1. Soit un objet *television* et une méthode *allumer* définie pour cet objet, lequel des énoncés suivants permet d'appeler la méthode allumer ?

```
a)  allumer.television()
b)  allumer.television
c)  television.allumer
d)  television.allumer()
```

Réponse : _____

2. Soit la classe **Incremente** suivante :

```
public class Incremente
{
    private int increment;
    private int petitPlus;

    public Incremente(int inc, int petit)
    {
        increment = inc;
        petitPlus = petit;
    }

    public int additionne(int n)
    {
        int somme;
        somme = n + increment + petitPlus;
        return somme;
    }
}
```

ou tout simplement :

```
return n + increment + petitPlus;
```

Quel résultat va-t-on obtenir suite à l'exécution de l'application suivante :

```
public class Test
{
    public static void main(String args[])
    {
        Incremente unObjet = new Incremente(10, 1);
        System.out.println("" + unObjet.additionne(5));
    }
}
```

Résultat : _____

3. Soit la classe suivante :

```
public class Cours
{
    private String sigle;    // sigle du cours
    private int eval;        // évaluation du cours

    public Cours(String sigle, int eval)   // constructeur
    {
        .......
    }

    public String appreciation()           // méthode d'instance
    {
        .......
    }
}
```

a) Complétez la déclaration de l'objet *cours2345* qui instancie la classe Cours avec "INFO 2345" comme sigle du cours et la valeur 3 comme évaluation :

```
_____________ cours2345 = ____________________________________________________
```

b) Complétez le code du constructeur *Cours* en tenant compte de ses paramètres :

```
public Cours(String sigle, int eval)
{
    _____________________________________________________________________
    _____________________________________________________________________
}
```

c) En supposant que, dans la méthode main, on ait l'instruction suivante :

```
System.out.println("J'aime Java " + cours2345.appreciation());
```

Donnez le code de la méthode *appreciation* qui retourne un message d'appréciation selon la valeur de l'attribut *eval* et en tenant compte du tableau suivant :

eval	message
1	pas du tout
2	un peu
3	beaucoup
4	à la folie

```
public String appreciation()
{
    _____________________________________________________________________
    _____________________________________________________________________
    _____________________________________________________________________
    _____________________________________________________________________
    _____________________________________________________________________
    _____________________________________________________________________
    _____________________________________________________________________
    _____________________________________________________________________
    _____________________________________________________________________
    _____________________________________________________________________
    _____________________________________________________________________
    _____________________________________________________________________
}
```

4. On a les classes suivantes :

```
1   public class Livre
2   {
3       private static int nbLivres = 0;
4       private String titre;
5       private int nbPages;
6       private int annee;
7
8       // constructeur d'initialisation
9       public Livre(String t, int nb, int a)
10      {
11          titre = t;
12          nbPages = nb;
13          annee = a;
14          nbLivres++;
15      }
16
17      private int getNbPages()
18      { return nbPages; }
19
20      public static String getTitre()
21      { return titre; }
22
23      public int getAnnee()
24      { return annee; }
25
26      public String affiche()
27      {
28          return titre + ", paru en " + annee + ", a " + nbPages + " pages";
29      }
30  } // fin Livre
```

```
40  public class TestLivre
41  {
42      public static void main(String args[])
43      {
44          Livre dictionnaire;
45          Livre roman = new Livre("Da Vinci code", 514, 2003);
46          Livre essai = new Livre();
46
48          System.out.println("Le roman a " + roman.getNbPages());
49
50          System.out.println("Le titre du roman est " + roman.getTitre());
51
52          System.out.println("Le dictionnaire est paru en " +
53              dictionnaire.getAnnee());
54
55          System.out.println(roman.affiche());
56
57          System.out.println(roman);
58
59          System.out.println("Le nombre de livres est " + Livre.nbLivres);
60      }
61  } // fin TestLivre
```

Expliquez les problèmes obtenus lors de la compilation (ou de l'exécution) des lignes suivantes :

21	
46	
48	
50	
53	
57	
59	

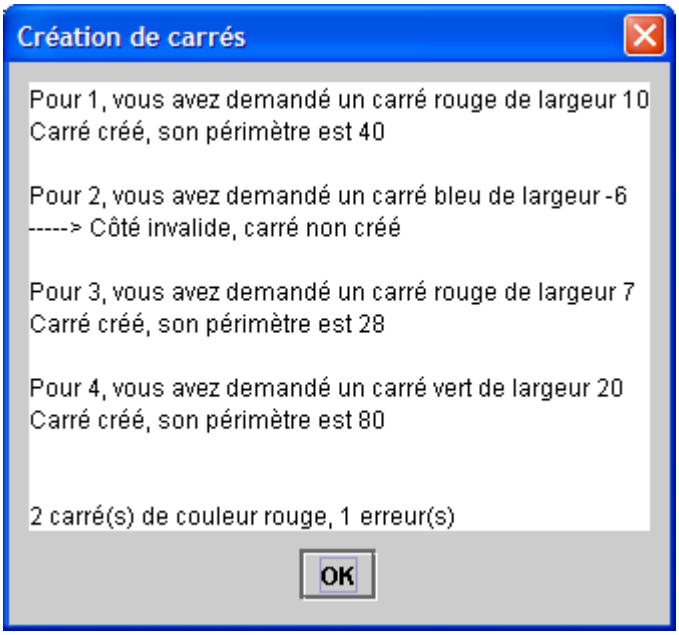
5. Soit la classe suivante :

```
public class Carre
{
    private String couleur;
    private int cote;                          // longueur d'un côté du carré
    private int perimetre;

    public Carre(String coul, int nb)          // constructeur
    {
        couleur = coul;
        cote = nb;
    }

    public void calculerPerimetre()
    {
        perimetre = cote * 4;
    }

    public String getCouleur()
    { return couleur; }

    public int getCote()
    { return cote; }

    public int getPerimetre()
    { return perimetre; }
}
```

Complétez la classe **TestCarre** afin de produire l'affichage ci-dessous. On doit :

- saisir au clavier la couleur et le côté de 4 objets Carre
- afficher la couleur et le côté de chaque carré demandé
- rejeter une valeur négative pour le côté en affichant un message approprié (on ne crée pas l'objet dans ce cas)
- calculer et afficher le périmètre de chaque objet Carre créé
- comptabiliser le nombre de carrés de couleur rouge et l'afficher
- comptabiliser le nombre d'erreurs et l'afficher

Résultats de l'exécution :



```
import javax.swing.*;
public class TestCarre
{
    public static void main(String args[])
    {
        final int NB_CARRES = 4;

        JTextArea affichage = new JTextArea();

        String laCouleur;     // pour lire la couleur d'un carré
        int longCote;         // pour lire la longueur du côté
        int nbRouge = 0;      // compteurs
        int nbErreurs = 0;

        Carre unCarre;        // déclaration d'une référence à un objet Carre

        for (int k = 1; k <= NB_CARRES; k++)
        {
            laCouleur = JOptionPane.showInputDialog(
                "Entrez la couleur du carré " + k);
            longCote = Integer.parseInt(JOptionPane.showInputDialog(
                "Entrez la longueur du côté pour le carré " + k));
            affichage.append("Pour " + k + ", vous avez demandé un carré " +
                        laCouleur + " de largeur " + longCote + "\n");

            // validation du côté
            if (_____________________________________________)
            {
                // erreur
                ____________________________________________________________
                ____________________________________________________________
                ____________________________________________________________
            }
            else
            {
                // création de l'objet unCarre
                // calcul et affichage de son périmètre, etc.
                ____________________________________________________________
                ____________________________________________________________
                ____________________________________________________________
                ____________________________________________________________
                ____________________________________________________________
                ____________________________________________________________
            }
        }

        // affichage des résultats
        ____________________________________________________________
        ____________________________________________________________
        ____________________________________________________________
        JOptionPane.showMessageDialog(null, affichage,
            "Création de carrés", JOptionPane.PLAIN_MESSAGE);
        System.exit(0);
    }
}
```

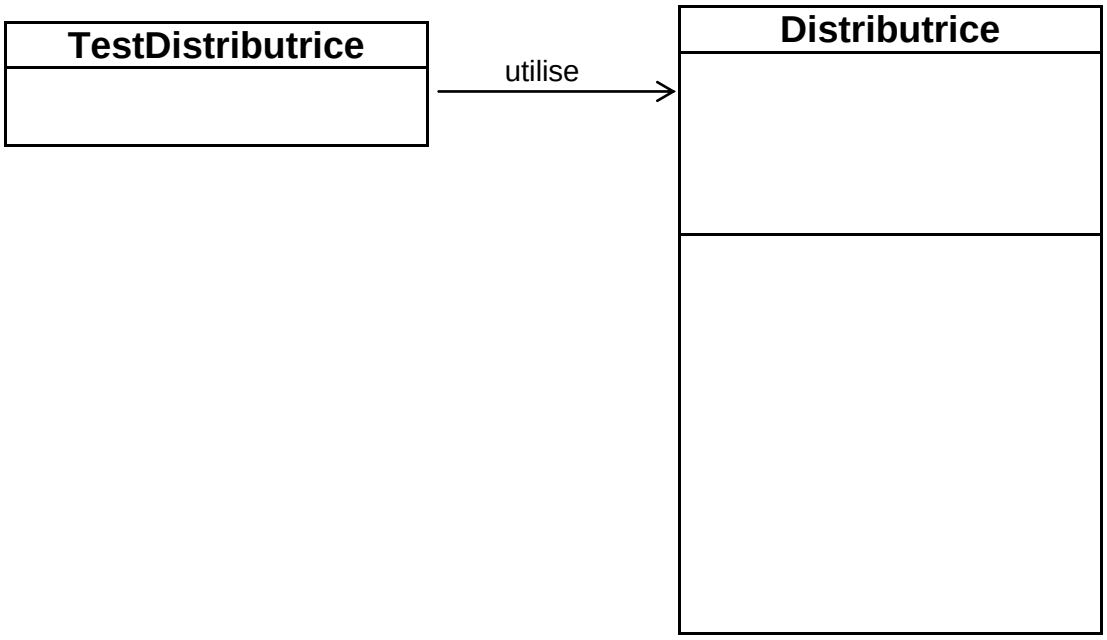
6. Créez une classe **Auto** avec :
   - les attributs *marque*, *couleur*, *bonEtat*
   - le constructeur qui affecte la marque et la couleur
   - les méthodes *rouler* (met bonEtat à true), *horsCircuit* (met bonEtat à false) et *toString* (qui retourne une phrase avec la marque, la couleur et l'état du véhicule (défectueux ou fonctionnel)

   Créez aussi une classe **TestAuto** qui crée une Volvo noire et qui fait l'appel de la méthode *rouler* suivi de l'affichage de l'objet (à l'aide de *toString*), puis de l'appel de *horsCircuit* et de l'affichage de l'objet à nouveau. Quels seront les résultats affichés ?

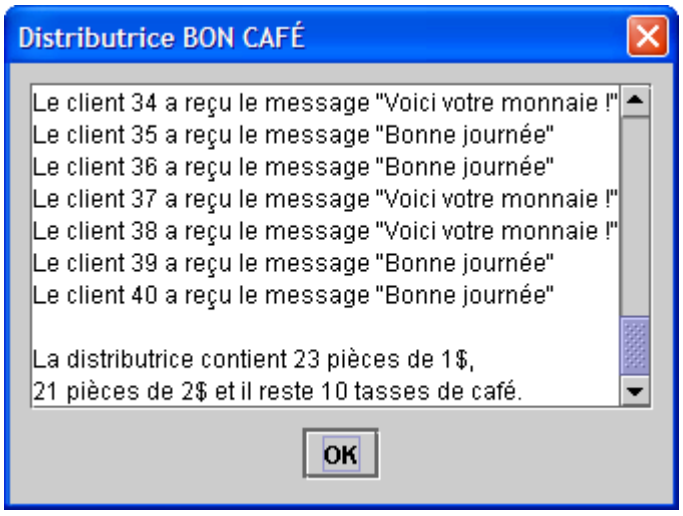
Résultats de l'exécution :

7. Sachant qu'on voudra créer dans une application des machines distributrices qui ne vendent que du café coûtant 1$ et qu'on pourra acheter le café en payant seulement avec des pièces de 1$ ou de 2$:
   a) Complétez la classe **Distributrice** avec :
      - ses 3 attributs
      - le constructeur sans paramètre et celui à 2 paramètres
      - les méthodes « *set* » et les méthodes « *get* »
      - la méthode *payerCafe* (qui doit retourner un des messages suivants : **Bonne journée** ou **Voici votre monnaie** ou **Désolé! je n'ai plus de 1$** ou **La pièce est refusée** ou **Désolé! il n'y a plus de café**)
      - la méthode *toString* qui retourne une chaîne donnant le contenu de la distributrice

   b) Complétez le diagramme de la classe **Distributrice**



   c) Complétez l'application **TestDistributrice** qui va créer une distributrice contenant au départ 25 pièces de 1$ et 50 tasses de café. L'application doit ensuite simuler l'achat de café par 40 clients à l'aide d'une boucle **for** (le client paie avec un 1$ lorsque le compteur de la boucle est un multiple de 3 ou de 5, sinon il paie avec un 2$). Faites afficher les messages concernant les ventes de café et l'état de la distributrice à la fin de la journée dans une zone de texte à défilement sur 10 lignes.



```
public class Distributrice
{
    // constantes
    public static final int UN_DOLLAR = 1;
    public static final int DEUX_DOLLARS = 2;

    // attributs d'instance



    // constructeurs



    // méthodes d'accès (get)



    // méthodes de mutation (set)
```

```
    // méthode toString
```

```
    // méthode payerCafe
```

```
import javax.swing.*;
public class TestDistributrice
{
    public static void main(String[] args)
    {
        JTextArea affichage = new JTextArea(10, 0);
        JScrollPane defilement = new JScrollPane(affichage);

        // création de la Distributrice


        // boucle d'achat de café par 40 clients


        // affichage des ventes et de l'état de la Distributrice

        JOptionPane.showMessageDialog(null, defilement,
            "Distributrice BON CAFÉ", JOptionPane.PLAIN_MESSAGE);
        System.exit(0);
    }
}
```

8. Supposons que la classe **FouFou** soit définie comme suit :

```
public class FouFou
{
    private int nb;
    private static String nom;

    public int getNb()              { return nb; }
    public static String getNom()   { return nom; }
    public void methode1()          { ......... ; }
    public static void methode2()   { ......... ; }
}
```

et soit la création de l'objet suivant : `FouFou unFou = new FouFou();`

Indiquez si les instructions suivantes sont correctes ou non :

a) `System.out.println("Le nombre est " + unFou.getNb());` _____
b) `System.out.println("Le nom est " + unFou.getNom());` _____
c) `unFou.methode1();` _____
d) `unFou.methode2();` _____
e) `System.out.println("Le nombre est " + FouFou.getNb());` _____
f) `System.out.println("Le nom est " + FouFou.getNom());` _____
g) `FouFou.methode1();` _____
h) `FouFou.methode2();` _____

9. Complétez la classe **Exemple** suivante, sachant que les objets **Exemple**, qui seront créés par la suite, doivent partager les attributs *somme* et *nbObjets*. L'attribut *nbObjets* permet de compter le nombre d'objets qui seront créés et l'attribut *somme* permet de faire la somme des différentes valeurs conservées dans l'attribut d'instance *valeur* des différents objets créés.

```
public class Exemple
{
    _____________________________________ nbObjets = 0;
    _____________________________________ somme = 0;
    _____________________________________ valeur;

    public Exemple(int nb)
    {
        ____________________________________________________;
        ____________________________________________________;
        ____________________________________________________;
    }

    public _________________ setValeur(int nb)
    {
        ____________________________________________________;
        ____________________________________________________;
        ____________________________________________________;
    }

    public _________________ getValeur()
    {
        ____________________________________________________;
    }
```

```
    public _________________ getNbObjets()
    {
        ____________________________________________________;
    }

    public _________________ getSomme()
    {
        ____________________________________________________;
    }

    public _________________ toString()
    {
        return "L'attribut valeur est de " + valeur + ", somme = " + somme +
               ", nombre d'objets créés = " + nbObjets;
    }
} // fin de la classe Exemple
```

Complétez maintenant l'application **TestExemple** :

```
public class TestExemple
{
    public static void main(String args[])
    {
        Exemple obj1 = new Exemple(5);
        System.out.println("objet 1 : " + obj1);

        Exemple obj2 = new Exemple(10);

        System.out.println("objet 1 : " + obj1);
        System.out.println("objet 2 : " + obj2);

        System.out.println("\nLa somme des attributs valeur = " +
                           ____________________________________);

        System.out.println("Le nombre d'objets Exemple = " +
                           ____________________________________);
    }
}
```

Quels seront les résultats affichés lors de l'exécution de la classe **TestExemple** ?

10. Complétez la classe **CompteEpargne** qui utilise une variable de classe privée pour stocker le *tauxAnnuelInteret* à 3.5% de tous les détenteurs d'un compte. Chaque objet de la classe contient la variable d'instance privée *soldeCompte* qui indique le montant dont l'épargnant dispose sur son compte. Fournissez les constructeurs (celui sans paramètre et celui avec paramètres), les méthodes *set*, les méthodes *get*, la méthode *toString* qui retourne le solde du compte dans un format monétaire et la méthode *calculerInteretMensuel* qui calcule le nouveau *soldeCompte* en lui ajoutant le résultat de la multiplication de sa valeur avec le *tauxAnnuelInteret*, divisé par 12.

    Complétez le programme pilote qui teste la classe **CompteEpargne** en instanciant deux objets **CompteEpargne**, *epargnant1* et *epargnant2*, ajustés respectivement à 2000 et 3000. Par la suite, modifiez le *tauxAnnuelInteret* à 4%, calculez et affichez les nouveaux soldes. Calculez et affichez les soldes du mois suivant avec un *tauxAnnuelInteret* de 5%.

```
import java.text.*;
public class CompteEpargne
{
}
```

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

        // avec taux d'intérêt à 4%

        // avec taux d'intérêt à 5%

    }
}
```

Résultats obtenus :

```
Solde de l'épargnant 1 : 2006,67 $
Solde de l'épargnant 2 : 3010,00 $
Solde de l'épargnant 1 : 2015,03 $
Solde de l'épargnant 2 : 3022,54 $
```