
TP 10 noté

- (a) Pour démarrer une session : utilisateur **licencep** et mot de passe **7002n****.
- (b) Pour démarrer *Processing* : clic sur la tête de coméléon en haut à droite → Développement → Processing.
- (c) La page VARI1 : **cedric.cnam.fr/~porumbed/vari1/**
- (d) Pour ouvrir un gestionnaire/navigateur de fichiers : clic sur la tête de caméléon → Système → Dolphin.
-

Exercice 1 (4 points) Soit le code ci-après qui a pour objectif de calculer la note minimale. Corriger les deux erreurs aux lignes indiquées en commentaire. Sauvegarder le programme corrigé dans un fichier **Exo1.pde**

```
float note1, note2, note3, note4;
note1=random(20);
note2=random(20);
note3=random(20);
note4=random(20);
int min=note1;
if(min>note2);           //il y une erreur sur l'exécution
    min=note2;           //de ce if
if(min>note3)
    note3=min;           //erreur de logique algorithmique ici
if(min>note4)
    min = note4;
printf("la note minimale est :"+min);
```

Exercice 2 (4 points) Copier le programme précédent dans un fichier **Exo2.pde**; modifiez ce nouveau fichier **Exo2.pde** pour le faire utiliser un tableau de 4 cases à la places des variables **note1**, **note2**, **note3**, **note4**.

Exercice 3 (2 points) Écrire dans un fichier **Exo3.pde** une fonction **detMaxVal(float[] tab)** qui prend comme argument un tableau de **float** et qui renvoie la valeur **maximale**. **Rappel** : le nombre de cases dans le tableau est **tab.length**. L'en-tête (ou la signature) de la fonction est :

```
float detMaxTab(float[] tab)
```

Attention : si n est inférieur à zéro, il faut renvoyer -1 et afficher « calcul impossible ».

Exercice 4 (1 point) Écrire dans un fichier **Exo4.pde** une fonction **detMaxValUnique(float[] tab)** qui prend comme argument un tableau de **float** et qui renvoie la valeur maximale unique. C'est à dire, la valeur renvoyée doit apparaître une seule fois dans le tableau. On suppose que toutes les valeurs sont positives; s'il n'y a pas une seule valeur unique, il faut renvoyer -1. Exemples :

- il faut renvoyer 9 pour **tab={3, 4, 9, 11, 8, 11}**
- il faut renvoyer 11 pour **tab={3, 4, 9, 11, 8, 9}**
- il faut renvoyer 3 pour **tab={3, 4, 4, 4, 8, 8}**
- il faut renvoyer -1 pour **tab={4, 4, 4, 4, 8, 8}**

Exercice 5 (2 points) Écrire un programme **Exo5.pde** qui affiche 9 cercles de rayon aléatoire $r < 100$, placés à des positions aléatoires sur une toile de taille 800×800 . Chaque cercle doit rester complètement à l'intérieur de la toile.

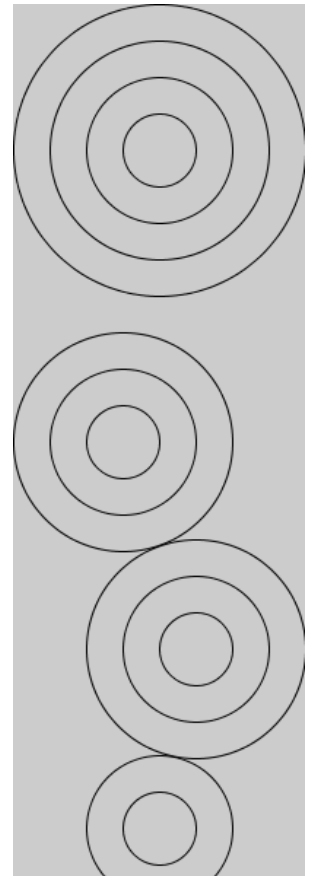
Exercice 6 (1 point) Continuer l'exercice précédent en ajoutant la contrainte suivante : les cercles ne doivent pas s'intersecter. On ne doit pas trouver un seul pixel couvert par deux cercles.

Exercice 7 (2 points) On considère un tableau `tab`, pre-rempli avec des valeurs entières entre 1 et 10. Le programme à droite génère un entier aléatoire `randInt`. Continuer le programme pour le faire afficher tous les indices où se trouve `randInt`. Par exemple, si le programme génère `randInt=9`, il devrait afficher “9 se trouve à l’indice 0” et “9 se trouve à l’indice 4”.

```
void setup() {
    int [] tab = {9, 3, 5, 6, 9};
    int randInt = (int)random(10);
    ....
    //à remplir
}
```

Exercice 8 (2 points) Écrire une méthode `dessinerCercles(float x, float y, int nbCercles)` qui permet de tracer `nbCercles` de centre (x,y) . Le plus petit cercle a le rayon 25, le deuxième le rayon 50, le troisième 75, ensuite 100, 125, etc. Faire fonctionner le programme ci-après pour obtenir la figure à droite.

```
.... dessinerCercles (...) {
    noFill(); //on indique que les cercles doivent être vides
    ....
}
void setup() {
    size(200,600);
    dessinerCercles(100,100,4);
    dessinerCercles(75,300,3);
    dessinerCercles(125,442,3);
    dessinerCercles(100,565,2);
}
```



Exercice 9 (3 points) Écrire dans un fichier `Exo9.pde` une fonction d’en-tête `float calculerIMC(float taille, float poids)` pour calculer l’Indice de Masse Corporelle (IMC) en fonction de la taille et du poids. La formule de calcul est $IMC = \frac{\text{poids}}{\text{taille}^2}$, voir des exemples à https://fr.wikipedia.org/wiki/Indice_de_masse_corporelle. Faire fonctionner le programme ci-après pour afficher “maigreur” si $IMC < 18.5$, “corpulence normale” si $18.5 \leq IMC \leq 25$, “surpoids” si $25 < IMC < 30$ ou “obésité” si $IMC \geq 30$.

```
----- calculerIMC (.....) {
    .....
}
void setup() {
    float taille_m = 1.5+random(0.4); //aléatoire entre 1.5m et 1.9m
    float poids_kg = 50+random(70); //aléatoire entre 50kg et 120kg
    float imc = calculerIMC(taille_m, poids_kg);
    println(" Si_poids="+poids_kg+"kg_et_taille=" +taille_m+" l'IMC_vaut_"+imc);
    if(imc<18.5){
        println(" maigreur");
    }else{
        //à remplir
    }
}
```

Déposer vos fichiers dans une archive ZIP en utilisant l’URL :
cedric.cnam.fr/~porumbed/vari1/tp/