Atividade 2 – Testes de Mutação

Discente: Gabriel Marques Costa
Docente: Glauco de Figueiredo Carneiro

1. Repositório escolhido

1.1. Informações

O repositório escolhido foi o PyGraph, ele é um repositório que tem como foco realizar a implementação dos ensinamentos teóricos de Grafos. Você pode acessá-lo no seguinte link: PyGraph.

1.2. Utilização

Para ter acesso ao repositório foi preciso utilizar as seguintes ferramentas: A ide VScode para rodar e realizar os testes, linguagem obrigatória Python e as bibliotecas: Pytest, Pytest-cov, mutmut e a obrigatória do código poetry.

2. Testes

2.1. Antes das alterações

Assim que foram instalados as dependências e requisitos para rodar os testes sem alterações no código foi possível perceber que ele apresentava: 58 testes que tinham êxito e 212 mutantes mortos e 79 que sobrevivem:

Relatório dos testes, o primeiro utilizando o comando “pytest -v” para realizar todos os testes do código e em seguida o comando “pytest --html=report.html” para ter o acesso ao HTML com as informações dos testes:

```
ED [ 86%]
tests/test_simple_graph.py::TestSimpleGraph::test_list_graph_edges PASSED
[ 87%]
tests/test_simple_graph.py::TestSimpleGraph::test_show_edge PASSED [ 89%]
tests/test_simple_graph.py::TestSimpleGraph::test_true_cycle_graph PASSED
[ 91%]
tests/test_simple_graph.py::TestSimpleGraph::test_false_cycle_graph PASSED
 [ 93%]
tests/test_simple_graph.py::TestSimpleGraph::test_loop PASSED      [ 94%]
tests/test_simple_graph.py::TestSimpleGraph::test_true_regular_graph PASSE
D [ 96%]
tests/test_simple_graph.py::TestSimpleGraph::test_false_regular_graph PASS
ED [ 98%]
tests/test_simple_graph.py::TestSimpleGraph::test_duplicated_edge PASSED [
100%]

========================== 58 passed in 0.21s ==========================
```

Summary

58 tests took 43 ms.

(Un)check the boxes to filter the results.

0 Failed, 58 Passed, 0 Skipped, 0 Expected failures, 0 Unexpected passes, 0 Errors, 0 Reruns

Result	Test
Passed	tests/test_graph.py::TestGraph::test_add_vertex
Passed	tests/test_graph.py::TestGraph::test_exception_add_vertex_duplicate
Passed	tests/test_graph.py::TestGraph::test_delete_vertex

Relatório dos mutantes, primeiro executando o comando “mutmut run” para realizar os testes de mutação em todo código:

```
🎉 Killed mutants.   The goal is for everything to end up in this bucket.
⏰ Timeout.          Test suite took 10 times as long as the baseline so were kil
ld.
ed.
🤔 Suspicious.       Tests took a long time, but not long enough to be fatal.
🙁 Survived.         This means your tests need to be expanded.
🔇 Skipped.          Skipped.

1. Running tests without mutations
⠏ Running...Done

2. Checking mutants
⠹ 212/212  🎉 133  ⏰ 0  🤔 0  🙁 79  🔇 0
```

2.2. Resultados após as alterações

Analisando o código foi possível ver apenas uma oportunidade de melhoria, onde foi identificado que uma das funções de teste estava inadequada em alguns parâmetros, sendo assim foi preciso atualizar a mesma:

```
def test_incidence_list():
    g = Graph()
    v1 = g.add_vertex("a")
    v2 = g.add_vertex("b")
    v3 = g.add_vertex("c")
    g.add_edge("a", "b", "ab")
    g.add_edge("b", "c", "bc")

    incidence = g.incidence_list()

    expected_incidence = [
        ValueBinding("a", "ab", 1),
        ValueBinding("a", "bc", 0),
        ValueBinding("b", "ab", 1),
        ValueBinding("b", "bc", 1),
        ValueBinding("c", "ab", 0),
        ValueBinding("c", "bc", 1),
    ]

    assert sorted(incidence) == sorted(
        expected_incidence), "A lista de incidência está incorreta."
```

Função utilizada para a melhoria dos testes:

```python
def incidence_list(self):
    """
    Método que retorna uma lista de objetos que contem
    a incidencia dos vertices com as arestas.

    Retorno:
    ----------
    incidence_list: List
        - Lista com os objetos de ligação(ValueBinding).
    """
    incidence_list = []
    for v in self.vertices.values():
        for e in self.edges:
            if e.vertex_a == v and e.vertex_b == v:
                incidence_list.append(
                    ValueBinding(v.get_value(), e.get_name(), 2)
                )
            elif e.vertex_a == v or e.vertex_b == v:
                incidence_list.append(
                    ValueBinding(v.get_value(), e.get_name(), 1)
                )
            else:
                incidence_list.append(
                    ValueBinding(v.get_value(), e.get_name(), 0)
                )
    return incidence_list
```

Assim obtendo um resultado satisfatório no novo teste do mutmut de:

```
🎉 Killed mutants.   The goal is for everything to end up in this bucket.
⏰ Timeout.          Test suite took 10 times as long as the baseline so were kil
l
ed.
🤔 Suspicious.       Tests took a long time, but not long enough to be fatal.
😕 Survived.         This means your tests need to be expanded.
🔇 Skipped.          Skipped.

1. Running tests without mutations
⠏ Running...Done

2. Checking mutants
⠹ 177/177  🎉 133  ⏰ 0  🤔 0  😕 44  🔇 0
```

3. Links Importantes

3.1. Link do repositório

https://github.com/GabrielCosta0109/Teste_Software_Mutantes_2024_Costa_Gabriel

3.2. Link deste documento

https://docs.google.com/document/d/1uXHKy-C7foClcy25rDcSUUbwoT0pyRhkvAd2xIxYAHg/edit?usp=sharing

3.3. Link do vídeo

https://drive.google.com/file/d/15i1oTmwfI8p92-jcH2mjBVzWSaIs84Tr/view?usp=sharing