1102 – Introduction à l'algorithmique et à la programmation

Cours : Paul Dorbec
TD-TP : + Pierrick Meignen, Pascal Morin, Albrecht Zimmermann.

Objectifs du Module

Dans le programme national :

> *Savoir décomposer un problème en sous-problèmes plus simples* et *définir des types simples pour structurer les données d'un problème en étant attentif aux critères de qualité de programmation*

1. **Algorithmique :** décrire la méthode employée avec des opérations élémentaires.
2. **Programmation :** transcrire l'algorithme dans un langage contraint, interprétable par une machine (ici le **C**)
3. **Qualité :** garantir un programme sans bug, et facile à maintenir et modifier.

Un algorithme

- description détaillée de la méthode pour atteindre un objectif.
  Ex : recette de cuisine, notice de montage d'un meuble, ...
- naturel mais pas toujours facile à expliciter
  Ex : conduire une voiture, trier sa main au tarot, reconnaître un chat, ...
- niveau de précision à définir, dépend un peu du contexte (de ce qu'on sait déjà faire)
  Ex : pour faire des lasagnes, faites une béchamelle...
- prend d'autant plus de sens quand la méthode n'est pas unique !

Exemple d'algorithme

L'entier n est-il premier ?

```
1 Si n est premier :
2       renvoyer vrai
3 sinon :
4       renvoyer faux
```

```
1 pour chaque i entre 2 et n :
2       si i divise n, renvoyer faux
3       sinon, continuer
4 si aucun ne divise n, renvoyer vrai
```

Exemple d'algorithme

L'entier n est-il premier ?

```
1 Si n est premier :
2       renvoyer vrai
3 sinon :
4       renvoyer faux
```

```
1 pour chaque i entre 2 et n :
2       si i divise n, renvoyer faux
3       sinon, continuer
4 si aucun ne divise n, renvoyer vrai
```

```
1 pour chaque i entre 2 et n :
2       pour chaque j entre 2 et n :
3           si i * j = n, renvoyer faux
4 si aucun ne divise n, renvoyer vrai
```

```
1 pour chaque i entre 2 et n :
2       mettre k à 0,
3       tant que k < n :
4           k devient k + i
5           si k = n, renvoyer faux
6 à l'issue de tout, renvoyer vrai
```

Exemple d'algorithme

L'entier n est-il premier ?

```
1 Si n est premier :
2       renvoyer vrai
3 sinon :
4       renvoyer faux
```

```
1 pour chaque i entre 2 et n :
2       si i divise n, renvoyer faux
3       sinon, continuer
4 si aucun ne divise n, renvoyer vrai
```

```
1 pour chaque i entre 2 et n :
2       pour chaque j entre 2 et n :
3           si i * j = n, renvoyer faux
4 si aucun ne divise n, renvoyer vrai
```

```
1 pour chaque i entre 2 et n :
2       mettre k à 0,
3       tant que k < n :
4           k devient k + i
5           si k = n, renvoyer faux
6 à l'issue de tout, renvoyer vrai
```

pour choisir, nécessaire de savoir ce qu'on sait déjà faire !

Exemple de programme C

L'entier *n* est-il premier ?

```
int est_premier(int n){
  for(int i=2; i<n; i++){
    if(n%i == 0){
      return 0;
    }
  }
  return 1;
}
```

- langage très contraint (mots clés limités)
- très précis
- lié par des conventions
  (0 = faux, 1 = vrai)
- possibilité d'utiliser d'autres bouts de programmes.

La compilation

Que devient ce bout de programme C ?

code source assembleur code binaire

[petite démo]

Compilation, conséquences

Conséquences

Le code obtenu n'est pas une traduction directe du code écrit.

Tout ce qui est :

- nom de variable
- indentation
- saut de ligne
- et autre mise en forme

c'est pour vous, pour permettre de travailler sur votre code.

Fin de l'intro

Au programme

- la syntaxe du C
- le processus de compilation (beaucoup en TP) et les bibliothèques
- les attentes de qualité
- et tout un tas de points de vigilance : portée des variables, passage par valeur, tableaux et structures...