Rapport

du Projet C++

WANG Hanshuo
WEI Yongtao

1. Description de l'application développée

Le thème de ce projet est "le monde d'après", nous espérons concevoir un programme qui soit à la fois lié à l'avenir et qui ait une signification pratique. Par conséquent, un programme qui prédit le développement du pays voit le jour.

Nous divisons la force d'un pays en trois modules principaux: force militaire, force technologique et économie. La force militaire est subdivisée en trois catégories: armée de terre, marine et force aérienne. Après avoir saisi les informations sur un pays, le programme calcule le taux de croissance de la force du pays dans chaque module. Lorsque nous saisissons les informations de plusieurs pays, le programme peut classer les pays en fonction du taux de croissance de la force scientifique et technologique. Vous pouvez également utiliser <,> , == pour comparer les taux de croissance de la vigueur de deux pays.

2. Mettre en valeur l'utilisation des contraintes

2.1 Class pays

```
class pays
{
public:
    pays(){}
    pays(string name);
    //pays(pays &_pays);
    ~pays();
    virtual int show_surface() const = 0;
    virtual int show_population() const = 0;
    virtual string show_name() const = 0;
    virtual double prediction() const = 0;
    virtual string evaluation() const = 0;
protected:
    string name;
};
```

C'est la classe de base, une classe virtuelle pure, cette classe contient toutes les méthodes utilisées par ses sous-classes, ainsi que le nom du pays.

2.2 Class paysInfo

```
class paysInfo : public pays
{
public:
    paysInfo();
    paysInfo(string _name, int surface,
                           int population);

    double prediction() const;
    string evaluation() const;
    string show_name() const;
    int show_surface() const;
    int show_population() const;

    ~paysInfo();
protected:
    string name;
    int surface;
    int population;
};
```

Puisque "pays" est une classe virtuelle pure, nous ne pouvons pas créer d'objets avec “pays”, nous utilisons donc ce sous-classe de pays, “paysInfo” pour créer un objet de classe contenant des informations de base sur le pays.

2.3 Class militaire

```
class militaire : public pays
{
public:
    //militaire();
    militaire(pays &country);
    ~militaire();
    virtual int show_surface() const = 0;
    virtual int show_population() const = 0;
    virtual string show_name() const = 0;
    virtual double prediction() const = 0;
    virtual string evaluation() const = 0;
protected:
    pays &_pays;
};
```

Cette classe purement virtuelle est également une sous-classe de “pays”. Nous l'utilisons pour représenter un module de force nationale: la force militaire.

2.4 Class aeriennes, terrestres et navales

```
class aeriennes: public militaire
{
public:
    aeriennes(pays &_pays, int population_ae, int nb_avion,
                           int rank_ae, int depense_ae);
    ~aeriennes();

    string show_name() const;
    double prediction() const;
    string evaluation() const;
    int show_surface() const;
    int show_population() const;

    friend ostream& operator<<(ostream & out, aeriennes &a);
    friend bool operator<(aeriennes &pays_1, aeriennes &pays_2);
    friend bool operator>(aeriennes &pays_1, aeriennes &pays_2);
    friend bool operator==(aeriennes &pays_1, aeriennes &pays_2);

private:
    int population_ae;
    int nb_avion;
    int rank_ae;
    int depense_ae;
};
```

Ces trois classes sont des sous-classes de “militaire”, elles ont des fonctions similaires mais correspondent à des armes différentes, je n'introduis donc que la classe “aeriennes”.
Le constructeur de “aeriennes” nous permet de saisir les informations de base de l'armée de l'air nationale: le nombre de soldats de l'armée de l'air, le nombre d'avions, les fonds investis cette année, et le classement mondial actuel. Grâce à la fonction de prédiction (), nous pouvons obtenir la valeur potentielle de croissance de la puissance navale du pays. Grâce à la fonction evaluation () et à la surcharge des opérateurs <<, nous pouvons sortir toutes les informations sur la puissance navale nationale actuelle. La surcharge des opérateurs <,>, == permet de comparer la valeur potentielle de deux objets “aériennes”.

2.5 Class technologie et economie

```
class technologie : public pays
{
public:
    technologie(pays &_pays, double frais_aug, double tech_population_aug,
                                                double rank_universite);
    ~technologie();
    double show_frais_aug() {return frais_aug;}
    double show_tech_population_aug() {return tech_population_aug;}
    double show_rank_universite() {return rank_universite;}
    double prediction() const;
    string evaluation() const;
    string show_name() const;
    int show_surface() const;
    int show_population() const;
    friend ostream& operator<<(ostream& out, technologie &pays);
    friend bool operator<(technologie &pays_1, technologie &pays_2);
    friend bool operator>(technologie &pays_1, technologie &pays_2);
    friend bool operator==(technologie &pays_1, technologie &pays_2);
private:
    pays &_pays;
    double frais_aug;
    double tech_population_aug;
    double rank_universite;
};
```

De même, “economie” et “technologie” sont également très similaires, nous nintroduisons donc que la technologie.
Pour cette classe, le constructeur initialise les variables privées de cette classe en passant des paramètres. En même temps, nous définissons trois fonctions pour imprimer des informations afin de fournir un accès indirect aux variables membres privées. Après cela, nous devons couvrir les cinq fonctions virtuelles pures dans la classe parente. Nous utilisons la surcharge d'opérateurs "<<" pour fournir un moyen pratique d'imprimer les informations d'évaluation et les informations de prévision. Enfin, nous utilisons trois fonctions de surcharge des opérateurs (<,>,=) pour comparer la force des deux pays.

2.6 Class rank

```
class ranks
{
public:
    ranks(vector<technologie> des_pays);
    ~ranks();
    string Meilleur_pays();
    vector<pair<string, float>> sort_pays();

private:
    vector<technologie> _des_pays;
};
```

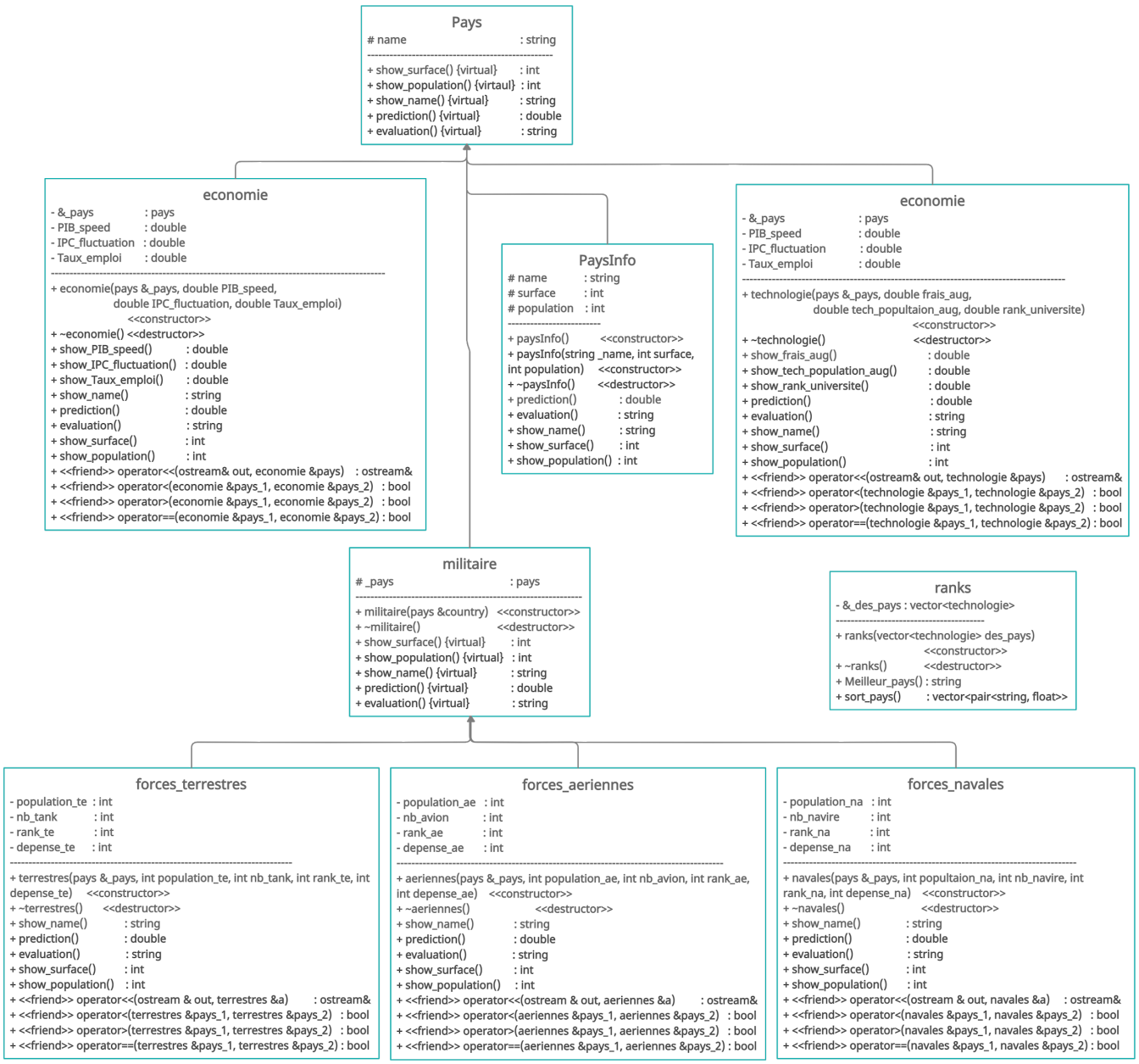
```
ranks::ranks(vector<technologie> des_pays):_des_pays(des_pays){}

ranks::~ranks(){}

//Trier par prévisions de pays
vector<pair<string, float>> ranks::sort_pays()
{
    map<string, float> technologie_rank;
    //Stockez le nom du pays et la valeur prévue correspondante dans le map
    for(auto &i:_des_pays)
    {
        technologie_rank.insert(pair<string,float>(i.show_name(),i.prediction()));
    }
    //Convertir le map en vecteur
    vector<PAIR> technologie_rank_vector(technologie_rank.begin(), technologie_rank.end());
    //Utilisez une structure personnalisée pour trier les éléments dans le vecteur
    sort(technologie_rank_vector.begin(), technologie_rank_vector.end(), CmpValue());

    return technologie_rank_vector;
}


string ranks::Meilleur_pays()
{
    //Fonction de classe d'appel
    vector<pair<string, float>> rank = sort_pays();
    stringstream out;

    cout << "pays\t" << "prediction\t"<< endl;

    //Imprimer les informations de commande de pays triées
    for(auto i:rank)
    {
        cout << i.first << "\t" << i.second << "\t" << endl;
    }
    //Imprimer le nom du pays avec la meilleure valeur prévue
    cout << "le Meilleur_pays est: " << rank[0].first << "avec" << rank[0].second << endl;

    return rank[0].first;
}
```

Pour la classe rank (), passez un vecteur de pays dans le constructeur et utilisez-le pour initialiser les variables de membre privé de cette classe. Nous définissons la fonction sort_pays () pour trier les pays. Cette fonction extrait les noms de pays et les valeurs prévues pour l'avenir en lisant des variables de membre privé, et les stocke dans le map sous la forme de paires clé-valeur, puis les trie via une structure personnalisée à des fins de comparaison. Nous avons également défini une fonction Meilleur_pays () pour afficher le nom du pays avec la meilleure valeur prédite. Cette fonction appelle la fonction sort_pays ().

2.7 TestCase

Nous avons utilisé un total de trois TEST_CASE et 32 assertions pour tester si les fonctions d'information d'impression (fonction show, fonction evaluation(), surcharge d'opérateur <<), surcharge d'opérateur <,>, == et class “rank” fonctionnent toutes correctement.

3. Diagramme UML de l'application



4. Exécution du code

```
yongtao@ubuntu:~/cpp/cpp_projet_try/tests$ ./testcase
pays_1:
1 soldats et 2 réservoirs dans l'armée de l'air;
Les dépenses de l'armée de l'air cette année sont de 1 euros; Classement mondial: 1
Taux de croissance estimé: 0.24

->pays_2
 @ pour l'instante:
   Le PIB_speed est:0.1
   Le IPC_fluctuation est:0.2
   Le taux de croissance de l'emploi est:0.4
   La situation économique est stable et les perspectives de développement sont bonnes!
 @ pour la future,le potentiel de croissance est: 0.3

->pays_3
 @ pour l'instante:
   Le PIB_speed est:0.2
   Le IPC_fluctuation est:0.2
   Le taux de croissance de l'emploi est:0.1
   Conditions économiques instables!
 @ pour la future,le potentiel de croissance est: 0.34


->pays_4
 @ pour l'instante:
   Le frais_aug est:0.5
   Le tech_population_aug est:0.4
   Le rank_universite est:0.1
   Bonnes perspectives d'évolution technologique et les perspectives de développement sont bonnes!
 @ pour la future,le potentiel de croissance est:3.8
```

```
->pays_5
 @ pour l'instante:
   Le frais_aug est:0.2
   Le tech_population_aug est:0.1
   Le rank_universite est:0.3
   Faibles perspectives de développement technologique!
 @ pour la future,le potentiel de croissance est:1.8

->pays_6
 @ pour l'instante:
   Le frais_aug est:0.5
   Le tech_population_aug est:0.5
   Le rank_universite est:0.4
   Bonnes perspectives d'évolution technologique et les perspectives de développement sont bonnes!
 @ pour la future,le potentiel de croissance est:4.8

->pays_7
 @ pour l'instante:
   Le frais_aug est:0.2
   Le tech_population_aug est:0.3
   Le rank_universite est:0.2
   Bonnes perspectives d'évolution technologique et les perspectives de développement sont bonnes!
 @ pour la future,le potentiel de croissance est:2.4

pays    prediction
pays_6  4.8
pays_4  3.8
pays_7  2.4
pays_5  1.8
===============================================================================
All tests passed (32 assertions in 3 test cases)
```

5. La partie de l'implémentation les plus fières

Nous pensons que le contenu le plus technique est la classe définie par "rank.hh". Cette classe utilise deux conteneurs: vecteur et map. Nous mettons de nombreux pays dans un vecteur et mettons les informations à comparer dans le map sous forme de paires clé-valeur. Après cela, nous convertissons le map en vecteur et comparons les valeurs avec la structure définie "CmpValue". Dans l'implémentation de la fonction sort_pays (), nous retournons un vecteur avec des objets de classe comme membres, afin que cette fonction puisse jouer un grand rôle lorsqu'elle est appelée par Meilleur_pays ().