Projekt zaliczeniowy – Języki skryptowe (Python)

Nazwa uczelni: Politechnika Świętokrzyska
Nazwa przedmiotu: Języki Skryptowe
Prowadzący: dr inż. Dariusz Michalski
Tytuł projektu: Gra w statki (tekstowa)
Autorzy: Jakub Jakubczyk, Grzegorz Huk
Grupa: Informatyka, rok II, 2ID12A
Data oddania: 24 czerwca 2025

2. Opis projektu

2.1 Cel projektu

Naszym celem było stworzenie prostej wersji gry w statki w języku Python. Chcieliśmy zapoznać się z modułami standardowymi, obsługą wyjątków, formatowaniem JSON.

2.2 Funkcje aplikacji

* Ustawianie automatyczne statków na planszy przez gracza i komputer
* Strzelanie po współrzędnych i informacja o trafieniu/pudle
* Zapis oraz odczyt stanu gry w pliku JSON
* Obsługa podstawowego interfejsu konsolowego

2.3 Zakres funkcjonalny

W projekcie zaimplementowaliśmy:

* Logikę planszy i statków (klasy Board i Ship)
* Serializację do JSON (moduł utils/serializers.py)
* Interfejs konsolowy
* Testy jednostkowe dla klasy Ship oraz przykładowe testy rysowania planszy

3. Struktura projektu

3.1 Opis plików i folderów

* src/board.py – klasa Board odpowiedzialna za planszę gry
* src/ship.py – klasa Ship reprezentująca statek i jego logikę
* src/decorators.py – dekoratory pomocnicze (np. do weryfikacji zakresu)
* src/ui.py – moduł obsługujący interfejs (konsola)
* src/utils/serializers.py – funkcje do zapisu/odczytu JSON
* tests/ – folder z testami jednostkowymi
* requirements.txt – lista użytych bibliotek (standardowe moduły)

3.2 Krótkie omówienie klas/modułów

* Board: przechowuje siatkę, przyjmuje strzały i sprawdza stan gry
* Ship: definiuje pozycję, długość oraz sprawdza trafienia
* ui: funkcje do wyświetlania menu i planszy, pobierania danych od użytkownika
* serializers: save_game(), load_game() – zapis/odczyt stanu gry

4. Technologie i biblioteki

* Python 3.x
* Moduły standardowe: json, datetime, os
* Dodatkowe: tkinter (opcjonalnie), functools, itertools

5. Sposób działania programu

5.1 Instrukcja uruchomienia

1. Sklonować repozytorium
2. Przejść do katalogu projektu
3. Uruchomić `pip install -r requirements.txt`
4. Uruchomić `python -m src.ui` lub `python src/ui.py`

5.2 Przykładowe dane wejściowe/wyjściowe

* Wejście: strzał na współrzędne 5,5
* Wyjście: "HIT" lub "MISS"

6. Przykłady kodu

6.1 Fragment funkcji funkcyjnej

```python
def fire(self, x, y):
    """Sprawdza, czy statek został trafiony."""
    if (x, y) in self.coordinates:
        self.hits.add((x, y))
        return True
    return False
```

6.2 Fragment klasy

```python
class Ship:
    def __init__(self, length, coordinates):
        self.length = length
        self.coordinates = coordinates
        self.hits = set()
```

6.3 Obsługa wyjątków

```python
try:
    x, y = map(int, input("Podaj współrzędne: ").split())
except ValueError:
    print("Nieprawidłowe dane, spróbuj ponownie.")
```

7. Testowanie

7.1 Opis sposobu testowania

* Użyto pytest do uruchomienia testów w folderze tests/
* Testowane przypadki: tworzenie statku, trafienie, zatopienie

7.2 Obsługa przypadków granicznych

* Strzał poza planszą – zwracany wyjątek
* Wielokrotne strzały w to samo miejsce – ignorowane

8. Wnioski

* Co się udało: zaimplementowaliśmy podstawową rozgrywkę, zapis stanu gry i testy.
* Co można było zrobić lepiej: rozbudować GUI, dodać grę sieciową, poprawić UX.
* Rozwinięte kompetencje: praca z modułami standardowymi, serializacja JSON, testowanie, podstawy UI.