Relatório Meta 1 Trabalho Prático

Unidade Curricular - Programação Orientada a Objetos

Ano Letivo 2022/2023

João Carvalho
a2019131769
Daniel Duarte
a2019131703
27 de novembro de 2022

Índice

1 - Introdução

Este trabalho prático tem como objetivo a construção de uma reserva natural povoada por uma variedade de animais que têm comportamento autónomo e variado, capazes de se deslocar pela reserva.

Neste trabalho existe interação entre um Simulador (Interface), Reserva e os animais e alimentos.

2 - Classes

```
class Interface{
public:
    Interface(Reserva *r):r(r){};
    ~Interface();

    void board();
    void run();

    string getInterfaceAsString() const;
    void readConstantes(vector<string> sepCommands);
    void loadCommandsFromKeyboard(string line);
    void loadCommandsFromFile(const string& filename);
    void commandAnimal(vector<string> sepCommands);
    void commandKill(vector<string> sepCommands);
    void commandFood(vector<string> sepCommands);
    void commandFeed(vector<string> sepCommands);
    void commandFeedID(vector<string> sepCommands);
    void commandNoFood(vector<string> sepCommands);
    void commandEmpty(vector<string> sepCommands);
    void commandSee(vector<string> sepCommands);
    void commandInfo(vector<string> sepCommands);
    void commandNext(vector<string> sepCommands);
    void commandAnim(vector<string> sepCommands);
    void commandVisAnim(vector<string> sepCommands);
    void commandStore(vector<string> sepCommands);
    void commandReStore(vector<string> sepCommands);
    //void commandLoad(vector<string> sepCommands);
    void commandSlide(vector<string> sepCommands);
    void commandExit(vector<string> sepCommands);

    bool isNumber(const string& str);
    vector<string> divideString(string str) const;

private:
    string interfaceStatus;
    Reserva* r;
    vector<string> commandsV;
};
```

Classe Interface

Classe utilizada para mostrar tudo no ecrã. Esta classe também é responsável por gerir a reserva através de composição, ou seja, é responsável por criar, destruir e gerir a reserva.

```
class Reserva{
public:
    Reserva(){};
    ~Reserva();

    void setNL(int nl);
    void setNC(int nc);

    int getNL() const;
    int getNC() const;

    //void criaAnimal(const char& abrv, int id);
    void criaAnimal(const char& abrv/*, int id*/, int y, int x);
    void criaAlimento(const char& abrv/*, int id*/, int y, int x);
    vector<Animal*> getAnimalVector();
    string getAnimalVectorAsString();
    vector<Alimento*> getAlimentoVector();
    string getAlimentosVectorAsString();

private:
    static int cnt;

    int NL;
    int NC;

    vector<Animal*> animais;
    vector<Alimento*> alimentos;
};
```

Classe Reserva

Esta classe é responsável por gerir, através de agregação, um vetor de animais e de alimentos.

```
class Animal{
public:
    Animal(char& abreviation, int y, int x);

    ~Animal();

    char getAbreviation() const;
    int getID() const;
    string getAnimalAsString() const;
    int getX() const;
    int getY() const;

    void setID(int number);
    void setX(int number);
    void setY(int number);
    void setInitialHealth(char abrv);
    void setLTime(char abrv);

    static void setAbreviation(string animalType, string character);
    static void setLife(string animalType, int health);
    static void setLifeTime(string animalType, int moments);
private:
    static string NCoelho;
    static int SCoelho;
    static int VCoelho;
    static string NOvelha;
    static int SOvelha;
    static int VOvelha;
    static string NLobo;
    static int SLobo;
    static int VLobo;
    static string NCanguru;
    static int SCanguru;
    static int VCanguru;
    static string NMisterio;
    static int SMisterio;
    static int VMisterio;
    char abreviation;
    int id;
    int initialHealth;
    int minHealth;
    int maxHealth;
    int health;
    int lifeTime;
    int starvingPoints;

    int y;
    int x;
};
```

Classe Animal

Classe que contem todos os detalhes acerca dos animais pertencentes a reserva.

```
class Alimento{
public:
    Alimento(char& abreviation/*, int id*/, int y, int x);
    ~Alimento();

    char getAbreviation() const;
    int getID() const;
    int  getlifeTime() const;
    int getNutValue() const;
    int getToxicity() const;
    string getSmell() const;
    int getY() const;
    int getX() const;

    string getAlimentosAsString() const;

    void setID(int number);


private:
    char abreviation;
    int id;
    int lifeTime;
    int nutValue;
    int toxicity;
    string smell;

    int y;
    int x;
};
```

Classe Alimento

Classe que contem todos os detalhes sobre os alimentos constituintes da reserva.

3 - Método de Implementação

Inicialmente o programa começa por perguntar ao utilizador o número de linhas e colunas que este deseja que a reserva tenha.

De seguida inicia se a fazer de COMANDOS, a qual consiste em que através da escrita de comandos por parte do utilizador ou através da leitura de ficheiros de texto que contem os mesmo, seja possível a criação de animais e alimentos, tão bem como, a visualização dos seus detalhes.

Após os comandos serem escritos e executados é representado na reserva todos os efeitos que estes causaram na mesma.