Introduction à C/C++.

Vincent Gaudillière

Polytech Nancy

2017 / 2018

Structure du cours

- 7 séances de 2h
- Séance = Cours (10-15 minutes) + TD
- Cours: présentation des notions importantes + syntaxe
- TD: exercices de difficulté variable (au choix)
- Langage: C (6 séances), C++ (1 séance)
- Contact: vincent.gaudilliere@inria.fr

C/C++: Notions de base.

Vincent Gaudillière

Polytech Nancy

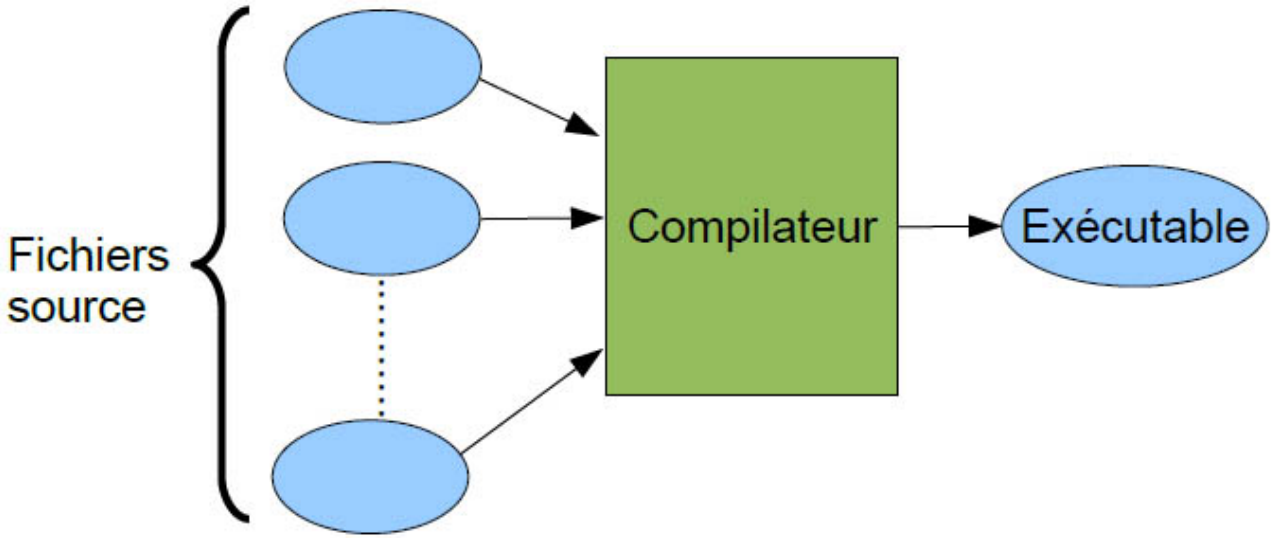
2017 / 2018

Le langage C.

- introduit au début des années 70.
- Un langage *classique* en informatique:
  - à la base de nombreux langages plus récents (C++, Java, C#, PHP),
  - syntaxe largement reprise.
- Très utilisé en programmation système:
  - noyaux d'OS (Windows, Linux, MacOS),
  - logiciels embarqués.
- Quelques caractéristiques:
  - portable ($\forall$ types d'architectures),
  - bas niveau (cf gestion des ressources matérielles).

Langages & traductions.

- Langage de l'ordinateur: **langage binaire** (0 et 1),
- **Langage de programmation**: C, C++, Java, Python, etc...,
- **Traduction** automatique → 2 types:
  - Avant l'exécution = la **compilation** → ex: C, C++, Java,
  - Pendant l'exécution = l'**interprétation** → ex: Python.



Hello world en C.

```c
#include <stdio.h>
#include <stdlib.h>

int main()
{
    printf("Hello world!\n");
    return 0;
}
```

La fonction `main`.

- Fonction principale du programme (exécutée en premier)

```
int main()
{
    // Instruction 1;
    // Instruction 2;
    // ...

    return 0;
}
```

- Les instructions, dans la fonction `main` comme dans n'importe quelle autre bloc d'instructions, sont exécutées séquentiellement.

Les variables en C.

- Types principaux: `char`, `int` (entiers), `float`, `double` (décimaux).
- Déclaration: *type* Nom_de_la_variable;
- Affectation: Nom_de_la_variable = donnée;

```
// Déclaration de la variable x
double x;

// Affectation de la valeur 1.5 à la variable x
x = 1.5;
```

- Initialisation = Déclaration + Affectation.

```
// Initialisation de la variable x
double x = 1.5;
```

Les variables en C.

- Opérations entre variables: $+$, $-$, $*$, $/$, $\%$
- Exemple: addition d'entiers en C

```
// Déclaration des variables
int a,b,c;

// Affectation des variables a et b
a = 1;
b = 3;

// Affectation de la variable c
c = a+b; // ici, la variable c prend la valeur 4
```

Deux fonctions importantes.

1. Affichage de données formatées : `printf`
2. Saisie de données formatées : `scanf`

- Ces fonctions permettent d'interagir avec l'utilisateur:
  - par l'intermédiaire du clavier (`scanf`),
  - par l'intermédiaire de l'écran (`printf`).
- format: %d (`int`), %f (`float`), %lf (`double`), %c (`char`), ...

```
float a;
printf("Entrez un nombre au clavier:\n");
scanf("%f",&a); // &a: adresse de la variable a
printf("Vous avez entré le nombre: %f.\n",a);
```

Expressions conditionnelles.

```
if(/* Proposition A */)
{
    // bloc d'instructions exécuté si A vraie
}
else if(/* Proposition B */)
{
    // bloc d'instructions exécuté si A fausse et B vraie
}
else
{
    // bloc d'instructions exécuté si A et B fausses
}
```

Expressions conditionnelles.

```
if(/* Proposition A */)
{
    // ...
}
else if(/* Proposition B */) // FACULTATIF
{
    // ...
}
else // FACULTATIF
{
    // ...
}
```

Expressions conditionnelles.

Opérateurs de comparaison (variables): ==, !=, >, <, >=, <=.
Opérateurs logiques (propositions): && (et), || (ou).

```
/* Petit jeu: faire deviner un nombre */
int nombreJoueur,nombreCache;
nombreCache = 7;

printf("Quel est, à votre avis, le nombre caché?\n");
scanf("%d",&nombreJoueur);
if(nombreJoueur==nombreCache){
    printf("Bravo, vous avez gagné!\n");
}
else{
    printf("Dommage, vous avez perdu...\n");
}
```

Boucles `while`.

```
while(/* Proposition A */)
{
    // bloc d'instructions exécuté tant que A vraie
}
```

Déroulement:

- (*) La proposition A est étudiée:
  - si A est vraie: le bloc d'instructions est exécuté, puis retour à (*),
  - si A est fausse: le bloc d'instructions est ignoré.

Boucles `for`.

```c
int i; // Ne pas oublier de déclarer la variable i.

for(i=0;i<10;i++)
{
    printf("%d ",i);
}
```

- Sur l'écran après la première itération: "0 ",
- Sur l'écran après la deuxième itération: "0 1 ",
- ...
- Sur l'écran à la fin de l'exécution: "0 1 2 3 4 5 6 7 8 9 ".

Utilisation d'*Eclipse CDT*.

- Créer un projet: **File** > **New** > **C project**,
- Appelez-le *TD1* (par exemple),
- Dans **Project type**: **Executable** > **Empty Project**,
- Dans **ToolChains**: **MinGW GCC**.

- Ecrivez votre code dans *TD1/src/*.c* .

- Compilation: **Project** > **Build Project** (attention aux erreurs!).

- Exécution: clic droit sur votre projet ou sur le fichier source, puis **Run As** > **Local C/C++ Application**.