

TP 6 VARI 1

Informations techniques PC Suse :

- Pour démarrer une session : utilisateur **licencep** et mot de passe **7002n***. Vous trouverez :
 - un gestionnaire de fichiers en haut à gauche placé dans le dossier personnel HOME
 - une icône lézard  en haut à droite pour accéder au menu.
- Pour démarrer *Processing* : clic sur l'icône lézard en haut à droite → Développement → Processing.
- Pour démarrer une *console* : clic sur l'icône lézard en haut à droite → Terminal → Konsole.
- Pour ouvrir un gestionnaire/navigateur de fichiers : clic sur l'icône lézard → Utilitaires → Dolphin.
- Pour lancer une commande : Alt + Space
- Pour modifier un fichier, clic droit sur le fichier → Ouvrir avec → Kate (ou autre éditeur de votre choix).

1 Commandes Linux et Réseaux

Démarrer une console/terminal en suivant les instructions au point (c) ci-dessus.

Exercice 1 Taper la commande suivante dans la console.

```
ip addr show
```

Elle affiche les adresses IP des interfaces réseau disponibles. Identifier l'adresse IP du réseau filaire, voir l'exemple encadré dans l'image ci-après.

```
licencep@samar39:~> ip addr show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid lft forever preferred lft forever
2: em1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast
    link/ether 5c:f9:dd:e7:2a:36 brd ff:ff:ff:ff:ff:ff
    inet 163.173.230.169/22 brd 163.173.231.255 scope global em1
        valid lft forever preferred lft forever
3: lxcbr1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue
    link/ether 16:bb:14:f0:e1:50 brd ff:ff:ff:ff:ff:ff
```

Exercice 2 Taper la commande suivante

```
ip route show
```

Elle affiche votre passerelle par défaut, voir première ligne. Vérifier si vous trouvez la même passerelle via la commande `/sbin/route -n`

2 Fonctions

Exercice 1 Écrire une méthode (fonction qui renvoie void) appelée `triangleRouge(int x0,int y0,int x1, int y1, int x2, int y2)` pour tracer un triangle à l'aide de 3 appels à la fonction `line(...)`. Les sommets sont indiqués par les coordonnées (x_0, y_0) , (x_1, y_1) , (x_2, y_2) . Remplir le code ci-après pour le faire fonctionner.

```
void triangleRouge (int x0,int y0,int x1, int y1, int x2, int y2){
    ....
}
void setup(){
    size(500,500);
    triangleRouge(10,10,100,50,50,100);
    triangleRouge(10,10,400,100,100,400);
}
```

Exercice 3 Taper la commande ci-après pour télécharger le fichier source `Toto.java`.

```
wget cedric.cnam.fr/~porumbed/vari1/Toto.java
```

Regarder le contenu du fichier `Toto.java` à l'aide de la commande `cat`, ou avec un éditeur comme `Kate`.

Pour compiler le programme Java, taper :

```
javac Toto.java
```

Et pour l'exécuter :

```
java Toto
```

Exercice 4 Donner une commande qui permet de trouver l'adresse IP de la machine `www.cnam.fr`. Indications : utiliser `ping`, `nslookup` ou `dig`.

Exercice 5 Taper la commande ci-après pour calculer 2^3 .
`irb<<<"2**3"`

Modifier la commande pour calculer $(2 + 7)^3 * (5 - 3)$. Vérifier le calcul avec une calculette (`kcalc`). Quelle méthode de calcul est la plus rapide ?

Exercice 2 Écrire une méthode `triangleRougeTab(int[] x, int[] y)` avec le même objectif que la méthode précédente, mais avec des arguments de type tableau. Remplir le code ci-après pour le faire fonctionner.

```
... triangleRougeTab(int [] x, int [] y) {
    ....
    //ajouter trois appels line(...)
    ...
}
void setup() {
    size(500,500);
    int [] a = new int [3];
    int [] b = new int [3];
    a[0] = 10 ; b[0]= 10;
    a[1] = 400; b[1]= 200;
    a[2] = 50 ; b[2]= 400;
    triangleRougeTab(a,b);
    triangleRougeTab(b,a);
}
```

Exercice 3 Écrire une fonction `perimetreTriangle(int[] x, int[] y)` qui renvoie le périmètre d'un triangle. Il faut déterminer les longueurs des trois côtés. Pour cela, vous pouvez utiliser le théorème de Pythagore, par exemple :
`float len01= sqrt((x[1] - x[0])2 + (y[1] - y[0])2).`

Exercice 4 Écrire une méthode `pentaTab(int[] x, int[] y)` (donc avec deux tableaux comme arguments) pour tracer un pentagone. Il faut généraliser `triangleRougeTab` pour relier le point $(x[0], y[0])$ au point $(x[1], y[1])$, ensuite relier ce dernier à $(x[2], y[2])$ et répéter cela avec $(x[3], y[3])$ et $(x[4], y[4])$. Remplir le programme ci-après pour le faire afficher un pentagone.

```
... pentaTab(int [] x, int [] y) {
    ....
}
void setup() {
    ...
    pentaTab(a, b);
}
```

3 Boucles

Exercice 1 Utiliser une boucle `for` pour afficher 15 fois le texte `Salut tout le monde!!!`.

Exercice 2 Écrire une boucle `for` pour tracer 35 rectangles de taille 200×100 placés à des positions aléatoires. **Indication** : `rect(random(100), random(100), 50, 20)` permet de tracer un rectangle de taille 50×20 à une position aléatoire avec les deux coordonnées entre 0 et 100, voir le programme ci-après.

```
void setup() {
    rect(random(100), random(100), 50, 20)
}
```

Exercice 3 Écrire une boucle `for` pour déterminer la valeur minimale d'un tableau de 7 entiers. Modifier le programme ci-après pour le faire afficher la valeur minimale du tableau, c.-à.-d. 1.

```
void setup() {
    int [] a = {9, 5, 1, 4, 3, 12, 9};
    int min = a[0];
    //calculer min
    println(min);
}
```

Exercice 4 Modifier la méthode `pentaTab(int[] x, int[] y)` pour afficher le pentagone à l'aide d'une boucle `for`. On suppose que les tableaux `x` et `y` ont une longueur de 5.

Exercice 5 Écrire une méthode `polygone(int[] x, int[] y)` qui généralise la méthode `pentaTab(...)` de l'exercice précédent. Elle devrait tracer un polygone de taille arbitraire. Initialiser dans la méthode `setup()` des tableaux avec des valeurs aléatoires (via, ex., `(int)random(...)`) et appeler `polygone(...)`. **Indication** : vous allez avoir besoin de déterminer la longueur des tableaux `x` et `y` et vous pouvez utiliser `x.length`.