GAMING CAMPUS PARIS

Projet StarCraft 2 en C++

Description du Projet

Le projet consiste à créer une simulation du jeu StarCraft 2, en se concentrant sur la faction des Zergs. Vous allez concevoir et implémenter différentes classes et héritages pour représenter les entités et les bâtiments Zergs, ainsi que leurs interactions. Voici un aperçu des principales composantes du projet :

1. **Couveuse** : La couveuse est la source de production des unités Zergs. Vous devrez créer une classe pour représenter la couveuse et permettre la création d'unités à partir de celle-ci.

```cpp
#pragma once

#include <iostream>
#include <vector>

class Couveuse {
private:
    std::vector<Unit> unitProductionQueue; // File d'attente pour la production d'unités
    int resourceCost; // Coût en ressources pour produire des unités

public:
    Couveuse(int cost) : resourceCost(cost) {
        // Initialisez ici les attributs de la couveuse
    }

    // Méthode pour ajouter une unité à la file de production
    void AddToQueue(const Unit& unit) {
        unitProductionQueue.push_back(unit);
    }

    bool CanProduce() const {
        // Mettez en place la logique pour vérifier si vous avez suffisamment de ressources, etc.
        return true;
    }

    void ProduceUnit() {
        if (CanProduce()) {
            // Mettez en place la logique pour produire une unité ici
            std::cout << "Production en cours..." << std::endl;
        }
        else {
            std::cout << "Impossible de produire l'unité. Ressources insuffisantes." << std::endl;
        }
    }
};
```

2. **Ouvrier** : Les ouvriers sont des unités de base qui peuvent être créées à partir de la couveuse. Ils peuvent ensuite être utilisés pour construire d'autres bâtiments Zergs ou collecter des ressources.

```cpp
#pragma once

#include <iostream>
#include <vector>

class Unit {
protected:
    std::string name;
    int health;
    int damage;

public:
    Unit(const std::string& n, int hp, int dmg) : name(n), health(hp), damage(dmg) {}

    // Implémentez ici les méthodes communes à toutes les unités
};
```

```cpp
class Ouvrier : public Unit {
private:
    int resourceCarried; // La quantité de ressources transportées par l'ouvrier

public:
    Ouvrier(const std::string& n, int hp, int dmg) : Unit(n, hp, dmg), resourceCarried(0) {}

    void CollectResources(int amount) {
        // Ajoutez le code pour la collecte de ressources ici
    }

    // Méthode pour construire une caserne
    Caserne BuildCaserne() {
        // Ajoutez le code pour construire une caserne ici
    }
};
```

3. **Caserne** : La caserne est un bâtiment qui peut être construit par les ouvriers. Elle permet la création d'unités de combat Zergs.

4. **Ressources** : Les Zergs ont besoin de ressources pour produire des unités et des bâtiments. Vous devrez mettre en place un système de collecte de ressources par les ouvriers.

```
class Ressources {
private:
    int quantity; // La quantité totale de ressources disponibles

public:
    Ressources(int initialQuantity) : quantity(initialQuantity) {}

    // Méthode pour obtenir la quantité de ressources
    int GetQuantity() const {
        return quantity;
    }

    void SetResources(int amount) {
        // Ajoutez le code pour ajouter/retirer des ressources ici
    }
};
```

Attentes du Projet

Lors de ce projet, nous attendons de vous :

- Implémentation des classes nécessaires pour représenter les couveuses, les ouvriers, les casernes et les ressources.

- Relation d'héritage appropriées entre les différentes unités Zergs.

- Système de production et de collecte de ressources fonctionnel.

- Interface utilisateur basique pour visualiser et interagir avec le monde du jeu (peut-être un mode textuel).

Avantages du Projet

La réalisation de ce projet présente plusieurs avantages :

- **Application Pratique** : Vous allez appliquer les concepts de programmation orientée objet et d'héritage dans un projet concret.

- **Amélioration des Compétences** : Vous gagnerez en expérience en travaillant sur un projet, ce qui renforcera vos compétences en programmation C++.

- **Créativité** : Vous aurez l'occasion de concevoir votre propre version du jeu StarCraft 2, en laissant libre cours à votre créativité.

Étapes d'Amélioration

Une fois le projet de base terminé, vous pouvez envisager les étapes suivantes pour améliorer votre projet :

1. **Ajout d'unités Zergs spéciales** : Intégrez de nouvelles unités Zergs spéciales avec des capacités uniques pour rendre le jeu plus intéressant (Mutalisk, Cafard etc..).

2. **Amélioration de l'IA** : Implémentez une intelligence artificielle pour les unités Zergs, leur permettant de prendre des décisions stratégiques (et automatisez le jeu).

3. **Interface Graphique Avancée** : Développez une interface graphique avancée pour améliorer l'expérience utilisateur.

Si vous avez besoin d'aide, ou si vous avez des questions, n'hésitez pas à me contacter sur discord : **vasrek**