# Pynguin
Tool Analysis

https://doi.org/10.1145/3510454.351682

Măierean Mircea - Ignat Dragoș - Ichim Vlad

Downloading the tool is straightforward and easy. First, ensure that you have **Python 3.10** installed on your system, as **the tool requires this specific Python version**. Next, it's highly recommended to create and activate a dedicated virtual environment to prevent potential conflicts with other Python packages. Once the virtual environment is active, simply install the tool by executing the command `pip install pynguin`. After completing these steps, you'll be ready to start using Pynguin.

Using the tool is also a good experience, we just need to run a command:

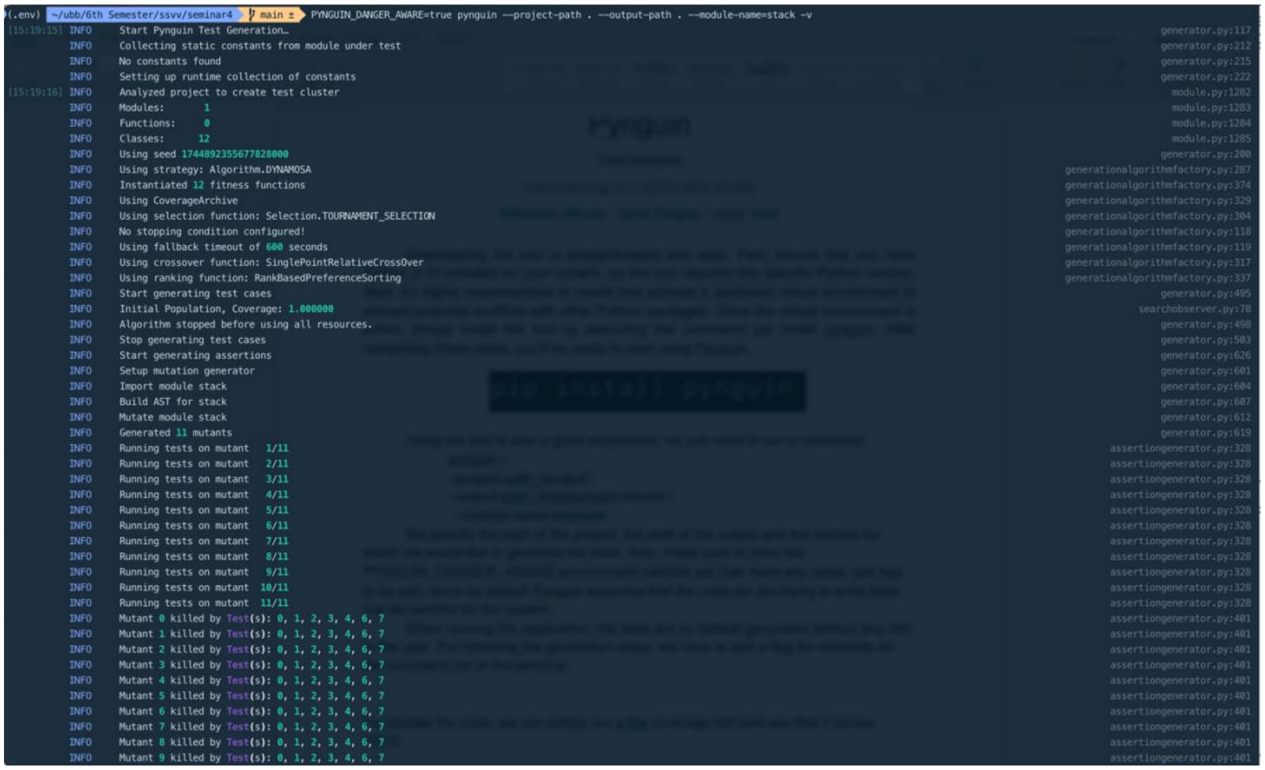
```
pynguin --project-path ./project --output-path ./tmp/pynguin-results \
         --module-name example
```

We specify the path of the project, the path of the output and the module for which we would like to generate the tests. Pynguin executes the module under test! Therefore, depending on what code is in that module, running Pynguin can cause serious harm to your computer, for example, wipe your entire hard disk! It is recommended to run the tests in a docker container, for preventing damage to the computer. As consent, the environment variable `PYNGUIN_DANGER_AWARE` has to be set (can have any value, just has to be set).

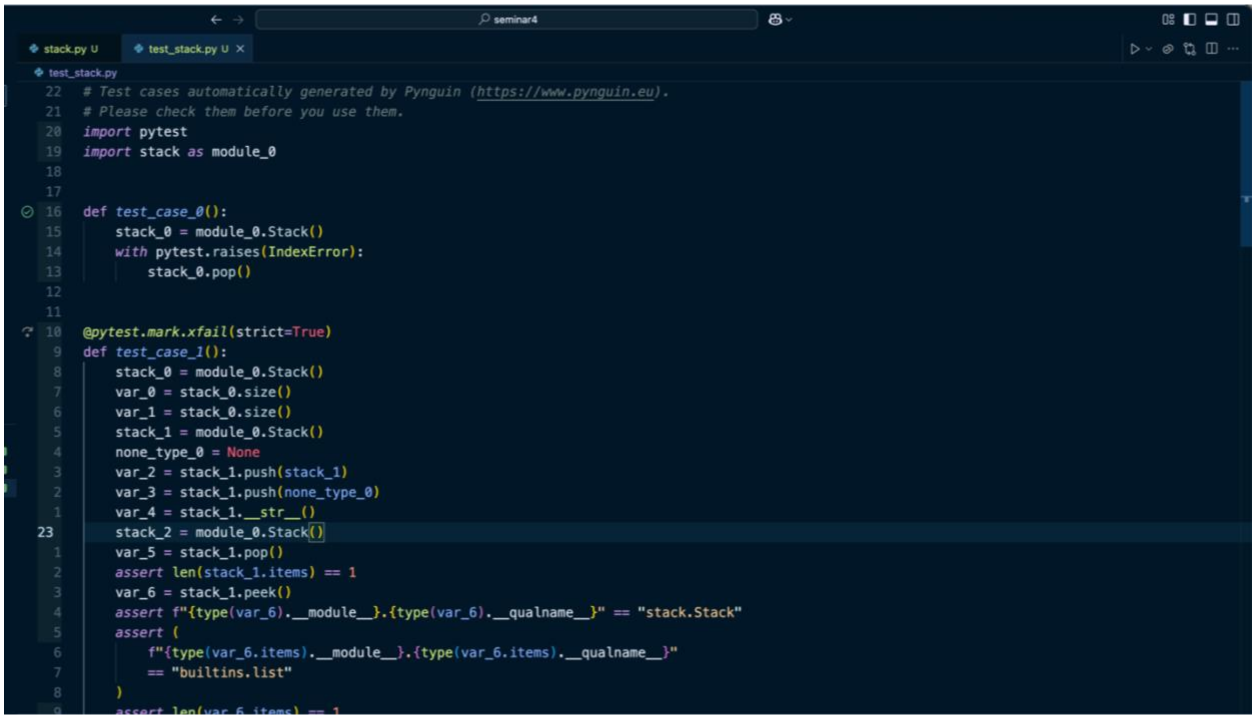
When running the application, the tests are by default generated without any info to the user. For following the generation steps, we must add a flag for verbosity on the command run in the terminal.

Using mutations, the tool creates unit tests that aim to have a code coverage of 100%. All these tests will be generated on a new file named `test_<module name>.py`. The number of tests depends on the number of lines and possible paths that exist in the code. The developer or the tester is still responsible to validate the tests and check that they are appropriate for the intended use case.

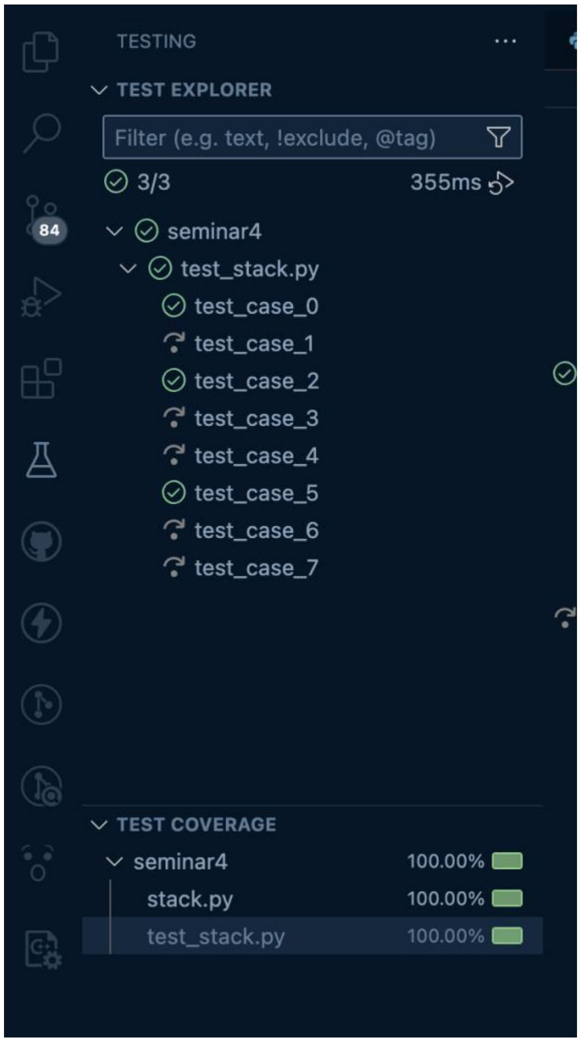
To validate the test coverage, we can simply run a coverage tool. It should score 100%



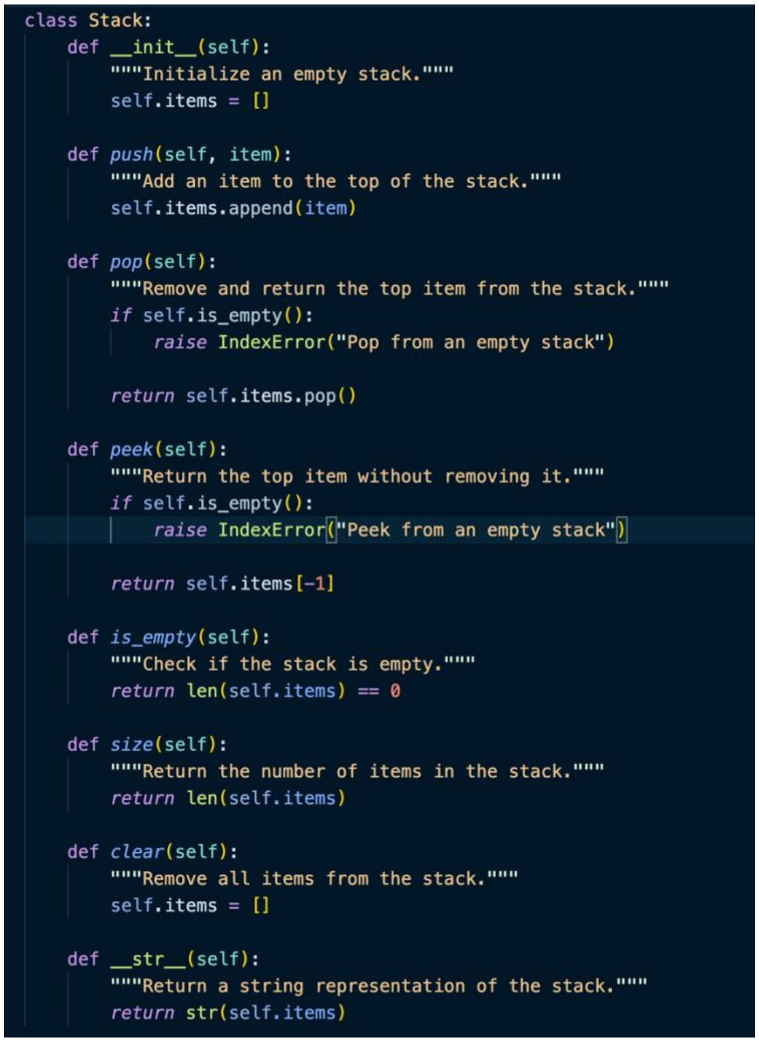
```
(.env) ~/ubb/6th Semester/ssvv/seminar4  main ± PYNGUIN_DANGER_AWARE=true pynguin --project-path . --output-path . --module-name=stack -v
[15:19:15] INFO     Start Pynguin Test Generation…                                              generator.py:117
           INFO     Collecting static constants from module under test                          generator.py:212
           INFO     No constants found                                                          generator.py:215
           INFO     Setting up runtime collection of constants                                  generator.py:222
[15:19:16] INFO     Analyzed project to create test cluster                                       module.py:1282
           INFO     Modules:       1                                                              module.py:1283
           INFO     Functions:     0                                                              module.py:1284
           INFO     Classes:      12                                                              module.py:1285
           INFO     Using seed 1744892355677828000                                              generator.py:200
           INFO     Using strategy: Algorithm.DYNAMOSA                       generationalgorithmfactory.py:287
           INFO     Instantiated 12 fitness functions                        generationalgorithmfactory.py:374
           INFO     Using CoverageArchive                                    generationalgorithmfactory.py:329
           INFO     Using selection function: Selection.TOURNAMENT_SELECTION generationalgorithmfactory.py:304
           INFO     No stopping condition configured!                        generationalgorithmfactory.py:118
           INFO     Using fallback timeout of 600 seconds                    generationalgorithmfactory.py:119
           INFO     Using crossover function: SinglePointRelativeCrossOver   generationalgorithmfactory.py:317
           INFO     Using ranking function: RankBasedPreferenceSorting       generationalgorithmfactory.py:337
           INFO     Start generating test cases                                                 generator.py:495
           INFO     Initial Population, Coverage: 1.000000                                searchobserver.py:78
           INFO     Algorithm stopped before using all resources.                               generator.py:498
           INFO     Stop generating test cases                                                  generator.py:503
           INFO     Start generating assertions                                                 generator.py:626
           INFO     Setup mutation generator                                                    generator.py:601
           INFO     Import module stack                                                         generator.py:604
           INFO     Build AST for stack                                                         generator.py:607
           INFO     Mutate module stack                                                         generator.py:612
           INFO     Generated 11 mutants                                                        generator.py:619
           INFO     Running tests on mutant  1/11                                      assertiongenerator.py:328
           INFO     Running tests on mutant  2/11                                      assertiongenerator.py:328
           INFO     Running tests on mutant  3/11                                      assertiongenerator.py:328
           INFO     Running tests on mutant  4/11                                      assertiongenerator.py:328
           INFO     Running tests on mutant  5/11                                      assertiongenerator.py:328
           INFO     Running tests on mutant  6/11                                      assertiongenerator.py:328
           INFO     Running tests on mutant  7/11                                      assertiongenerator.py:328
           INFO     Running tests on mutant  8/11                                      assertiongenerator.py:328
           INFO     Running tests on mutant  9/11                                      assertiongenerator.py:328
           INFO     Running tests on mutant 10/11                                      assertiongenerator.py:328
           INFO     Running tests on mutant 11/11                                      assertiongenerator.py:328
           INFO     Mutant 0 killed by Test(s): 0, 1, 2, 3, 4, 6, 7                    assertiongenerator.py:401
           INFO     Mutant 1 killed by Test(s): 0, 1, 2, 3, 4, 6, 7                    assertiongenerator.py:401
           INFO     Mutant 2 killed by Test(s): 0, 1, 2, 3, 4, 6, 7                    assertiongenerator.py:401
           INFO     Mutant 3 killed by Test(s): 0, 1, 2, 3, 4, 6, 7                    assertiongenerator.py:401
           INFO     Mutant 4 killed by Test(s): 0, 1, 2, 3, 4, 6, 7                    assertiongenerator.py:401
           INFO     Mutant 5 killed by Test(s): 0, 1, 2, 3, 4, 6, 7                    assertiongenerator.py:401
           INFO     Mutant 6 killed by Test(s): 0, 1, 2, 3, 4, 6, 7                    assertiongenerator.py:401
           INFO     Mutant 7 killed by Test(s): 0, 1, 2, 3, 4, 6, 7                    assertiongenerator.py:401
           INFO     Mutant 8 killed by Test(s): 0, 1, 2, 3, 4, 6, 7                    assertiongenerator.py:401
           INFO     Mutant 9 killed by Test(s): 0, 1, 2, 3, 4, 6, 7                    assertiongenerator.py:401
```



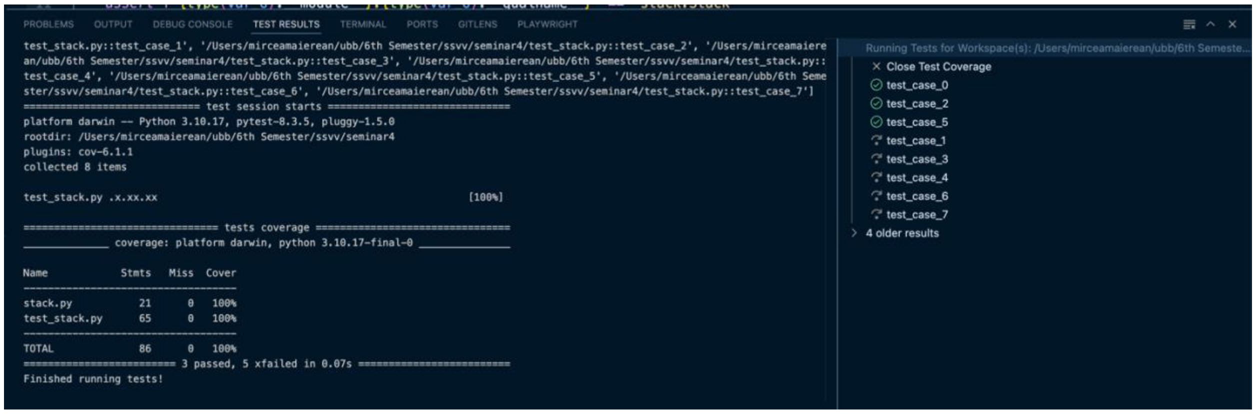
```python
# Test cases automatically generated by Pynguin (https://www.pynguin.eu).
# Please check them before you use them.
import pytest
import stack as module_0


def test_case_0():
    stack_0 = module_0.Stack()
    with pytest.raises(IndexError):
        stack_0.pop()


@pytest.mark.xfail(strict=True)
def test_case_1():
    stack_0 = module_0.Stack()
    var_0 = stack_0.size()
    var_1 = stack_0.size()
    stack_1 = module_0.Stack()
    none_type_0 = None
    var_2 = stack_1.push(stack_1)
    var_3 = stack_1.push(none_type_0)
    var_4 = stack_1.__str__()
    stack_2 = module_0.Stack()
    var_5 = stack_1.pop()
    assert len(stack_1.items) == 1
    var_6 = stack_1.peek()
    assert f"{type(var_6).__module__}.{type(var_6).__qualname__}" == "stack.Stack"
    assert (
        f"{type(var_6.items).__module__}.{type(var_6.items).__qualname__}"
        == "builtins.list"
    )
    assert len(var_6.items) == 1
```



```
seminar4        100.00% 🟩
  stack.py      100.00% 🟩
  test_stack.py 100.00% 🟩
```



```python
class Stack:
    def __init__(self):
        """Initialize an empty stack."""
        self.items = []

    def push(self, item):
        """Add an item to the top of the stack."""
        self.items.append(item)

    def pop(self):
        """Remove and return the top item from the stack."""
        if self.is_empty():
            raise IndexError("Pop from an empty stack")

        return self.items.pop()

    def peek(self):
        """Return the top item without removing it."""
        if self.is_empty():
            raise IndexError("Peek from an empty stack")

        return self.items[-1]

    def is_empty(self):
        """Check if the stack is empty."""
        return len(self.items) == 0

    def size(self):
        """Return the number of items in the stack."""
        return len(self.items)

    def clear(self):
        """Remove all items from the stack."""
        self.items = []

    def __str__(self):
        """Return a string representation of the stack."""
        return str(self.items)
```



```
test_stack.py::test_case_1', '/Users/mirceamaierean/ubb/6th Semester/ssvv/seminar4/test_stack.py::test_case_2', '/Users/mirceamaiere
an/ubb/6th Semester/ssvv/seminar4/test_stack.py::test_case_3', '/Users/mirceamaierean/ubb/6th Semester/ssvv/seminar4/test_stack.py::
test_case_4', '/Users/mirceamaierean/ubb/6th Semester/ssvv/seminar4/test_stack.py::test_case_5', '/Users/mirceamaierean/ubb/6th Seme
ster/ssvv/seminar4/test_stack.py::test_case_6', '/Users/mirceamaierean/ubb/6th Semester/ssvv/seminar4/test_stack.py::test_case_7']
============================= test session starts ==============================
platform darwin -- Python 3.10.17, pytest-8.3.5, pluggy-1.5.0
rootdir: /Users/mirceamaierean/ubb/6th Semester/ssvv/seminar4
plugins: cov-6.1.1
collected 8 items

test_stack.py .x.xx.xx                                                   [100%]

================================ tests coverage ================================
_______________ coverage: platform darwin, python 3.10.17-final-0 _______________

Name            Stmts   Miss  Cover
-----------------------------------
stack.py           21      0   100%
test_stack.py      65      0   100%
-----------------------------------
TOTAL              86      0   100%
========================= 3 passed, 5 xfailed in 0.07s =========================
Finished running tests!
```

```
Running Tests for Workspace(s): /Users/mirceamaierean/ubb/6th Semeste...
  × Close Test Coverage
  ✓ test_case_0
  ✓ test_case_2
  ✓ test_case_5
    test_case_1
    test_case_3
    test_case_4
    test_case_6
    test_case_7
> 4 older results
```