

**TD 3 : Structures de contrôles : instructions conditionnelles, boucles simples**

Dans les exercices suivants, vous préciserez le nom, l'entrée et la sortie de vos programmes. Par exemple, pour un programme calculant le maximum de deux nombres, vous écrirez :

```
// Programme maximum
// Entrée: deux entiers a et b de type int
// Sortie: le maximum de a et de b, stocké dans la variable max
int max;
if ( a >= b ) {
    max = a;
} else {
    max = b;
}
```

Si vous vous sentez à l'aise, mettez votre programme sous la forme d'une fonction :

```
int maximum(int a, int b) {
    if ( a >= b ) {
        return a;
    } else {
        return b;
    }
}
```

**INSTRUCTIONS CONDITIONNELLES : IF****Exercice 1.**

- (1) Écrire un programme qui prend en entrée l'âge d'une personne (un entier donc) et dont la sortie est `true` si elle est majeure et `false` sinon.
- (2) Modifier votre programme pour que la la sortie soit une chaîne de caractères (type `string`) contenant le tarif SNCF appliqué à cette personne : "`enfant`" (moins de 11 ans), "`jeune`" (12-27 ans), "`senior`" (60 ans ou plus) ou "`plein tarif`" sinon.

**Exercice 2 (À faire chez vous).**

- (1) Le magasin Honeydukes est ouvert de 10h à 12h et de 14h à 19h. Écrire un programme qui prend en entrée l'heure sous forme d'un entier et dont la sortie est `true` si le magasin est ouvert et `false` sinon.
- (2) Si ce n'est pas déjà le cas, modifier votre programme pour qu'il y ait au plus un `if`.
- (3) Si ce n'est pas déjà le cas, modifier votre programme pour qu'il n'y ait pas de `if`.  
Indication : remarquer que l'expression booléenne qui sert de condition au `if` a déjà la valeur recherchée (voir aussi le cours : « Erreurs classiques avec les conditionnelles »).

**INSTRUCTIONS ITÉRATIVES : BOUCLE WHILE****Exercice 3 (Programme mystère).**

- (1) Exécuter pas à pas le fragment de programme suivant pour  $a = 3$  et  $b = 11$  :

```
while ( b >= a ) {
    b = b - a;
}
```

- (2) Pour  $a$  et  $b$  entiers positifs quelconques, quelle est la valeur de  $b$  après la boucle ?

**Exercice 4** (logarithme entier en base 2).

- (1) Écrire un programme dont l'entrée est un entier positif  $n$  et qui calcule le plus petit  $i$  de la forme  $2^k$  tel que  $n \leq i$ .  
Indication : partir de  $i=1$  et le multiplier par 2 tant que nécessaire.
- (2) Modifier votre programme pour que la sortie soit  $k$  tel que  $i = 2^k$  (où  $i$  est l'entier défini à la question précédente).

INSTRUCTIONS ITÉRATIVES AVEC COMPTEUR : BOUCLES **WHILE** ET **FOR****Exercice 5.**

On rappelle les exemples de programmes vus en cours pour afficher les nombres de 1 à 10 avec respectivement une boucle **while** et une boucle **for** :

```
int i = 1;
while ( i <= 10 ) {
    cout << i << endl;    // Affiche la valeur de i
    i = i + 1;
}
```

```
for ( int i = 1; i <= 10; i = i + 1 ) {
    cout << i << endl;    // Affiche la valeur de i
}
```

- (1) Dans chacun de ces deux programmes, entourer la déclaration du compteur, l'initialisation, la condition et l'incrémentation.
- (2) Adapter le premier exemple pour afficher les nombres pairs inférieurs ou égaux à 10, en commençant par 0. De même avec le deuxième exemple.
- (3) Même chose pour afficher les nombres de 10 à 1 ;
- (4) Même chose pour afficher les nombres entiers de carré inférieur à 10 ;
- (5) Même chose pour calculer la valeur de la somme  $1^2 + 2^2 + \dots + 10^2$ .

À votre avis, dans chacun des cas ci-dessus, laquelle des deux formes est la plus naturelle ?

## POUR ALLER PLUS LOIN

**Exercice ♣ 6** (Nombres premiers).

- (1) Écrire une fonction qui prend en argument (entrée) un entier  $n$  et teste si  $n$  est un nombre premier (c'est-à-dire renvoie **true** si  $n$  est premier et **false** sinon).
- (2) Écrire une fonction qui prend en argument un entier  $n$  et affiche tous les nombres premiers entre 1 et  $n$ .
- (3) Écrire une fonction qui affiche les  $n$  premiers nombres premiers.

**Exercice ♣ 7** (Dates).

- (1) Écrire une fonction qui prend en entrée une date sous la forme de trois entiers jour / mois / année, et teste si c'est une date valide. Pour l'instant, on ignore les années bissextiles. Par exemple :  
— `date_valide(28, 5, 1973)` renvoie **true**  
— `date_valide(31, 2, 2015)` renvoie **false**
- (2) Écrire une fonction qui prend en argument une date valide sous la forme de trois entiers et affiche la date du lendemain. Par exemple :  
— `jour_suivant(18, 12, 2017)` affiche Le jour suivant est le 19 12 2018.

- (3) Reprendre la question 1 en considérant les années bissextiles (une année est bissextile si elle est divisible par 4, mais pas 100 sauf si elle est divisible par 400).
- (4) Écrire une fonction qui prend en argument une date valide et renvoie le jour de la semaine de cette date.
- (5) Projet Euler- 19 : Combien de 1<sup>er</sup> du mois ont été des dimanches au XX<sup>e</sup> siècle ?