Processing Cheat sheet (JAVA)zz

Convention de nommage :

Les noms des variables doivent respecter les conventions suivantes (d'après SUN):

1. ***1ère lettre en minuscule***
2. ***Mélange de minuscule, majuscule avec la première lettre de chaque mot en majuscule***
3. ***Donner des noms simples et descriptifs***
4. *Ne pas commencer les noms avec '$' ou '_' bien que ce soit possible.*
5. ***Variable d'une seule lettre (pour un usage local)***
    - ***int** : i, j, k, m, et n*
    - ***char** : c, d, et e*
    - ***boolean** : b*

De plus, d'une manière générale :

- ***N'utiliser que les lettres [a-z] et [A-Z] et [0-9]** : Ne pas utiliser de tiret '-', d'underscore '_', ou d'autres caractères ($, *, accents, ...).*

(Voir : http://www.loribel.com/java/normes/nommage.html)

Déclaration de variables

```
// Déclaration d'une variable de type int
// La valeur initiale n'est pas définie
int nombreColisDepart;

// Dans ce cas la valeur initiale est 9
int capaciteCamion = 9;

// Déclaration d'une chaine de caractères
String hello = "Hello World";

// Définition d'une variable de type caractère
char c = 'c';
```

Assigner des variables

```
// Assigner une valeur à une variable existante
nombreColisDepart = 34;

// Incrémenter une variable
```

```
colisDansCamion++;
// alternative :
colisDansCamion += 1;
colisDansCamion += 2; // Ajouter 2 à la variable colisDansCamion

// Décrémenter une variable
nombreColisDepart--;
```

Tests

```
// Égalité de 2 variables
colisDansCamion == capaciteDuCamion

// Non Égalité
colisDansCamion != capaciteDuCamion

// Inférieur à
colisDansCamion < capaciteDuCamion

// Supérieur ou égale à
colisDansCamion >= capaciteDuCamion

// ET logique
(colisDansCamion < capaciteDuCamion) && (nombreColisDepart > 0)

// Ou logique
(colisDansCamion < capaciteDuCamion) || (nombreColisDepart > 0)

// Négation
!((colisDansCamion < capaciteDuCamion) || (nombreColisDepart > 0))
```

Conditions

```
// Si le camion n'est pas plein
if(colisDansCamion < capaciteDuCamion){
  // Alors ....
}

// Si le nombre de clois dans le camion est égale à 0
if( colisDansCamion == 0 ){
  // Alors
  println("Le camion est vide");
} else {
  //Sinon
  println("Le camion n'est pas vide");
```

```
}

// Si le nombre de clois dans le camion est égale à 0
if( colisDansCamion == 0 ){
  // Alors
  println("Le camion est vide");
} else if(colisDansCamion < capaciteDuCamion){
  //Sinon
  println("Le camion n'est pas encore plein");
} else {
  //Sinon
  println("Le camion est plein");
}
```

Boucles

```
// Boucle faire tant que
while((colisDansCamion < capaciteDuCamion) && (nombreColisDepart > 0)){
  /*
  Ce block est répété tant que la condition ((colisDansCamion <
capaciteDuCamion) && (nombreColisDepart > 0)) est vraie
  */
}

// Boucle pour
for(int i=0; i<10 ; i++){
  /*
  Ce block est répété 10 fois
  */
}
```

Tableaux

```
// Déclare un tableau de 6 entier
// Le contenu du tableau n'est pas défini
int[] qty = new int[6];

// Déclare un table de 3 entier
// Le contenue est définit
int[] espaceDisponible = { 5 , 8 ,9 };
```

```
// /!\ l'indice des tableaux commence à 0

// Modifier la première valeur d'un tableau
espaceDisponible[0] = 5;

// /!\ La dimension d'un tableau est fixe
espaceDisponible[5] = 5; // Cause une erreure car on accède à un élément
qui n'existe pas
```

Fonctions et procédures

```
// Fonction
boolean camionEstVide(){
  return (colisDansCamion == 0);
}

// Procédure (Une procédure ne retourne pas de résultat)
void viderLeCamion(){
  while(!camionEstVide()){
      dechargerUnColis();
  }
}

// Appel d'un procédure / fonction sans paramètres
viderLeCamion();

// Pramètres d'une fonction/procédure
int square(int valeur){
  return valeur * valeur;
}

// Appel d'une procédure / fonction avec paramètres
int resultat = square(5);
```

Scope d’une variable

```
void maFonction(){
  // On déclare la variable i
  // Cette variable a été déclarée dans la fonction "maFonction"
  // Sa portée est donc limité à cette fonction
  int i = 0;
  /*
```

```
  ....
  ....
  */
  // La variable i n'existe plus à partir de cette ligne
}
i = 3; // Error : la variable i n'existe pas
```