

-  Secrets
-  ABAP
-  Apex
-  C
-  C++
-  CloudFormation
-  COBOL
-  C#
-  CSS
-  Flex
-  Go
-  HTML
-  Java
-  JavaScript
-  Kotlin
-  Objective C
-  PHP
-  PL/I
-  PL/SQL
-  Python
-  RPG
-  Ruby
-  Scala
-  Swift
-  Terraform
-  Text
-  TypeScript
-  T-SQL
-  VB.NET
-  VB6
-  XML



Python static code analysis

Unique rules to find Bugs, Vulnerabilities, Security Hotspots, and Code Smells in your PYTHON code

All rules 216

 Vulnerability 29

 Bug 55

 Security Hotspot 31

 Code Smell 101


Tags ▾



Search by name... 


Functions should not have too many lines of code
 Code Smell
Track uses of "NOSONAR" comments
 Code Smell
Track comments matching a regular expression
 Code Smell
Statements should be on separate lines
 Code Smell
Functions should not contain too many return statements
 Code Smell
Files should not have too many lines of code
 Code Smell
Lines should not be too long
 Code Smell
Methods and properties that don't access instance data should be static
 Code Smell
New-style classes should be used
 Code Smell
Parentheses should not be used after certain keywords
 Code Smell
Track "TODO" and "FIXME" comments that do not contain a reference to a person
 Code Smell
Module names should comply with a naming convention

Identity comparisons should not be used with cached typed

Analyze your code typed

 Code Smell

 Major 

 suspicious

Identity operators `is` and `is not` check if the same object is on both sides, i.e. `a is b` returns `True` if `id(a) == id(b)`.

Integers, bytes, floats, strings, frozensets and tuples should not be compared with identity operators because the result may not be as expected. If you need to compare these types you should use instead equality operators `==` or `!=`.

The CPython interpreter caches certain builtin values for integers, bytes, floats, strings, frozensets and tuples. For example, the literal `1` will create the same object as `int("1")`, which means that `1 is int("1")` is `True`. However this works only by chance as other integer values are not cached, for example `int("1000") is 1000` will always be `False`. This behavior is not part of Python language specification and could vary between interpreters. CPython 3.8 even [warns about comparing literals using identity operators](#).

The only case where using the `"is"` operator with a cached type is ok is with `"interned"` strings. Note however that interned strings don't necessarily have the same identity as string literals.

This rule raises an issue when at least one operand of an identity operator:

- is of type `int`, `bytes`, `float`, `frozenset` or `tuple`.
- or it is a string literal.

Noncompliant Code Example

```
def literal_comparison(param):
    param is 2000 # Noncompliant

literal_comparison(2000) # will return True
literal_comparison(int("2000")) # will return False






() is tuple() # Noncompliant. Always True
(1,) is tuple([1]) # Noncompliant. Always False

from sys import intern

with open("test.txt") as f: # test.txt contains "blabl
    text = f.read()
    intern(text) is "blabla\n" # Noncompliant. Always Fals
```

Compliant Solution

```
def literal_comparison(param):
    param == 2000
```

 Code Smell
Comments should not be located at the end of lines of code  Code Smell
Lines should not end with trailing whitespaces  Code Smell
Files should contain an empty newline at the end  Code Smell
Long suffix "L" should be upper case  Code Smell

```
literal_comparison(2000) # will return True
literal_comparison(int("2000")) # will return True
```

```
() == tuple() # Always True
(1,) == tuple([1]) # Always True
```

```
from sys import intern
```

```
with open("tmp/test.txt") as f: # test.txt contains "b
    text = f.read()
intern(text) is intern("blabla\n") # Always True
```

See

- [Why does Python 3.8 log a SyntaxWarning for 'is' with literals?](#) - Adam Johnson
- [Equality vs identity](#) - Trey Hunner
- [Python documentation - sys.intern](#)

Available In:

sonarlint  | sonarcloud  | sonarqube 