LP. Langage C++.Python

I/ Comparaison de leur style

Exemple : afficher du texte en C++ et en Python

Tout ce qui est en italique sont des commentaires -très importants en programmation- et ne font pas partie de l'exécution du programme.

```cpp
#include <iostream> // Bibliothèque nécessaire pour pouvoir afficher
// le "#include" est pratiquement l'équivalent d'un "import" en Python

// Fonction obligatoire, le programme commence TOUJOURS ici,
// des restes du langage C pour (beaucoup) simplifier.
int main()
{
    // Affiche le résultat dans la console
    std :: cout << "Coucou, je viens d'avoir " << 18 << " ans";

    // Si le programme renvoie zéro, c'est que tout va bien
    // Il y a donc un système de contrôle qu'il n'y a pas en Python
    return 0;
}
```

Le programme se résume en une ligne avec Python sans avoir à importer une quelconque bibliothèque (et même sans avoir à mettre d'espaces entre les données).

```python
print("coucou, je viens d'avoir", 18, "ans")
```

Exemple : les structures de contrôles en C++ et en Python.

En langage C++

```cpp
#include <iostream>


int main()
{
    int x = 10;

    if (x == 10) {
        std :: cout<<"x vaut 10";
    }
    else {
        std :: cout<<"x ne vaut pas 10";
    }

    return 0;
}
```

En langage Python

```python
# Soit x un nombre entier valant ici 10
x = 10


if x == 10 :
    print("x vaut 10")
else :
    print("x ne vaut pas pas 10)")
```

```cpp
#include <iostream>

int main()
{
    // Déclaration de la variable i
    int i;

    for(i = 0; i < 10 ; i = i + 1) {
        std :: cout << i << "\n";
    }

    return 0;
}
```

```python
for i in range(10) :
    print(i)
```

```cpp
#include <iostream>

int main()
{
    // Déclaration de la variable i et
    // affectation de sa valeur
    int i = 0;

    while (i < 10) {
        std :: cout << i << "\n";
        i = i + 1;
    }

    return 0;
}
```

```python
i = 0

while i < 10 :
    print(i)
    i = i + 1
```

Hormis les quelques spécificités du C++, les différences sont mineures entre les deux langages et c'est bien cela le plus important.

- Des accolades pour déterminer les blocs en C++ et l'indentation en Python.
- Des « ; » en C++ et le « saut de ligne » en Python pour distinguer les différentes instructions.

Il y a toutefois deux différences très importantes entre les deux langages :

- C++ est un langage dit « compilé » et le programme ne s'exécutera que s'il n'y a aucune erreur de détectée en amont avec sa traduction en langage machine par le compilateur. C++ est donc un langage dont les programmes s'exécutent rapidement car traduits en langage machine en amont.
  Python lit le programme « à la volée » et les erreurs apparaissent à son exécution.
- C++ est un langage à typage statique, il faut impérativement déclarer le type de la variable (nombre entier, flottant, chaîne de caractères etc.) comme en algorithmique alors que Python, à typage dynamique, se charge tout seul de déterminer le « meilleur » type à décider pour chaque variable.
  Quelques types en C++ :

  int : nombre entier

  float : nombre réel

  double : nombre réel à double précision

  string : chaîne de caractères

Ces deux approches différentes induisent des conséquences notables lors de la sécurisation et l'exécution des programmes en fonction du langage utilisé, chacun ayant ses avantages et inconvénients.

II/ Quelques bugs historiques (et à venir)

Le vol 501 d'Ariane V : le 4 juin 1996, la fusée Ariane V a été détruite en vol 37 secondes après son lancement. Déjà utilisé avec Ariane IV, le système a eu un comportement inattendu avec Ariane V. Initialement codée sur 8 octets, la valeur de l'accélération horizontale était convertie, durant le processus de contrôle, en un nombre entier codé sur 2 octets.

La valeur de l'accélération étant trop grande, la conversion en nombre entier a échoué, provoquant une réaction en chaîne dans le pilote automatique qui a abouti à l'autodestruction de la fusée.

Le redémarrage du Boeing 737 : utilisant un système informatique actuellement dépassé (32 bits alors que tout ordinateur est en 64 bits depuis plusieurs années !), un rapport préconisait un redémarrage de l'avion tous les 248 jours pour éviter le pire si le bug a lieu en plein vol.

Il s'agirait là encore d'un problème de débordement d'entier 32-bit déclenché après 2^{31} centisecondes (248,55 jours) de fonctionnement continu, 2^{31} étant le nombre de secondes dans 248 jours multipliés par 100 (un compteur en centièmes de seconde).

Le redémarrage des Airbus 350 : même principe de dépassement de capacité qu'avec les Boeing : https://www.developpez.com/actu/271538/Un-bogue-logiciel-de-l-Airbus-A350-oblige-les-compagnies-aeriennes-a-redemarrer-les-avions-toutes-les-149-heures/

Le bug de l'an 2038 : Les machines UNIX suivent la norme POSIX qui spécifie notamment le temps écoulé en secondes depuis le 1er janvier 1970. De nombreux systèmes de fichiers codent ce temps sur 32 bits par un entier signé (le signe permet d'accéder aux dates antérieures à 1970). Au bout de 2^{31} secondes soit un peu plus de 68 ans, beaucoup de systèmes risquent de repasser 68 ans avant 1970 soit en 1901 !

III/ Langages de programmation populaires actuels

Il est extrêmement difficile de classer les langages de programmation. Plusieurs critères entrent en ligne de compte, entre les préférences des développeurs, l'utilisation en entreprise, les langages les plus plébiscités dans les moteurs de recherche …

Python occupe une place de choix, les langages C/C++ restant toujours aussi estimés malgré leur ancienneté.

Plus d'informations ici : https://www.esilv.fr/numerique-le-classement-des-langages-de-programmation-les-plus-populaires/

Ce qui est important de comprendre, c'est que tout langage de programmation est adapté dans plus de 90% des projets. Il n'y a pas de « meilleur » langage et même son élaboration future semble très compromise. Il ne doit pas devenir un barrage dans l'élaboration d'un projet. D'autre part, il existe de très nombreuses convergences entre eux ; avoir une approche stratégique plutôt que technique est la clé pour appréhender les différents langages.

L'apprentissage de langages de programmation selon *Thomas Pernin* :

« Idéalement, il ne faudrait donc pas apprendre un premier langage, mais au moins deux ou trois en même temps, de types variés. L'expérience montre que c'est pédagogiquement très difficile. Ainsi, ce qui se passe usuellement, c'est que le débutant apprend un langage, puis, à force de sueur, de larmes et de sang, il finit par se débarrasser des mauvais réflexes qu'il a acquis. C'est à ce prix que s'obtient la vraie maîtrise ».