

Exercice 1 :

Un championnat de football est une suite de matchs de football, dans lesquels se confrontent plusieurs équipes sur plusieurs journées. Chaque équipe joue plusieurs matchs contre plusieurs équipes. Les équipes cumulent des points dans un championnat selon les règles suivantes :

- une équipe qui perd un match, cumule **0 point**.
- une équipe qui gagne un match, cumule **3 points**.
- un match nul (même nombre de buts dans pour les deux équipes) implique un cumul de **1 point** pour les deux équipes.

A la fin de tous les matchs, L'équipe qui aura le nombre de point le plus élevé sera consacrée « équipe championne ».

Travail à faire :

On veut créer un programme pour le suivi d'un championnat de football, pour cela, on réalisera deux classes : **MatchFoot** et **Championnat**

1. La classe **MatchFoot** modélise un match entre deux équipes A et B. Elle dispose des membres suivants :
 - **equipeA**, **equipeB** de type chaîne de caractères représentant les noms des deux équipes
 - **nbrButA**, **nbrButB**, deux entiers qui représentent respectivement le nombre de buts de **equipeA** et **equipeB**
 - un constructeur ayant comme paramètres deux chaînes de caractères : le nom de l'équipe A et celui de l'équipe B.
 - les méthodes getters nécessaires (pas de setter)
 - La méthode **marquerBut(String eq)** : deux valeurs possibles pour **eq** : "A" pour rajouter un but pour **EquipeA** et "B" pour rajouter un but **EquipeB**
2. La classe **Championnat** dispose des propriétés suivantes :
 - **tabMatches** : tableau d'objets de type **MatchFoot**
 - **nbrMatches** : le nombre de match dans **tabMatches**
 - **tabEqPts** : tableau de deux dimensions de type String qui contient dans chaque ligne, le nom d'une équipe et le nombre de points de cette équipe dans le championnat, exemple :

Nom équipe		Nombre de points	
0	"Algérie"	"4"	
1	"France"	"0"	
2	"Brésil"	"4"	

La classe **Championnat** définit les méthodes suivantes :

- un constructeur sans paramètre qui initialise le tableau **tabMatches** (40 matchs maximum), et **tabEqPts** (10 équipes maximum)
- une méthode **void ajouterMatch(MatchFoot mf)** : rajoute un objet match à **tabMatches**, cette méthode doit gérer la taille maximale du tableau **tabMatches**
- une méthode **void listeEquipes()** : qui remplit la 1^{ère} colonne du tableau **tabEqPts** par les équipes participantes au championnat sans répétition.
- **int pointParEquipe (String eq)** : qui pour une équipe **eq** donnée, calcule nombre de points cumulés dans le championnat. (suivant les règles citées au début de l'exercice)
- Une méthode **void calculeNbPoint ()** : qui fait appel à la méthode **pointParEquipe**, pour remplir la 2^{ème} colonne du tableau **tabEqPts**.
- Une méthode **toString()** qui retourne une chaîne de caractère qui représente le tableau **tabEqPts** une fois remplie. Exemple :

Equipe	Points
Algérie	4
France	0
Brésil	4

- Pour tester la solution, dans la méthode **main** doit disposer de :
 - Un objet **Championnat** initialement vide.

- Trois (03) objets **MatchFoot** entre trois 3 équipes : "Algérie", "France", "Brésil" avec des scores (résultat de match) de votre choix. Ces trois matchs composent le championnat
- Affichage du tableau des points des équipes participantes au championnat

Exercice2 :

Donner, **avec justification**, le résultat de l'exécution du programme suivant :

```
1    class A{
2        int x=1;
3        A(){ }
4        A(int t){x=t-1;}
5        void f(){System.out.println("Je suis f, je travaille pour A, x="+x);}
6    }

7    class B extends A{
8        int y;
9        B(){ }
10       B(A aa){super(aa.x); y=aa.x-2;}
11       void f(){System.out.println("Je suis f, je travaille pour B, x="+x+", y="+y);}
12       void g(){ }
13   }

14   public class TestAB {
15       public static void main(String[] args){
16           A a=new A();
17           a.f();
18           A b=new B(a);
19           b.f();
20       }
21   }
```

Corrigé type de l'examen P00

Exercice n°1: (17/17)

class MatchFoot {	3/3
private String equipeA, equipeB;	0.25
private int nbrButA, nbrButB;	0.25
public MatchFoot(String equipeA, String equipeB){	0.5
this.equipeA=equipeA;	
this.equipeB=equipeB;	
}	
public String getEquipeA() { return equipeA;}	0.25
public String getEquipeB() { return equipeB;}	0.25
public int getNbrButA() { return nbrButA;}	0.25
public int getNbrButB() { return nbrButB;}	0.25
public void ajouterBut(String eq){	
if (eq.equals("A")) nbrButA++;	0.5
else if (eq.equals("B")) nbrButB++;	0.5
}	
}	
class Championnat{	12/12
public static final int MAX_MATCH_IN_TAB=40;	
private int nbrMatch; // Nombre de match dans le tableau tabMatches	0.5
private MatchFoot[] tabMatches; // tableaux des matchs du championnat	0.5
private String tabEqPts[][]; // tableau des équipes participantes au championnat	0.5
public Championnat(){	1
tabMatches=new MatchFoot[MAX_MATCH_IN_TAB];	
tabEqPts=new String[10][2];	
}	
public void AjouterMatch(MatchFoot mf){	1
int i=nbrMatch;	
if(i<MAX_MATCH_IN_TAB){	
tabMatches[i]=mf;	
nbrMatch++;	
}	
}	
public void ListeEquipe(){	3
int k=0;	
for(int i=0;i<nbrMatch;i++){	
boolean trouve1=false, trouve2=false;	
int j=0;	
while(tabEqPts[j][0]!=null && !trouve1){	
if(tabMatches[i].getEquipeA().equals(tabEqPts[j][0])) trouve1=true;	
j++;	
}	
j=0;	
while(tabEqPts[j][0]!=null && !trouve2){	
if(tabMatches[i].getEquipeB().equals(tabEqPts[j][0])) trouve2=true;	
j++;	
}	
if (!trouve1) { tabEqPts[k][0]=tabMatches[i].getEquipeA(); k++; }	
if (!trouve2) { tabEqPts[k][0]=tabMatches[i].getEquipeB(); k++; }	
}	
}	
private int pointParEquipe(String eq){	2.5
int nbrPoint=0;	
for(int i=0;i<nbrMatch;i++){	
if(tabMatches[i].getEquipeA().equals(eq)) {	
if(tabMatches[i].getNbrButA()==tabMatches[i].getNbrButB()) nbrPoint++;	
else if(tabMatches[i].getNbrButA()>tabMatches[i].getNbrButB())nbrPoint+=3;	
}	
}	

