

# Éléments de programmation : tableaux, notions de C et Java

Valeur d'accueil et de reconversion en informatique (VARI1)

Daniel Porumbel (dp.cnam@gmail.com)

<http://cedric.cnam.fr/~porumbed/vari1/>

Écrire un programme **Processing** :

- A** déclarer trois variables `note1`, `note2`, `note3` de type `int`
- B** initialiser les trois notes à des valeurs aléatoires
  - afficher “succès” si toutes les notes sont supérieures à 10,
  - sinon afficher “échec”

il peut y avoir plusieurs solutions...

Penser au passage à l'échelle, à un programme facile à modifier si on ajoute plus notes.

- 1 Traduire le programme en C (presque copier-coller !)
- 2 Utiliser un **tableau de notes**
  - C : `int notes[3];`
  - Processing : `int[] notes = new int[3];`
- 3 Saisir les notes au clavier sous C

Écrire un programme **Processing** :

- A** déclarer trois variables `note1`, `note2`, `note3` de type `int`
- B** initialiser les trois notes à des valeurs aléatoires
  - afficher “succès” si toutes les notes sont supérieures à 10,
  - sinon afficher “échec”

Il peut y avoir plusieurs solutions...

Penser au passage à l'échelle, à un programme facile à modifier si on ajoute plus notes.

1 Traduire le programme en C (presque copier-coller !)

2 Utiliser un tableau de notes

```
C : int notes[3];
```

```
Processing : int[] notes = new int[3];
```

3 Saisir les notes au clavier sous C

Écrire un programme **Processing** :

- A** déclarer trois variables `note1`, `note2`, `note3` de type `int`
- B** initialiser les trois notes à des valeurs aléatoires
  - afficher “succès” si toutes les notes sont supérieures à 10,
  - sinon afficher “échec”

Il peut y avoir plusieurs solutions...

Penser au passage à l'échelle, à un programme facile à modifier si on ajoute plus notes.

- 1** Traduire le programme en C (presque copier-coller !)
- 2** Utiliser un **tableau de notes**

```
C : int notes[3];  
Processing : int[] notes = new int[3];
```
- 3** Saisir les notes au clavier sous C

Écrire un programme **Processing** :

- A** déclarer trois variables `note1`, `note2`, `note3` de type `int`
- B** initialiser les trois notes à des valeurs aléatoires
  - afficher “succès” si toutes les notes sont supérieures à 10,
  - sinon afficher “échec”

Il peut y avoir plusieurs solutions...

Penser au passage à l'échelle, à un programme facile à modifier si on ajoute plus notes.

- 1** Traduire le programme en C (presque copier-coller !)
- 2** Utiliser un **tableau de notes**

```
C : int notes[3];
```

```
Processing : int[] notes = new int[3];
```

- 3** Saisir les notes au clavier sous C

Écrire un programme **Processing** :

- A** déclarer trois variables `note1`, `note2`, `note3` de type `int`
- B** initialiser les trois notes à des valeurs aléatoires
  - afficher “succès” si toutes les notes sont supérieures à 10,
  - sinon afficher “échec”

Il peut y avoir plusieurs solutions...

Penser au passage à l'échelle, à un programme facile à modifier si on ajoute plus notes.

- 1** Traduire le programme en C (presque copier-coller !)
- 2** Utiliser un **tableau de notes**

```
C : int notes[3];
```

```
Processing : int[] notes = new int[3];
```

- 3** Saisir les notes au clavier sous C

# Solution Processing classique

```
int note1=9;
int note2=19;
int note3=14;
int min=note1;
if (min>note2)
    min=note2;
if (min>note3)
    min=note3;
if (min<=10)
    println ("échec");
else
    println ("succès");
```

# Solution Processing avec tableau

```
int [] note = new int [3];
note [0]=9;
note [1]=19;
note [2]=14;
int min=note [0];
if (min>note [1])
    min=note [1];
if (min>note [2])
    min=note [2];
if (min<=10)
    println ("échec");
else
    println ("succès");
```



# Solution C classique (sans tableau)

```
void main() {  
    int note1=6;  
    int note2=9;  
    int note3=15;  
    int min=note1;  
    if (min>note2)  
        min=note2;  
    if (min>note3)  
        min=note3;  
    if (min<=10)  
        printf("échec");  
    else  
        printf("succès");  
}
```

fichier test.c  
(extension .c, pas .cpp)

à compiler (en terminal) avec `gcc -w test.c -o exec`

# Version C avec tableau

```
void main() {  
    int note[3];  
    note[0]=9;  
    note[1]=19;  
    note[2]=14;  
    int min=note[0];  
    if (min>note[1])  
        min=note[1];  
    if (min>note[2])  
        min=note[2];  
    if (min<=10)  
        printf("échec");  
    else  
        printf("succès");  
}
```

# Version C avec lecture clavier

```
void main(){
    int note[3];
    char saisie[10];           //chaîne de caractères
    gets(saisie);             //fonction gets: lecture chaîne au clavier
    note[0] = atoi(saisie);   //fonction atoi: conversion vers entier
    gets(saisie);
    note[1] = atoi(saisie);
    gets(saisie);
    note[2] = atoi(saisie);
    int min=note[0];
    if (min>note[1])
        min=note[1];
    if (min>note[2])
        min=note[2];
    if (min<=10)
        printf("échec");
    else
        printf("succès");
}
```

On doit l'exécuter dans la console/terminal :

Se placer avec `cd ...` dans le même dossier que le programme

compiler avec `gcc -w nomDuProg.c -o exec`

lancer avec `./exec`

# Un programme Processing

```
float exam = 10.2; ←  
float tp1=2, tp2=10, tp3=2, tp4=15;  
float min = tp1;  
if (tp2 < min) min = tp2;  
if (tp3 < min) min = tp3;  
if (tp4 < min) min = tp4;  
float tp = (tp1 + tp2 + tp3 + tp4 - min) / 3;  
float noteFinale = exam * 6 / 10 + tp * 4 / 10;  
if ( (exam >= 7) && (noteFinale > 10) )  
    println ( "réussi" );  
else  
    println ( "échoué" );
```

déclaration et  
initialisation de  
variable réelle

Ce programme résout un des exercices du TP 5, rappelé par la suite. Soit les variables *exam*, *tp1*, *tp2*, *tp3* et *tp4* initialisées au début du programme pour indiquer une note d'examen et resp. 4 notes de TP. Une note d'examen inférieure à 7 est éliminatoire. La plus petite note des quatre notes de TP est ignorée. La note finale est calculée ainsi : l'examen compte pour 60% et la moyenne des trois notes de TP restantes pour 40%. Exemple : si *exam* = 10, *tp1* = 5, *tp2* = *tp3* = *tp4* = 12, on obtient  $10 \cdot 0.6 + 12 \cdot 0.4 = 10.8$ . Si la note finale est supérieure à 10, afficher "réussi", sinon afficher "échoué".

# Le même programme en C

```
void main() { ←  
    float exam = 10;  
    float tp1=2, tp2=10, tp3=2, tp4=15;  
    float min = tp1;  
    if (tp2 < min) min=tp2;  
    if (tp3 < min) min=tp3;  
    if (tp4 < min) min=tp4;  
    float tp=(tp1+tp2+tp3+tp4-min)/3;  
    float noteFinale=exam*6/10+tp*4/10;  
    if ( (exam>=7)&&(noteFinale>10))  
        printf("réussi");  
    else  
        printf("échoué");  
}
```

On définit la fonction `main()`  
qui est exécutée en premier

Le corps de la fonction `main` est presque identique au programme Processing précédent !

# Le même programme en Java

```
class Notes {
    public static void main (String [] args) {
        float exam = 10;
        float tp1=2, tp2=10, tp3=3, tp4=15;
        float min = tp1;
        if (tp2 < min) min=tp2;
        if (tp3 < min) min=tp3;
        if (tp4 < min) min=tp4;
        float tp=(tp1+tp2+tp3+tp4-min)/3;
        float noteFinale=exam*6/10+tp*4/10;
        if ( (exam>=7)&&(noteFinale>10))
            System.out.println ("réussi");
        else
            System.out.println ("échoué");
    }
}
```

# Tableaux

- Un tableau est une séquence d'éléments (variables) qui occupent des cases mémoire contiguës (un bloc sans trou)
- Sous C/C++/Java/Processing les éléments doivent avoir le même type, ex., que des `int`, ou que `char` (caractères)
  - Pas vrai en `python` ou `ruby`
- on accède à un élément du tableau à l'aide de son indice (premier élément : indice 0)

Un tableau avec 6 éléments

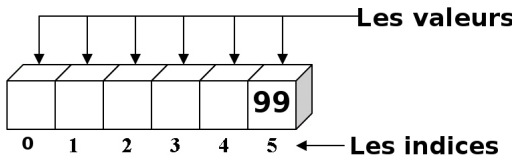
```
int[] tab;  
tab= new int[6];  
tab[5] = 99;
```

# Tableaux

- Un tableau est une séquence d'éléments (variables) qui occupent des cases mémoire contiguës (un bloc sans trou)
- Sous C/C++/Java/Processing les éléments doivent avoir le même type, ex., que des `int`, ou que `char` (caractères)
  - Pas vrai en `python` ou `ruby`
- on accède à un élément du tableau à l'aide de son indice (premier élément : indice 0)

Un tableau avec 6 éléments

```
int [] tab;  
tab= new int [6];  
tab[5] = 99;
```





# Des tableaux sous Processing

Rappel déclaration tableau de 3 variables/cases :

```
int [] tab = new int [3];
```

- 1 Écrire un programme pour calculer le minimum du tableau, ainsi que son indice dans le tableau
  - Exemple : si `tab={4, 7, 5}`, alors `min=4` et **indice=0**, c-à-d, `min` se trouve à la première case (indice 0)
  - Exemple : si `tab={6, 5, 8}`, alors `min=5` et **indice=1**, c-à-d, `min` se trouve à la 2ème case (indice 1)
- 2 Continuer le programme du point 1 pour trier le tableau
- 3 Continuer le programme précédent pour calculer la valeur médiane (centrale), ex., la valeur médiane de 5, 3,9 est 5
- 4 Continuer le programme du point 1 (oublions les tris) pour calculer la valeur médiane
- 5 Continuer le programme précédent pour afficher l'indice de la valeur médiane dans le tableau de départ.

# Des tableaux sous Processing

Rappel déclaration tableau de 3 variables/cases :

```
int [] tab = new int [3];
```

- 1 Écrire un programme pour calculer le minimum du tableau, ainsi que son indice dans le tableau
- 2 Continuer le programme du point 1 pour trier le tableau
- 3 Continuer le programme précédent pour calculer la valeur médiane (centrale), ex., la valeur médiane de 5, 3,9 est 5
- 4 Continuer le programme du point 1 (oublions les tris) pour calculer la valeur médiane
- 5 Continuer le programme précédent pour afficher l'indice de la valeur médiane dans le tableau de départ.

# Des tableaux sous Processing

Rappel déclaration tableau de 3 variables/cases :

```
int [] tab = new int [3];
```

- 1 Écrire un programme pour calculer le minimum du tableau, ainsi que son indice dans le tableau
- 2 Continuer le programme du point 1 pour trier le tableau
- 3 Continuer le programme précédent pour calculer la valeur médiane (centrale), ex., la valeur médiane de 5, 3,9 est 5
- 4 Continuer le programme du point 1 (oublions les tris) pour calculer la valeur médiane
- 5 Continuer le programme précédent pour afficher l'indice de la valeur médiane dans le tableau de départ.

# Des tableaux sous Processing

Rappel déclaration tableau de 3 variables/cases :

```
int [] tab = new int [3];
```

- 1 Écrire un programme pour calculer le minimum du tableau, ainsi que son indice dans le tableau
- 2 Continuer le programme du point 1 pour trier le tableau
- 3 Continuer le programme précédent pour calculer la valeur médiane (centrale), ex., la valeur médiane de 5, 3,9 est 5
- 4 Continuer le programme du point 1 (oublions les tris) pour calculer la valeur médiane
- 5 Continuer le programme précédent pour afficher l'indice de la valeur médiane dans le tableau de départ.

# Des tableaux sous Processing

Rappel déclaration tableau de 3 variables/cases :

```
int [] tab = new int [3];
```

- 1 Écrire un programme pour calculer le minimum du tableau, ainsi que son indice dans le tableau
- 2 Continuer le programme du point 1 pour trier le tableau
- 3 Continuer le programme précédent pour calculer la valeur médiane (centrale), ex., la valeur médiane de 5, 3,9 est 5
- 4 Continuer le programme du point 1 (oublions les tris) pour calculer la valeur médiane
- 5 Continuer le programme précédent pour afficher l'indice de la valeur médiane dans le tableau de départ.