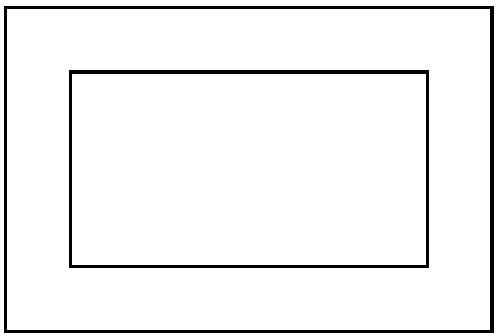
Quelques programmes en mode graphique

1. Tracer 3 cercles concentriques sur une toile de taille 700×700
2. Tracer deux rectangles emboités :
   - le premier de taille 500×300
   - le deuxième de taille 400×200
3. Tracer deux cercles à l'intérieur d'un rectangle. Les deux cercles doivent se toucher
4. Utiliser l'instruction `arc(....)` pour tracer un demi-cercle.

Quelques programmes en mode graphique

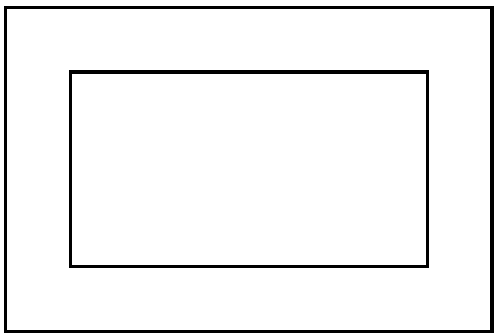
1. Tracer 3 cercles concentriques sur une toile de taille 700×700
2. Tracer deux rectangles emboités :
   - le premier de taille 500×300
   - le deuxième de taille 400×200



3. Tracer deux cercles à l'intérieur d'un rectangle. Les deux cercles doivent se toucher
4. Utiliser l'instruction `arc(....)` pour tracer un demi-cercle.

Quelques programmes en mode graphique

1. Tracer 3 cercles concentriques sur une toile de taille 700×700
2. Tracer deux rectangles emboités :
   - le premier de taille 500×300
   - le deuxième de taille 400×200



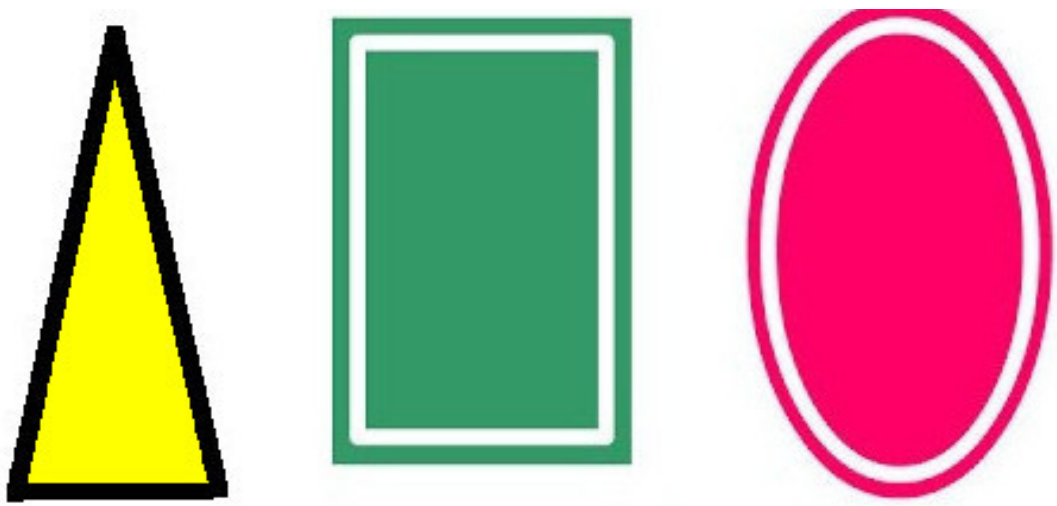
3. Tracer deux cercles à l'intérieur d'un rectangle. Les deux cercles doivent se toucher
4. Utiliser l'instruction `arc(....)` pour tracer un demi-cercle.

Des dessins en couleurs

- La couleur du dessin suivant est réglé par l'instruction `stroke(rouge,vert,bleu)` où `rouge`, `vert` et `bleu` representent des nombres de 0 à 255
- Utiliser `fill(rouge,vert,bleu)` pour indiquer la couleur de remplissage de la figure (ellipse, rectangle, triangle)
- Tester aussi `noStroke();` et `strokeWeight(...);`
- Réaliser un par un les dessins ci-dessous

Des dessins en couleurs

- La couleur du dessin suivant est réglé par l'instruction `stroke(rouge,vert,bleu)` où `rouge`, `vert` et `bleu` representent des nombres de 0 à 255
- Utiliser `fill(rouge,vert,bleu)` pour indiquer la couleur de remplissage de la figure (ellipse, rectangle, triangle)
- Tester aussi `noStroke();` et `strokeWeight(...);`
- Réaliser un par un les dessins ci-dessous



La boucle `pour` ou `for`

Objectif : tracer plusieurs cercles à des positions aléatoires

- Rappel : `random(x)` renvoie une valeur aleatoire de 0 à `x`
- Syntaxe boucle : `for(int i=1;i<=100;i++){...}`

1. Ajouter de la couleur : le premier cercle doit être rouge et ensuite la couleur doit glisser graduellement vers bleu
2. Mélanger les figures : d'abord un cercle, ensuite un rectangle, ensuite un cercle, rectangle, cercle, rectangle, ...
3. Écrire un nouveau programme pour réaliser un dégradé

La boucle `pour` ou `for`

Objectif : tracer plusieurs cercles à des positions aléatoires

- Rappel : `random(x)` renvoie une valeur aleatoire de 0 à `x`
- Syntaxe boucle : `for(int i=1;i<=100;i++){...}`

1. Ajouter de la couleur : le premier cercle doit être rouge et ensuite la couleur doit glisser graduellement vers bleu
2. Mélanger les figures : d'abord un cercle, ensuite un rectangle, ensuite un cercle, rectangle, cercle, rectangle, ...
3. Écrire un nouveau programme pour réaliser un dégradé

La boucle `pour` ou `for`

Objectif : tracer plusieurs cercles à des positions aléatoires

- Rappel : `random(x)` renvoie une valeur aleatoire de 0 à `x`
- Syntaxe boucle : `for(int i=1;i<=100;i++){...}`

1. Ajouter de la couleur : le premier cercle doit être rouge et ensuite la couleur doit glisser graduellement vers bleu
2. Mélanger les figures : d'abord un cercle, ensuite un rectangle, ensuite un cercle, rectangle, cercle, rectangle, ...
3. Écrire un nouveau programme pour réaliser un dégradé

Plan

1. Les premiers programmes/dessins en langage `Processing`

2. Comprendre l'ordinateur : notions d'architecture
   - Les couches et les fonctions de base d'un ordinateur
   - Mémoires et Processeur
   - Du langage `processing` à la couche matérielle

Comprendre la machine informatique

Pour comprendre le fonctionnement des ordinateurs, on va étudier :

1. architecture des ordinateurs
2. systèmes d'exploitation (prochain cours)
3. réseaux

- Le terme « machine » ou «ordinateur » est à prendre dans son sens le plus large : « machine électronique capable d'exécuter des opérations arithmétiques et logiques »
- Il peut désigner aussi bien un ordinateur de bureau ou portable (PC, Mac), un serveur de calcul ou encore un terminal mobile de type tablette ou smartphone.

1. Les premiers programmes/dessins en langage `Processing`

2. Comprendre l'ordinateur : notions d'architecture
   - Les couches et les fonctions de base d'un ordinateur
   - Mémoires et Processeur
   - Du langage `processing` à la couche matérielle

Machine informatique : couches génériques

Couche logicielle

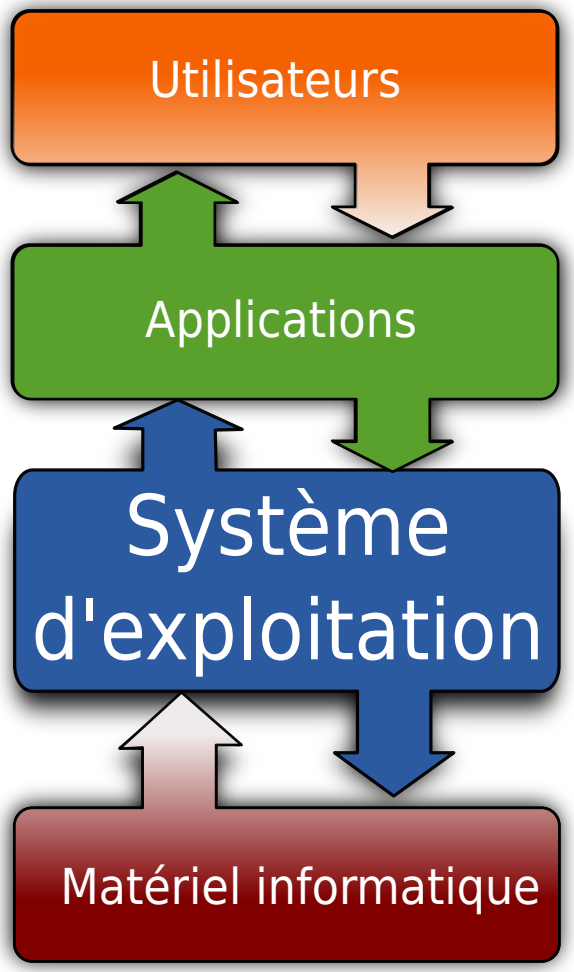
des programmes qui permettent à l'utilisateur de réaliser des taches, y compris `processing` et `java`

Couche Système d'Exploitation

dirige l'utilisation des ressources de la machine par les programmes de la couche logicielle

Couche Matérielle

la machine physique y compris le processeur (CPU), la mémoire vive (RAM), disques durs, clés USB, imprimantes, etc.



Fonctions du système d'exploitation (OS)

Le système d'exploitation : aussi appelé OS, de l'anglais Operating System.

OS = Interface entre le matériel et les logiciels

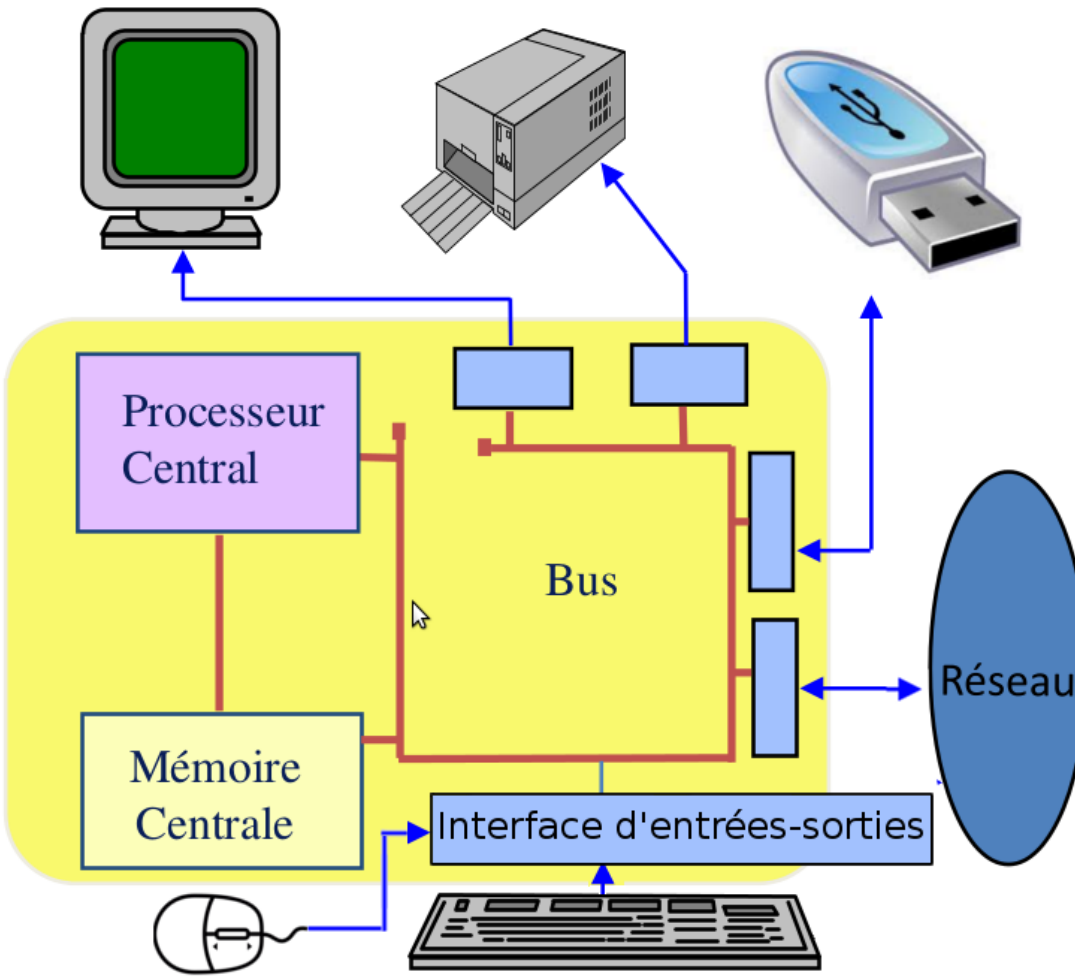
Ressources matérielles : Processeur, mémoire vive (RAM), fichiers, réseaux, interface graphique utilisateur, périphériques, disques durs, contrôle d'accès pour plusieurs usagers simultanément, etc.

Principaux OS

- Linux/Unix ses distributions (Ubuntu, Suze, Debian)
- Windows 95, Windows Vista, Windows 7, etc.
- MacOS, Android (basés sur des noyaux Linux)

Composants couche physique (hardware)

- le processeur exécute les instructions machine, c'est le cerveau du système
- les mémoires vives (RAM, cache) stockent les données et les instructions
- le bus permet le transfert de données entre les différents composants
  - souvent implémenté sur la carte mère.



- les périphériques : disques durs, clés USB, imprimantes, moniteur (écran), clavier, souris, cartes d'extension (graphique), manettes de jeu, lecteurs de CD/DVD, etc.