NOM & PRENOM : .....

Groupe : .....

Note : ...../20

Exercice 1 : (8pts)

1. Enumérer les utilisations possibles du mot clé *super avec une bref explication :*
   List the possible uses of the keyword super with a brief explanation

   super.methode_name(): Lorsque la classe parent et la classe dérivées ont le même nom de méthode(redéfinition ou la surcharge), on utilise le mot clé java super dans la classe enfant pour accéder à la méthode de la superclasse.

   super.attribut_name(): Lorsque le parent et les classes dérivées ont le même nom de variable, on peut utiliser le mot-clé java super pour accéder la variable de la superclasse.

   Super(), super(args) : avec et sans argument, pour appeler le constructeur de la classe mère dans une classe fille

2. Que signifier le mot « *final* » devant une méthode ?
   What does the word "final" mean before a method?

   Une méthode final ne peut être redéfinie (overrided) dans une classe fille

3. Citer les différents modificateurs de visibilité (accessibilité)
   List the different visibility modifiers (accessibility)
   public , protected, private, par defaut (aucun indicateur)

4. Vérifier si les codes suivants contiennent des erreurs ou s'exécutent correctement et pour chaque erreur indiquer sa ligne et proposer une correction.
   Check if the following code contains errors or executes correctly and for each error indicate its line and suggest a correction.

// code 1
public class A{
1. int a;
2. A(int x){a=x;}
}
3. class B extends A{
4. int b;
}
	Add a default constructor in calss A or add the following contructor to class B
	
	B(){super(b)}

// code 2
1. public int f1(int[10] T){
2. T = new int[15];
3. for (int i=0;i <=T.length;i++)
4. T[i]=T[i-1]+T[i+1];
return T;
}
	5. public int [] f1(int[] T){
	6. T = new int[15];
	7. for (int i=1;i <T.length-1;i++)
	8. T[i]=T[i-1]+T[i+1];
	
	return T;
	}

//code 3
public class A{
1. int e;
2. A(int x){e=x;}
}
class Main1{
3. public static void main(String[] args)
{
4. A a =new A();
}}
	Int x=2;
	
	A a =new A(x);// A a =new A(4);

Exercice 2 (12 pts):

On cherche à modéliser les polynômes creux (كثير حدود مبعثر) de degré درجة inférieur à un maximum (100) sous la forme d'une liste contenant ses coefficients (معاملات):

$p(x) = a0 + a4\,x^4 + a2\,x^2 + a7\,x^7 + \cdots + an\,x^n$, **pour ce faire : للقيام بذلك اتبع الخطوات التالية**

I. Monôme (أحادي الحد):

a. **Ecrire une classe Monome permettant de modèliser un monôme $a\,x^i$, défini par le couple(ثنائية) d'attributs (degré, coefficient).**

b. **Ecrire les constructeurs nécessaires de la classe Monome et ses *getter* et *setters* : Les coefficients et les degrés peuvent être s'initialisent à 1 ou passés en paramètres. Ainsi, les degrés <0 ou > 100 doivent être rend respectivement à 0 et à 100. الدرجات <0 أو >100 يجب إرجاعها إلى 0 و100 على التوالي.**

c. **Ecrire la méthode *valMonome*, calculant la valeur du monôme (حساب قيمة أحادي الحد) pour le paramètre x.**

d. **Ecrire une méthode *Display* qui permet d'afficher un Monome de sa forme traditionnelle.**

```
public class Monome {

private  double c;   private int d;

//Constructors' code here
public Monome() {  c=1;d=1;    }

    public Monome(double c, int d) {
        this.c = c;
        this.d = d;
        if(d>100)this.d=100;
        if(d<0)this.d=0;
    }

// getter and setter code here


    public double getC() {return c;  }

    public int getD() { return d; }

    public void setD(int d) { this.d = d;
    if(d>100)this.d=100;
    if(d<0)this.d=0;
    }

    public void setC(double c) { this.c = c;  }
```

```
// valMonome method code here

double valMonome(double x){  return c*Math.pow(x, d);  }


// Display method code here


    void afficher(){  System.out.print(c+"x^"+d+" "); }


}
```

II. polynôme sous forme de liste de monôme :

a. **Ecrire une classe ListMonome qui englobe une liste d'objets Monome.**

b. **La classe ListMonome contient des constructeurs qui initialisent un objet ListMonome de sorte qu'il contient un seul monôme, par défaut ou passé en paramètre.**

c. **La classe ListMonome contient une méthode *Search*(int degre) qui renvoie -1 si un monôme de degré i ne se trouve pas dans la liste et renvoie son index dans la liste dans le cas contraire.**

d. **La classe ListMonome contient une méthode Remove, permet de supprimer le monôme de degré i de la liste des monômes. Si, ce monôme n'existe pas dans la liste, un message d'erreur doit être affiché.**

e. **Ecrire une méthode Display qui permet d'afficher un polynôme de sa forme traditionnel.**

f. **La classe ListMonome contient une méthode Add, permet d'ajouter un monôme à la liste des monômes. Si un monôme de même degré existe déjà dans la liste, la méthode Add modifie le monôme existant en additionnant le nouveau coefficient et l'ancien. Par exemple : $3x^4 + 2\,x^4 = 5\,x^4$**

g. **Ecrire la méthode valListMonome, calculant la valeur du Polynome pour le paramètre x.**

h. **L'accès aux éléments du polynômes se réalise par la méthode *getMonome*(int d) qui permet de trouver et renvoyer le monome de degré *d*. Un message d'erreur doit être s'affiché si le monome indiqué n'existe pas dans la liste, et dans ce cas la méthode renvoie un monome de coeficient 0.**

```
class ListMonome
{
ArrayList <Monome> L=new ArrayList <Monome>();
//Constructors' code here
public ListMonome() { Monome e=new Monome();    L.add(e);}

public ListMonome(Monome e) { L.add(e); }

// search code here
int Rechercher(int i){
      int index=-1;
      for(int j=0;j<L.size();j++)
      {
          if(L.get(j).getD()==i){index=j;}
      }
      return index;
     }

// Remove code here
void remove(int i){
        int index = Rechercher(i);
         if(index!=-1) L.remove(index);
         else System.out.print("error");
     }

// Display code here
void afficher(){
      for(int i=0;i<L.size();i++){L.get(i).afficher();System.out.print(" + ");}
     }
```

```
// Add code here
void Remplir(Monome e) {
     int ed=e.getD(); double ec=e.getC();
     int i=Rechercher(ed);
     Monome newe=new Monome();
    if( i!=-1) {
        newe.setC(ec+L.get(i).getC());
        newe.setD(ed);
        L.set(i,newe);

    }else L.add(e);
 }
```

```
// valListMonome code here
double valListMonome(double x){
         double s=0;
         for(int i=0;i<L.size();i++) s=s+L.get(i).valMonome(x);
     return s;}
```

```
// getMonome
Monome getMonome(int d){
          int index= Rechercher(d);
         if( index!=-1) return L.get(index);
         else return new Monome(0,0);
      }
```

III. On veut créer une classe Monome2d héritée de la classe Monome, de sorte que les instances de Monome2d sont des monômes de deuxième degré max: $p(x) = a0 + a1\,x^1 + a2\,x^2$.

a. **Créer la classe Monome2d avec ses constructeurs nécessaires qui permettent d'initialiser un monôme de deux dimensions (en forçant le polynôme d'être de deuxième degré).**
b. **Quelles sont les méthodes de la classe Monome qu'on doit redéfinir et adapter dans la classe Monome2d ? (Les méthodes qui peuvent violer la contrainte d'être un monome de 2èmme degré) ? Redéfinir ces méthodes.**

```
public class Monome2d extends Monome{
    public Monome2d() {
        }

    public Monome2d(double c, int d) {
        if(d>2)setD(2);
        else setD(d);
        setC(c);
    }

    @Override
    public void setD(int d) {
        if(d>2) super.setD(2);
        else super.setD(d);
    }




}
```

Pour un test, dans une méthode main, créer un objet ListMonome et insérer le polynôme suivant : $p(x) = 1\,x \;+2\,x^2+3\,x^3\;+4\,x^4+5n\,x^5$. Afficher sa valeur pour x=8.

```
public static void main(String[] args) {


for(int i=2;i<6;i++)Ll.Remplir(new Monome(i,i));
    Ll.afficher();
    System.out.println(Ll.valListMonome(2));


}
```