Exercice1: les interfaces

1. Ecrivez une interface Forme avec les méthodes abstraites suivantes :
   - perimetre() : Renvoie le périmètre de la forme;
   - aire() : Renvoie l'aire de la forme.
2. Ecrivez une classe Carre implémentant l'interface Forme avec les attributs suivants :
   - cote : Le côté du carré.

   La classe Carre doit disposer des constructeurs suivants :
   - Carre();
   - Carre(cote).

   La classe Carre doit contenir
   - des accesseurs (get) et mutateurs (set) pour les différents attributs
   - perimetre() : Donne le périmètre du carré;
   - aire() : Donne l'aire du carré;
   - toString() : Donne une représentation du carré.
3. Ecrivez une classe Cercle implémentant l'interface Forme avec les attributs suivants :
   - rayon : Le rayon du cercle.

   La classe Cercle doit disposer des constructeurs suivants :
   - Cercle();
   - Cercle(rayon).

   La classe Cercle doit contenir
   - des accesseurs (get) et mutateurs (set) pour les différents attributs
   - perimetre() : Donne le périmètre du cercle;
   - aire() : Donne l'aire du cercle;
   - toString() : Donne une représentation du cercle.
4. Ecrivez aussi une classe TestForme afin de tester les classes.
   - stocker plusieurs objets différents dans un tableau
   - appliquer toString() sur chaque objet

Exercice 2 Générer, une exception surveillée de type Exception

Considérons le cas d'un compte qui est défini par un code et un solde et sur lequel, on peut verser un montant, retirer un montant et consulter le solde.

```
public class Compte {
private int code;
private float solde;
public void verser(float mt){
solde=solde+mt;}
public void retirer(float mt)throws Exception{
solde=solde-mt;}
public float getSolde(){
return solde;}}
```

Utilisation de la classe Compte

```
import java.util.Scanner;
public class Application {
public static void main(String[] args) {
Compte cp=new Compte();
Scanner clavier=new Scanner(System.in);
System.out.print("Montant à verser:");
float mt1=clavier.nextFloat();
cp.verser(mt1);
System.out.println("Solde Actuel:"+cp.getSolde());
System.out.print("Montant à retirer:");
float mt2=clavier.nextFloat();
cp.retirer(mt2); }}
```

Traiter les exceptions suivantes :

1. créer une nouvelle Exception nommée **SoldeInsuffisantException** avec un constructeur qui prend en paramètre le message a afficher, en héritant de la classe Exception, nous créons une exception surveillée. Pour créer une exception non surveillée, vous pouvez hériter de la classe RuntimeException. utiliser cette exception.

2. Supposons que l'on ne doit pas accepter un montant négatif dans la méthode retirer. On devrait générer une exception de type **MontantNegatifException.** Il faut d'abord créer cette nouvelle exception

3. une autre exception non surveillée est générer dans le cas ou on saisie une chaine de caractères et non pas un nombre. Cette exception est de type **InputMismatchException**, générée par la méthode nextFloat() de la classe Scanner. Nous devrions donc faire plusieurs catch dans la méthode main