Université de Carthage
École Supérieure de la Statistique et de l'Analyse de l'Information

Mme Aïcha El Golli Jabbes

Année Universitaire **2022-2023**

Durée : 30 minutes **Quizz2 Java 1** $2^{ème}$ année

Nom et prénom : .....

Groupe : CD

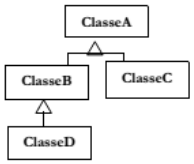
---

1. (9 points) Soient les deux classes suivantes. Qu'affichera le code suivant ?

```
public class Classe1 {
protected String m;
public Classe1() {this.m="allo"; System.out.println("const mere");}
public String toString() {return m;}
public void methode() {
        System.out.println("methode mere "+ this);
}
public static void main(String args[]) {
Classe2 f = new Classe2(17) ;
System.out.println(f) ;
f.methode();
Classe1 e= new Classe1() ;
System.out.println(e) ;
e.methode() ;
e=f;
e.methode();
e.m="hello";
System.out.println(e);
}}
class Classe2 extends Classe1{
        public int n;
        public Classe2(int n) { this.n=n;
        System.out.println("const fille");
        }
        public String toString() {return super.toString()+" "+ n;}
        public void methode() {
                n++;
                System.out.println("methode fille "+ this);
        }
        public void test() { System.out.println("Fille.test");}
}
```

Solution: const mere (1pt)
const fille (1pt)
allo 17 (1pt)
methode fille allo 18 (1pt)
const mere (1pt)
allo (1pt)
methode mere allo (1pt)
methode fille allo 19 (1pt)
hello 19 (1pt)

2. (6 points) soit la hiérarchie de classes suivante :




```
public class ClasseA {
protected int n;
        ClasseA(int n){this.n=n;}
        public String nom(){return("ClasseA");}
        public int getN(){return(n);}
        public String toString(){return(nom()+" n="+ this.n);}
}
```

```
public class ClasseB extends ClasseA {
protected char m;
        ClasseB(int n, char m){super(n); this.m=m;}
        public String nom() {return("ClasseB");}
        public int getM() {return(m);}
        public String toString() {return(nom()+" "+super.toString()+" M="+ this.m);}
}
```

```
public class ClasseC extends ClasseA {
        ClasseC(int n){super(n); }
}
```

```
public class ClasseD extends ClasseB {
protected String nom;
        public ClasseD(int n, char m, String nom) {super(n,m); this.nom=nom;}
        public String nom(){return "ClasseD";}
        public String getNom(){return nom;}
}
```

1) Soit la déclaration suivante :

```
ClasseB b=new ClasseD(2, 't', "hello");
```

Indiquez si les instructions suivantes sont correctes ou pas (ce qui se passe à la compilation et à l'exécution) et, pour les instructions qui vous semblent correctes, indiquez ce qui serait affiché, si il y a affichage.

	Compilation	Exécution
1) `System.out.println(b.nom());`	Ok **(0,5pt)**	`ClasseD` **(1pt)**
2) `System.out.println(b.getNom());`	Not ok **(1pt)**	
3) `ClasseC c= b;`	Not ok **(1pt)**	
4) `ClasseD d=(ClasseD)b ;` `System.out.println(d);`	Ok **(0,5pt)** Ok **(0,5pt)**	Ok **(0,5pt)** `ClasseD ClasseD n=2 M=t` **(1pt)**

3. (5 points) Soit la classe Livre suivante :

```
public class Livre {
        private String titre; /** son titre (une chaîne de caractères) ; **/
        private String auteur;/**son auteur (une chaîne de caractères) ; **/
        private int nbpages;
        ...}
```

Écrire la redéfinition de la méthode equals qui permet de vérifier si deux livres sont les mêmes ou pas.

Solution:

```
public boolean equals(Object o) { //1pt
        if (o==this)return true; //1pt
        if (!(o instanceof Livre)) return false; //1pt
        Livre r= (Livre)o; //0.5pt
        return(titre.equals(r.getTitre())&& //1.5pt
                                (auteur.equals(r.auteur()))&&
                                nbpages==r.getNbpages());
                }
```