Aide-mémoire – C++

https://grader.soi.ch/workshop/

Entrée/sortie

```
#include <soi>

signed main() {
  int a = read_int();
  int b = read_int();
  int reponse = a + b;
  print(reponse);
}
```

- `read_int()` lit le prochain nombre dans l'entrée.
- `print(x)` écrit la valeur de x dans la sortie.
- `print("salut")` écrit « salut » dans la sortie.
- `print(1, 2, 3, 4)` écrit les quatre nombres « 1 2 3 4 » dans la sortie.

Débogage

```
int x=4;
dbg(x);
dbg(x + 3);
```

Sortie :

```
[task.cpp:5 (main)] x = 4 (int)
[task.cpp:6 (main)] x + 3 = 7 (int)
```

« task.cpp » : fichier ; « 5 », resp. « 6 » : numéro de ligne ; « main » : fonction ; « x », resp. « x+3 » : sortie ; « 4 », resp. « 7 » : valeur ; « int » : type.

Variables

Les noms de variable doivent commencer par une lettre et peuvent ensuite être composés de lettres, de chiffres et de tirets bas *(underscores)*. Toutes les variables ont un type.

- Déclare les variables peu avant leur première utilisation.
- Assigne-leur la bonne valeur.

```
int n = read_int(); // lecture
int somme = 0; // la somme de rien du tout est 0
int index = -1; // toujours un index non valable
bool fini = false; // vérifier si on a déjà fini
bool a_t_on_deja_termine; // un peu long
set<pair<int, int>> sommets_actifs; // plus tard...
```

Instructions conditionnelles

If (si)

```
if (a == b) {
    print("a et b sont égaux");
}
```

If – else (si – sinon)

```
if (a == b) {
    print("a et b sont égaux");
} else {
    print("a et b diffèrent");
}
```

Imbrication d'instructions conditionnelles

```
if (a == b) {
    print("a et b sont égaux");
} else {
    if (a > b) {
        print("a est plus grand que b");
    } else {
        print("b est plus grand que a");
    }
}
```

Else if (sinon, si)

```
if (a == b) {
    print("a et b sont égaux");
} else if (a > b) {
    print("a est plus grand que b");
} else {
    print("b est plus grand que a");
}
```

Ligne directrice : essaie d'imbriquer le moins possible ton code.

Conditions

Les conditions sont des expressions qui renvoient une valeur de vérité, `true` ou `false`. Voici quelques expressions pour tester les relations entre deux nombres a et b :

- `a==b` est `true` si a et b sont égaux et `false` sinon ;
- `a!=b` est `true` si a et b sont inégaux et `false` sinon ;
- `a<b` est `true` si a est plus petit que b et `false` sinon ;
- `a>b` est `true` si a est plus grand que b et `false` sinon ;
- `a<=b` est `true` si a est plus petit ou égal à b et `false` sinon ;
- `a>=b` est `true` si a est plus grand ou égal à b et `false` sinon.

Tu peux combiner plusieurs valeurs de vérité avec des opérateurs logiques :

- `!a` est `true` si a est `false` et `false` sinon ;
- `a&&b` est `true` si a et b sont tous deux `true` et `false` sinon ;
- `a||b` est `true` si a ou b est `true` (ou les deux) et `false` si les deux sont `false`.

L'ordre des opérations est comme suit :
`!` > opérateurs numériques > `==`, `!=` > `&&` > `||`.

Boucles while

Cette instruction exécute répétitivement le corps de la boucle tant que la condition renvoie `true`.

```
int i = 0;
while (i < 10) {
    print("le carré de ", i, " est ", i*i);
    ++i;
}
```

Une boucle peut être interrompue prématurément grâce à l'instruction **`break`**.

```
int i = 0;
while (true) { // exécute cette boucle pour toujours
    print("je vérifie si le carré de ", i, " est 144");
    if (i*i == 144)
        break;
    i++;
}
print("réponse:", i);
```

Boucles for

```
for (int i = 0; i < 10; ++i) {
    print("le carré de ", i, " est ", i*i);
}
```

Les trois parties sont facultatives.

```
int i = 0; // i peut être utilisé après la boucle
for (; i < 10; ++i) {
    if (ce_i_est_special(i))
        break;
}
print(i); // imprime la valeur du i spécial ou 10
```

Boucles for-ever : alternative à `while(true)`

```
int i = 0;
for (;;) { // boucle infinie...
    i = next(i);
    if (ma_condition_speciale(i)) {
        break; // ... en tout cas jusqu'à cette fin
    }
    print("i actuel:", i);
}
```

Exemple : test de primalité

Sortie : Détermine si un nombre donné n est un nombre premier.

```
bool est_premier = true;
for (int i = 2; i < n; i++) {
    if (n % i == 0) {
        est_premier = false;
        break;
    }
}
```

Ordre des opérations

1. `!`
2. `*`, `/`, `%`
3. `+`, `-`
4. `<`, `>`, `<=`, `>=`
5. `==`, `!=`
6. `&&`
7. `||`