Dokumentacja Projektu Symulacji Ekosystemu

# Temat Projektu

Symulacja ekosystemu, w której różne gatunki zwierząt poruszają się po planszy, polują na siebie, konsumują zasoby, oraz wchodzą w interakcje z otoczeniem i między sobą. Symulacja kończy się, gdy na planszy pozostaje tylko jeden gatunek zwierząt.

## **Skład Grupy Projektowej**

* Lider: Kacper Mondzelewski
* Członkowie:
  + Dominik Krakowiak
  + Kacper Mondzelewski

## **Opis Zadania Symulacji**

Celem projektu jest stworzenie symulacji ekosystemu, w której różne gatunki zwierząt poruszają się po planszy, konsumują zasoby, polują na siebie nawzajem oraz używają swoich unikalnych umiejętności. Plansza jest wypełniona różnymi zasobami (trawa, krzaki z owocami, grzyby) oraz kamieniami. Zwierzęta mają swoje statystyki takie jak siła, zdrowie, szybkość oraz unikalne zdolności. Symulacja trwa w turach, podczas których każde zwierzę wykonuje swoje akcje. Symulacja kończy się, gdy na planszy pozostaje tylko jeden gatunek zwierząt.

### **Kluczowe funkcje symulacji:**

1. **Poruszanie się zwierząt:** Zwierzęta poruszają się po planszy w losowych kierunkach, unikając powrotu na poprzednie pole, o ile to możliwe.
2. **Konsumowanie zasobów:** Zwierzęta konsumują zasoby, które napotkają na swojej drodze, co wpływa na ich statystyki.
3. **Polowanie:** Drapieżniki atakują i zadają obrażenia roślinożercom znajdującym się na sąsiadujących polach.
4. **Unikalne zdolności:** Niektóre zwierzęta mają specjalne zdolności, które mogą aktywować podczas symulacji.
5. **Aktualizacja bonusów i efektów:** Zwierzęta mogą otrzymywać bonusy do statystyk oraz być dotknięte efektami (np. trucizna).

### **Opis Działania Funkcji w Kodzie**

#### **Struktura Effect**

* **Opis**: Struktura Effect reprezentuje efekt czasowy, który wpływa na zwierzę przez określoną liczbę tur.
* **Pola**:
  + damagePerTurn: Ilość obrażeń zadawanych na turę.
  + duration: Czas trwania efektu w turach.

#### **Klasa Animal**

**Opis**: Bazowa klasa dla wszystkich zwierząt w symulacji.

* **Pola**:
  + type: Typ zwierzęcia.
  + symbol: Symbol zwierzęcia na planszy.
  + strength: Siła zwierzęcia.
  + health: Zdrowie zwierzęcia.
  + speed: Szybkość zwierzęcia.
  + position: Aktualna pozycja zwierzęcia na planszy.
  + effects: Lista aktywnych efektów działających na zwierzę.

**Najważniejsze Funkcje**:

* **Konstruktor**:

Animal(std::string type, char symbol, int strength, int health, int speed)

Inicjalizuje zwierzę z podanymi parametrami.

* **move**:

virtual void move(std::vector<std::vector<char>>& board, int steps)

Przesuwa zwierzę po planszy o określoną liczbę kroków.

* **hunt**:

virtual void hunt(std::vector<std::vector<char>>& board, std::vector<std::shared\_ptr<Animal>>& prey)

Poluje na inne zwierzęta (do nadpisania przez klasy pochodne).

* **consumeResource**:

void consumeResource(char resourceSymbol)

Konsumuje zasób na planszy i modyfikuje odpowiednie statystyki zwierzęcia.

* **applyEffects**:

void applyEffects()

Zastosowuje aktywne efekty na zwierzęciu, zmniejszając zdrowie i redukując czas trwania efektów.

* **addEffect**:

void addEffect(const Effect& effect)

Dodaje nowy efekt do aktywnych efektów zwierzęcia.

#### **Klasa Carnivore**

**Opis**: Klasa pochodna Animal reprezentująca drapieżnika.

**Najważniejsze Funkcje**:

* **hunt**:

void hunt(std::vector<std::vector<char>>& board, std::vector<std::shared\_ptr<Animal>>& animals) override

Poluje na inne zwierzęta w sąsiednich komórkach, zmniejszając ich zdrowie i usuwając je z planszy, jeśli zdrowie spadnie do zera.

#### **Klasa Herbivore**

**Opis**: Klasa pochodna Animal reprezentująca roślinożercę.

**Najważniejsze Funkcje**:

* **useUniqueAbility**:

virtual void useUniqueAbility()

Używa unikalnej zdolności (do nadpisania przez klasy pochodne).

#### **Klasa Wolf**

**Opis**: Klasa reprezentująca wilka.

* **Konstruktor**:

Wolf() : Carnivore("Wilk", 'W', 30, 60, 7) {}

Inicjalizuje wilka z określonymi statystykami.

#### **Klasa Eagle**

**Opis**: Klasa reprezentująca orła.

* **Konstruktor**:

Eagle() : Carnivore("Orzel", 'E', 20, 30, 9) {}

Inicjalizuje orła z określonymi statystykami.

**Najważniejsze Funkcje**:

* **hunt**:

void hunt(std::vector<std::vector<char>>& board, std::vector<std::shared\_ptr<Animal>>& animals) override

Poluje na inne zwierzęta z większym zasięgiem niż standardowy drapieżnik.

#### **Klasa Viper**

**Opis**: Klasa reprezentująca żmiję.

* **Konstruktor**:

Viper() : Carnivore("Zmija", 'V', 15, 40, 5) {}

Inicjalizuje żmiję z określonymi statystykami.

**Najważniejsze Funkcje**:

* **hunt**:

void hunt(std::vector<std::vector<char>>& board, std::vector<std::shared\_ptr<Animal>>& animals) override

Poluje na inne zwierzęta, dodatkowo zadając im efekt trucizny.

#### **Klasa Bear**

**Opis**: Klasa reprezentująca niedźwiedzia.

* **Konstruktor**:

Bear() : Carnivore("Niedzwiedz", 'B', 50, 80, 4) {}

Inicjalizuje niedźwiedzia z określonymi statystykami.

#### **Klasa Bison**

**Opis**: Klasa reprezentująca bizona.

* **Konstruktor**:

Bison() : Herbivore("Bizon", 'Z', 40, 100, 3) {}

Inicjalizuje bizona z określonymi statystykami.

#### **Klasa Sparrow**

**Opis**: Klasa reprezentująca wróbla.

* **Konstruktor**:

Sparrow() : Herbivore("Wrobel", 'S', 10, 15, 10) {}

Inicjalizuje wróbla z określonymi statystykami.

**Najważniejsze Funkcje**:

* **useUniqueAbility**:

void useUniqueAbility() override

Ma 50% szans na uniknięcie ataku.

#### **Klasa Hare**

**Opis**: Klasa reprezentująca zająca.

* **Konstruktor**:

Hare() : Herbivore("Zajac", 'H', 15, 35, 7) {}

Inicjalizuje zająca z określonymi statystykami.

**Najważniejsze Funkcje**:

* **useUniqueAbility**:

void useUniqueAbility() override

Ma 30% szans na uniknięcie ataku.

#### **Klasa RoeDeer**

**Opis**: Klasa reprezentująca jelenia.

* **Konstruktor**:

RoeDeer() : Herbivore("Jelen", 'D', 25, 55, 8) {}

Inicjalizuje jelenia z określonymi statystykami.

#### **Klasa Resource**

**Opis**: Klasa reprezentująca zasoby na planszy.

* **Konstruktor**:

Resource(std::string name, char symbol)

Inicjalizuje zasób z podaną nazwą i symbolem.

**Najważniejsze Funkcje**:

* **display**:

void display() const

Wyświetla informacje o zasobie.

### **Podsumowanie Najważniejszych Funkcji**

* **move**: Przesuwa zwierzę po planszy.
* **hunt**: Poluje na inne zwierzęta.
* **consumeResource**: Konsumuje zasób na planszy.
* **applyEffects**: Zastosowuje aktywne efekty na zwierzęciu.
* **addEffect**: Dodaje nowy efekt do aktywnych efektów zwierzęcia.
* **useUniqueAbility**: Używa unikalnej zdolności zwierzęcia.

Opisane struktury i klasy oraz ich kluczowe funkcje stanowią fundament symulacji ekosystemu, umożliwiając realistyczne interakcje między różnymi gatunkami zwierząt i zasobami.