

Initiation à la programmation en Java

Jean-Luc Baril
LIB, Université de Bourgogne
B.P. 47 870, 21078 DIJON-Cedex France

May 22, 2025

Contents

1	Introduction au langage Java	4
1.1	Contexte	4
1.2	Premier programme	4
1.3	Première variable	4
1.4	Première formule	5
2	Les entrées/sorties	7
2.1	Le cours	7
2.2	Exercices corrigés	7
3	Les conditionnelles	10
3.1	Le cours	10
3.2	Exercices corrigés	10
4	Les répétitives	17
4.1	Le cours	17
4.2	Exercices corrigés	17
5	Bilan n°1	22
5.1	Le point méthode	22
5.2	Exercices	22
6	Les tableaux.	24
6.1	Le cours	24
6.2	Exercices corrigés	24
7	Les chaînes de caractères.	29
7.1	Le cours	29
7.2	Exercices corrigés	29
8	Les fonctions.	35
8.1	Le cours	35
8.2	Exercices corrigés	36
9	La récursivité.	41
9.1	Le minimum à savoir	41
9.2	Exercices corrigés	41

10 Bilan n°2	44
10.1 Le point méthode	44
10.2 Exercices corrigés	44
11 Corrections des bilans	46
11.1 Bilan n°1	46
11.2 Bilan n°2	52

Ce document propose d'aider le lecteur à apprendre les bases de programmation en langage Java. Il est constitué de plusieurs chapitres développant chacun un concept fondamental: Entrées/Sorties, Conditionnelles, Répétitives, Tableaux, Chaînes de caractères, Fonctions, Récursivité. Pour chaque concept, un résumé du cours, un *minimum à savoir* et une liste d'exercices corrigés sont proposés. Il s'agit d'un document complémentaire au cours d'Initiation à la programmation Java accessible en cliquant sur le lien <http://jl.baril.u-bourgogne.fr/coursinfo.pdf>. Avant de commencer, vous devez télécharger le fichier Lire.java à l'adresse <http://jl.baril.u-bourgogne.fr/licence1.html> et le compiler dans le répertoire de travail en exécutant la commande **javac Lire.java**.

1 Introduction au langage Java

1.1 Contexte

Le langage Java, créé par James Gosling et son équipe chez Sun Microsystems en 1995, est un langage de programmation orienté objet. Java a été conçu pour être simple, portable et sûr, avec une syntaxe similaire à celle de C++ mais avec une gestion simplifiée de la mémoire.

Historique

Le but initial au début des années 1990 était de créer un langage de programmation pour les appareils électroniques grand public tels que les décodeurs de télévision. Cependant, avec l'émergence d'Internet, le projet a évolué pour devenir un langage adapté au développement d'applications web.

En 1995, Java a été officiellement lancé avec la promesse "Write Once, Run Anywhere" (écrire une fois, exécuter partout).

Utilité et Applications

Java est utilisé dans une vaste gamme d'applications, allant des applications web et mobiles aux systèmes embarqués et aux grandes applications d'entreprise. Sa portabilité, sa robustesse et sa sécurité en font un choix populaire pour les développeurs. Par exemple, Spotify, Instagram, WhatsApp sont toutes codées en Java pour leur versions Android.

1.2 Premier programme

Maintenant qu'on en sait un peu plus sur ce langage, et qu'on mesure à quel point il est omniprésent dans notre quotidien, il est temps de programmer !

Un programme Java se structure de la manière suivante :

```
public class monpremierprogramme // Nom du programme
{
    public static void main (String[] args) //Programme principal
    {
        //declaration de variables
        //instructions //commentaires
    }
}
```

Compiler

- Dans un fichier texte, taper le code ci-dessus
- Sauvegarder le fichier texte dans un fichier **monpremierprogramme.java**
- Compiler le fichier: **javac monpremierprogramme.java**
(création du fichier monpremierprogramme.class)
- Exécuter le programme: **java monpremierprogramme**

1.3 Première variable

Une variable est un contenant ayant un nom (identificateur) qui peut contenir une information d'un certain type (entier, nombre décimal, texte, ...). Attention, toute variable d'un programme doit être déclarée avant son utilisation, en début de programme par exemple.

Les types en Java

Il existe huit types de base en Java :

- des types **entier** : byte, short, int, long
- un type **caractère** : char
- un type **booléen** : boolean
- deux types **flottants** : float et double.

Lorsque l'on déclare une variable, la machine Java nous donne en retour une zone mémoire d'un certain nombre d'octets, qui peut contenir le type voulu.

Type de base	Type	Nombre de bits
boolean	booléen	32 (effectifs)
byte	entier	8
short	entier	16
int	entier	32
long	entier	64
float	virgule flottante	32
double	virgule flottante	64
char	caractère	16

Par exemple, pour affecter à une variable *i* la valeur 3.14, nous avons besoin du type *float*.

```
public class monpremierprogramme
{
    public static void main (String[] args)
    {
        float i ;
        i = 3.14 ;
    }
}
```

Notons aussi qu'il faut mettre ';' à la fin de chaque instruction puis passer à la ligne.

1.4 Première formule

Maintenant, voici la syntaxe Java utilisée pour les opérateurs, afin de pouvoir écrire des formules :

- L'addition, la soustraction, la multiplication, la division : +, -, *, /
- La division entière et le reste de la division entière : /, %
- Les comparaisons : <, >, ==, <=, >=, !=
- Affectation : =
- Le Non : !
- Le ET logique : &&
- Le Ou (exclusif) : ^
- Le Ou (Inclusif) : ||

Expressions booléennes

Le type boolean permet uniquement deux valeurs: **true**, **false**. Les opérateurs booléens permettent des opérations sur ces valeurs.

Exemples:

Mathématique	Java
$a = b$	<code>a == b</code>
$a \geq b$	<code>a >= b</code>
$x \in [a, b]$	<code>x >= a && x <= b</code>
$x \in]a, b]$	<code>x > a && x <= b</code>
$x \in]a, b] \cap [c, d[$	<code>(x > a && x <= b) && (x >= c && x < d)</code>
$x \in]a, b] \cup [c, d[$	<code>(x > a && x <= b) (x >= c && x < d)</code>
$x \notin]a, b] \cup [c, d[$	<code>(x <= a x > b) && (x < c x >= d)</code>

Maintenant que vous connaissez les bases de la syntaxe Java, il est temps de s'exercer !

2 Les entrées/sorties

2.1 Le cours

Cette partie entraîne aux entrées et sorties, c'est à dire à communiquer avec l'ordinateur pendant l'exécution du programme. On distingue deux actions:

- Affichage des résultats à l'écran (sortie)
- Saisie d'informations au clavier (entrée)

Sorties

Ci-dessous les différentes façons d'afficher à l'écran :

Pour afficher à l'écran le message *Bonjour* :

```
System.out.print("Bonjour"); //sans retour à la ligne
System.out.println("Bonjour"); //avec retour à la ligne
```

Pour afficher à l'écran le contenu de la variable *x* :

```
System.out.print(x); //sans retour à la ligne
System.out.println(x); //avec retour à la ligne
```

Pour afficher *Bonjour* et le contenu de la variable *x* :

```
System.out.print("Bonjour"+x); //sans retour à la ligne
System.out.println("Bonjour"+x); //avec retour à la ligne
```

Entrées

Ci-dessous les différentes manières de demander une entrée à l'utilisateur de votre programme : Pour saisir un entier au clavier et le mettre dans la variable *x* :

```
x=Lire.i();
```

Pour saisir un réel au clavier et le mettre dans la variable *x* :

```
x=Lire.d();
```

Pour saisir un caractère au clavier et le mettre dans la variable *x* :

```
x=Lire.c();
```

2.2 Exercices corrigés

Exercice 3: Périmètre. Ecrire un programme qui calcule le périmètre d'un triangle rectangle dont on connaît l'hypoténuse et l'un des côtés de l'angle droit.

```
public class Ex3
{
    public static void main(String args[])
    {
        double x,y,z;
```

```

x = Lire.d();
z = Lire.d();
if(z>=x)
{
    y = Math.sqrt(z*z-x*x);
    System.out.println("périmètre : "+(x+y+z));
}
}
}

```

Exercice 4: Conversion secondes. Ecrire un programme qui permet d'exprimer un nombre de secondes saisi par l'utilisateur en heures, minutes et secondes.

```

public class Ex4
{
    public static void main(String args[])
    {
        System.out.println("Donner un nb de secondes : ");
        int s = Lire.i();

        int h = s/3600;
        int tmp = s%3600;
        int m = tmp/60;
        s = tmp%60;
        System.out.println("Ce nb de secondes vaut : "+h+" h "+m+" min "+s+"
            secondes");

        /* autre solution
        int h = s/3600;
        System.out.print(h+" h ");
        h = s%3600;
        int m = h/60;
        s = h%60;
        System.out.println(m+" min "+s+" s");
        */
    }
}

```

Exercice 5: Utilisation d'un boolean. Ecrire un programme qui permet de saisir un nombre entier puis qui affiche **true** si le nombre appartient à l'intervalle [1,10] et **false** sinon (ne pas utiliser de conditionnelle).

```

public class Ex5
{
    public static void main(String args[])
    {
        int x;
        boolean bool;
        System.out.println("Entrer un entier");
        x = Lire.i();
        bool=x >= 1 && x <= 10;
        System.out.println(bool);
    }
}

```

Exercice 6: Majeurs. Ecrire un programme qui permet de saisir les âges de 5 personnes puis qui affiche **true** si au moins 4 personnes sont adultes (âge > 18) et **false** sinon (ne pas utiliser de conditionnelle).

```
public class Ex6
{
    public static void main(String args[])
    {
        int a1,a2,a3,a4,a5;
        boolean bool;
        System.out.println("age 1 : ");
        a1 = Lire.i();
        System.out.println("age 2 : ");
        a2 = Lire.i();
        System.out.println("age 3 : ");
        a3 = Lire.i();
        System.out.println("age 4 : ");
        a4 = Lire.i();
        System.out.println("age 5 : ");
        a5 = Lire.i();
        bool=(a1>18 && a2>18 && a3>18 && a4>18) || (a1>18 && a2>18 && a3>18
            && a5>18) || (a1>18 && a2>18 && a4>18 && a5>18) || (a1>18 && a3>18
            && a4>18 && a5>18) || (a2>18 && a3>18 && a4>18 && a5>18);
        System.out.println(bool);
    }
}
```

Exercice 7: Subsidiaire. Ecrire un programme qui affiche **true** si une valeur donnée x par l'utilisateur vérifie $x \in [0, 10]$ et $x \notin [4, 5]$.

```
public class Ex6
{
    public static void main(String args[])
    {
        double x;
        boolean bool;
        System.out.println("age 1 : ");
        x = Lire.d();
        bool=(x>=0 && x<=10) && (x<4 || x>5);
        System.out.println(bool);
    }
}
```

3 Les conditionnelles

3.1 Le cours

Cette partie permet de s'entraîner sur les instructions conditionnelles. Ces outils permettent au programme d'exécuter les instructions dans leurs accolades (") si une condition est respectée. Si la condition est fausse, le programme ne considère pas les instructions et passe à la suite.

En voici les principales :

La structure **if** permet de discuter en fonction d'une condition (expression booléenne). Le **else** est facultatif.

```
if(condition)
{ //instructions si la condition est true }
```

```
if(condition)
{ //instructions si la condition est true }
else
{ //instructions si la condition est false }
```

Remarque : si le bloc d'instructions dans le IF contient une seule instruction alors les accolades sont facultatives.

La structure **switch** permet de discuter en fonction du contenu d'une variable. L'ajout d'un **break** permet de sortir immédiatement de la structure.

```
switch(x)
{ case 3 : //bloc d'instructions 1
  case 5 : //bloc d'instructions 2
  case 8 : //bloc d'instructions 3
}
```

Ici, le programme effectue les blocs 1, 2 et 3 si x vaut 3, et seulement les blocs 2 et 3 si x vaut 5, et seulement le bloc 3 si x=8.

```
switch(x)
{ case 3 : //instructions si la variable x vaut 3 ;
      break;
  case 5 : //instructions si la variable x vaut 5 ;
}
```

Ici, le programme effectue le bloc 1 seulement si x vaut 3, et toujours seulement les blocs 2 et 3 si x vaut 5.

Remarques : - pour l'instant, on se limitera aux variables *int* et *char* dans le switch.
- L'instruction *break* sera permise UNIQUEMENT dans un switch.

3.2 Exercices corrigés

Exercice 1: Maxi. Ecrire un programme qui

- affiche le maximum de deux entiers saisis par l'utilisateur,
- affiche le maximum de trois entiers saisis par l'utilisateur,
- affiche en ordre croissant trois entiers saisis par l'utilisateur.

```

public class Max
{
    public static void main(String args[])
    {
        int x,y,z,max;
        System.out.println("Donner nb 1 :");
        x = Lire.i();
        System.out.println("Donner nb 2 :");
        y = Lire.i();
        System.out.println("Donner nb 3 :");
        z = Lire.i();

        //2
        if(x>y) max=x;
        else max=y;
        if (z>max) max=z;
        System.out.println("max = "+max);

        //3
        if(x == max) {
            if (y <= z) System.out.println("ordre croissant = "+y+" "+z+" "+x);
            else System.out.println("ordre croissant = "+z+" "+y+" "+x);
        }
        else {
            if(y == max) {
                if (x <= z) System.out.println("ordre croissant = "+x+" "+z+" "+y);
                else System.out.println("ordre croissant = "+z+" "+x+" "+y);
            }
            else {
                if (x <= y) System.out.println("ordre croissant = "+x+" "+y+" "+z);
                else System.out.println("ordre croissant = "+y+" "+x+" "+z);
            }
        }
    }
}

```

Exercice 2: Age. Ecrire un programme qui permet de saisir l'âge de 5 personnes puis qui affiche **il n'y a pas de mineurs** si les 5 âges sont supérieurs ou égaux à 18, qui affiche **il y a un seul mineur** si un seul âge est inférieur à 18, et qui affiche **il y a au moins deux mineurs** sinon.

```

public class Mineur
{
    public static void main(String args[])
    {
        int age, mineur=0;
        System.out.println("Donner un age :");
        a = Lire.d();
        if(a<18)
            mineur = mineur +1;
        System.out.println("Donner un age :");
        a = Lire.d();
        if(a<18)

```

```

    mineur = mineur +1;
System.out.println("Donner un age :");
a = Lire.d();
if(a<18)
    mineur = mineur +1;
System.out.println("Donner un age :");
a = Lire.d();
if(a<18)
    mineur = mineur +1;
System.out.println("Donner un age :");
a = Lire.d();
if(a<18)
    mineur = mineur +1;
if(mineur==0)
    System.out.println("Il n y a pas de mineur");
else
    if(mineur==1)
        System.out.println("Il y a 1 seul mineur");
    else
        System.out.println("Il y au moins deux mineur");

```

Exercice 3: Equation. Ecrire un programme qui trouve les solutions de l'équation $ax^2 + bx + c = 0$.

```

public class Equation
{
public static void main(String args[])
{
    double a, b, c, discri;
System.out.println("Donner a :");
a = Lire.d();
System.out.println("Donner b :");
b = Lire.d();
System.out.println("Donner c :");
c = Lire.d();
if (a == 0 && b == 0 && c == 0)
    System.out.println("infinité de solutions réelles");
else
    if (a == 0 && b == 0) {
        System.out.println("pas de solution");
    }
    else
        if (a == 0)
            System.out.println("x vaut "+ (-c / b));
        else
            {
                discri = b * b - 4 * a * c;
                if (discri < 0)
                    System.out.println("pas de solution réelle");
                else
                    if (discri == 0)
                        System.out.println("x a comme solution réelle "+(-b / (2 *
                            a)));
                    else

```

```

        System.out.println("x a comme solutions réelles "+(-b +
            Math.sqrt(discriminant)) / (2 * a) +" et "+(-b -
            Math.sqrt(discriminant)) / (2 * a));
    }
}

```

Exercice 4: Chiffre romain. Ecrire un programme qui convertit un entier entre 1 et 5 dans son écriture romaine (1 → I, 2 → II, 3 → III, 4 → IV, 5 → V).

```

public class Romain
{
    public static void main(String args[])
    {
        int x;
        do
        {
            System.out.println("Entrer un entier entre 1 et 5");
            x = Lire.i();
        }while(x<1 || x>5);
        switch(x)
        {case 1: {System.out.println("I");break;}
        case 2: {System.out.println("II");break;}
        case 3: {System.out.println("III");break;}
        case 4: {System.out.println("IV");break;}
        case 5: {System.out.println("V");break;}
        }
    }
}

```

Exercice 5: Calculatrice. Ecrire un programme qui simule le fonctionnement d'une calculatrice à 4 opérations en utilisant l'instruction **switch ...case** pour le choix de l'opérateur.

```

public class Calcul
{
    public static void main(String args[])
    {
        double x,y;
        char o;
        System.out.println("Donner nb 1 :");
        x = Lire.d();
        System.out.println("Donner nb 2 :");
        y = Lire.d();
        System.out.println("Donner l'opérateur :");
        o = Lire.c();
        switch(o) {
            case '+': System.out.println("le résultat est "+(x+y)); break;
            case '-': System.out.println("le résultat est "+(x-y)); break;
            case '*': System.out.println("le résultat est "+(x*y)); break;
            case '/': if (y == 0)
                System.out.println("impossible");
                else
                    System.out.println("le résultat est "+(x/y));
        }
    }
}

```

```

        break;
    default : System.out.println("opérateur incorrect");
    }
}
}

```

Exercice 6: Cible. Proposer un programme permettant le tirage aléatoire de 4 points (x, y) avec $x, y \in [0, 1[$, et l’affichage du nombre de ces points situés dans un disque de centre $(0.5, 0.5)$ et de rayon 0.5.

```

public class Cible
{
    public static void main(String args[])
    {
        double x,y,d;
        int s = 0;
        x = Math.random();
        y = Math.random();
        System.out.println("x = "+x+" y = "+y);
        d = Math.sqrt((x-0.5)*(x-0.5) + (y-0.5)*(y-0.5));
        if(d < 0.5)
            s=s+1;
        x = Math.random();
        y = Math.random();
        System.out.println("x = "+x+" y = "+y);
        d = Math.sqrt((x-0.5)*(x-0.5) + (y-0.5)*(y-0.5));
        if(d < 0.5)
            s=s+1;
        x = Math.random();
        y = Math.random();
        System.out.println("x = "+x+" y = "+y);
        d = Math.sqrt((x-0.5)*(x-0.5) + (y-0.5)*(y-0.5));
        if(d < 0.5)
            s=s+1;
        x = Math.random();
        y = Math.random();
        System.out.println("x = "+x+" y = "+y);
        d = Math.sqrt((x-0.5)*(x-0.5) + (y-0.5)*(y-0.5));
        if(d < 0.5)
            s=s+1;
        System.out.println("nb de pts dans la cible : "+s);
    }
}

```

Exercice 7: Lendemain. Ecrire un programme qui calcule la date du lendemain d’une date saisie par trois entiers, **jour** pour le numéro de jour dans la semaine, **mois** pour le numéro de mois, et **annee** pour l’année. Le mois devra être affiché en toutes lettres. On ne tiendra pas compte des années bissextiles.

```

public class Lendemain2
{
    public static void main(String args[])
    {
        int jour ,mois ,annee;
    }
}

```

```

System.out.println("Jour?");
jour = Lire.i();
System.out.println("Mois?");
mois = Lire.i();
System.out.println("Annee?");
annee = Lire.i();

int jourLendemain = jour;
int moisLendemain = mois;
int anneeLendemain = annee;

if (mois == 1 || mois == 3 || mois == 5 || mois == 7 || mois == 8 ||
    mois == 10 || mois == 12)
    if (jour < 31)
        jourLendemain = jour + 1;
    else
        if (jour == 31)
            {
                jourLendemain = 1;
                if (mois == 12)
                    {
                        anneeLendemain = annee+1;
                        moisLendemain = 1;
                    }
                else
                    moisLendemain = mois + 1;
            }
        else
            System.out.println("jour incorrect");
if (mois == 4 || mois == 6 || mois == 9 || mois == 11)
    if (jour < 30) jourLendemain = jour + 1;
    else
        {
            if (jour == 30)
                {
                    jourLendemain = 1;
                    moisLendemain = mois + 1;
                }
            else
                System.out.println("jour incorrect");
        }
}
if (mois == 2)
    if (jour < 28)
        jourLendemain = jour + 1;
    else
        {
            if(jour==28)
                {
                    jourLendemain = 1;
                    moisLendemain = mois + 1;
                }
        }
}

```

```
    else
      System.out.println("jour incorrect");
  }
}
```

4 Les répétitives

4.1 Le cours

Cette partie permet d'aborder les boucles répétitives *for*, *while* et *do*.

Boucle *for*

La boucle *for* permet de répéter plusieurs instructions en faisant varier un indice d'une valeur donnée (initialisation) à une autre valeur donnée (condition) en incrémentant ou décrémentant cet indice. La syntaxe est la suivante :

```
for(initialisation; condition; incrementation)
{ /* Instructions */ }
```

Boucle *while*

Tant que la condition est vraie (true), le programme exécute les instructions de la boucle *while*.

- La condition est vérifiée avant d'exécuter les instruction.
- Il est **nécessaire** de modifier l'objet de la condition à l'intérieur de la boucle pour que la boucle puisse s'arrêter.

```
while(condition)
{ /*Instructions*/ }
```

Boucle *do*

Le programme exécute les instructions de la boucle *do*, puis vérifie ensuite la condition pour éventuellement continuer.

- La condition est vérifiée après avoir exécuté les instruction.
- Il est là aussi **nécessaire** de modifier l'objet de la condition à l'intérieur de la boucle pour que la boucle puisse s'arrêter.

```
do
{ /*Instructions*/ }
}while(condition);
```

4.2 Exercices corrigés

Exercice 1: Majeurs. Ecrire un programme qui permet de saisir l'âge de 20 personnes puis qui indique le nombre d'adultes et d'enfants.

```
public class Majeurs
{
    public static void main(String args[])
    {
        int a, c = 0;
        for (int i = 0; i < 20; i++) {
```

```

        System.out.println("age : ");
        a = Lire.i();
        if (a >= 18) c = c+1;
    }
    System.out.println("Il y a : "+c+" adultes et "+(20-c)+" enfants");
}
}

```

Exercice 2: Moyenne pondérée. Ecrire un programme qui calcule la moyenne pondérée d'une suite de notes. Les notes ainsi que leur coefficients sont saisis au clavier jusqu'à ce que la note -1 soit entrée. Les notes sont des réels entre 0 et 20, les coefficients sont des entiers entre 1 et 10.

```

public class Notes
{
    public static void main(String args[])
    {
        double n=0,snp=0;
        int c=0,sc=0;

        do {
            do {
                System.out.println("note entre 0 et 20 ou -1 pour quitter: ");
                n = Lire.d();
            }
            while (n != -1 && (n < 0 || n > 20));
            if (n != -1)
            {
                do {
                    System.out.println("coef entre 1 et 10: ");
                    c = Lire.i();
                }
                while (c < 0 || c > 10);
                snp = snp+c*n;
                sc = sc+c;
            }
        }
        while (n != -1);
        if (snp != 0)
            System.out.println("moyenne : "+snp/sc);
        else
            System.out.println("pas de note");
    }
}

```

Exercice 3: Loto. Ecrire un programme qui génère aléatoirement six nombres entiers différents entre 1 et 49 inclus.

```

public class Loto
{
    public static void main(String args[])
    {
        int x1, x2, x3, x4, x5, x6;
        x1 = (int)(Math.random()*49+1);
    }
}

```

```

do x2 = (int)(Math.random()*49+1); while (x2 == x1);
do x3 = (int)(Math.random()*49+1); while (x3 == x2 || x3 == x1);
do x4 = (int)(Math.random()*49+1); while (x4 == x3 || x4== x2 || x4 ==
    x1);
do x5 = (int)(Math.random()*49+1); while (x5 == x4 || x5 == x3 || x5==
    x2 || x5 == x1);
do x6 = (int)(Math.random()*49+1); while (x6 == x5 || x6 == x4 || x6
    == x3 || x6== x2 || x6 == x1);
System.out.println(x1+" "+x2+" "+x3+" "+x4+" "+x5+" "+x6);
}
}

```

Exercice 4: Pi. Ecrire un programme qui calcule pour n donné la valeur de $1 + \frac{1}{4} + \frac{1}{9} + \frac{1}{16} + \frac{1}{25} + \dots + \frac{1}{n}$ en utilisant successivement les boucles **for**, **while** et **do ...while**. En déduire une valeur approchée de π sachant que $\frac{\pi^2}{6} = \sum_{i=1}^{\infty} \frac{1}{i^2}$.

```

public class Pi1
{
    public static void main(String args[])
    {
        int n;
        double x;
        x = 0;
        System.out.println("Donner n");
        n = Lire.i();
        for (int i = 1; i <= n; i++) {
            x = x + 1.0/(i*i); // Ne pas faire la division entière 1/(i*i)
        }
        System.out.println("Valeur approchée de pi : " + Math.sqrt(6*x));
    }
}

```

```

public class Pi2
{
    public static void main(String args[])
    {
        int n,i;
        double x;
        x = 0;
        System.out.println("Donner n");
        n = Lire.i();
        i=1;
        while(i<=n)
        {
            x = x + 1.0/(i*i); // Ne pas faire la division entière 1/(i*i)
            i=i+1;
        }
        System.out.println("Valeur approchée de pi : " + Math.sqrt(6*x));
    }
}

```

```

public class Pi3
{
    public static void main(String args[])
    {
        int n,i;
        double x;
        x = 0;
        System.out.println("Donner n");
        n = Lire.i();
        i=1;
        do
        {
            x = x + 1.0/(i*i); // Ne pas faire la division entière 1/(i*i)
            i=i+1;
        }
        while(i<=n);
        System.out.println("Valeur approchée de pi : " + Math.sqrt(6*x));
    }
}

```

Exercice 5: PGCD. Ecrire un programme pour calculer, par la méthode d'Euclide, le PGCD de deux nombres entiers.

```

public class pgcd
{
    public static void main(String args[])
    {
        int a,b;
        System.out.println("Entrer a");
        a = Lire.i();
        System.out.println("Entrer b");
        b = Lire.i();
        while(a%b != 0)
        {
            r=a%b;
            a=b;
            b=r;
        }
        System.out.println("pgcd = "+b);
    }
}

```

Exercice 6: Pierre - Feuille - Ciseau. Ecrire un programme qui permet de jouer à Pierre - Feuille - Ciseau contre l'ordinateur. Le jeu d'arrête lorsque l'un des deux joueurs a 5 points; le nom du gagnant doit s'afficher à l'écran.

```

public class pierrefeuilleciseau
{
    public static void main(String args[])
    {
        int joueur, ordi, ptjoueur=0,ptordi=0;
        do
        {
            ordi=(int)(1+3*Math.random());

```

```
System.out.println("Tapez 1 pour pierre, 2 pour feuille et 3 pour  
ciseau");  
joueur=Lire.i();  
if((ordi==1 && joueur==3)|| (ordi==2 && joueur==1)|| (ordi==3 &&  
joueur==2))  
    ptordi=ptordi+1;  
if((joueur==1 && ordi==3)|| (joueur==2 && ordi==1)|| (joueur==3 &&  
ordi==2))  
    ptjoueur=ptjoueur+1;  
}  
while(ptjoueur<5 && ptordi<5);  
if(ptjoueur==5)  
    System.out.println("Vous avez gagné");  
else  
    System.out.println("L'ordinateur a gagné");  
}  
}
```

5 Bilan n°1

5.1 Le point méthode

Pour faire ces exercices, entraînez-vous à développer vos propres solutions. Pour ce faire, les corrections sont données en page , à la fin de ce livre.

5.2 Exercices

Exercice 1: Joli dessin.

On souhaite afficher pour tout $n \geq 4$ le dessin suivant:

```
[frame=single]
* * * ...
 * * ...
* * * ...
 * * ...
* * * ...
. . .
. . .
```

Exercice 2: Combinaisons C_n^p .

Question 1. Ecrire un programme permettant de calculer $n!$. En déduire un programme permettant de calculer $C_n^p = \frac{n!}{p!(n-p)!}$.

Question 2. Vérifier alors la formule (sans utiliser *Math.pow*) $\sum_{p=0}^n C_n^p = 2^n$

Question 3. Refaire le programme du calcul de C_n^p pour qu'il fonctionne pour des valeurs de n plus grandes. Remarquons que : $C_n^p = \frac{n!}{p!(n-p)!} = \frac{n(n-1)\dots(n-p+1)(n-p)\dots3\ 2\ 1}{p!(n-p)(n-p-1)\dots3\ 2\ 1}$. Donc on a : $C_n^p = \frac{n}{1} \cdot \frac{n-1}{2} \cdot \frac{n-2}{3} \dots \frac{n-i+1}{i} \dots \frac{n-p+1}{p}$.

Exercice 3: Base 2.

Ecrire un programme qui réalise la décomposition binaire d'un entier positif. Exemple: la décomposition binaire de 26 est 11010. On pourra utiliser une chaîne de caractères pour stocker la décomposition binaire.

Exercice 4: Jeu.

Ecrire un programme qui permet à un utilisateur de rechercher un nombre entier aléatoire entre 1 et 100 généré par la machine. On imposera au plus 7 coups et on affichera à chaque coup si le nombre proposé par l'utilisateur est trop grand ou trop petit.

Exercice 5: Approximation de π .

Nous présentons dans cet exercice deux formules permettant d'obtenir une approximation de π .

Méthode 1 : La formule de Wallis (1665) $\frac{\pi}{2} = \frac{2}{1} \cdot \frac{2}{3} \cdot \frac{4}{3} \cdot \frac{4}{5} \cdot \frac{6}{5} \cdot \frac{6}{7} \dots$ Ecrire un programme permettant de trouver une approximation de π en utilisant les 1000 premiers facteurs de ce produit.

Méthode 2 : La formule de Viète (1593) $\frac{2}{\pi} = \frac{\sqrt{2}}{2} \cdot \frac{\sqrt{2+\sqrt{2}}}{2} \cdot \frac{\sqrt{2+\sqrt{2+\sqrt{2}}}}{2} \dots$

Exercice 6: Les nombres parfaits, amis et d'Armstrong.

Nombres parfaits: Un entier naturel est un nombre parfait s'il est égal à la somme de ses diviseurs propres. Par exemple, les diviseurs de 6 sont 1,2,3,6; ses diviseurs propres sont 1,2,3; la somme de ses diviseurs propres est $1 + 2 + 3 = 6$; 6 est donc un nombre parfait. Ecrire un programme qui affiche tous les nombres parfaits inférieurs à 10000.

Nombres amis : Deux entiers naturels sont des nombres amis si chacun d'entre eux est égal à la somme des diviseurs propres de l'autre. Ecrire un programme qui affiche la liste des couples de nombres amis inférieurs à 10000.

Nombres d'Armstrong : On appelle nombre d'Armstrong, un entier naturel autre que 0 et 1 qui a la propriété d'être égal à la somme des cubes de ses chiffres. Ecrire un programme permettant d'afficher tous les nombres d'Armstrong compris entre 2 et 1000.

Exercice 7: Evolution d'une urne.

Soit une urne contenant deux boules, une noire et une rouge. On veut simuler le schéma suivant: on tire une boule, si celle-ci est rouge on remet cette boule dans l'urne et on ajoute une autre boule rouge; si on tire une boule noire on la remet dans l'urne et on ajoute deux autres boules noires. Le but de l'exercice est d'écrire un programme qui affiche à chaque étape le nombre de boules de chaque couleur.

6 Les tableaux.

6.1 Le cours

Dans cette partie, on va manipuler des objets permettant de stocker plusieurs éléments du même type : les tableaux.

On déclare un tableau, c'est à dire qu'on lui réserve une adresse mémoire avec cette syntaxe :

- Tableau 1D: Type des éléments `tab[]`;
- Tableau 2D: Type des éléments `tab[][]`;

Le nombre d'éléments du tableau est appelé la taille du tableau. Les éléments sont repérés par un ou plusieurs indices de type `int` (ou `char`) et le nombre d'indices est la dimension du tableau. Pour définir la taille d'un tableau, on utilise la syntaxe :

- Tableau 1D: `tab=new Type des éléments[Nbre cases]`;
- Tableau 2D: `tab=new Type des éléments[nblig][nbcoll]`;

Par exemple :

Déclaration d'un tableau 1D de 10 cases contenant des entiers :

```
int tab [] ;
tab=new int [10] ;
```

Qui donne :

1	3	5	7	9	0	2	4	6	8
---	---	---	---	---	---	---	---	---	---

0 1 2 3 4 5 6 7 8 9 Indices du tableau

On accède à la i -ième case du tableau `tab` en appelant `tab[i-1]` .

Déclaration d'un tableau 2D 2 lignes et 10 colonnes contenant des entiers :

```
int tab [] [] ;
tab=new int [2] [10] ;
```

Ce qui crée :

1	3	5	7	9	0	2	4	6	8	0
0	2	4	6	8	1	3	5	7	9	1

0 1 2 3 4 5 6 7 8 9 Indices du tableau

On accède à la case située sur la i -ième ligne et j -ième colonne en appelant `tab[i-1][j-1]` .

Attention : Les indices pour repérer les cases (donc les lignes et les colonnes) des tableaux commencent à 0.

6.2 Exercices corrigés

Exercice 1 : Tableau à une dimension. Soit `Tab` un tableau à une dimension de taille 10 contenant des entiers. On souhaite écrire un programme qui propose à l'utilisateur un menu composé de la liste des

opérations ci-dessous:

- 1) Remplir aléatoirement le tableau Tab avec 10 entiers compris entre 1 et 10 inclus.
- 2) Afficher le tableau Tab à l'écran.
- 3) Décaler d'une case à droite chaque élément du tableau (le dernier élément est déplacé dans la première case).
- 4) Afficher toutes les permutations circulaires du tableau.
- 5) Echanger le contenu de deux cases consécutives lorsque l'entier de la première case est supérieur à celui de la seconde. On fera les échanges en parcourant une seule fois le tableau de la gauche vers la droite.
- 6) Déduire de la question 5 une méthode pour trier les éléments du tableau dans l'ordre croissant.

```
public class Tableau1D
{
    public static void main (String [] args)
    {
        int i, j, choix, aux, Tab[] = new int [10];
        boolean trie;
        do
        { System.out.print("donnez votre choix:");
          choix=Lire.i();
          switch (choix)
          { case 1: for(i=0;i<=9;i++) Tab[i]=1+(int)(Math.random()*10);
                break;
            case 2: for(i=0;i<=9;i++) System.out.print(Tab[i]+" ");
                System.out.println();
                break;
            case 3: aux=Tab[9];
                for(i=9;i>=1;i--) Tab[i]=Tab[i-1];
                Tab[0]=aux;
                break;
            case 4: for(i=0;i<=9;i++) System.out.print(Tab[i]+" ");
                System.out.println();
                for(j=1;j<=8;j++)
                { aux=Tab[9];
                  for(i=9;i>=1;i--) Tab[i]=Tab[i-1];
                  Tab[0]=aux;
                  for(i=0;i<=9;i++) System.out.print(Tab[i]+" ");
                  System.out.println();
                }
                break;
            case 5: for(i=0;i<9;i++)
                    if(Tab[i]>Tab[i+1])
                    { aux=Tab[i]; Tab[i]=Tab[i+1]; Tab[i+1]=aux; }
                break;
            case 6: trie=false;
                while(!trie)
                { trie=true;
                  for(i=0;i<9;i++)
                  if(Tab[i]>Tab[i+1])
                  { trie=false;
                    aux=Tab[i]; Tab[i]=Tab[i+1]; Tab[i+1]=aux;
                  }
                }
          }
        }
    }
}
```

```

    } //end switch
}
while (choix >=1 && choix <=6);
}
}

```

Exercice 2 : Jeu du master mind. On veut écrire le programme qui permet de jouer au master mind contre l'ordinateur. Le principe du jeu est le suivant: l'ordinateur place dans 5 trous, 6 clous de couleurs différentes choisies parmi 8 possibles. Le joueur doit trouver la bonne combinaison en proposant des combinaisons de 5 clous de couleurs différentes. A chaque essai, l'ordinateur indiquera le nombre de clous *bien placés* (même place et même couleur) et le nombre de clous *mal placés* (bonne couleur mais mauvaise place). Le jeu s'arrête lorsque le joueur a trouvé la bonne combinaison ou lorsque le nombre d'essai a atteint 12.

```

public class Master
{
public static void main (String [] args)
{ boolean yest=True;
int joueur[]=new int [5];
ordi[]=new int [5];
couleur[]=new int [8];
v0,v1,v2,v3,v4,i,indice,bplaces,mplaces,essai=0; //Question 1
v0=(int)(Math.random()*8);
do { v1=(int)(Math.random()*8);
} while(v1==v0);
do { v2=(int)(Math.random()*8);
} while(v2==v0 || v2==v1);
do { v3=(int)(Math.random()*8);
} while(v3==v0 || v3==v1 || v3==v2);
do { v4=(int)(Math.random()*8);
} while(v4==v0 || v4==v1 || v4==v2 || v4==v3);
ordi[0]=v0; ordi[1]=v1; ordi[2]=v2; ordi[3]=v3; ordi[4]=v4;
for(i=0;i<=7;i++) //Question 1 Solution 'Futée'
couleur[i]=i+1;
for(i=0;i<=4;i++)
{indice=(int)((8-i)*Math.random()); // entier aléatoire dans [0,7-i]
ordi[0]=couleur[indice];
temp=couleur[indice]; // on échange la case indice avec la case
d'indice 7-i
couleur[indice]=couleur[7-i];
couleur[7-i]=temp;
} //Question 2
do{ for(i=0;i<=4;i++)
{ yest=False;
do{
System.out.print("Donnez un entier entre 0 et 7: ");
joueur[i]=Lire.i();
for(j=0;j<i;j++)
if(joueur[j]==joueur[i])
yest=True;
}
while(yest);
}
}
}

```

```

} // Question 3
bplaces=0;
for(i=0; i<=4; i++)
    if(ordi[i]==joueur[i])
        bplaces=bplaces+1;
mplaces=0;
for(i=0; i<=4; i++)
    for(j=0; j<=4; j++)
        if(ordi[i]==joueur[j] && i!=j)
            mplaces=mplaces+1;
System.out.println("bien places " + bplaces + ", mal places " +
    mplaces);
essai++;
} while(bplaces!=5 && essai<=12);
if(bplaces==5)
    System.out.println("Gagne");
else
    System.out.println("Perdu");
}
}

```

Exercice 3 : Triangle de Pascal (deux solutions). Ecrire un programme permettant de construire et d'afficher les n premières lignes du triangle de Pascal (n étant choisi par l'utilisateur). Rappelons que l'on a la formule $C_n^p = C_{n-1}^p + C_{n-1}^{p-1}$.

```

public class Pascal1
{
    public static void main (String [] args)
    { int n, i, j, C[][];
      System.out.print("Donnez n");
      n=Lire.i();
      C = new int [n][n];
      C[0][0]=1; System.out.println(C[0][0]);
      for(i=1; i<n; i++)
      { C[i][0]=1; C[i][i]=1;
        for(j=1; j<=i-1; j++) C[i][j]=C[i-1][j]+C[i-1][j-1];
        for(j=0; j<=i; j++) System.out.print(C[i][j]+"\\t");
        System.out.println();
      }
    }
}

public class Pascal2
{
    public static void main (String [] args)
    { int n, i, j, C[];
      System.out.print("Donnez n:"); n=Lire.i();
      C = new int [n];
      C[0]=1; System.out.println(C[0]);
      for(i=1; i<n; i++)
      { C[i]=1;
        for(j=i-1; j>=1; j--) C[j]=C[j]+C[j-1];
        for(j=0; j<=i; j++) System.out.print(C[j]+"\\t");
      }
    }
}

```

```

        System.out.println();
    }
}
}

```

Exercice 4 : Lancers de dés. On lance simultanément deux dés à 6 faces (numérotés de 1 à 6) et on considère la somme des deux chiffres obtenus. On effectue n fois (n choisi par l'utilisateur inférieur à 100) cette opération. Ecrire un programme permettant de stocker dans un tableau la fréquence d'apparition de chaque somme possible. Modifier ce programme pour qu'à partir du tableau des fréquences, il affiche un tableau à deux dimensions simulant l'histogramme des fréquences. Par exemple, si le tableau des fréquences est :

Somme des 2 dés	2	3	4	5	6	7	8	9	10	11	12
Fréquence d'apparition	0	4	1	3	0	0	1	4	3	2	5

on doit obtenir:

```

                *
*           *   *
*  *       * *  *
*  *       * *  * *
* * *     * * * * *
-----
2 3 4 5 6 7 8 9 10 11 12
-----

```

```

public class LancerDes
{
    public static void main (String [] args)
    {
        int n, i, j, u, max, F[]=new int [13];
        for(i=0;i<=12;i++) F[i]=0;
        System.out.print("Donnez n");
        n=Lire.i();
        for(i=1;i<=n;i++)
        {
            u=1+(int)(6*Math.random()+1)+(int)(6*Math.random());
            F[u]++;
        }
        max=F[2];
        for(i=3;i<=12;i++) if(max<F[i]) max=F[i];
        for(j=max;j>=1;j--)
        {
            for(i=2;i<=12;i++)
                if(F[i]>=j) System.out.print("* ");
            else System.out.print(" ");
            System.out.println();
        }
        for(i=2;i<=12;i++) System.out.print("_ ");
        System.out.println();
        for(i=2;i<=12;i++)
            if (i<10) System.out.print(i+" ");
            else System.out.print(i+" ");
        System.out.println();
        for(i=2;i<=12;i++) System.out.print("_ ");
        System.out.println();
    }
}

```

7 Les chaînes de caractères.

7.1 Le cours

Cette partie se concentre sur l'utilisation des chaînes de caractères (type String). Ce sont des suites ordonnées de caractères (type char). Par exemple : "abcdefghijkl000lmnopqrst123uvwxyz" est une chaîne de caractères (entourée de guillemets "..."). Les caractères sont classés par le code ASCII.

Voici les principales actions sur les chaînes de caractères :

Déclaration d'une chaîne de caractère mot, initialisation de mot à la chaîne vide puis affectation :

```
String mot;  
mot="";  
mot="Bonjour";
```

Accès à la *i*-ième lettre du mot :

```
mot.charAt(i-1)
```

Longueur de la chaîne de caractère mot :

```
mot.length()
```

Comparaison de deux chaînes de caractères mot1 et mot2 :

```
if(mot1.equals(mot2))
```

Juxtaposition de deux chaînes (concaténation) :

```
chaine3 = chaine1+chaine2
```

Remarque : Il y a beaucoup d'autres méthodes sur les chaînes de caractères, mais elles relèvent de l'optimisation d'actions que l'on peut déjà faire avec les 5 méthodes de base citées ci-dessus. Il convient alors de maîtriser d'abord ces basiques.

7.2 Exercices corrigés

Exercice 1: Traitement de chaînes. Ecrire un programme qui propose le menu suivant et exécute le traitement demandé.

1. Saisir un mot
2. Donner le miroir du mot
3. Remplacer toutes les occurrences d'une lettre par une autre
4. Supprimer toutes les occurrences d'une lettre donnée dans la chaîne
5. Déterminer si le mot est un palindrome.
6. Quitter le programme.

```
public class chainechar  
{  
    public static void main(String args[])  
    {  
        String mot;  
        int choix ;  
        char anc, nv;  
        String lettre, res;
```

```

char let;

mot="";
do
{
    System.out.println("1- saisir mot");
    System.out.println("2- afficher miroir");
    System.out.println("3- remplacer un caractere");
    System.out.println("4- supprimer une lettre");
    System.out.println("5- verifier si palindrome");
    System.out.println("6- quitter");

    do { System.out.println("Votre choix (1-6):");
        choix=Lire.i();
    }
    while (choix<1 || 6<choix);

    switch (choix)
    { case 1 :
        System.out.print("Entrer un mot : "); mot=Lire.S();
        break;
      case 2 :
        for(int i=mot.length()-1;i>=0;i--)
            System.out.print(mot.charAt(i));
        System.out.println();
        break;
      case 3 :
        System.out.print("Entrer lettre a remplacer: ");
        anc=Lire.c();
        System.out.print("Entrer lettre remplacement: ");
        nv=Lire.c();
        res="";
        for(int i=0; i<mot.length();i++)
            if (mot.charAt(i)==anc)
                res = res + nv;
            else
                res = res + mot.charAt(i);
        System.out.println(res);
        mot=res;
        break;
      case 4 :
        System.out.print("Entrer lettre a supprimer : ");
        let=Lire.c();
        res = "";
        for (int i=0; i<mot.length();i++)
            if (mot.charAt(i)!=let)
                res = res+mot.charAt(i);
        System.out.println(res);
        mot=res;
        break;
      case 5 :

```

```

        res = "";
        for(int i=mot.length()-1;i>=0;i--)
            res = res + mot.charAt(i);
        if (res.equals(mot))
            System.out.println(mot + " est un palindrome");
        else
            System.out.println(mot + " n est pas un palindrome
                                (+res+)");
        break;
    }
} while (choix !=6);
}
}

```

Exercice 2: Numéro de sécurité sociale. Un numéro de sécurité sociale est une chaîne de 13 caractères. Il se compose de la façon suivante:

- le premier chiffre indique le sexe (1 ou 2),
- les deux suivants indiquent l'année de naissance,
- les deux d'après indiquent le mois de naissance,
- le sixième et septième chiffre réunis indiquent le département de naissance,
- les trois suivants indiquent le code de la commune de naissance,
- les trois derniers spécifient le numéro de dossier.

Ecrire un programme qui demande à l'utilisateur de fournir un numéro de sécurité sociale, jusqu'à ce qu'il soit correct, c'est à dire qu'il vérifie les conditions suivantes:

- il comporte exactement 13 caractères,
- chaque caractère est un chiffre,
- le premier chiffre, indiquant le sexe, est 1 ou 2
- le numéro de mois de naissance est compris entre 1 et 12.

```

public class NumSS
{
    public static void main(String[] args)
    {
        String num;
        boolean correct;
        int a,b,c;
        do
        {
            System.out.println("Entrer un numero de securite sociale : ");
            num = Lire.S();
            correct = true;

            if (num.length() != 13)
            {
                correct = false; System.out.println("longueur
                    incorrecte");}
            else
            {
                for(int i=0;i<num.length();i++)
                {
                    char caract =num.charAt(i);
                    if (caract<'0' || caract >'9')
                    {
                        correct = false; System.out.println((i+1)+" eme
                            caract pas correct");}
                }
            }
        }
    }
}

```

```

    }

    if (!num.charAt(0)=='1' && !num.charAt(0)=='2')
    {
        correct = false; System.out.println("sexe incorrect");}

    a=num.charAt(3)-'0';
    b=num.charAt(4)-'0';
    c=a*10+b;
    if (c<1 || c>12)
    {
        correct = false; System.out.println("mois incorrect");}

    }

} while (!correct);
}
}

```

Exercice 3: Code secret. Le ROT13 est un code très simple où on remplace chaque lettre de l'alphabet par celle qui vient 13 rangs plus loin, en rebouclant de Z à A : A devient N, B devient O, L devient Y, M devient Z, N devient A, etc. Comme $13+13=26$ et qu'il y a 26 lettres dans l'alphabet, le même code fonctionne pour chiffrer et déchiffrer. Ecrire un programme qui effectue le chiffrement ROT13 sur les chaînes entrées par l'utilisateur.

```

public class Rot13
{
    public static void main(String[] args)
    {
        String chE;
        String chS="";

        System.out.println("Entrer un texte a coder :");
        chE = Lire.S();

        chS="";
        for(int i=0;i<chE.length();i++)
        {
            if(chE.charAt(i)+13>'z')
                chS=chS+(char)(chE.charAt(i)-13);
            else
                chS=chS+(char)(chE.charAt(i)+13);
        }
        System.out.println(chE+" codee = "+chS);
    }
}

```

Exercice 4: Cryptage. Ecrire un programme qui demande à l'utilisateur de saisir 26 lettres pour chiffrer les 26 lettres de l'alphabet (et vérifie qu'il ne rentre pas deux fois la même), puis permet de chiffrer ou déchiffrer des chaînes avec ce code.

```

public class Codage3
{
    public static void main(String[] args)
    {

```

```

String alphabet ="abcdefghijklmnopqrstuvwxy";
String convers="";
String mot;
String res;

char c;
int nb;
boolean ok;
char rep;

nb=alphabet.length();
for (int i=0;i<nb;i++)
{ do
  { ok = true;
    System.out.print("Code pour "+alphabet.charAt(i) +": ");
    c=Lire.c();
    for (int j=0;j<i;j++)
      if (c==convers.charAt(j))
        { ok = false; System.out.println("lettre
          incorrecte");}
    }while (!ok);
  convers=convers+c;
}
System.out.println();
for(int i=0;i<nb;i++) System.out.print(alphabet.charAt(i)+" ");
System.out.println();
for(int i=0;i<nb;i++) System.out.print(convers.charAt(i)+" ");
System.out.println();
do
{ System.out.println("Entrer le texte a coder :");
  mot=Lire.S();
  System.out.println("Saisir (c) pour coder et (d) pour decoder
    : ");
  rep=Lire.c();

  res="";
  for (int i=0;i<mot.length();i++)
  { int pos=0;

    if (rep=='c')
    { while (alphabet.charAt(pos)!=mot.charAt(i)) pos++;
      res = res+convers.charAt(pos);
    }
    else
    {while (convers.charAt(pos)!=mot.charAt(i)) pos++;
      res = res+alphabet.charAt(pos);
    }
  }
}
System.out.println(mot+" ==> "+res);

```

```
        System.out.println("voulez-vous recommencer (o/n)
        ?");rep=Lire.c();
    } while (rep=='o');
}
}
```

8 Les fonctions.

8.1 Le cours

Une **fonction** est un bout de code qui a un **nom** et qui peut être appelée depuis le reste du programme.

Exemples de fonctions prédéfinies: `Math.pow(a,b)`, `Math.abs(a)`, `Math.random()`, `Math.sqrt(a)`, ...

L'intérêt des fonctions est multiple :

- Meilleure lisibilité et concision (cela évite d'écrire plusieurs fois le même code)
- Niveau d'abstraction plus élevé (un nom remplace une partie complexe du programme)
- Gain de place mémoire
- Modification plus aisée
- Partage de fonctions avec des bibliothèque de fonction)
- Récursivité (voir chapitre 9)

On déclare une fonction à l'extérieur du **main** mais à l'intérieur de **class**. Il est conseillé de les déclarer juste après le programme principal :

```
public class programme
{
    public static void main(String args [])
    {
        // Programme principal
    }

    // Déclaration des fonctions
}

```

Il y a deux types de fonction ; celles qui renvoient une valeur et celles qui ne renvoient rien (mais peuvent servir à afficher quelque chose à l'écran, par exemple).

Déclaration d'une fonction *toto* qui renvoie une valeur entière, à partir de deux entiers donnés *a* et *b* :

```
public int toto(int a,int b)
{ //instruction
    return w; //retour du résultat
}

```

Déclaration d'une fonction qui ne renvoie pas de valeur, à partir de deux entiers donnés *a* et *b* :

```
public void toto(int a,int b)
{ //instruction
}

```

- Lors de l'exécution de la fonction, les **paramètres formels**, c'est à dire ceux de la définition (ici, le *a* et le *b*) sont remplacés par les **paramètres effectifs** utilisés dans le programme principal.
- Les paramètres d'une fonction de types int, char, double, boolean, String **ne peuvent pas être modifiés** au cours de l'exécution d'une fonction. On dit que les paramètres sont **passés par valeur**.
- Les paramètres d'une fonction de types tableau 1D ou 2D **peuvent être modifiés**. On dit que les paramètres sont **passés par références ou par variable**.
- Une variable définie dans une fonction s'appelle une **variable locale**. Elle est connue uniquement dans la fonction lors de son exécution.

8.2 Exercices corrigés

Exercice 2: Les entêtes. Ecrire les entêtes des fonctions suivantes.

- sur les entiers: test si un nombre est pair, calcul du nombre des diviseurs, pgcd
- sur les réels: calcul de la racine carrée, moyenne de deux réels
- sur les mots: test si palindrome, calcul du miroir, nombre d'occurrences d'une lettre
- sur les tableaux à une dimension d'entiers: action de saisie, affichage, calcul du min.

```
boolean estPair(int nb)
int nbDiviseur(int nb)
int pgcd(int a, int b)

double racine(double nb)
double moy(double a, double b)

boolean palindrom(String mot)
String miroir(String mot)
int nbOcc(String mot, char c)

int[] saisie() ou void saisie(int[] t)
void affichage(int[] t)
int mini(int[] t)
```

Exercice 3: Les entêtes. Ecrire un programme permettant d'effectuer les principales opérations arithmétiques sur les nombres réels. Le menu suivant sera proposé à l'utilisateur et le traitement de chaque choix fera l'objet d'une fonction.

- x saisie du premier opérande
- y saisie du deuxième opérande
- + addition x+y
- - soustraction x-y
- * multiplication x*y
- / division x/y
- p puissance x^y
- = affichage de x et y
- q quitter le programme

Le résultat de chaque opération doit être affiché et stocké dans x.

```
public class Operations
{ // version sans optimisation
    public static double saisie()
```

```

{ double val;
  System.out.print("Entrer une valeur : ");
  val=Lire.d();
  return val;
}

public static double addition(double a, double b)
{ return a+b;}

public static double soustraction(double a, double b)
{ return a-b; }

public static double multiplication(double a, double b)
{ return a*b; }

public static double division(double a, double b)
{ return a/b; }

public static double puissance(double a, double b)
{ return Math.pow(a, b); }

public static void affichage(double a, double b)
{   System.out.println("x="+a+" et y="+b);   }

public static void main(String[] args) {
    double x=0, y=0;
    char choix;

do
{   // menu
    System.out.println("x  saisie de 1er operande");
    System.out.println("y  saisie de 2eme operande");
    System.out.println("+  addition");
    System.out.println("-  soustraction");
    System.out.println("*  multiplication");
    System.out.println("/  division");
    System.out.println("^  puissance");
    System.out.println("="  affichage de x et y");
    System.out.println("q  quitter le programme");

    // saisie choix utilisateur
    System.out.print("Entrer votre choix : ");
    choix=Lire.c();

    // traitement du choix
    switch (choix)
    {
        case 'x' : x=saisie();
                  break;
        case 'y' : y=saisie();
                  break;
    }
}

```

```

        case '+' : x=addition(x,y);affichage(x,y);
                break;
        case '-' : x=soustraction(x,y);affichage(x,y);
                break;
        case '*' : x=multiplication(x,y);affichage(x,y);
                break;
        case '/' : x=division(x,y);affichage(x,y);
                break;
        case '^' : x=puissance(x,y);affichage(x,y);
                break;
        case '=' : affichage(x,y);
    }
} while (choix!='q');
}
}

```

Exercice 4 : cogitus Dans le jeu du cogitus, on dispose d'un damier de 7*7 cases et de 30 pions sur lesquels sont écrits des chiffres positifs. Le jeu consiste à poser ces pions sur les cases de telle sorte que chaque ligne et chaque colonne soit valide. Une ligne est valide si chacune des sommes des pions consécutifs sur la ligne est inférieure ou égale à 11. Idem pour une colonne. Ecrire les fonction suivantes:

1. initialisation du damier avec des cases vides.
2. affichage du damier
3. test de validité de la ligne numéro i du damier.
4. test de validité de la colonne numéro j du damier.
5. test si un pion de valeur v peut être mis sur une certaine case du damier.

Ecrire le programme principal qui permet à l'utilisateur de jouer au cogitus : à chaque coup, l'ordinateur génère un chiffre entre 1 et 9 aléatoirement et le joueur doit proposer un placement de ce chiffre sur la grille après quoi l'ordinateur vérifie que la configuration est toujours valide (si ce n'est pas le cas, on redemande le placement du chiffre) et ainsi de suite jusqu'à ce que les 30 chiffres soient placés ou que le joueur abandonne la partie. A la fin de la partie, on demande au joueur s'il souhaite rejouer.

```

public class cogitus
{
    public static void main (String args [])
    {
        int [][] t=new int[7][7];
        int l; // numero de la ligne
        int c; // numero de la colonne
        int val; // valeur à placer dans la case
        char rep; // reponse utilisateur
        int cpt; // compteur

        ini(t);
        cpt=0;
        do
        {affichage(t); // affichage du contenu de la grille
          val = (int)(Math.random()*9+1); // generation de la valeur
            aleatoirement
          System.out.println("La valeur à placer est "+val);
          cpt++;
        }
        do
    }
}

```

```

    {
        System.out.println("saisir un numéro de ligne");
        System.out.println("saisir un numéro de colonne");
    } while (testcase(t, val, l, c) == false);
    System.out.println("Voulez-vous continuer o/n ?");
    rep = Lire.c();
} while (cpt < 30 && rep == 'o');
}
public static void ini(int [][] t)
{
    for (int j = 0; j < 7; j++)
        for (int i = 0; i < 7; i++)
            t[j][i] = 0;
}

public static void affichage (int [][] t)
{
    for (int j = 0; j < 7; j++)
    {
        for (int i = 0; i < 7; i++) System.out.print(t[j][i] + " ");
        System.out.println("");
    }
}

public static boolean validligne (int [][] t, int i)
{
    boolean valid = true; // valid est une variable locale
    int som = 0;
    int j = 0;
    while (j < t.length && valid)
    {
        if (t[i][j] != 0)
        {
            som = som + t[i][j];
            if (som > 11) { valid = false; }
        }
        else som = 0;
        j++;
    }
    return valid;
}

public static boolean validcolonne (int [][] t, int j)
{
    boolean valid = true;
    int som = 0;
    int i = 0;
    while (i < t.length && valid == true)
    {
        if (t[i][j] != 0)
        {
            som = som + t[i][j];
            if (som > 11) valid = false;
        }
        else som = 0;
        i++;
    }
    return valid;
}
}

```

```
public static boolean testcase (int [][] tab, int v, int i, int j)
{boolean res=true;
  if (tab[i][j]!=0)
    res=false;
  else
  { tab[i][j]=v;
    if (validligne(tab,i)==false || validcolonne(tab,j)==false)
      { res=false;
        tab[i][j]=0;
      }
  }
  return res;
}
}
```

9 La récursivité.

9.1 Le minimum à savoir

Une fonction est récursive lorsque la fonction s'appelle elle-même.

Une fonction récursive doit toujours avoir un **cas de base** sans appel récursif, et un **cas général** où intervient la récursivité.

Voici une fonction récursive pour calculer la factorielle en utilisant la formule récursive $n! = (n - 1)! \cdot n$.

```
public static int facto(int n)
{ if(n==0) // Cas de base
  return 1;
  else
  return n*facto(n-1); //Cas général
}
```

Les avantages de la récursivité sont :

- résoudre certains problèmes de manière plus intuitive et concise
- possibilité de décomposition de problèmes complexes en sous-problèmes plus simples.

Les désavantages sont alors :

- consommation excessive de mémoire (pile) par les appels récursifs si la profondeur de récursion est trop grande
- parfois moins efficaces en termes de temps d'exécution par rapport aux solutions itératives, surtout si elles effectuent des calculs redondants.

9.2 Exercices corrigés

Exercice 1 : Calcul du pgcd Ecrire une fonction récursive qui renvoie le pgcd de deux entiers en utilisant la méthode d'Euclide.

```
public static int pgcd(int a, int b)
{ int temp;
  if(a<b)
  {temp=a;
   a=b;
   b=temp;
  }
  if(a%b==0)
  return b
  else
  return pgcd(b,a%b);
}
```

Exercice 2 : Fibonacci Ecrire une fonction récursive qui renvoie le nombre de Fibonacci f_n définie par $f_0 = 0$, $f_1 = 1$, et $f_n = f_{n-1} + f_{n-2}$ pour $n \geq 2$.

```
public static int fibonacci(int n)
```

```

{ if(n==0)
  return 0;
  else
    if(n==1)
      return 1;
    else
      return fibonacc(n-1)+fibonacci(n-2);
}

```

Exercice 3 : Coefficient binomial Ecrire la définition d'une fonction récursive qui renvoie le coefficient binomial défini par: $C_n^k = 1$ si $n = 0$ ou $k = 0$ ou $k = n$, et $C_n^k = C_{n-1}^k + C_{n-1}^{k-1}$ sinon.

```

public static int binom(int n,int k)
{ if(n==0 || k==0 ||k==n)
  return 1;
  else
    return binom(n-1,k)+binom(n-1,k-1);
}

```

Exercice 4 : Palindrome Un mot est un palindrome si sa première lettre est égale à la dernière lettre et si le mot restant en supprimant la première et dernière lettre est aussi un palindrome. Ecrire une fonction récursive qui renvoie true/false selon que le mot entré en paramètre est un palindrome ou non.

```

public static boolean palin(String mot, int deb, int fin)
{
  if(deb>=fin)
    return true
  else
    if(mot.charAt(deb)==mot.charAt(fin))
      return palin(mot, deb+1, fin-1);
    else
      return false;
}

```

Appeler la fonction dans le main :

```

mot="totot";
if(palin("totot",0,mot.length()-1))
  System.out.println("palindrome")
else
  System.out.println("pas palindrome");

```

Exercice 5 : Recherche dans un tableau Ecrire une fonction récursive qui renvoie true/false selon qu'une valeur donnée se trouve ou non dans un tableau trié.

```

public static boolean rech(int[] tab, int val, int deb, int fin)
{ int milieu;
  if(deb>fin)
    return false;
  else
    {
      milieu=(deb+fin)/2;
      if(tab[milieu]==val)
        return true;
    }
}

```

```
    else
    if(tab[milieu]>val)
        return rech(tab,val, deb, milieu-1);
    else
        return rech(tab,val,milieu+1,fin);
    }
}
```

Appeler la fonction dans le main :

```
if(rech(T,elmt,0,T.length-1))
    System.out.println("La valeur est dans le tableau");
else
    System.out.println("La valeur n'est pas dans le tableau");
```

10 Bilan n°2

10.1 Le point méthode

Les exercices suivants permettent de s'entraîner en condition d'examen sur tous les chapitres, avec les corrections à la fin. Essayez de trouver vos propres solutions !

10.2 Exercices corrigés

Exercice 1 : Approximation du nombre d'Euler

Le nombre d'Euler e est égal à $\sum_{k=0}^{\infty} \frac{1}{k!}$. Donner une approximation de e en calculant la somme des n premiers termes de la suite $(\frac{1}{k!})_{k \geq 0}$.

Exercice 2 : Le crible d'Eratosthène

Il existe une méthode de détermination de tous les nombres premiers compris entre 1 et n , connue sous le nom de crible d'Eratosthène. Elle consiste à dresser une liste de tous les nombres entiers considérés et à y rayer tous les multiples d'autres entiers. Plus précisément, on procède ainsi :

- on raye le 1 (qui, par définition, n'est pas un nombre premier),
- on recherche, à partir du dernier nombre premier considéré (la première fois, on convient qu'il s'agit du 1), le premier nombre non rayé (on peut montrer qu'il est premier). Il devient, à son tour, le dernier nombre premier considéré et on raye tous ses multiples,
- on répète le traitement précédent jusqu'à ce que le nombre premier considéré soit supérieur à la racine carrée de n . On peut alors démontrer que tous les nombres non premiers ont été rayés de la liste. Écrire un programme exploitant cette méthode pour rechercher tous les nombres premiers compris entre 1 et une valeur n fournie en donnée.

Exercice 3: Récursivité

Ecrire une fonction calculant la valeur de la fonction d'Ackermann A définie pour $m \geq 0$ et $n \geq 0$ par:

- $A(m, n) = A(m - 1, A(m, n - 1))$ pour $m > 0$ et $n > 0$,
- $A(0, n) = n + 1$ pour $n \geq 0$,
- $A(m, 0) = A(m - 1, 1)$ pour $m > 0$.

Exercice 4: Les mots de Fibonacci

La suite (S_n) des mots de Fibonacci est définie par $S_1 = 1, S_2 = 0$, et pour tout $n > 2$, le mot S_n par la concaténation des mots S_{n-1} et S_{n-2} (dans cet ordre). Faire un programme Java permettant d'afficher le mot S_n lorsque n est saisi par l'utilisateur. Refaire ce programme en utilisant la récursivité.

Exercice 5: La conjugaison

Ecrire un programme qui lit au clavier un verbe du premier groupe (il s'assurera qu'il est bien terminé par *er*) et qui en affiche la conjugaison au présent de l'indicatif. On supposera qu'il s'agit d'un verbe régulier. Autrement dit, on admettra que l'utilisateur ne fournit pas un verbe tel que manger (dans ce cas, le programme affichera nous mangons !). Les résultats se présenteront ainsi :

```
Donnez un verbe regulier du premier groupe
:dire
*** il ne se termine pas par er - donnez-en un autre :
chanter
je chante
tu chantes
il/elle chante
nous chantons
```

vous chantez
ils/elles chantent

Exercice 6: Tri de mots

Ecrire un programme qui lit une suite de mots au clavier et qui les affiche triés par ordre alphabétique. On supposera que ces mots ne contiennent que des lettres non accentuées (majuscules ou minuscules). On ne saisira pas plus de 10 mots.

11 Corrections des bilans

11.1 Bilan n°1

Exercice 1

```
public class Dessin
{
    public static void main (String [] args)
    {
        int n,i,j;
        do
        {
            System.out.println("Entrer la valeur de n:");
            n=Lire.i();
        }
        while(n<4);
        for (i=1;i<=n;i++)
        {
            for(j=1;j<=n;j++)
                if((i+j)%2==0)
                    System.out.print("*");
                else
                    System.out.print(" ");
            System.out.println();
        }
    }
}
```

Exercice 2

Question 1

```
public class Combinaison
{
    public static void main (String [] args)
    {
        int n,p,i,j,Cnp,factn,factp,factnp;
        System.out.println("Entrer la valeur de n:");
        n=Lire.i();
        System.out.println("Entrer la valeur de p:");
        p=Lire.i();

        factn=1;
        for(i=1;i<=n;i++)
            factn=factn*i;

        factp=1;
        for(i=1;i<=p;i++)
            factp=factp*i;

        factnp=1;
```

```

    for(i=1;i<=n-p;i++)
        factnp=factnp*i;

    Cnp=factn/(factp*factnp);
    System.out.println(Cnp);
}
}

```

Question 2

```

public class Verif
{
    public static void main (String [] args)
    {
        int n,p,i,j,Cnp,factn,factp,factnp,Somme,puis;
        do
        {System.out.println("Entrer la valeur de n:");
          n=Lire.i();
        }
        while(n<0);
        factn=1;
        for(i=1;i<=n;i++)
            factn=factn*i;

        Somme=0;
        for(p=0;p<=n;p++)
        {
            factp=1;
            for(i=1;i<=p;i++)
                factp=factp*i;

            factnp=1;
            for(i=1;i<=n-p;i++)
                factnp=factnp*i;

            Cnp=factn/(factp*factnp);
            Somme=Somme+Cnp;
        }
        puis=1;
        for(i=1;i<=n;i++)
            puis=puis*2;
        if(Somme==puis)
            System.out.println("Formule juste");
        else
            System.out.println("Formule fausse");
        }
}

```

Question 3

```

public class Rapide
{

```

```

public static void main (String [] args)
{
    int n,p,i;
    double Cnp;

    System.out.println("Entrer la valeur de n:");
    n=Lire.i();

    System.out.println("Entrer la valeur de p:");
    p=Lire.i();

    Cnp=1;
    for(i=1;i<=p;i++)
        Cnp=Cnp*(n-i+1)/i;

    System.out.println(Cnp);
}
}

```

Exercice 3

```

public class Binaire
{
    public static void main (String [] args)
    {
        int n;
        String binaire;
        System.out.println("Entrer la valeur de n:");
        n=Lire.i();

        binaire="";
        do
        {
            if(n%2==1)
                binaire="1"+binaire;
            else
                binaire="0"+binaire;
            n=n/2;
        }
        while(n!=0);
        System.out.println(binaire);
    }
}

```

Exercice 4

```

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

```

```

int ordi ,joueur ,coups;
ordi=(int) (1+100*Math.random());
coups=0;
do
{
    System.out.println("Proposer votre nombre");
    joueur=Lire.i();
    coups=coups+1;
    if(joueur>ordi)
        System.out.println("Votre nbre est trop grand");
    if(joueur<ordi)
        System.out.println("Votre nbre est trop petit");
    }
while(joueur!=ordi && coups<7);

if(joueur==ordi)
    System.out.println("Bravo vous avez trouve en "+coups+"coups");
else
    System.out.println("Perdu");
}
}

```

Exercice 5

Méthode 1

```

public class Wallis
{
    public static void main (String [] args)
    {
        int n,i,num,denom;
        double p;
        System.out.println("Entrer la valeur de n:");
        n=Lire.i();

        p=1;
        num=0;
        denom=1;
        for(i=1;i<=n;i++)
        {
            if(num<denom)
                num=num+2;
            else
                denom=denom+2;
            p=p*num/denom;
        }

        System.out.println(2*P);
    }
}

```

Méthode 2

```
public class Viete
{
    public static void main (String [] args)
    {
        int n,i,denom;
        double P,num;
        System.out.println("Entrer la valeur de n:");
        n=Lire.i();

        P=1;
        num=0;
        denom=2;
        for(i=1;i<=n;i++)
        {
            num=Math.sqrt(2+num);
            P=P*num/denom;
        }

        System.out.println(2/P);
    }
}
```

Exercice 6

Nombres parfaits

```
public class Parfait
{
    public static void main (String [] args)
    {int n,s,i;

        for(n=1;n<=10000;n++)
        {
            s=0;
            for(i=1;i<=n/2;i++)
                if(n%i==0)
                    s=s+i;
            if(s==n)
                System.out.print(n+" ");
        }

        System.out.println();
    }
}
```

Nombres amis

```
public class Amis
{
    public static void main (String [] args)
    {
        int n,s,i,somme;
```

```

for(n=1;n<=10000;n++)
{
    s=0;
    for(i=1;i<=n/2;i++)
        if(n%i==0)
            s=s+i;
    somme=0;
    for(i=1;i<=s/2;i++)
        if(s%i==0)
            somme=somme+i;
    if(somme==n)
        System.out.println(n+" est "+s+" sont amis");
}
}
}

```

Nombres d'Armstrong

```

public class Armstrong
{
    public static void main (String [] args)
    {
        int n,somme,unite,dizaine,centaine;
        for(n=2;n<=1000;n++)
        {
            unite=n%10;
            dizaine=((n-unite)/10)%10;
            centaine=n/100;
            somme=unite*unite*unite+dizaine*dizaine*dizaine+centaine*centaine*centaine;
            if(somme==n)
                System.out.print(n+" ");
        }
        System.out.println();
    }
}

```

Exercice 7

```

public class Urne
{
    public static void main (String [] args)
    {
        int n,i,rouge,noire,alea;
        System.out.println("Entrer le nombre de tirages");
        n=Lire.i();
        rouge=1;
        noire=1;
        for(i=1;i<=n;i++)
        {
            alea=(int)(1+(rouge+noire)*Math.random());
            if(alea<=rouge)

```

```

    rouge=rouge+1;
else
    noire=noire+2;
System.out.println("Rouge="+rouge+" et Noire="+noire);
}
}
}

```

11.2 Bilan n°2

Exercice 1

```

//NOMBRE EULER
public class nombre_euler
{
    public static void main (String [] args)
    {
        int n, i;
        double somme=0.0;
        System.out.print("donnez n");
        n=Lire.i();
        for(i=0;i<=n;i++)
            somme=somme+ 1.0/fact(i);
        System.out.println(somme);
    }

    public static int fact(int k)
    {
        if(k==0)
            return 1;
        else
            return k*fact(k-1);
    }
}

//NOMBRE EULER, 2eme SOLUTION

public class nombre_euler_bis
{
    public static void main (String [] args)
    {
        int n, i, denominateur=1;
        double somme=1.0;
        System.out.print("donnez n");
        n=Lire.i();
        for(i=1;i<=n;i++)
        {
            denominateur=denominateur*i;
            somme=somme+ 1.0/denominateur;
        }
        System.out.println(somme);
    }
}

```

Exercice 2

```
public class eratosthene
{   public static void main (String [] args)
    {   int n, i, j, t[], racine;
        System.out.print("donnez n : ");
        n=Lire.i();
        t=new int[n+1];
        t[1]=0;
        for(i=2;i<=n;i++)
            t[i]=i;
        racine=(int) Math.sqrt(n);
        i=2;
        while(i<=racine)
        {   if( t[i]!=0)
            {   j=2;
                while (i*j<=n)
                {   t[i*j]=0;
                    j++;
                }
            }
            i++;
        }
        for(i=1;i<=n;i++)
            if(t[i]!=0)
                System.out.print(t[i]+" ");
        System.out.println();
    }
}
```

Exercice 3

```
public static int A(int m,int n)
{   if(m==0)
    return n+1;
    else
    if(n==0)
    return A(m-1,1);
    else
    return A(m-1,A(m,n-1));
}
```

Exercice 4

```
Itératif
public class mot_fibo // Sans récursivité
{   public static void main (String [] args)
    {   int n;
        String mot="", mot1="1", mot2="0";
        System.out.print("donnez n : ");
    }
}
```

```

n=Lire.i();
for(i=3;i<=n;i++)
{ mot=mot1+mot2;
  mot1=mot2;
  mot2=mot;
}
if(n==1)
  mot=mot1;
if(n==2)
  mot=mot2;
System.out.println("Le mot de Fibonacci de taille "+n+" est "+ mot);
}

```

Récurusif

```

public class mot_fibo // Avec récursivité
{ public static void main (String [] args)
  { int n;
    String mot="";
    System.out.print("donnez n : ");
    n=Lire.i();
    System.out.println("Le mot de Fibonacci de taille "+n+" est "+
      Fibo(n));
  }

  public static String Fibo(int n)
  { if (n==0)
    return "0";
    else
    if (n==1)
    return "1";
    else
    return Fibo(n-1)+Fibo(n-2);
  }
}

```

Exercice 5

```

public class conjugaison
{ public static void main (String [] args)
  { String verbe="",racine="";
    int i, longueur;
    do
    { System.out.print("Donnez un verbe regulier du premier groupe : ");
      verbe=Lire.S();
      longueur=verbe.length();
    }
    while( verbe.charAt(longue-2)!='e' || verbe.charAt(longue-1)!='r' );
    for(i=0;i<=longue-3;i++)
      racine=racine+verbe.charAt(i);
  }
}

```

```

        System.out.println("je "+racine+"e");
        System.out.println("tu "+racine+"es");
        System.out.println("il/elle "+racine+"e");
        System.out.println("nous "+racine+"ons");
        System.out.println("vous "+racine+"ez");
        System.out.println("ils/elles "+racine+"ent");
    }
}

```

Exercice 6

```

// Remarques :
// (1) Le programme n'est pas tres efficace, vous etes invitez l'ameliorer
// (2) Dans cette version les majuscules sont 'plus petites' que les
    minuscules

public class tri_mots
{
    public static void main (String [] args)
    { int n,i,j;
      boolean trie;
      String aux="",liste [];
      System.out.print("donnez n:");
      n=Lire.i();
      liste= new String[n];
      for (i=0;i<n;i++)
      { System.out.print("donnez le "+i+"eme mot:");
        liste[i]=Lire.S();
      }
      for(i=1;i<n;i++)
      for(j=0;j<n-1;j++)
      if( liste[j].compareTo(liste[j+1])>0 )
      { aux= liste[j];
        liste[j]=liste[j+1];
        liste[j+1]=aux;
      }
      for (i=0;i<n;i++)
      System.out.println(liste[i]);
    }
}

```