Paul Clas 1846912

Gaspar Faure 1903623

TP5

LOG1000

10 avril 2018

École Polytechnique

Répartition des points de l’évaluation [/80]:

Respect des formats et consignes demandées pour la rédaction du rapport. [/4]

E1) Une mauvaise odeur dans les attributs [/32]

1) [/2]

Le fait d’avoir des attributs qui pourrait remplir le rôle d’attribut dans une autre classe dans une même classe est une odeur grave. Cela rend la classe grosse, ce que l’on veut éviter.

2) [/2]

Il faudrait extraire la classe équipe car « lorsqu’on se retrouve avec une classe qui fait le travail qui devrait être fait par deux classes. Il faut créer une nouvelle classe et déplacer les attributs et méthodes pertinents vers la nouvelle classe. » tutoriel.pdf

3) [/6]

Les methodes qui seront impactées par ce changement sont :

- Joueur(string, string, int, string, string);

- string getNomEquipe();

- string getPaysEquipe();

- void associerEquipe(string, string);

- void saveJoueur(string);

- void afficher();

- Joueur\* trouverJoueur(string, string,string);

4) [/4]

Les attributs qu’il faut modifier par ce changement sont :

- string teamName;

- string teamPays;

5) [/6]

Étape

Description

Extraire Classe

On créer la classe Équipe a laquelle on donne les attributs teamName et teamPays et on ajoute un pointeur vers equipe pour chaque joueur.

Supprimer attributs/paramètres inutile

On supprime teamName et teamPays de Joueur()et de trouverJoueur().

Création des attributs dans la classe Équipe.

Creation des attributs teamName et teamPays.

Création constructeur dans la classe Équipe.

Création d’un constructeur Équipe avec les paramètres teamName et teamPays

Création des méthodes dans la classe Équipe.

Création des getNomEquipe() et getPaysEquipe() dans la classe Équipe.

6) [/6]

7) [/4]

8) [/2]

E2) Une mauvaise odeur dans les méthodes [/22]

1) [/2]

Les deux mauvaises odeurs sont : méthode longue et code dupliqué.

Les méthodes longues sont des odeurs graves, car elles nuisent beaucoup à la compréhension du code, il faut plus de temps et d’analyse du code pour comprendre ce qu’il veut dire et faire. Le code dupliqué, en plus de rendre la méthode encore plus longue, rend la méthode plus complexe en apparences

2) [/6]

|  |  |
| --- | --- |
| Étape | Description |
| Extraire Méthode | Extraire une méthode que nous appellerons extraireChaine() de la fonction trouverJoueur() |
| Utiliser la nouvelle méthode | Remplacer les utilisations abusives de la boucle for par Une utilisation de la fonction extraireChaine() |

3) [/6]

string extraireChaine(int & debutLecture, string ligne) {

string retour;

int i;

for (i = debutLecture ; i < ligne.length() ; i++) {

if (ligne[i] != ',') {

retour += ligne[i];

} else {

break;

}

}

debutLecture = i;

return retour;

}

// Trouver un Joueur avec son nom dans la base de données DB

Joueur\* Joueur::trouverJoueur (string DB, string nom,string prenom) {

ifstream fichier(DB.c\_str(), ios::in); // Ouvrir le fichier "DB.txt"

Joueur\*tmp=NULL;

if(fichier)

{

string line;

// Lire les Joueurs, un Joueur par ligne dans la base de données (DB)

while (getline(fichier, line)) {

// Récupérer le nom de joueur

int i = 0;

string nomJoueurDB = extraireChaine(i,line);

// Si le Joueur qu'on lit actuellement est celui qu'on cherche

if (nomJoueurDB == nom) {

// Récupérer le prenom du joueur

string prenomJoueurDB = extraireChaine(++i, line);

if (prenomJoueurDB == prenom) {

// Récupérer le nom de l'age

string ageJoueurDB = extraireChaine(++i, line);

// Récupérer le nom de l'équipe

string teamNameDB = extraireChaine(++i, line);

// Récupérer le pays de l'équipe

string teamPaysDB = extraireChaine(++i, line);

// Créer un objet de type Joueur avec les informations récupérées

Joueur \*a = new Joueur(nomJoueurDB, prenomJoueurDB,atoi(ageJoueurDB.c\_str()), teamNameDB, teamPaysDB);

// Fermer la base de données

fichier.close();

// Retourner l'Joueur sélectionné

return a;

} // end if (prenomJoueurDB == prenom)

} //end if (nomJoueurDB == nom)

}// end while

fichier.close();

}// end if(fichier)

return NULL;

}

4) [/4]

5) [/2]

E3) Utilisation des variables [/22]

1) [/6]

|  |  |  |  |
| --- | --- | --- | --- |
|  | Span | Duree de vie | Portee des variables |
| DBFile | (39 + 20)/2 = 29,5 | 73 – 12 = 61 | 80-12= 68 |
| choix | (8+0+54)/3 = 20,6 | 78 – 13 = 65 | 80-13 = 67 |
| Joueur | (27+0+9+1+8+0+7+0)/8= 6,5 | 73 – 11 = 62 | 80-11 = 69 |

2) [/2]

La variable DBFile profitera le plus de la restructuration car cette variable possède le plus grand span pour une porte de variable et de vie très grande.

3) [/4]

|  |  |
| --- | --- |
| Étape | Description |
|  |  |

4) [/4]

5) [/4]

6) [/2]