Projet LU2IN002 - 2020-2021

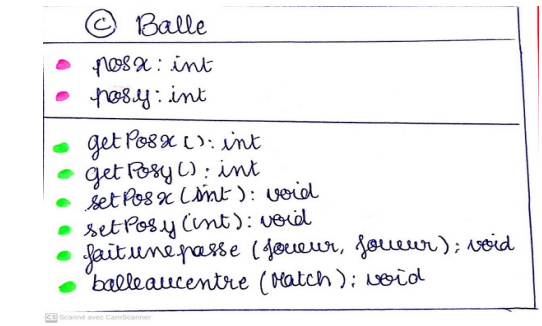
Numéro du groupe de TD/TME : Groupe 7

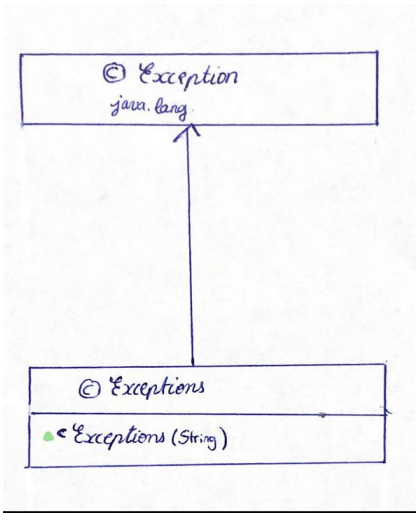
Nom : VEDANAYAGAME	*Nom :* MASTOR	*Nom :*
Prénom : Souveda	*Prénom :* Assia	*Prénom :*
N° étudiant : 28600150	*N° étudiant :* 28602563	*N° étudiant :*

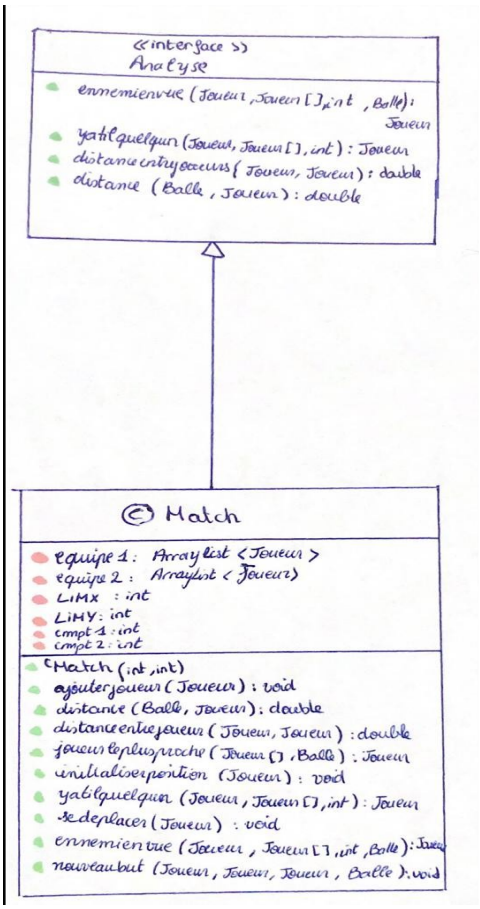
Thème choisi (en 2 lignes max.)
Simulation d'un match de handball ayant lieu dans un club où deux équipes s'affrontent

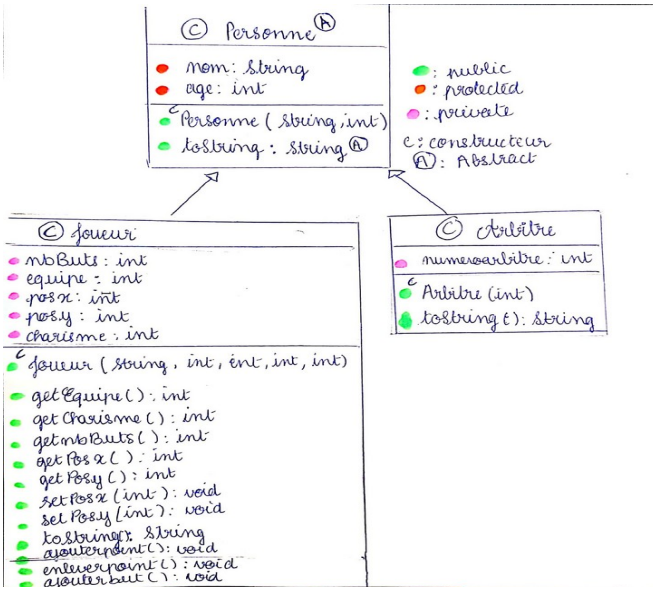
Description des classes et de leur rôle dans le programme (2 lignes max par classe)
Personne: classe mère des classes Joueur et Arbitre Joueur: modèlise un joueur qui joue dans le match Arbitre: modélise un arbitre du match Match: contient toutes les méthodes permettant le déroulement d'un match Balle: modélise une balle, avec sa position sur le terrain Club: permet l'inscription des joueurs et des arbitres pour le match TestMatch: simule/ teste le développement d'un match

Schéma UML des classes vision fournisseur (dessin "à la main" scanné ou photo acceptés)









Checklist des contraintes prises en compte:	*Nom(s) des classe(s) correspondante(s)*
Classe contenant un tableau ou une ArrayList	Match, TestMatch, Club
Classe avec membres et méthodes statiques	Club
Classe abstraite et méthode abstraite	Personne
Interface	Match (interface Analyse)
Classe avec un constructeur par copie ou clone()	Club
Définition de classe étendant Exception	Exceptions
Gestion des exceptions	Testmatch
Utilisation du pattern singleton	

Présentation de votre projet (max. 2 pages) : texte libre expliquant en quoi consiste votre projet.

Notre projet consiste à modéliser le déroulement d'un match dans un club, la classe Club illustrant l'inscrption des joueurs et des arbitres pour le match. Pour cela, nous nous placons sur un terrain. Deux équipes s'affrontent. Ces équipes sont constituées à partir de la classe Club qui contient toutes les méthodes nécessaires. Chacun des joueurs aura une postion x et une postion y, qui réprésentent son postionnement sur le terrain.

Un arbitre sera là pour assurer le bon déroulement du match. Il vérfiera si les joueurs sont bien sur le terrain, qu'ils ne font pas de faute éliminatoire…

La position des joueurs sur Le terrain est initialisée alétoirement.
Nous avons créé une méthode "joeuuerleplusproche" dans le match qui permet de savoir lequel des joueurs est le plus proche de la balle. Ce dernier récupère donc la balle au début du match.
Ensuite, plusieurs méthodes ont été créées permettant à ce dernier de faire la passe à un joueur de son équipe. Sinon, la balle est reprise par un de ses adversaires. L'attribut equipe de la classe Joueur permet de savoir à quelle équipe un joueur appartient. En fonction de celle-ci, Le joueur se déplacera soit vers le gardien g1 (de l'équipe 1) soit vers Le gardien g2 (de l'équipe 2).
Une fois arrivé près de la cage, la position de la balle est changée aléatoirement, permettant de modéliser un but. Cette position est comparée à celle du gardien en question afin de savoir si l'équipe a marqué un but ou pas. Des compteurs ont été créés pour pouvoir compter le nombre de buts de chacune des équipes. Lorsqu'une des équipes aura marqué 5 buts, la match s'arrête et nous auront un gagnant.

Copier / coller vos classes et interfaces à partir d'ici :

//Personne

```
public abstract class Personne {

protected String nom;
protected int age;


protected Personne (String nom, int age){


this.nom=nom;
this.age=age;
}


public abstract String toString();
}
```

//classe joueur

```
public class Joueur extends Personne  {


private int nbButs;
private int charisme;
private int equipe;
private int posx, posy;
```

```
public Joueur (String nom, int age, int nbButs,int equipe){
super(nom,age);
this.nbButs=nbButs;
this.charisme=charisme;
this.equipe=equipe;
}

public int getEquipe(){
return equipe;
}

public int getnbButs(){
return nbButs;
}



public int getCharisme(){
return charisme;
}
public void setPosx(int pos){
this.posx=pos;
}
public void setPosy(int pos){
this.posy=pos;
}

public int  getPosx(){
return posx;
}

public int  getPosy(){
return posy;
}

public String toString(){
return nom+","+age+"ans. je fais partie de l'équipe numéro "+equipe+" et j'ai une position de
x:"+posx+" y: "+posy;
```

```
}



public void enleverpoint(){
this.charisme=charisme-1;

}

public void ajouterpoint(){
this.charisme=charisme+1;

}

public void ajouterbut(){
this.nbButs=nbButs+1;
}



}
```

//classe Balle

```
public class Balle{

private int posx;
private int posy;

public int getPosx(){

return posx;
}

public int getPosy(){
return posy;

}
```

```
public void setPosx(int newposx){

this.posx=newposx;

}

public void setPosy(int newposy){
this.posy=newposy;

}

public void faitunepasse(Joueur a, Joueur b){
this.posx=b.getPosx();
this.posy=b.getPosy();

}

public String toString(){
return "position x:"+posx+" position y:"+posy;

}

public void balleaucentre(Match m1){
this.posx=((int)(m1.LIMX/2)); //on pose la balle au centre du terrain
this.posy=((int)(m1.LIMY/2));
}

}
```

// interface ANALYSE

```
public interface analyse {
public Joueur ennemienvue(Joueur jqdlb, Joueur joueurt[],int distancemax,Balle balle);
public Joueur yatilquelqun( Joueur jlpp,Joueur joueurt[],int distancemax);
public double distanceentrejoueurs(Joueur a, Joueur b);
public double distance(Balle balle, Joueur j);
```

```
}
```

//classe match

```
import java.util.*;
import java.lang.Math;

public class Match implements analyse {
private ArrayList<Joueur> equipe1=new ArrayList<Joueur>();
private ArrayList< Joueur > equipe2=new ArrayList<Joueur>();
public int LIMX=70;
public int LIMY=100;
public int cmpt1;
public int cmpt2;
public Match(int cmpt1, int cmpt2) {
this.cmpt1=cmpt1;
this.cmpt2=cmpt2;
}



public void ajouterJoueur(Joueur i){
if (i instanceof Joueur){

if (i.getEquipe()==1) {
if (equipe1.size()<7){
equipe1.add(i);
}
}

else {
if (equipe2.size()<7){
equipe2.add(i);
}
}

}
}
```

```
public double distance(Balle balle, Joueur j){
return Math.sqrt(( Math.pow((double) ( (balle.getPosx())-(j.getPosx()) ),2.0 ) )   +
(Math.pow((double) ( balle.getPosy()-j.getPosy() ),2.0))   ); //formule de la distance entre 2 points

}


public double distanceentrejoueurs(Joueur a, Joueur b){
double diffx=Math.abs((double)(a.getPosx()-b.getPosx()));
double diffy=Math.abs((double)(a.getPosy()-b.getPosy()));
double tot= Math.sqrt(Math.pow(diffx,2.0)+Math.pow(diffy, 2.0));
return tot;
}



public Joueur joueurleplusproche(Joueur tab[],Balle b){
Joueur leplusproche=tab[0];
int i;
for(i=1;i<tab.length;i++){
if (distance(b,tab[i])<distance(b,leplusproche)){
leplusproche=tab[i];
}
}
return leplusproche;
}

public void initialiserposition(Joueur i){
i.setPosx((int)(Math.random()*LIMX));
i.setPosy((int)(Math.random()*LIMY));

}
```

```
public Joueur yatilquelqun( Joueur jlpp,Joueur joueurt[],int distancemax){
if (jlpp.getEquipe()==1){ // si le joueur le plus proche fait partie de l'équipe 1 alors:

for (int i=0;i<joueurt.length;i++){ //on parcourt la liste des joueurs
if(joueurt[i].getEquipe()==1){ // si le joueur fait egalement parti de l'équipe 1
if ((distanceentrejoueurs(jlpp,joueurt[i])<distancemax)&&(jlpp.getPosy()>joueurt[i].getPosy())){ // si la distance entre ces deux joueurs est inférieure à la distance maximum possible pour faire une passe et que le second joueur est plus avancé dans le terrain
return joueurt[i];
}
}
}
return jlpp;
}

else{
for (int i=0;i<joueurt.length;i++){ //on parcourt la liste des joueurs
if(joueurt[i].getEquipe()==2){ // si le joueur fait egalement parti de l'équipe 1
if ((distanceentrejoueurs(jlpp,joueurt[i])<distancemax)&&(jlpp.getPosy()<joueurt[i].getPosy())){ // si la distance entre ces deux joueurs est inférieure à la distance maximum possible pour faire une passe et que le second joueur est plus avancé dans le terrain ( de l'autre coté)
return joueurt[i];
}
}
}
return jlpp;
}

}

public void sedeplacer(Joueur joueur){
if (joueur.getEquipe()==1){
joueur.setPosy(joueur.getPosy()-1);
}
else {
joueur.setPosy(joueur.getPosy()+1);
}

}
```

```
public Joueur ennemienvue(Joueur jqdlb, Joueur joueurt[],int distancemax,Balle balle){
if( jqdlb.getEquipe()==1){ //si jqdlb fais parti de l'equipe 1
for (Joueur j:equipe2){
if( distanceentrejoueurs(jqdlb, j)<=distancemax){ // et que un joueur de l'equipe adverse est près
balle.faitunepasse(jqdlb,j); // alors le joueur de l'équipe adverse lui prend la balle
return j; // et on retroune ce joueur
}
}
return jqdlb; } //sinon on retourne le joueur de base


else {
for (Joueur j:equipe1){
if( distanceentrejoueurs(jqdlb, j)<=distancemax){
balle.faitunepasse(jqdlb,j);
return j;
}
}
return jqdlb;
}
}


public void nouveaubut(Joueur jqdlb,Joueur g2, Joueur g1, Balle b) {
if (jqdlb.getEquipe()==1){
b.setPosx((int)(Math.random()*(LIMX+40))); //le plus 5 c'est pour permettre qu'il y ai faute de
temps en temps, le tire est mal fait et la balle sort
b.setPosy((int)(Math.random()*(30+40)));
if (b.getPosx()!=g2.getPosx() && b.getPosy()!=g2.getPosy()&& b.getPosy()<g2.getPosy()) {
this.cmpt1++;
System.out.println ("but marqué par l'equipe 1");
jqdlb.ajouterpoint();
jqdlb.ajouterbut();
}
else {
System.out.println ("but raté par equipe 1");
jqdlb.enleverpoint();
}
}
```

```
else {
b.setPosx((int)(Math.random()*(LIMX)));
b.setPosy((int)((LIMY-(30)+(Math.random()*((LIMY+40)-(LIMY-(30)))))));
if (b.getPosx()!=g1.getPosx() && b.getPosy()>g1.getPosy()&& b.getPosy()!=g1.getPosy()) {
this.cmpt2++;
System.out.println ("but marqué par l'equipe 2");
jqdlb.ajouterpoint();
jqdlb.ajouterbut();
}
else {
System.out.println ("but raté par l'equipe 2");
jqdlb.enleverpoint();
}
}
}
}
```

// classe club

```
import java.util.*;
public class Club {
private  ArrayList< Personne > membrejoueur=new ArrayList<Personne>();
private ArrayList< Arbitre > membrearbitre=new ArrayList<Arbitre>();


private static  int nbMaxJoueur=30;
private static int nbMaxArbitre=6;
private static String nomduclub="Queen club";


public Club( Club c){


this.nbMaxJoueur= c.nbMaxJoueur;
this. nbMaxArbitre=c.nbMaxArbitre;
this.nomduclub= c. nomduclub;
}


public Club(){
this.nbMaxJoueur= 200;
this. nbMaxArbitre=30;


}
```

```
public static String getnom(){
return nomduclub;
}


public void ajouterMembreJoueur(Joueur p) throws Exceptions{

if(p.age<15) {
throw new Exceptions("l'age doit etre supérieur à 15 ans, il ne peut donc pas etre inscrit ");
}
if (membrejoueur.size()<nbMaxJoueur) {
membrejoueur.add(p);
}

else {
System.out.println("Il n'y a plus de place !");
}

}

public void ajouterMembreArbitre(Arbitre p) throws Exceptions{

if(p.age<15) {
throw new Exceptions("l'age doit etre supérieur à 15 ans, il ne peut donc pas etre inscrit");
}
if (membrearbitre.size()<nbMaxArbitre) {
membrearbitre.add(p);
}

else {
System.out.println("Il n'y a plus de place !");
}

}



}
```

//classe Arbitre

```
public class Arbitre extends Personne {

private int numeroarbitre;

public Arbitre(String nom,int age,int numeroarbitre){
super(nom,age);
this.numeroarbitre=numeroarbitre;
}
public String toString(){
return "Il y a une faute";
}

}
```

// classe TestMatch

```
import static java.lang.Math.*;
import java.lang.Math;
public class Testmatch {

public static void main(String [] args){
Club notreclub=new Club(); //On créer un club
Club autreClub=new Club(notreclub); //On le clone
Match m1= new Match(0,0); //on créer un match

Joueur j1=new Joueur("joueur 1",18,0,1,5); //Puis des joueurs
Joueur j2=new Joueur("joueur 2",15,0,1,9);
Joueur j3=new Joueur("joueur 3",16,0,1,1);
Joueur j4=new Joueur("joueur 4",17,0,1,3);
Joueur j5=new Joueur("joueur 5",17,0,1,4);
Joueur j6=new Joueur("joueur 6",18,0,2,7);
Joueur j7=new Joueur("joueur 7",15,0,2,9);
Joueur j8=new Joueur("joueur 8",16,0,2,1);
Joueur j9=new Joueur("joueur 9",17,0,2,0);
```

```
Joueur j10=new Joueur("joueur 10",17,0,2,1);
Joueur g1=new Joueur ("gardien 1",19,0,1,8);
Joueur g2=new Joueur("gardien 2",20,0,2,8);
Arbitre a1=new Arbitre("Bernard",16,1); //Et un arbitre

Joueur joueurt[]={j1,j2,j3,j4,j5,j6,j7,j8,j9,j10};

for( int i=0;i<joueurt.length;i++){ //ajoute des membres au club si ils ont plus de 15 ans
try{
notreclub.ajouterMembreJoueur(joueurt[i]);
}catch( Exceptions e){ // sinon leve une exception
System.out.println(e.getMessage());
}
}

try{
notreclub.ajouterMembreArbitre(a1); // on inscrit l'arbitre dans le club si il a plus de 15 ans
}catch( Exceptions t){ // sinon leve une exception
System.out.println(t.getMessage());
}


Balle b = new Balle();
b.balleaucentre(m1); //on pose la balle au centre
int tailledescages=20;
int distancemax=30; //distance maximum entre les joueurs pour pouvoir faire une passe
int distancedescages=100;
int fautes=0;   // nombres de fautes

for (int j=0;j<joueurt.length;j++) {
m1.ajouterJoueur(joueurt[j]);
m1.initialiserposition(joueurt[j]); //on initialise la position des joueurs
}

g1.setPosx((int)(Math.random()*m1.LIMX)); //puis celle des gardiens
g1.setPosy(m1.LIMY-20);
g2.setPosx((int)(Math.random()*m1.LIMX));
g2.setPosy(20);
```

```
while((m1.cmpt1<5)&&(m1.cmpt2<5) ){ //Tant que aucune équipe ne marque 5 buts
Joueur jqdlb= m1.joueurleplusproche(joueurt,b); //On identifie le joueur le plus proche de la balle
b.setPosx(jqdlb.getPosx()); // et il récupere la balle ( la position du joueur devient aussi celle de la balle)
b.setPosy(jqdlb.getPosy());

Joueur jqdlb2;

if((jqdlb.getPosy()>distancedescages)||(Math.abs(m1.LIMY-jqdlb.getPosy())>distancedescages)){ // Si le joueur est trop loin des cages
jqdlb2=m1.yatilquelqun(jqdlb, joueurt, distancemax); //il regarde si il y a un joueur de son equipe autour de lui
if (jqdlb2==jqdlb){ //si il n'y a personne
jqdlb2=m1.ennemienvue(jqdlb2,joueurt,30,b); //on regarde si il y a un joueur de l'autre équipe pas loin qui pourrais lui prendre la balle
jqdlb=jqdlb2;

if(jqdlb2==jqdlb){ //si il n'y a toujours personne
m1.sedeplacer(jqdlb2); // alors il avance d'une case
jqdlb=jqdlb2;
}
}
else{
jqdlb=jqdlb2; // sinon, le joueur fait la passe et le joueur qui détient la balle devient son cohéquipier.
}
}

else { //Si le joueur est assez près alors il tente un but
m1.nouveaubut(jqdlb, g2, g1, b);
if (b.getPosx()<0 || b.getPosx()>m1.LIMX ||  b.getPosy()<0 || b.getPosy()>m1.LIMY){ // si la balle sort du terrain , il y a alors faute.
System.out.println(a1.toString());
fautes++;
}
}
}
System.out.println("Il y a eu "+fautes+" fautes durant ce match.\n\n");
```

```
if (m1.cmpt1>m1.cmpt2) {
for (int i=0; i<joueurt.length;i++) {
if (joueurt[i].getEquipe()==1)
System.out.println("Le joueur"+ i +" a marqué "+joueurt[i].getnbButs()+" et a maintenant "
+joueurt[i].getCharisme()+" points de charisme");


}
System.out.println("\nL'équipe 1 a au total marqué "+m1.cmpt1+" buts");
System.out.println("L'équipe 2 a au total marqué "+m1.cmpt2+" buts");
System.out.println("le vainceur est donc l'équipe 1\n");
}
else{
for (int i=0; i<joueurt.length;i++) {
if (joueurt[i].getEquipe()==2)
System.out.println("Le joueur"+ i +" a marqué "+joueurt[i].getnbButs()+" et a maintenant "
+joueurt[i].getCharisme()+" points de charisme");


}
System.out.println("\nL'équipe 1 a au total marqué "+m1.cmpt1+" buts");
System.out.println("L'équipe 2 a au total marqué "+m1.cmpt2+" buts");
System.out.println("le vainceur est donc l'équipe 2\n");


}
}
}
```

//classe Exception

```
public class Exceptions extends Exception{

public Exceptions(String message){

super(message);
}
```

```
}
```