Contrôle de programmation Java n^{o}1 – SUJET A – 3 pages

durée 1h30 – justifiez vos réponses

indiquez le sujet sur votre copie

1. (4 points) On définit

```
1  public class UneClasse {
2  private static int ind=0 ;
3  private  int[] t;
4  public UneClasse(int i,int n)
5      { this.ind=i ; this.t = new int[n]; }
6  public UneClasse(){}
7  public void setInd(int i){this.ind=i;}
8  public void setT(int x, int j){this.t[j]=x;}
9  public void setT(int x){this.t[this.ind]=x;}
10 public int getInd(){return this.ind;}
11 public int[] getT(){return this.t;}
12 public String affichage()
13     {return ("t["+ ind+"] vaut "+ this.t[ind];}
14 }
```

puis

```
1  class EssaiUneClasse {
2  public static void main(String args[]) {
3  UneClasse u; UneClasse w;
4  u = new UneClasse(3,7);
5  w = new UneClasse(5, 10);
6  for (int k=0;k<u.getT().length;k++) u.setT(1,k);
7  for (int k=0;k<w.getT().length;k++) w.setT(k+1,k);
8   u.setT(-1);
9  System.out.println (u.getT().length);
10 System.out.println (w.getT().length)
11 System.out.println(u.affichage());
12 System.out.println (w.affichage()); }}
```

Quel affichage obtient-on ? JUSTIFIEZ.

2. (4 points) On définit

```
1  class DeuxNombres {
2  public int a1 ;
3  public int a2;
4  public DeuxNombres(int x, int y) { this.a1 = x ;this.a2 = y ; }
5  public void setA1(int n){this.a1=n;}
6  public void setA2(int n){this.a2=n;}
7  public int getA1(){return this.a1;}
8  public int getA2(){return this.a2;}
9  public void afficher(){return (this.getA1()+ " et "+ this.getA2());}
10 }
```

2.1 Comment appelle-t-on ce qui est défini en ligne 4 ?

2.2 L'instruction suivante est-elle valide (JUSTIFIEZ)?

```
DeuxNombres u = new DeuxNombres();
```

2.3 Quel sera l'affichage provoqué par l'exécution de la méthode suivante (JUSTIFIEZ)?

```
1  public static void main(String args[]) {
2  DeuxNombres u = new DeuxNombres(1,2);
3  DeuxNombres v = u;
4  DeuxNombres w = new DeuxNombres(1,2);
5  if (u==v) System.out.println ( "1") ;
6  if (w==u) System.out.println ( "2") ;
7  u.setA1(7); v.setA2(5);
8  System.out.println(u.afficher());
9  if (u==v) System.out.println ( "3") ;
10 }
```

3. (5 points) Un joueur est défini par un nom, un nombre de points de vie et un nombre de potions. Un joueur doit avoir au moins 1 point de vie pour continuer à jouer mais il peut gagner un point de vie par potion utilisée; il perd quand il n'a plus de point de vie.

Ecrire la classe `Joueur` avec

- ses attributs
- une méthode d'instance `void gagner(int s)` qui augmente le nombre de points de vie de l'objet de l'entier s paramètre
- une méthode de classe `boolean perdre(int perdu, Joueur j)` qui
  - de retirer le nombre `perdu` des points de vie du `Joueur` j s'il lui reste au moins un point de vie
  - sinon qui utilise le nombre de potions du `Joueur` j nécessaires pour garder un point de vie si c'est possible
  - sinon qui met à 0 le nombre de points de vie du `Joueur` j
  - et qui renvoie `true` si le `Joueur` j peut continuer à jouer et `false` sinon.

4. (7 points)

4.1 Définir une classe `Horaire` permettant de représenter un horaire avec

- les attributs privés caractérisant les heures et minutes (ce sont des entiers)
- les accesseurs en lecture et écriture
- le constructeur à deux paramètres
- une méthode d'instance `int duree(Horaire h)` qui renvoie la durée en minutes entre l'objet et le paramètre
- une méthode d'instance `boolean estAvant( Horaire h)` qui renvoie true si et seulement si l'objet est antérieur au paramètre.

4.2 Définir une classe `Vol` avec

- les attributs publiques caractérisant les villes de départ et d'arrivée (de type `String`) et les horaires de départ et d'arrivée de type `Horaire`
- le constructeur à quatre paramètres
- une méthode d'instance `boolean correspondancePossible(Vol v)` qui renvoie vrai s'il est possible de prendre le `Vol` paramètre après le `Vol` objet avec au moins une heure d'escale (vérifier les horaires et les villes)
- une méthode de classe `int dureeTotale(Vol v, Vol w)` qui, si la correspondance est possible entre v et w, renvoie la durée du voyage depuis la ville de départ jusqu'à la ville d'arrivée, et sinon qui renvoie -1.