GitHub Repository: github.com/zran28320-lgtm/CS449-Solitaire

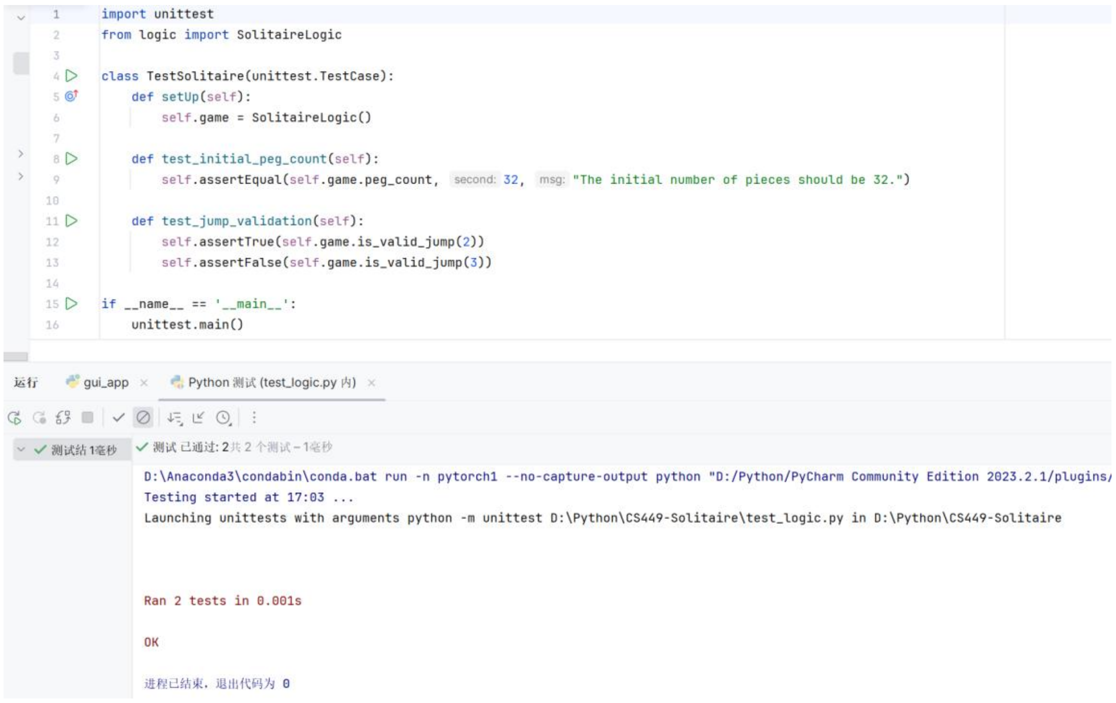
Sprint #0 Report

1. Key Decisions of the Solitaire Project

Object-oriented programming language	Python
GUI library	Tkinter
IDE (Integrated Development Environment)	PyCharm
xUnit framework	unittest
Programming style guide	Google Python Style Guide
Project hosting site	Github.com

2. Unit testing

(1) The Screenshot of Program Execution



```python
import unittest
from logic import SolitaireLogic

class TestSolitaire(unittest.TestCase):
    def setUp(self):
        self.game = SolitaireLogic()

    def test_initial_peg_count(self):
        self.assertEqual(self.game.peg_count,  second: 32,  msg: "The initial number of pieces should be 32.")

    def test_jump_validation(self):
        self.assertTrue(self.game.is_valid_jump(2))
        self.assertFalse(self.game.is_valid_jump(3))

if __name__ == '__main__':
    unittest.main()
```

运行
gui_app
Python 测试 (test_logic.py 内)
测试结 1毫秒
测试 已通过: 2共 2 个测试 – 1毫秒

```
D:\Anaconda3\condabin\conda.bat run -n pytorch1 --no-capture-output python "D:/Python/PyCharm Community Edition 2023.2.1/plugins/
Testing started at 17:03 ...
Launching unittests with arguments python -m unittest D:\Python\CS449-Solitaire\test_logic.py in D:\Python\CS449-Solitaire



Ran 2 tests in 0.001s

OK

进程已结束，退出代码为 0
```

(2) The Source Code of Program

logic.py:

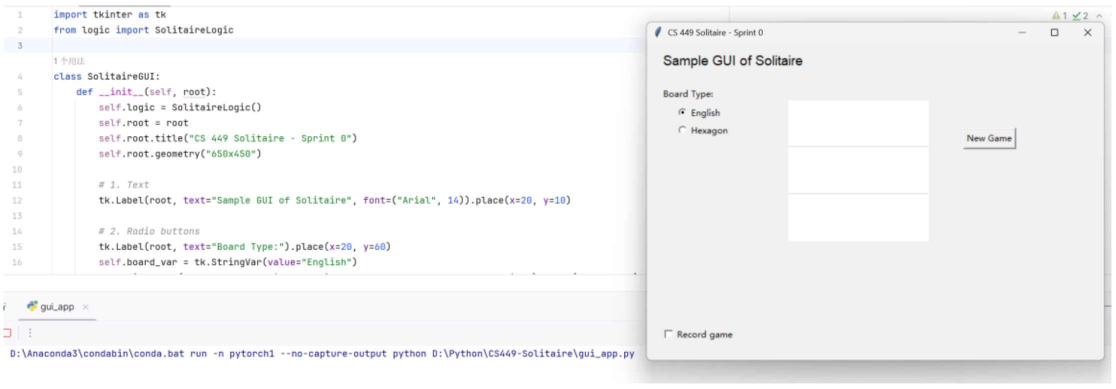
```python
class SolitaireLogic:
    def __init__(self):
        self.board_size = 7
        self.peg_count = 32

    def reset_game(self):
        self.peg_count = 32
        return True

    def is_valid_jump(self, distance):
        return distance == 2
```

test_logic.py:

```python
import unittest
from logic import SolitaireLogic

class TestSolitaire(unittest.TestCase):
    def setUp(self):
        self.game = SolitaireLogic()

    def test_initial_peg_count(self):
        self.assertEqual(self.game.peg_count, 32, "The initial
number of pieces should be 32.")

    def test_jump_validation(self):
        self.assertTrue(self.game.is_valid_jump(2))
        self.assertFalse(self.game.is_valid_jump(3))

if __name__ == '__main__':
    unittest.main()
```

3. GUI programming

(1) The Screenshot of Program Execution



```python
import tkinter as tk
from logic import SolitaireLogic

1 个用法
class SolitaireGUI:
    def __init__(self, root):
        self.logic = SolitaireLogic()
        self.root = root
        self.root.title("CS 449 Solitaire - Sprint 0")
        self.root.geometry("650x450")

        # 1. Text
        tk.Label(root, text="Sample GUI of Solitaire", font=("Arial", 14)).place(x=20, y=10)

        # 2. Radio buttons
        tk.Label(root, text="Board Type:").place(x=20, y=60)
        self.board_var = tk.StringVar(value="English")
```

gui_app

```
D:\Anaconda3\condabin\conda.bat run -n pytorch1 --no-capture-output python D:\Python\CS449-Solitaire\gui_app.py
```

CS 449 Solitaire - Sprint 0

Sample GUI of Solitaire

Board Type:

[✓ English] Hexagon

New Game

☐ Record game

(2) The Source Code of Program

gui_app.py:

```python
import tkinter as tk
from logic import SolitaireLogic

class SolitaireGUI:
    def __init__(self, root):
        self.logic = SolitaireLogic()
        self.root = root
        self.root.title("CS 449 Solitaire - Sprint 0")
        self.root.geometry("650x450")

        # 1. Text
        tk.Label(root, text="Sample GUI of Solitaire",
font=("Arial", 14)).place(x=20, y=10)

        # 2. Radio buttons
        tk.Label(root, text="Board Type:").place(x=20, y=60)
        self.board_var = tk.StringVar(value="English")
        tk.Radiobutton(root, text="English",
variable=self.board_var, value="English").place(x=40, y=85)
        tk.Radiobutton(root, text="Hexagon",
variable=self.board_var, value="Hexagon").place(x=40, y=110)

        # 3. Lines
        self.canvas = tk.Canvas(root, width=200, height=200,
bg="white", highlightthickness=1)
        self.canvas.place(x=200, y=80)

        self.canvas.create_line(0, 66, 200, 66, fill="lightgray")
        self.canvas.create_line(0, 133, 200, 133, fill="lightgray")

        # 4. Check box
        self.record_var = tk.BooleanVar()
        tk.Checkbutton(root, text="Record game",
variable=self.record_var).place(x=20, y=400)
        tk.Button(root, text="New Game",
command=self.reset).place(x=450, y=120)

    def reset(self):
        if self.logic.reset_game():
            print("Game Logic Reset!")

if __name__ == "__main__":
    root = tk.Tk()
    app = SolitaireGUI(root)
    root.mainloop()
```