RAPPORT
PROJET JAVA – SEMESTRE 1

GROUPE - AUBRY HERMAL

07/01/2015

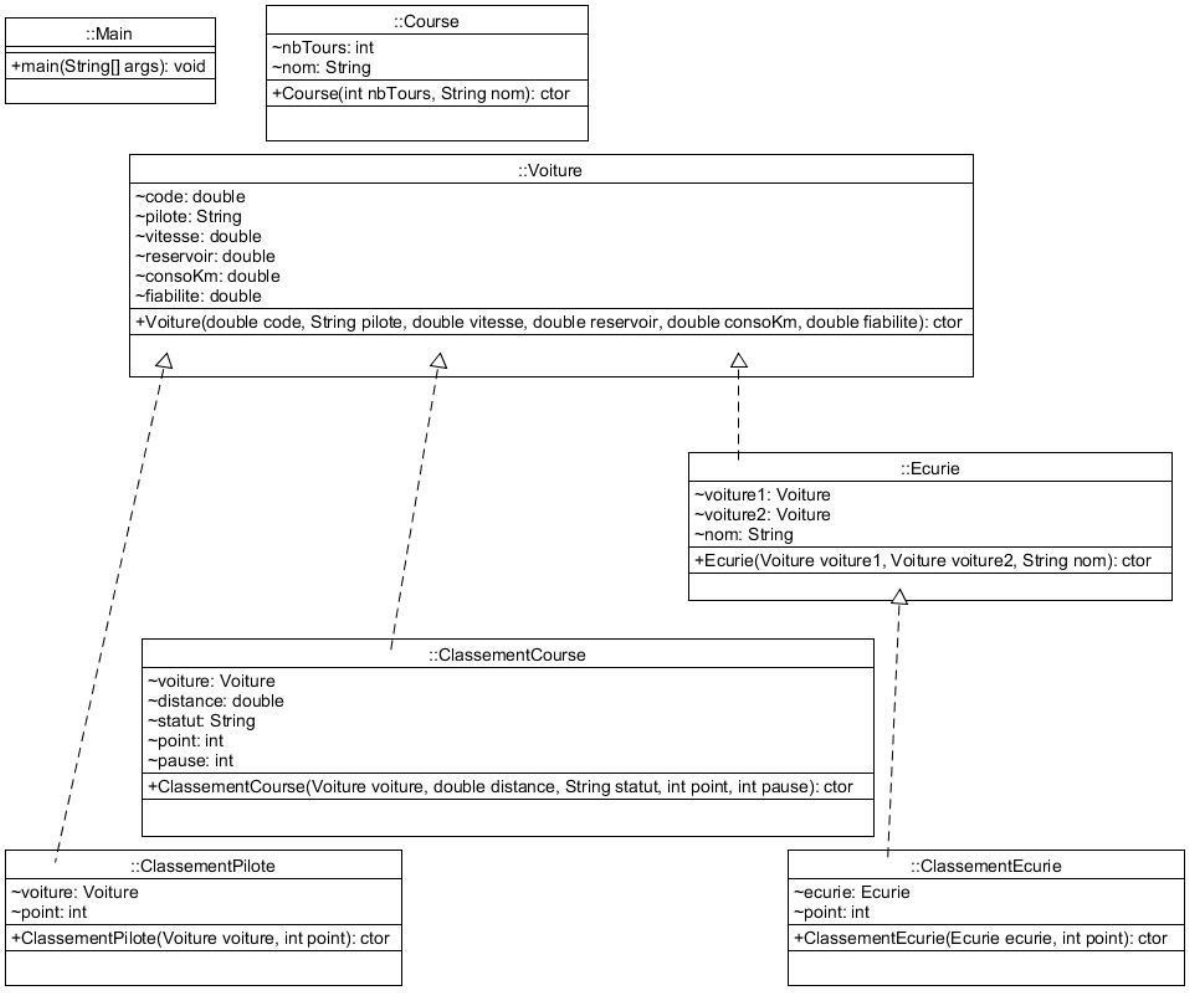
TABLE DES MATIÈRES

Table des matières

Présentation du projet

Le projet nous ayant été donné de faire, en utilisant le langage Java était d'écrire un programme permettant de simuler les courses d'une saison de Formule 1. Les concurrents se composant de 10 écuries de 2 voitures chacune. Chaque voiture ayant un réservoir se vidant à chaque fois qu'il y a déplacement, des chances de tomber en panne ou de rentrer dans une autre voiture lors d'un dépassement afin de rendre la simulation « plus vivante ».

DIAGRAMME DE CLASSES



PRESENTATION DU PROJET

ETAPES DE REALISATION DU PROJET

Nous avons afin de mener à bien le projet, divisé notre travail en plusieurs étapes. Tout d'abord nous avons fait un premier diagramme de classe sur lequel nous nous sommes basé afin de commencer la partie codage.

Nous avons ensuite dû procéder à quelques changements au niveau du diagramme car nous avons ajouté d'autres attributs à plusieurs de nos classes, tels que le statut pour savoir si une voiture est en course ou hors course par exemple.

Ensuite, nous sommes passés à la réalisation des classes en Java à l'aide d'Eclipse, l'IDE que nous avons tous les deux utilisé. Puis après la création des classes, nous avons commencé à créer nos premiers constructeurs. Nous avons donc crée les circuits, les pilotes, les écuries etc. On a ensuite procédé à des tests pour vérifier que les voitures « avançaient » bien avant d'implémenter les collisions lors des dépassements, les arrêts aux stands, les avaries et les changements de courses.

Enfin, nous avons terminé avec la gestion du classement par pilote et par constructeur (écurie) au fil de la course.

DOCUMENTATION DU CODE SOURCE

Verification du réservoir

```
//Vérification du réservoir

//Le réservoir diminue
Course[i].voiture.reservoir += -Course[i].voiture.consoKm;

//Si il n'a plus assez d'essence pour terminer ce tour
if (Course[i].voiture.reservoir - Course[i].voiture.consoKm < 0){
    Course[i].voiture.reservoir += 30;

    //La voiture rentre aux stands
    Course[i].statut = "Stand";
    System.out.println(Course[i].voiture.pilote +" rentre aux stands pour faire le plein d'essence");
}
```

La vérification du réservoir se fait après chaque boucle, on vérifie donc si la consommation d'essence est supérieure moins le réservoir de la voiture est supérieur à 0. Si c'est le cas on passe à la suite, sinon le véhicule rentre dans les stands et rempli son réservoir.

Arrêt aux stands

```
//La voiture est aux stands
if (Course[i].statut == "Stand"){
    Course[i].pause +=1;

    //La pause de 10 secondes est finie
    if (Course[i].pause >= 10){

        //Le pilote repart en course
        Course[i].statut = "En course";
        Course[i].pause = 0;
    }
}
```

Lorsqu'une voiture s'arrête aux stands voici ce qu'il se passe :

A chaque tour de boucle on incrémente pause de 1 si la condition Stand est vérifiée. Une fois pause supérieure ou égale à 10 on fait sortir la voiture des stands et elle reprend la course.

Dépassement et collision

```
//Dépassement
if (Course[j].distance >= Course[j-1].distance){

    //Les 2 voitures sont en course, donc on effectue un test de collision
    if ((Course[j].statut == "En course" || Course[j].statut == "Avarie") && (Course[j-1].statut == "En course" || Course[j-1].statut == "Avarie")){
        double collision = Math.random();

        //Si il y a collision
        if (collision < 0.0025){

            //Les deux pilotes ne sont plus en course
            Course[j].statut = "Course terminée";
            Course[j-1].statut = "Course terminée";
            Course[j].distance = 0;
            Course[j-1].distance = 0;
            System.out.println("Collision entre "+ Course[j].voiture.pilote + " et " + Course[j-1].voiture.pilote);
        }
    }

    //Echange des positions
    ClassementCourse aux = Course[j];
    Course[j] = Course[j-1];
    Course[j-1] = aux;
}
```

Si la voiture à une position j a parcouru une distance supérieure ou égale à une voiture en position j-1, alors il y a dépassement. Lorsqu'il y a dépassement on effectue un test de collision, collision est une variable à laquelle on assigne un nombre aléatoire compris entre 0 et 1. Si ce nombre aléatoire est inférieur à 0,0025 alors on a collision, les statuts des deux voitures sont alors mis à jour et la course est terminée pour eux. Si il n'y a pas collision on échange simplement les positions des deux voitures. On n'effectue pas de test de collision lorsqu'une voiture dépasse une voiture aux stands.

Avarie et panne

```
//Test avarie
double avarie = Math.random();
if(avarie < Course[i].voiture.fiabilite){

    //La voiture subit une avarie
    Course[i].statut = "Avarie";
    System.out.println(Course[i].voiture.pilote +" subit une avarie !");
}

//Test panne
double panne = Math.random();
if(panne < Course[i].voiture.fiabilite){

    //La voiture subit une panne
    Course[i].statut = "Course terminée";
    Course[i].distance = 0;
    System.out.println(Course[i].voiture.pilote +" subit une panne, il n'est plus dans la course !");
}
```

Les tests d'avarie et de panne sont assez similaires au test de collision, en effet on applique aussi la méthode de la variable à valeur aléatoire et l'on regarde si la fiabilité de la voiture sur laquelle on fait le test est supérieur à cette variable générée, on commence par l'avarie qui donne le statut avarie qui ne s'appliquera qu'au prochain avancement de ce véhicule. On vient ensuite au test de panne, si la fiabilité de la voiture est inférieure à la panne alors la course pour cette voiture est terminée, et elle n'avancera plus.

```
//La voiture avance
if (Course[i].statut == "En course"){
    Course[i].distance += Course[i].voiture.vitesse;
} else if (Course[i].statut == "Avarie"){
    Course[i].distance += 0.8*Course[i].voiture.vitesse;
}
```

On voit ici qu'au début de chaque avancé si la voiture est en course mais qu'elle a le statut avarie, elle n'avancera qu'à 80% de sa vitesse habituelle.

Classement

```
//Tri du classement des pilotes
for (int i=0 ;i<19;i++){
    for (int j=19;i < j;j--){
        if(TotalPilote[j].point >= TotalPilote[j-1].point){
            ClassementPilote aux = TotalPilote[j];
            TotalPilote[j] = TotalPilote[j-1];
            TotalPilote[j-1] = aux;
        }
    }
}
```

Le tri du classement des pilotes s'effectue de la manière suivante : Si le pilote à la position j a plus de points que le pilote à la position j-1, on inverse leurs positions et on continue les tests avec le pilote j-2 etc. C'est un tri à bulle.

CONCLUSION

Conclusion

Le projet Java s'est déroulé plutôt corectement , nous avons eu quelques difficultés dans un premier temps pour décider comment savoir qu'une voiture venait juste de passer un tour ou alors si elle était toujours dans ce tour afin de diminuer la réserve de carburant à l'interieur du réservoir et ensuite procéder aux tests.

De même pour les collisions nous avions besoin des dépassements et donc d'ordonner la position des voitures en fonction de leur distance parcouru, nous avons fini par résoudre ce problème en triant grâce au tri à bulle et en testant à chaque nouveau tour par la suite la collision

Nous pourrions améliorer la « simulation » en changeant la taille des circuits, en donnant différents types de pneus par exemple ayant des charactéristiques différentes, en implémentant un système météorologique augmentant plus ou moins les risques d'accident et l'usure des pneus. Nous pourrions ajouter d'autres circuits aussi afin de créer des saisons aléatoire en piochant dans une liste de circuits. Une interface graphique pourrait aussi être la bienvenue. C'est une liste non exhaustive, mais déjà importante de ce que nous pourrions changer pour rendre ce projet meilleur.