LABORATOIRE 3 | *Informatique 1*
TESTS ET BOUCLES

Hes·so VALAIS WALLIS
Haute Ecole d'Ingénierie
Hochschule für Ingenieurwissenschaften

But du laboratoire (2 heures)

1. Le but de ce laboratoire et de se familiariser avec les modifications du flux séquentiel d'un programme, en utilisant des conditions et des boucles. Nous introduirons aussi la classe *FunGraphics* qui est une librairie de dessin permettant de dessiner dans une fenêtre graphique.
2. La durée de labo estimée est de **deux périodes**.
3. Vous trouverez ce laboratoire sur http://inf1.begincoding.net. Vous y trouverez également la solution dès la semaine prochaine.

Partie 1 - Exemples simples de tests et de boucles

Tâche 1 (boucle)

1. Créez un projet *Lab3* comme vu dans le labo précédent.
2. Ajoutez une classe *Task1* à votre projet et écrivez le programme ci-dessous (qui se trouve également dans la série 3):

```
public class Task1 {
     public static void main(String[] args) {
         int x = 0;
         for (int i = 0; i < 10; x++) {
             System.out.println("Number : " + x);
         }
     }
}
```

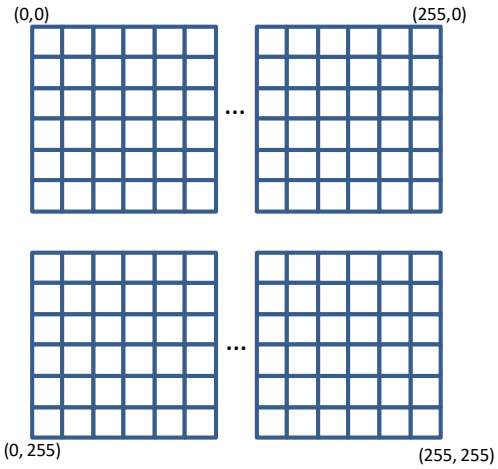
3. Exécutez votre programme.
4. Que se passe-t-il ? Expliquez

Tâche 2 (If et switch)

1. Télécharger le fichier *lab-exp.zip* depuis le site et extrayez les fichiers « .java » dans le répertoire *src* de votre projet. Suivez la procédure d'installation pour la libraire *FunGraphics* disponible sur le site du cours (voir l'onglet *FunGraphics*).
2. Ouvrez le programme `SimpleCalculator.java` et exécutez-le.
3. Comprenez son fonctionnement.
4. Pour l'instant, il ne comporte que 3 opérations : l'addition, la soustraction et la multiplication. Modifiez le code source pour rajouter l'opération de division.
5. Remplacez les tests *if* par un opérateur *switch*.
6. (Facultatif) Rajoutez maintenant une nouvelle option dans votre programme, qui va afficher les deux nombres en binaire. Le principe est de lire les bits du chiffre un après l'autre et de les afficher, grâce à une boucle. Pour ne lire que le premier bit, il faut faire un masque qui ne conserve que le premier bit (calculer un AND avec le nombre binaire 1). Ensuite, il faut décaler le résultat tout à droite du bit, et ensuite on peut afficher ce nombre comme un *int* (vu qu'il ne reste que le dernier bit qui n'est peut-être pas nul, on affiche 0 ou 1).
7. Rajouter une boucle qui englobe tout le programme, afin de répéter toutes les opérations.
8. Rajouter maintenant une condition qui permette de sortir de la boucle quand on le désire. Pour cela, rajoutez un nouvel opérateur dans la liste des opérateurs, nommé *quit* et qui, lorsqu'il est sélectionné, quitte la boucle (quelle instruction utilisez-vous ?).

Partie 2 - Exemples simples utilisant la classe *FunGraphics*

La classe *FunGraphics* se charge de vous fournir une surface sur laquelle vous pouvez dessiner librement. Vous trouverez cette librairie sur le site web du cours avec les installations pour l'installer.



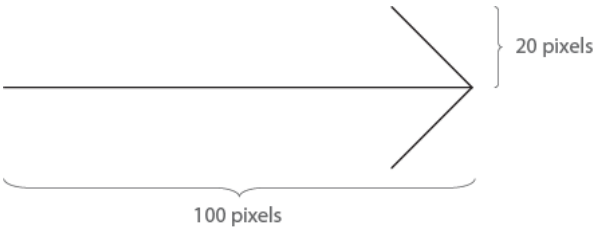
L'ouverture d'une fenêtre de taille *width x height* se réalise à l'aide du code suivant (dont nous verrons la signification la semaine prochaine) :

```
FunGraphics display = new FunGraphics(width, height);
```

On peut ensuite dessiner à l'aide des méthodes `display.setPixel(int x, int y)` qui font apparaître un point à la position (x, y). Attention, la position (0,0) se trouve en haut à gauche de l'écran.

Tâche 3 (Dessin d'une flèche)

1. Ouvrez le fichier *DrawArrow.java*.
2. Exécutez le programme. Il va créer une fenêtre graphique et dessiner un pixel noir en position [200,200].
3. Codez maintenant un programme qui va dessiner une flèche horizontale qui pointe à droite, avec une longueur de 100 pixels et une tête de 20 pixels, comme dans le dessin ci-dessous. Il vous faudra utiliser plusieurs boucles et utiliser la méthode *setPixel()*.
    a. Dessinez d'abord le trait horizontal du corps de la flèche avec une boucle.
    b. Dans un deuxième temps, dessinez chaque branche de la flèche à l'aide d'une boucle.

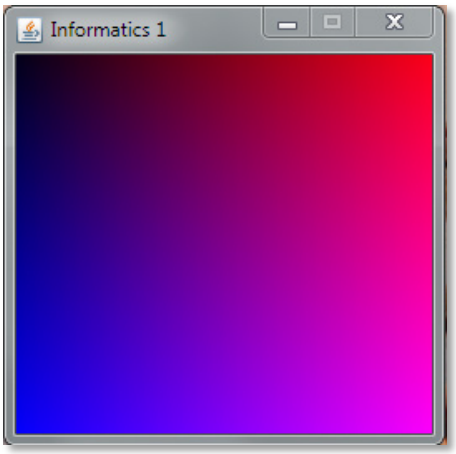


Tâche 4 (Dessin d'un rectangle)

1. Créez une nouvelle classe *Task4* basée sur le modèle ci-dessus avec une interface graphique
2. Écrivez maintenant un code qui va dessiner un rectangle plein bleu, centré en position [200,200], d'une largeur de 300 pixels et d'une hauteur de 200 pixels. L'idée est d'utiliser deux boucles imbriquées qui vont balayer les pixels correspondants au rectangle et qui vont les peindre avec la bonne couleur.
3. Pour peindre le pixel en position [x,y] en bleu, vous pouvez utiliser la méthode suivante :

```
display.setPixel(x, y, Color.blue);
```

Tâche 5 (Dégradé rouge-bleu)



1. Faites une nouvelle classe avec une interface graphique.
2. La fenêtre graphique créée doit faire 256 pixels par 256 pixels.
3. Le but de cet exercice est de dessiner une palette d'un dégradé de couleurs (bleu et rouge). Pour ce faire, il faut utiliser une première boucle pour faire le dégradé horizontal de rouge et une seconde pour faire le dégradé vertical.
4. Si un pixel est à la position [x,y], on va le peindre avec une valeur de bleu de x et une valeur de rouge de y. La valeur verte du pixel restant toujours à zéro.
5. Pour peindre le pixel en position [x,y] d'une certaine couleur, on utilise :

```
display.setPixel(x, y, new Color(RED, GREEN, BLUE));
```

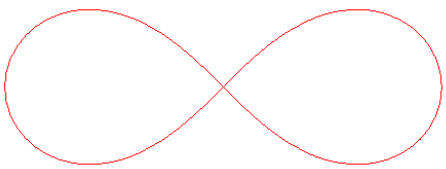
Où *RED*, *GREEN* et *BLUE*, sont des valeurs entières entre 0 et 255.La valeur 0 signifie qu'un composant de la couleur est absente, tandis que 255 signifie que la couleur est à son intensité maximale.

Tâche 6 (*Facultatif* – Dessin d'un disque)

1. Écrivez maintenant un code qui va créer une fenêtre graphique et y dessiner un disque bleu rempli, centré en position [200,200] et d'un rayon de 100 pixels, sur fond rouge.
2. Une possibilité pour résoudre ce problème et de créer deux boucles qui vont balayer tous les pixels de l'image, les peindre en bleu s'ils sont dans le disque (remplissent la condition de distance par rapport au centre du cercle) et en rouge sinon.

Tâche 7 (*Facultatif* – Plus difficile)

1. Le lemniscate de Bernoulli est une courbe mathématique dont le tracé est le suivant :



2. Dessinez cette courbe où x et y sont donnés par l'équation paramétrique suivante (*a* est une constante) :

$$\begin{cases} x(t) = a\,\dfrac{\sin t}{1+\cos^2 t} \\ y(t) = a\,\dfrac{\sin t \cos t}{1+\cos^2 t} \end{cases}$$