())
        POWER = int(input("Enter Power : ").strip())
        MODULO = int(input("Enter Modulo : ").strip())
    except ValueError:
        print("Invalid literal for integer")

    print(bin_exp_mod(BASE, POWER, MODULO))
```

OUTPUT

```
C:\Users\student\Desktop\202001277>py -m pylint test.py
************* Module test
test.py:28:0: C0304: Final newline missing (missing-final-newline)
test.py:1:0: C0114: Missing module docstring (missing-module-docstring)
test.py:1:16: C0103: Argument name "a" doesn't conform to snake_case naming style (invalid-name)
test.py:1:19: C0103: Argument name "n" doesn't conform to snake_case naming style (invalid-name)
test.py:1:22: C0103: Argument name "b" doesn't conform to snake_case naming style (invalid-name)
test.py:16:4: C0103: Variable name "r" doesn't conform to snake_case naming style (invalid-name)

-------------------------------------------------------------------
Your code has been rated at 6.25/10 (previous run: 5.00/10, +1.25)
```

Analysis:

- All errors are true

Error types:

- **Missing final newline.**

  We have to add a new line when our code is complete.
- **Missing module docstring.**

  At the start of the code we add a string(comment) which indicates the use of our programme.

- **Snake_case naming style**

  It shows that we have to name our variable in proper format.