Initiation à la programmation avec Java

Ce livre vous permettra de comprendre les fondements de la programmation et de l'algorithmique. Vous acquerrez des bases en programmation qui vous permettront d'aborder n'importe quel langage dans les meilleures conditions. Tous les aspects essentiels seront vus : les modèles de programmation, les éléments de lexique et de syntaxe, les outils, l'organisation du code, l'accès aux bases de données et les tests.

Introduction

Plan du cours

- Les fondements de la programmation

1. Qu'est-ce qu'un programme ? Qu'est-ce qu'un langage ? Les différents paradigmes. Quel langage pour quelle application ?
2. Les compilateurs. Les exécutables.
3. Les responsabilités d'un programmeur.
4. Qu'est-ce qu'un algorithme ?
5. Les besoins auxquels répond un algorithme.
6. Le concept de pseudo-langage.

Travaux pratiques Présentation de différents langages (Java, C#, Visual Basic, C, C++). Ecriture d'un premier algorithme en pseudo-langage. » Genèse d'un premier programme

- Ecriture d'un programme simple : syntaxe et instructions.

1. Compilation et exécution du programme.
2. Qu'est-ce qu'une librairie ? Son rôle, son usage.
3. Travaux pratiques
4. Découverte de l'environnement de développement et d'exécution. Ecriture, compilation et exécution d'un premier programme.

- Règles de programmation

1. Convention de nommage.
2. Convention syntaxique.
3. Utilisation des commentaires. Pourquoi commenter les développements ?
4. Améliorer la lisibilité des programmes : indentation du code, découpage du code...

- Les variables

1. Qu'est-ce qu'une variable ?
2. Pourquoi typer une variable ?
3. Les types primitifs : entiers, chaînes de caractères, nombres réels, autres.
4. Déclaration, définition et initialisation d'une variable.
5. Les constantes.
6. Saisie, affichage, affectation, conversion de type.
7. Organiser ses données sous forme de tableaux.
8. Les types évolués : enregistrement, matrice, arbre.

Travaux pratiques Ecriture de plusieurs programmes simples manipulant les variables.

- Opérateurs et expressions

1. Les différents opérateurs (multiplicatif, additif, comparaison, égalité, logique, affectation).
2. Combinaison d'opérateurs.
3. Expression booléenne.
4. Travaux pratiques
5. Manipulation des opérateurs et des expressions booléennes.

- Les structures de contrôle

1. Les sélections alternatives (si, si-alors-sinon, sélection cas).
2. Les blocs d'instructions (notion de Début... Fin).
3. Les boucles itératives (tant que-répéter, répéter-jusqu'à, pour-de- à).
4. Imbrication des instructions.
5. Les commentaires.

Travaux pratiques Utilisation des structures de contrôle pour implémenter un algorithme.

- Les procédures et les fonctions

1. Définitions : procédure, fonction.
2. Pourquoi sont-elles incontournables en programmation (réutilisabilité, lisibilité...) ?
3. Le passage de paramètres.
4. Le code retour d'une fonction.
5. Sensibilisation aux limites du passage de la valeur d'une variable.
6. Notion de passage par adresse.
7. Appel de fonctions.

- Introduction à la programmation objet

1. Les concepts associés à la programmation objet : classe, attribut, méthode, argument.
2. La modélisation objet à partir des exigences fonctionnelles.
3. Introduction aux bonnes pratiques d'organisation de conception et d'organisation d'un programme.

Travaux pratiques Illustration des concepts objets.

- L'accès aux bases de données

1. Organisation et stockage des données.
2. Les traitements de base (connexion, requêtes, récupération des données).
3. Application cliente et serveur de données.
4. Affichage et manipulation des données dans l'application cliente.

Travaux pratiques Création d'un formulaire de recherche d'informations dans une base de données.

- Maintenance, débogage et test des programmes

1. Savoir lire et interpréter les différents messages d'erreurs.
2. Utiliser un débogueur : exécuter un programme pas à pas, points d'arrêts, inspecter les variables pendant l'exécution.
3. Prévoir les tests unitaires.

Travaux pratiques Utilisation d'un débogueur pour contrôler l'exécution des programmes.

Slides

Ce cour est dispensé sous forme de slides:

inj_2017.pdf

Les variables

Exercices

Exercice 1 Les noms de variables

Quel est sont les nom de variables correctes:

- fonction-1
- _MOYENNE_du_MOIS_
- 3e_jour
- limite_inf.
- lim_supérieure
- __A_
- _
- a
- 3

Exercice 2 circulation de données

Ecrire un programme qui permute et affiche les valeurs de trois variables A, B, C de type entier qui sont entrées au clavier :
A ==> B , B ==> C , C ==> A

Correction:

```
public static void main(String [] args)
{
  String A=args[0];
  String B=args[1];
  String C=args[2];
  System.out.println(B);
  System.out.println(C);
  System.out.println(A);
}
```

Exercice 3: Les résistances

Soit trois resistance R1,R2 et R3 ecrivez la resistance totale en série et en parallèle.

```
public static void main(String[] args) {
    double R1=0;
    double R2=0;
    double R3=0;
    System.out.println("En série"+(R1+R2+R3));
    System.out.println("En parallère"+(R1*R2*R3)/(R1*R2+R1*R3+R2*R3));


}
```

Exercice 4 : La TVA

Ecrire un programme qui calcule le prix TTC (type double) d'un article à partir du prix net (type int) et du pourcentage de TVA (type int) à ajouter. Utilisez la formule suivante en faisant attention aux priorités et aux conversions automatiques de type:

Prix TTC=Prix Net+Prix net*TVA/100

```
public static void main(String [] args)
{
    int prixnet=4;
    int tva=20;
    double prixnetDouble=prixnet;
    double tvaDouble=tva;
    double prixTTCDouble=prixnetDouble+prixnetDouble*(tvaDouble/100);
    System.err.println("Prix TTC:"+prixTTCDouble);
}
```

Exercice 5: l'horloge

Écrivez un programme FormatHour qui prend en paramètre un nombre de secondes et qui permet le formater en heures-minutes-secondes.

Voici un exemple d'exécution du programme :

```
> java FormatHour 5208 minutes 40 secondes
> java FormatHour
> java FormatHour Hello
> java FormatHour 252177 heures 17 secondes
```

Comme vous pouvez le voir, si on ne fournit pas un paramètre lors de l'appel du programme ou si le paramètre fourni n'est pas un entier, le programme ne fait rien du tout.

```
public class FormatHour
{
    public static void main (String[] args)
    {
        if (args.length == 1)
        {
            try
            {
                // Calcul des valeurs
                int seconds = Integer.parseInt (args[0]);

                int hours = seconds / 3600;
                seconds = seconds % 3600;

                int minutes = seconds / 60;
                seconds = seconds % 60;

                // Affichage
                if (hours != 0)
                {
                    System.out.print (hours + " heures ");
                }
                if (minutes != 0)
                {
                    System.out.print (minutes + " minutes ");
                }
                if (seconds != 0)
                {
                    System.out.print (seconds + " secondes");
                }
                System.out.println();
            }
            catch (NumberFormatException exception){}
        }
    }
```

```
}
```

Exercice 6 La bibliotheque

Nous allons d'abord modéliser une personne qui est potentiellement un auteur de livre. Ce dernier a :

- un nom
- un prénom
- une année de naissance
- un booléen pour savoir si la personne est un auteurs ou pas

```
public class Person
{
public String nom;
public String prenom;
public int anneeNaissance;
public boolean autheurs;
}
```

Nous avons maintenant les livres. Les livres sont des structures ayant un numéro isbn (une chaine de caractère), un titre et peuvent être écrit par plusieurs auteurs (un tableau d'auteurs).

```
public class Book
{
public String title;
public String isbn;
public Person[] authors;

}
```

Exercice 7 Les départements

L'idée est de modéliser l'administration territorial Francaise qui est composé:

- De ville ayant un nom et un numéro INSEE (Chaine de caractère) unique
- Des communautés de commune qui sont des regroupements de ville ayant un nom.

- Des départements qui sont des regroupements de commune ayant un nom et un numéro (chaine de caractère unique).
- Les régions qui sont un regroupement de commune ayant un numéro ISO (chaine de caractère).
- Le pays qui est un regroupement de régions

Implémenter les opérateurs d'initialisation, equals et toString pour chacune de ces entités.

Les structures de contrôles

L'idée est de s'entrainer avec les structures de contrôles

New Page

Exercice sur les IfThenElse

Exercice IfThen

Considérez la séquence d'instructions suivante:

```
if (A>B) System.out.println ("premier choix \n"); else
if (A>10) System.out.println ("deuxième choix \n");
if (B<10) System.out.println ("troisième choix \n");
else System.out.println ("quatrième choix \n");
```

a) Copiez la séquence d'instructions en utilisant des tabulateurs pour marquer les blocs if - else appartenant ensemble.

b) Déterminez les réponses du programme pour chacun des couples de nombres suivants et vérifiez à l'aide de l'ordinateur.

- A=10 et B=5
- A=5 et B=5
- A=5 et B=10
- A=10 et B=10
- A=20 et B=10
- A=20 et B=20

Exercice IfThenElse

Considérez la séquence d'instructions suivante:

```
if (A>B)
if (A>10)
System.out.println ("premier choix \n"); else if (B<10)
System.out.println ("deuxième choix \n"); else
if (A==B) System.out.println ("troisième choix \n");
else System.out.println ("quatrième choix \n");
```

a) Copiez la séquence d'instructions en utilisant des tabulateurs pour marquer les blocs if - else appartenant ensemble.

b) Pour quelles valeurs de A et B obtient-on les résultats:
premier choix, deuxième choix, ... sur l'écran?

c) Pour quelles valeurs de A et B n'obtient-on pas de réponse sur l'écran?

d) Notez vos réponses et choisissez vous-mêmes des valeurs pour A et B pour les vérifier l'aide de l'ordinateur.

Exercice maximum

Ecrivez un programme qui lit trois valeurs entières (A, B et C) au clavier et qui affiche la plus grande des trois valeurs, en utilisant:

- if - else et une variable d'aide MAX
- les opérateurs conditionnels et une variable d'aide MAX

Exercice Tri

Ecrivez un programme qui lit trois valeurs entières (A, B et C) au clavier. Triez les valeurs A, B et C par échanges successifs de manière à obtenir :

val(A) val(B) val(C)

Affichez les trois valeurs.

Exercice signe

Ecrivez un programme qui lit deux valeurs entières (A et B) au clavier et qui affiche le signe du produit de A et B sans faire la multiplication.

Exercice abs

Ecrivez un programme qui lit deux valeurs entières (A et B) au clavier et qui affiche le signe de la somme de A et B sans faire l'addition. Utilisez la fonction abs de la classe Math
== Exercice second degrée==

Ecrivez un programme qui calcule les solutions réelles d'une équation du second degré ax2+bx+c = 0 en discutant la formule:
X1,X2= (-b +/- Math.sqrt(b*b-4ac)) /2a

Utilisez une variable d'aide D pour la valeur du discriminant b2-4ac et décidez à l'aide de D, si l'équation a une, deux ou aucune solution réelle. Utilisez des variables du type int pour A, B et C.

Considérez aussi les cas où l'utilisateur entre des valeurs nulles pour A; pour A et B; pour A, B et C. Affichez les résultats et les messages nécessaires sur l'écran.

Exercice sur les boucles et les tableaux

Boucle et tableau 1

Ecrire un programme qui lit la dimension N d'un tableau T du type int (dimension maximale: 50 composantes), remplit le tableau par des valeurs entrées au clavier et affiche le tableau.

Calculer et afficher ensuite la somme des éléments du tableau.

Boucle et tableau 2

Ecrire un programme qui lit la dimension N d'un tableau T du type int (dimension maximale: 50 composantes), remplit le tableau par des valeurs entrées au clavier et affiche le tableau.

Boucle et tableau 3

Effacer ensuite toutes les occurrences de la valeur 0 dans le tableau T et compter les éléments restants. Afficher le tableau résultant.

Boucle et tableau 4

Ecrire un programme qui lit la dimension N d'un tableau T du type int (dimension maximale: 50 composantes), remplit le tableau par des valeurs entrées au clavier et affiche le tableau.

Ranger ensuite les éléments du tableau T dans l'ordre inverse sans utiliser de tableau d'aide. Afficher le tableau résultant.

Idée: Echanger les éléments du tableau à l'aide de deux indices qui parcourent le tableau en commençant respectivement au début et à la fin du tableau et qui se rencontrent en son milieu.

Boucle et tableau 5

Ecrire un programme qui lit la dimension N d'un tableau T du type int (dimension maximale: 50 composantes), remplit le tableau par des valeurs entrées au clavier et affiche le tableau.

Copiez ensuite toutes les composantes strictement positives dans un deuxième tableau TPOS et toutes les valeurs strictement négatives dans un troisième tableau TNEG. Afficher les tableaux TPOS et TNEG.

Boucle et tableau 6

Ecrire un programme qui lit les dimensions L et C d'un tableau T à deux dimensions du type int (dimensions maximales: 50 lignes et 50 colonnes). Remplir le tableau par des valeurs entrées au clavier et afficher le tableau ainsi que la somme de tous ses éléments.

Boucle et tableau 7

Ecrire un programme qui lit les dimensions L et C d'un tableau T à deux dimensions du type int (dimensions maximales: 50 lignes et 50 colonnes). Remplir le tableau par des valeurs entrées au clavier et afficher le tableau ainsi que la somme de chaque ligne et de chaque colonne en n'utilisant qu'une variable d'aide pour la somme.

Boucle et tableau 8

Ecrire un programme qui transfère un tableau M à deux dimensions L et C (dimensions maximales: 10 lignes et 10 colonnes) dans un tableau V à une dimension L*C.

Exemple:

a	b	c

d	e	f

en

a	d
b	e
c	f

Maximum et minimum des valeurs d'un tableau

Ecrire un programme qui détermine la plus grande et la plus petite valeur dans un tableau d'entiers A. Afficher ensuite la valeur et la position du maximum et du minimum. Si le tableau contient plusieurs maxima ou minima, le programme retiendra la position du premier maximum ou minimum rencontré.

Insérer une valeur dans un tableau trié

Un tableau A de dimension N+1 contient N valeurs entières triées par ordre croissant; la (N+1)ième valeur est indéfinie. Insérer une valeur VAL donnée au clavier dans le tableau A de manière à obtenir un tableau de N+1 valeurs triées.

```
boolean hasBeenInserted = false;
    int[] resultat = new int[arr.length + 1];
    int index = 0;
    int toBeInserted = 6;
    for (int i = 0; i < arr.length; i++) {
        int valeur = arr[i];
        if (valeur >= toBeInserted && !hasBeenInserted) {

            resultat[index] = toBeInserted;
            index++;
            hasBeenInserted = true;
        }

        resultat[index] = valeur;
```

```
    index++;
  }
```

Exercice sur les boucles

Calcul de Factorielle

Essayer d'écrire le code permettant de calculer l'operation factorielle.
Cette opération prend un entier en parametre et :
* renvoie 1 si le parametre est vaut 0
* renvoie n*(n-1)*(n-2)...*1. Par exemple 5!=5*4*3*2*1

```
public class Fact {


public static void main(String [] args)
{
String param=Utils.readLine();
int nb=Utils.convertStringToInt(param);
int factorielle=1;
for (int i = 2; i <= nb; i++)
{
factorielle *= i;
}
System.out.println(factorielle);
}
}
```

Dessin

Ecrire une methode static prennant en parametre :
* nbTiret un nombre
* nbEspace un nombre
* nbMotif un nombre

L'idée étant d'afficher au plus nb caractère composé de nbTiret de fois le caractère - et nbEspace de fois le caractere espace.
Par exemple pour nbMotif=3, nbTiret=3 et nbEspace=2 on auras la chaine suivante <--- --- --- >

```
public static void drawLine (int nm, int nt, int nb)
{
for (int i = 0; i < nm; i++)
{
// Tirets
for (int j = 0; j < nt; j++)
{
System.out.print ("-");
}
// Espaces
for (int j = 0; j < nb; j++)
{
System.out.print (" ");
}
}
System.out.println();
}
```

Palindrome

Enoncée

Ecrire une methode static prennant en parametre :
* str une chaine de caractère

L'idée étant d'afficher vrai si la chaine de caractère est un palindrome. Un palindrome étant une chaine de caractère se lisant de la même façon du début à la fin que de la fin vers le début.
Par exemple radar est un palindrom.
Dans l'exercice suivant :

- les espaces sont ignorée 'rad ar' est un palindrom
- la fonction replace, lenght et charAt de la classe String peuvent être utilisé

La méthode conseillé est d'utilisé la méthode donnée ci joint qui élimine les espaces d'une chaine de caractère

Solution

```
public static boolean isPalindrome (String s)
{
s = s.replace(" ","");
for (int i = 0; i < s.length() / 2; i++)
{
if (s.charAt (i) != s.charAt (s.length() - 1 - i))
{
return false;
}
}
return true;
}
```

Enoncée

Maintenant l'idée est d'écrire une methode renvoyant le miroir d'une chaine de caractère. Par exemple le mot 'caractère' a travers cette méthode doit renvoyer 'erètcarac'

```
public static String reverse (String str)
{
String s = "";
for (int i = str.length() - 1; i >=0; i--)
{
s += str.charAt (i);
}
return s;
}
```

Enoncée

Écrivez une méthode de signature

public static String merge (String s, String t)

qui permet de fusionner les deux chaines de caractères s et t. Fusionner deux chaines de caractères s1s2s3... et t1t2t3... consiste à construire une nouvelle chaine de caractères qui sera construite en prenant une lettre à la fois dans chacune des chaines. La fusion sera la chaine s1t1s2t2s3t3...

Faites bien attention que les deux chaines n'ont pas forcément les mêmes longueurs. Par exemple, l'appel

merge ("Hello", "Bonjour");

renvoie la chaine de caractères HBeolnljoour.

On utiliseras la méthode static max de la classe Math qui renvoie le maximum de deux entier.

Solution

```
public static String merge (String s, String t)
{
String ret = "";
int max = Math.max (s.length(), t.length());
for (int i = 0; i < max; i++)
{
if (i < s.length()) ret += s.charAt (i);
if (i < t.length()) ret += t.charAt (i);
}
return ret;
}
```

PairImpair

Enoncée

Ecrire une methode static prennant en paramètre un tableau d'entier et imprimant le nombre d'entier impair du tableau.
On utiliseras la propriété que x % 2 vaut zero si x est pair

Solution

```
public static int nbOfOddValues (int[] tab)
{
int cnt = 0;
for (int i = 0; i < tab.length; i++)
{
if (tab[i] % 2 != 0)
{
cnt++;
}
}
return cnt;
}
```

Livret A

Enoncée

Le but est d'écrire une méthode prennant en paramètre :

- une somme d'argent
- un taux de livret A
- un nombre indiquant le nombre d'année de placement

Cette méthode doit calculer la somme d'argent placé au taux sus cité l'argent placé en début d'année:

Solution

```
public static double computeLivretA (double money,double interet, int nbyear)
{
```

```
for (int i=0;i<nbyear;i++) money+=money*interet/100;
return money;
}
```

Enoncée

Le but est d'écrire une méthode renvoyant le nombre d'année qu'il faut pour que de l'argent placé atteigne montant demandé.
Par exemple pour passer de 100€ à 1000€ avec un taux à 10% la méthode doit renvoyer 7.

Solution

```
public static int livretA(double money, double target, double interest)
{
double balance = 0;
int years = 0;

while (balance < target)
{
balance += money; // début année, j'ajoute argent sur le compte
balance += balance * interest / 100; // fin année, j'ajoute intérêts
years++; // une année écoulée
}

return years;
}
```

Pierre Feuille Ciseaux

Enoncée

Dans ce jeux, l'utilisateur choisie un objet parmis une feuille, une pierre et un cieaux.
L'ordinateur fait deux meme.
Le résultat est que la feuille gagne sur le cailloux, qui gagne sur le ciceaux, qui gagne sur la feuille.
Pour reussir, il faut prendre la methode Math.random pour avoir un nombre aléatoire.

Solution

```
public class PierreFeuilleCiseaux{
public static void main (String args[]) {
String arme[] = {"la pierre", "les ciseaux", "la feuille"};
String message[] = { arme[0] + " casse " + arme[1], arme[2] + " recouvre " + arme[0], arme[1]
+ " coupent "+ arme[2] };
int resultat[] = {0, 2, 1};
// affichage menu
for (int i=0;i<3;i++){
System.out.println(i + ". " + arme[i]);
}
// choix du joueur 1 : l'humain
int j1 = -1;
do{
j1 = Utils.readInt(); // ascii

}while(j1 < 0 || j1 > 2);
int j2 = (int)(Math.random()*3); // choix du joueur 2 : la machine
System.out.println("J'ai choisi " + arme[j2]);
if ( j1 == j2 ) { System.out.println("On a choisi pareil ! " );System.exit(0); }
int i = j1 + j2 - 1;
System.out.println(message[i]);
if (j1 == resultat[i]) {
System.out.println("Tu gagnes" );
} else {
System.out.println("Je gagne" );
}
}
}
```

Les images

L'idée est de coder des filtres sur les images (vecteur composé de pixel rouge, vert, bleue et transparence).
Nous allons commencer par une superposition d'image (addition de deux images):
La spécification de la fonction est static Image add(Image original,Image original2, float alpha1,float alpha2);
et l'image de destination est rougeDestination=alpha1*rougeOriginal1+alpha2*rougeOriginal2 (et ceci pour toutes les couleurs)
En sus, sont prévue 3 filtres:
- un passage en niveau de gris d'une image (moyenne des 3 couleurs)
- un passage en niveau de gris de meilleurs qualité (0.21 * rouge+ 0.71 * vert+ 0.07 * bleu)

- un filtrage gaussien (application de la matrice 121).
Pour faire cela, nous allons utiliser 2 classes:

```
public class Pixel {
public int alpha, red, green, blue;
public static Pixel createPixel(int alpha,int red,int green,int blue)
{
Pixel p=new Pixel();
p.alpha=alpha;
p.red=red;
p.green=green;
p.blue=blue;
return p;
}
}
```

et

```
import java.awt.Color;
import java.awt.image.BufferedImage;
import java.io.File;
import java.io.IOException;

import javax.imageio.ImageIO;

public class Image {

public int width;
public int height;
private BufferedImage data;

public static Image createImage(Image myimage)
{
Image result= new Image();
result.height=myimage.height;
result.width=myimage.width;
result.data=new BufferedImage(myimage.data.getWidth(), myimage.data.getHeight(),
myimage.data.getType());
return result;
}
```

```
public static Pixel getPixel(Image myimage,int i,int j)
{
Pixel pixel=new Pixel();
pixel.alpha = new Color(myimage.data.getRGB(i, j)).getAlpha();
pixel.red = new Color(myimage.data.getRGB(i, j)).getRed();
pixel.  green = new Color(myimage.data.getRGB(i, j)).getGreen();
pixel.blue = new Color(myimage.data.getRGB(i, j)).getBlue();
return pixel;
}
private static int colorToRGB(Pixel p) {

int newPixel = 0;
newPixel += p.alpha;
newPixel = newPixel << 8;
newPixel += p.red; newPixel = newPixel << 8;
newPixel += p.green; newPixel = newPixel << 8;
newPixel += p.blue;

return newPixel;

}
public static void setPixel(Image myImage, int i, int j, Pixel p)
{
int newPixel = colorToRGB(p);

// Write pixels into image
myImage.data.setRGB(i, j, newPixel);

}
public static void saveImage(Image myimage,String filename)
{
try {

File imageFile = new File(filename);
ImageIO.write(myimage.data, "jpg", imageFile);
} catch (IOException e) {
// TODO Auto-generated catch block
e.printStackTrace();
}
//"jpg" is the format of the image
```

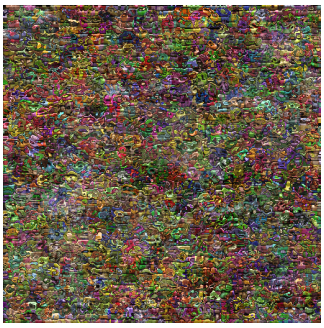
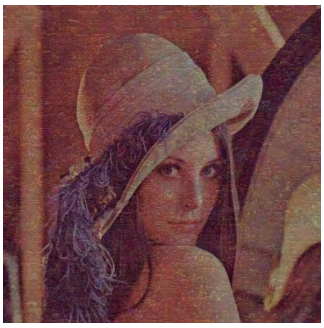
```
//imageFile is the file to be written to.



}
public static Image readImage(String fileName)
{
Image result=new Image();
try {
result.data = ImageIO.read(new File(fileName));
} catch (IOException e) {
throw new RuntimeException(e);
}
result.width = result.data.getWidth();
result.height = result.data.getHeight();

return result;
}
}
```

Résultats des exercices:

	Image de base	Image de composition	Résultat
Addition d'image			
Floue			

Solution

```
public class Main {
private static Image avg(Image original) {

int alpha, red, green, blue;
Pixel newPixel;

Image avg_gray =Image.createImage(original);

for(int i=0; i<original.width; i++) {
for(int j=0; j<original.height; j++) {

Pixel p=Image.getPixel(original, i, j);

// Get pixels by R, G, B
alpha = p.alpha;
red = p.red;
green = p.green;
blue = p.blue;

int newColor = (red + green + blue)/3;

// Return back to original format
newPixel = Pixel.createPixel(newColor, newColor, newColor, newColor);

// Write pixels into image
Image.setPixel(avg_gray,i, j, newPixel);

}
}

return avg_gray;

}
private static Image avglum(Image original) {

int alpha, red, green, blue;
Pixel newPixel;

Image avg_gray =Image.createImage(original);
```

```
for(int i=0; i<original.width; i++) {
for(int j=0; j<original.height; j++) {

Pixel p=Image.getPixel(original, i, j);

// Get pixels by R, G, B
alpha = p.alpha;
red = p.red;
green = p.green;
blue = p.blue;

int newColor = (int) (0.21 * red + 0.71 * green + 0.07 * blue);

// Return back to original format
newPixel = Pixel.createPixel(newColor, newColor, newColor, newColor);

// Write pixels into image
Image.setPixel(avg_gray,i, j, newPixel);

}
}

return avg_gray;

}
private static Image add(Image original,Image original2, float alpha1,float alpha2) {

int alpha, red, green, blue;
Pixel newPixel;

Image avg_gray =Image.createImage(original);

for(int i=0; i<original.width; i++) {
for(int j=0; j<original.height; j++) {

Pixel p1=Image.getPixel(original, i, j);
Pixel p2=Image.getPixel(original2, i, j);

// Get pixels by R, G, B
alpha = 1;
```

```
red = (int) (p1.red*alpha1+p2.red*alpha2);
green = (int) (p1.green*alpha1+p2.green*alpha2);
blue = (int) (p1.blue*alpha1+p2.blue*alpha2);

// Return back to original format
newPixel = Pixel.createPixel(alpha, red, green, blue);
// Write pixels into image
Image.setPixel(avg_gray,i, j, newPixel);

}
}

return avg_gray;

}
private static Image gauss(Image original) {

int alpha, red, green, blue;
Pixel newPixel;

Image avg_gray =Image.createImage(original);

for(int i=1; i<original.width-1; i++) {
for(int j=1; j<original.height-1; j++) {

Pixel p1=Image.getPixel(original, i-1, j-1);
Pixel p2=Image.getPixel(original, i-1, j);
Pixel p3=Image.getPixel(original, i-1, j+1);
Pixel p4=Image.getPixel(original, i, j-1);
Pixel p5=Image.getPixel(original, i, j);
Pixel p6=Image.getPixel(original, i, j+1);
Pixel p7=Image.getPixel(original, i+1, j-1);
Pixel p8=Image.getPixel(original, i+1, j);
Pixel p9=Image.getPixel(original, i+1, j+1);

// Get pixels by R, G, B
alpha =
(p1.alpha+p2.alpha+p3.alpha+p4.alpha+2*p5.alpha+p6.alpha+p7.alpha+p8.alpha+p9.alpha)/10;
red = (p1.red+p2.red+p3.red+p4.red+2*p5.red+p6.red+p7.red+p8.red+p9.red)/10;
green =
```

```
(p1.green+p2.green+p3.green+p4.green+2*p5.green+p6.green+p7.green+p8.green+p9.green)/10;
blue = (p1.blue+p2.blue+p3.blue+p4.blue+2*p5.blue+p6.blue+p7.blue+p8.blue+p9.blue)/10;

// Return back to original format
newPixel = Pixel.createPixel(alpha, red, green, blue);

// Write pixels into image
Image.setPixel(avg_gray,i, j, newPixel);

}
}

return avg_gray;

}
public static void main(String [] args)
{
Image i=Image.readImage("C:\\PGM\\Scic\\project\\testimage\\lena30.jpg");
Image avgimage=avglum(i);
Image.saveImage(avgimage, "C:\\PGM\\Scic\\project\\testimage\\lena30avglum.jpg");
Image avgimage2=avg(i);
Image.saveImage(avgimage2, "C:\\PGM\\Scic\\project\\testimage\\lena30avg.jpg");
Image gauss=gauss(i);
Image.saveImage(gauss, "C:\\PGM\\Scic\\project\\testimage\\lena30gauss.jpg");
Image texture=Image.readImage("C:\\PGM\\Scic\\project\\testimage\\texture.png");
Image addImage=add(i,texture,0.5f,0.2f);
Image.saveImage(addImage, "C:\\PGM\\Scic\\project\\testimage\\addImage.jpg");

}
}
```

Les fonctions

Valeur absolue

Ecrire une méthode qui étant donné un nombre réel, renvoie sa valeur absolue

```
public static double valeurAbsolue(double a)
    {
         if( a > 0)
         {
              return a;
         }
         else
         {
              return -a;
        }
   }
```

recherche

Ecrire une méthode qui, étant donné un tableau de nombres entiers et un nombre entier quelconque, teste la présence de ce nombre dans ce tableau.

Lorsque vous verrez un exercice sur les méthodes avec les mots teste, ou vérifie, ou permet de savoir si, sachez que vous avez affaire une méthode de type booléen. Mais attention, ne vous concentrez pas sur ces mot ou expressions pour dire qu' une méthode est de type booléenne. Il s' agit surtout de réfléchir un peu en se posant la question suivante. L' énoncé de l' exercice pose-t-elle une question à laquelle il faut forcément répondre par vrai ou par faux ? Si c' est le cas, alors, la méthode est de type booléenne.

Ici, l' énoncé permet de se poser la question suivante : Ce nombre entier est-il dans le tableau de nombres entiers ? Réponse = vrai ou faux. Donc, méthode de type booléenne. Et le type booléen en java utilise le mot clé boolean.

```
    public static boolean nombreDansTableau(int a, int [] tab)
     {
```

```
          for(int i = 0; i < tab.length; i++)
          {
                if(tab[i] == a)
                {
                      return true;
                }
          }
          return false;
    }
```

Écrire les fonctions suivantes.

- Une fonction qui renvoie la plus grande de deux valeurs de type int.
- Une fonction qui répète un même mot un certain nombre de fois auchoix.
- Une fonction, construite à partir de la fonction Math.random, quitire au sort un nombre entier entre deux bornes données enarguments.
- Une fonction qui décide s'il est possible de construire un triangle avectrois segments de mesures données.

Maximum de deux valeurs

```
int maximum(int n1, int n2)
{
if (n1>n2)
return n1;
else
return n2;
}
```

Fonction qui repete un certains nombres de mots

```
String repetition_mot(String mot, int nb_repetitions)
{
String mot_repete = " ";
for (int i=1;i<=nb_repetitions;i++)
{
mot_repete = mot_repete + mot;
}
return mot_repete;
}
```

Fonction qui génère un nombre entre deux bornes

```
int entier_aleatoire(int inf, int sup)
{
return (int) (Math.random()*(sup – inf) + inf);
}
```

Fonction est ce un triangle

```
boolean triangle_possible(double c1, double c2, double c3)
{
if (c1 > c2 + c3) return false;
if (c2 > c1 + c3) return false;
if (c3 > c1 + c2) return false;
return true;
}
```

Porté d'une variable

Déterminer la portée de chaque variable dans le programme suivant. L’utilisation qui est faite de ces variables est-elle cohérente avec cette portée ? Si non, comment corriger ce programme ?

```
nt z, y;
void v (double x)
{
double u;
u = x * x;
z = (int) x;
}
void main ()
{
double t;
y = 4;
t = 1 / (double) y;
v(t);
println(u);
}
```

Correction

```
int z, y;
static double v (double x)
{
double u;
u = x * x;
z = (int)x;
return u;
}
void main ()
{
double t;
y = 4;
t = 1 / (double)y;
println(v(t));
}
```

Moyenne

Ecrire un programme se servant d'une fonction MOYENNE du type float pour afficher la moyenne arithmétique de deux nombres réels entrés au clavier.