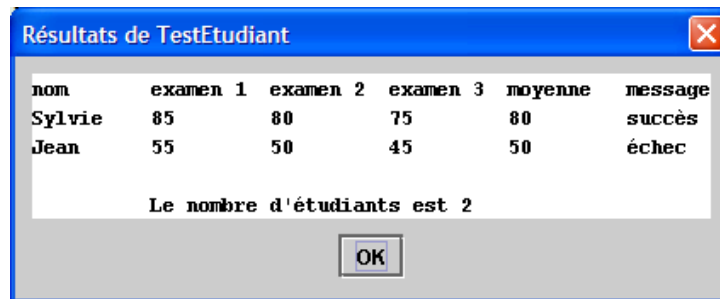


- Créez une classe **TestEtudiant** qui crée un étudiant avec le constructeur sans paramètre et fixe ses attributs (Sylvie, 85, 80, 75) avec les méthodes « set » et un deuxième étudiant (Jean, 55, 50, 45) avec le constructeur avec paramètres. L'application doit afficher les résultats suivants :

This image shows a single sheet of white paper with horizontal blue or grey ruling lines. The lines are evenly spaced and run across the width of the page. There are approximately 20 lines visible. The paper has a slight shadow on the right side, suggesting it's resting on a surface.

[illegible]

2. Nous avons dans un fichier la liste des étudiants d'un groupe (le nombre maximum possible d'étudiants est de 24). Faites le projet qui :
  - amène ce fichier en mémoire de façon à construire un tableau d'objets, dont les objets sont de classe **Etudiant**
  - modifie la 2<sup>ème</sup> note du 4<sup>ème</sup> étudiant par la note de 85
  - affiche le tableau des étudiants avec leur moyenne et message
  - affiche le nom du meilleur étudiant avec sa moyenne
  - affiche le nombre d'étudiants

Résultats du groupe					
nom	examen 1	examen 2	examen 3	moyenne	message
Julie	80	80	80	80	succès
Marc	100	100	100	100	succès
Martin	78	79	80	79	succès
Karine	51	85	55	63	succès

Le meilleur

Le meilleur étudiant est Marc  
avec une moyenne de 100

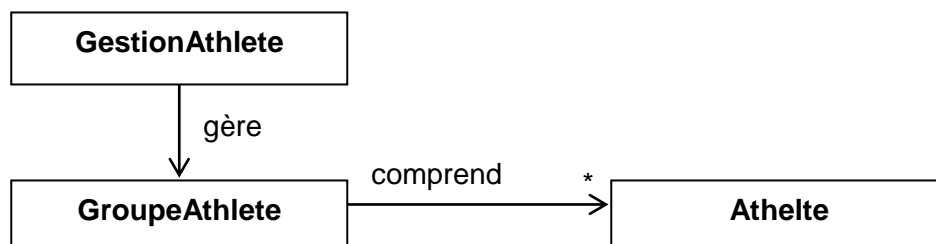
Le nombre d'étudiants du groupe

Il y a 4 étudiants dans le groupe

## Contenu de *athletes.txt* :

```
Julie      080 080 080
Marc       100 100 100
Martin     078 079 080
Karine     051 060 055
```

Le projet va donc comporter une classe **GroupeAthlete** permettant de représenter un groupe d'athletes avec ses méthodes et une classe **GestionAthlète** contenant l'application.



```
import java.io.*;
javax.swing.*;
java.awt.Font;
public class GroupeEtudiants
{
    public static final int MAX_ETUD = 24;
    private _____;
    private int nbEtud;          // le nombre d'étudiants dans le fichier

    public GroupeEtudiants() throws IOException
    {
        BufferedReader fichier = new BufferedReader(new FileReader("notes.txt"));
        String ligne;
        String nom;
        int note1, note2, note3;
        int k = 0;

        ligne = fichier.readLine();
        while (_____)
        {
            _____
            _____
            _____
            _____
            _____
            _____
        }
        fichier.close();

        nbEtud = _____; /* aller chercher l'attribut
                                static de la classe Etudiant */
    }

    public void afficherLesEtudiants()
    {
        _____
        _____
        _____
        _____
        _____
        _____
        _____
    }
}
```

```
public void afficherLeMeilleur()
{
    int maxMoy = -1;        // la moyenne la plus haute
    int maxEtud = -1;       // la position du meilleur étudiant
    int moyEtud;

    _____
    _____
    _____
    _____
    _____
    _____
    _____
    _____
    _____
    _____
    _____
}

// Cette méthode retourne l'étudiant placé dans le tableau à l'indice ind
public _____ getEtudiant(int ind)
{
    return _____;
}

public int getNbEtud()
{
    return nbEtud;
}
}
```

[illegible]

3. Soit la classe **Compte** suivante :

```
public class Compte
{
    private int nb;

    public Compte()          { nb = 1; }
    public Compte(int n)     { nb = n; }
    public void setNb(int n) { nb = n; }
    public int getNb()       { return nb; }
}
```

Quels résultats seront affichés lors de l'exécution du programme suivant :

```
public class Test
{
    public static void main(String args[])
    {
        Compte monCompte = new Compte();
        int nbFois = 0;

        for (int i = 0; i < 10; i++)
            augmenter(monCompte, nbFois);

        System.out.println("Attribut nb de monCompte = " + monCompte.getNb());
        System.out.println("nbFois = " + nbFois);
    }

    public static void augmenter(Compte c, int nbFois)
    {
        c.setNb(c.getNb() + 1);
        nbFois++;
    }
}
```

---

---

---

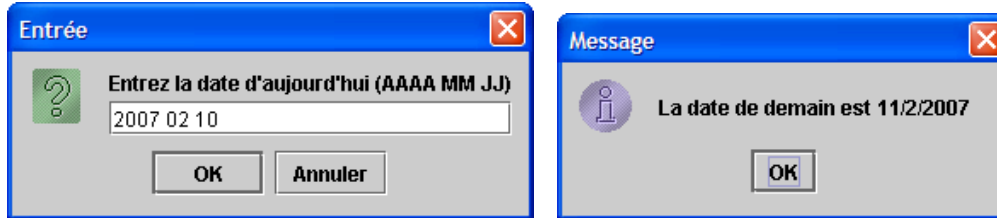
---

---

---

---

4. Vous devez faire un projet qui, à partir d'une date lue au clavier, va afficher la date du lendemain. Afin de réaliser ce projet, vous décidez d'améliorer la définition de la classe **Date** en fournissant le service permettant de lire au clavier une date et le service permettant d'obtenir la date du lendemain (en vous servant des méthodes privées *determinerNbJoursMois* et *estBissextile*). Fournissez aussi le service *toString*.



Complétez la classe **Date** et l'application **DateLendemain** :

```
import javax.swing.*;
public class Date
{
    private int jour;
    private int mois;
    private int an;

    public Date()
    {
        jour = mois = 1;
        an = 2000;
    }

    public Date(int j, int m, int a)
    {
        jour = j;
        mois = m;
        an = a;
    }

    // les méthodes set et get habituelles
    . . . . .

    public void lire()
    {
        String ligne;
        ligne = JOptionPane.showInputDialog(
            "Entrez la date d'aujourd'hui (AAAA MM JJ) : ");

        // permet de définir les attributs de l'objet courant (c'est-à-dire
        // l'objet qui sera utilisé pour appeler cette méthode d'instance)
        _____ = Integer.parseInt(ligne.substring(0,4));
        _____ = Integer.parseInt(ligne.substring(5,7));
        _____ = Integer.parseInt(ligne.substring(8));
    }

    private boolean estBissextile()
    {
        return ( _____ % 4 == 0 && _____ % 100 != 0 ) || ( _____ % 400 == 0 );
    }
}
```

```

public _____ lendemain()
{
    Date dateDemain = new Date();

    if ( _____ != determinerNbJoursMois())
    {
        _____;
        _____;
        _____;
    }
    else if ( _____ == 12)
    {
        _____;
        _____;
        _____;
    }
    else
    {
        _____;
        _____;
        _____;
    }
    return _____;
}

private int determinerNbJoursMois()
{
    int nbJours;
    int tabJrMois[] = {31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};

    if ( _____ == 2 && estBissextile())
        nbJours = 29;
    else
        nbJours = tabJrMois[_____];

    return nbJours;
}

public String toString()
{
    return jour + "/" + mois + "/" + an;
}
} // fin de la classe Date

```

```

// Voici l'application
import javax.swing.*;
public class DateLendemain
{
    public static void main(String args[])
    {
        _____
        _____
        _____
        _____
        JOptionPane.showMessageDialog(null,
            "La date de demain est " + _____);
        System.exit(0);
    }
}

```



5. Soient les 3 chaînes suivantes :

```
String s1 = "Youpie! un rave...";  
String s2 = s1;  
String s3 = "Youpie! un rave...";
```

Assumant que `s1` et `s3` sont à des adresses-mémoire différentes, quels sont les résultats des expressions suivantes ?

- a) `s1 == s2` \_\_\_\_\_
- b) `s2 == s3` \_\_\_\_\_
- c) `s1.equals(s2)` \_\_\_\_\_
- d) `s2.equals(s3)` \_\_\_\_\_
- e) `s1.compareTo(s2)` \_\_\_\_\_
- f) `s2.compareTo(s3)` \_\_\_\_\_

6. Sachant que 2 noms ont été lus au clavier et déposés dans les variables `nom1` et `nom2`, faites afficher les 2 noms par ordre alphabétique

---

---

---

---

---

---

---

---

---

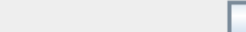
---

7. Écrivez un programme en Java qui lit plusieurs lignes de texte contenues dans un fichier **document.txt** et affiche le nombre total de mots. On veut aussi savoir le pourcentage de mots qui comportent **2** lettres. Les mots sont séparés par un (ou plusieurs) des caractères suivants : espace ( ), apostrophe ('), virgule (,), point (.), point-virgule (;), deux-points (:), point d'interrogation (?), point d'exclamation (!) ou caractère de fin de ligne (**\n**).

Si le fichier **document.txt** contient ceci :

Lignes de texte pour exercice !

Les lignes sont séparées par un (ou plusieurs) des caractères suivants : espace, apostrophe, virgule, point, point-virgule; deux-points: point d'interrogation? point d'exclamation! ou caractère de fin de ligne.

[illegible]