Les bases de l'Informatique
Formes de programmation

Thierry Vaira

BTS SN

v1.0 - 7 juillet 2017

Programmation

Dans le domaine de l'informatique, la **programmation** est l'ensemble des activités qui permettent l'écriture des programmes informatiques.

- L'immense majorité des programmes qui s'exécutent sur nos ordinateurs, téléphones et autres outils électroniques sont écrits dans des langages de programmation dits **impératifs** : les lignes du programme sont **exécutées les unes après les autres**. Chaque ligne du programme effectue soit une opération simple, soit exécute une fonction qui est elle-même une suite d'opérations simples.
- On approfondira :
  - la **programmation procédurale** qui se fonde sur le concept d'appel procédural. Une **procédure**, aussi appelée **routine**, **sous-routine** ou **fonction** contient simplement une série d'étapes à réaliser.
  - La **programmation orientée objet** (POO) qui consiste en la définition et l'interaction de briques logicielles appelées **objets**. Un objet représente un concept, une idée ou toute entité du monde physique, comme une voiture, une personne ou encore une page d'un livre.

Autres formes de programmation

Ils existent de nombreuses autres formes de programmation :

- la programmation déclarative dont :
  - La programmation logique qui définit les applications à l'aide d'un ensemble de faits élémentaires les concernant et de règles de logique leur associant des conséquences plus ou moins directes.
  - La programmation fonctionnelle qui considère le calcul en tant qu'évaluation de fonctions mathématiques.
- La programmation par contrat dans lequel le déroulement des traitements est régi par des règles, appelées des assertions, qui forment.
- La programmation par contraintes qui permet de résoudre des problèmes combinatoires de grandes tailles tels que les problèmes de planification et d'ordonnancement.

➠ Langages fonctionnels (Lisp, Haskell, OCaml, ...), langages de programmation logique (Prolog), langages de programmation par contrat (Eiffel), ...

Conception

- La phase de conception définit le but du programme.
- Si on fait une rapide analyse fonctionnelle d'un programme, on détermine essentiellement :
  - les données qu'il va traiter (données d'entrée)
  - la méthode employée (appelée l'algorithme)
  - et le résultat qu'il produit (données de sortie)

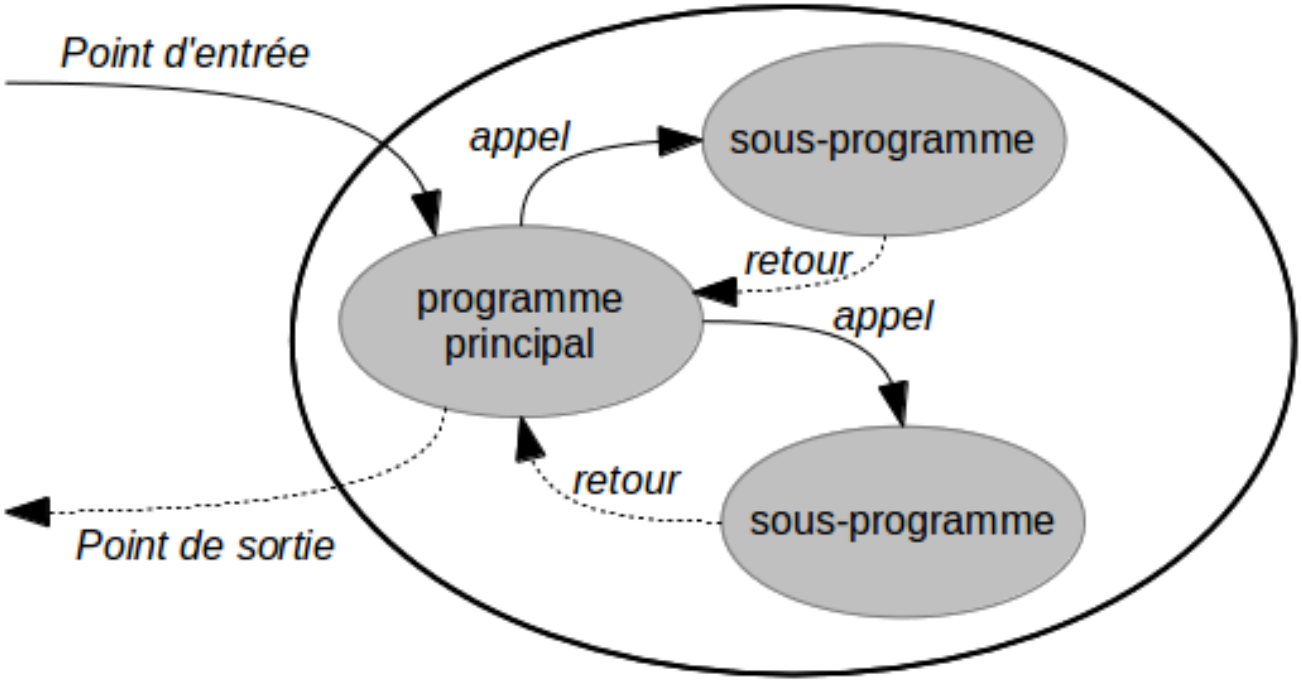
L'approche par fonctions

- Décomposer un problème en sous problèmes :
  - Ceci conduit souvent à diminuer la complexité d'un problème et permet de le résoudre plus facilement.
- Éviter de répéter plusieurs fois les mêmes lignes de code :
  - Ceci facilite la résolution de bogues mais aussi le processus de maintenance.
- Généraliser certaines parties de programmes :
  - La décomposition en module permet de constituer des sous-programmes réutilisables dans d'autres contextes.

Structure d'un programme

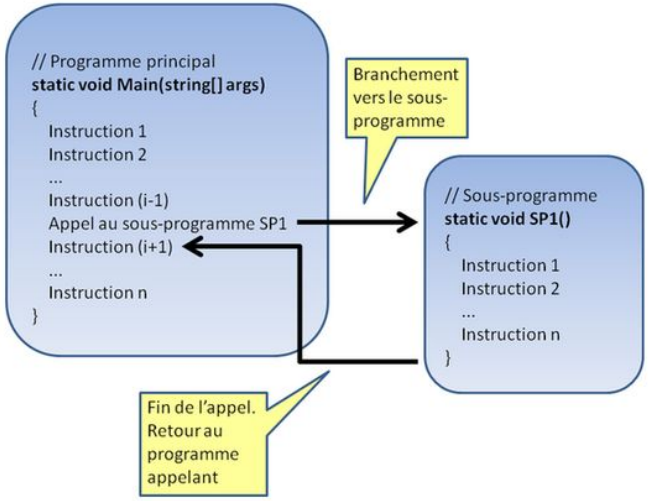
➠ Dans le cas d'une approche fonctionnelle, un programme n'est plus une simple séquence d'instructions mais est constitué :

- D'un ensemble de **sous-programmes** et
- D'un et un **seul programme principal** : unique et obligatoire.



Déroulement d'un programme

- L'exécution du programme commence par l'exécution du programme principal
- L'appel à un sous programme permet de déclencher son exécution, en interrompant le déroulement séquentiel des instructions du programme principal
- Le déroulement des instructions du programme reprend, dès que le sous programme est terminé, à l'instruction qui suit l'appel



Les sous-programmes

➟ On distingue deux types de sous-programmes :

- Les **fonctions**
  - Sous-programme qui retourne **une et une seule valeur** : permet de ne récupérer qu'un résultat.
  - Par convention, ce type de sous-programme ne devrait pas interagir avec l'environnement (écran, utilisateur).
- Les **procédures**
  - Sous-programme qui permet de récupérer de **0 à *n* résultats**
  - Par convention, ce type de sous-programme peut interagir avec l'environnement (écran, utilisateur).
- Cette distinction ne se retrouve pas dans tous les langages de programmation !
  - Par exemple, le C/C++ n'admet que le concept des fonctions qui serviront à la fois pour les fonctions et les procédures.

Définir une fonction

Algorithme :

```
Fonction poserQuestion(q : Chaîne de
     caractères) : Caractère
Donnée(s) : q la question
Résultat : Lit la réponse et retourne
     'V' pour vrai, sinon 'F' pour faux
Variable locale r : Caractère

Début
    Ecrire q
    Répéter
        Ecrire "(V)rai ou (F)aux ?"
        Lire r
    TantQue (r<>'V' ET r<>'F')
    Retourner r
Fin
```

En C++ :

```
char poserQuestion(string q)
{
    char r;

    cout << q;
    do
    {
        cout << "(V)rai ou (F)aux ? ";
        cin >> r;
    }
    while (r!='V' && r!='F');
    return r;
}
```

Appeler une fonction

Algorithme :

```
...
score <- 0
question <- "1. ADN signifie Anti-
     Démangeaison-Nasale ?"
reponse <- poserQuestion(question)
Si reponse = 'F'
    Alors score <- score - 1
    Sinon score <- score + 1
FinSi

question <- "2. La programmation c'est
     facile ?"
...
```

En C++ :

```
...
score = 0;
question = "1. ADN signifie Anti-
     Démangeaison-Nasale ?";
reponse = poserQuestion(question);
if (reponse == 'F')
{
   score = score - 1;
}
else
{
   score = score + 1;
}

question = "2. La programmation c'est
     facile ?";
...
```

Bibliothèque logicielle

- Une **bibliothèque logicielle** est un **ensemble de fonctions** utilitaires, regroupées et mises à disposition afin de pouvoir être utilisées sans avoir à les réécrire.
- Les bibliothèques logicielles ne sont pas complètement des « exécutables » car elles ne possèdent pas de programme principal et par conséquent ne peuvent pas être exécutées directement.
- Les bibliothèques logicielles sont :
  - une interface de programmation (API, *Application Programming Interface*) ;
  - les composants d'un kit de développement logiciel (SDK, *Software Developement Kit*) ;
  - parfois regroupées en un *framework*, de façon à constituer un ensemble cohérent et complémentaire de bibliothèques.
- Exemples :
  - Fichier `.DLL` (*Dynamic Link Library*) ou Bibliothèque de liens dynamiques (Windows).
  - Fichier `.so` (*Shared Object*) ou Bibliothèque dynamique (Linux).

L'approche orientée objet

- Décomposer un problème en objets et les faire interagir entre eux :
  - Un **objet** est caractérisé par le rassemblement, au sein d'une même **unité d'exécution**, d'un ensemble de **propriétés** (constituant son **état**) et d'un **comportement**
  - La notion de **propriété** est matérialisée par un **attribut** qui est une variable locale à l'objet
  - La notion de **comportement** est matérialisée par un ensemble de **méthodes** qui sont ses sous-programmes
  - La **classe** est le modèle (le « moule ») pour créer des objets logiciels.
- On distinguera les langages capables de faire la programmation orientée objet (POO) :
  - C : approche fonctionnelle seulement
  - C++, Java, PHP5, ... : langages orienté objet

Exemple d'objet : une lampe

- Une lampe est **caractérisée par** :
  - Sa **puissance** (une **propriété** ➠ un **attribut**)
  - Le **fait qu'elle soit allumée ou éteinte** (un **état**)
- Au niveau **comportement**, les **actions possibles** sur une lampe sont :
  - L'**allumer** (une **méthode**)
  - L'**éteindre** (une autre **méthode**)

Coder une classe

En Java :

```java
public class Lampe
{
    private int puissance;
    private boolean estAllumee;

    public void allumer()
    {
        this.estAllumee = true;
    }
    public void eteindre()
    {
        this.estAllumee = false;
    }
}
```

En C++ :

```cpp
class Lampe
{
    private:
        int puissance;
        bool estAllumee;

    public:
        void allumer()
        {
            this->estAllumee = true;
        }
        void eteindre()
        {
            this->estAllumee = false;
        }
};
```

La Salle Avignon

Structure d'un programme orienté objet

➠ Dans le cas d'une approche orientée objet, un programme n'est plus une simple séquence d'instructions mais est constitué :

- D'un ensemble d'**objets** s'échangeant des **messages** (i.e. appel d'une méthode) et
- D'un et un **seul programme principal** : unique et obligatoire.