

Algorithmique

Cette fiche de révision NSI appartient à « l'Algorithmique ». Les notions suivantes sont abordées: résoudre un problème par l'automatisation, structures de contrôle.

Un algorithme est une suite finie et non ambiguë d'opérations ou d'instructions permettant de résoudre un problème.

Il est dit *correct* lorsque, pour chaque instance du problème, il se termine en produisant la bonne sortie, c'est-à-dire qu'il résout le problème posé.

On mesure l'efficacité d'un algorithme notamment par son temps d'exécution, par la précision des résultats obtenus, par sa scalabilité (son aptitude à être efficacement parallélisé).

Résoudre un problème par l'automatisation

- Concerne des problématiques récurrentes.
- Peuvent être résolues par le même traitement.
- C'est uniquement un ensemble de règles logiques.

Un algorithme est découpé en une ou plusieurs fonctions.

Chaque fonction est composée de plusieurs parties:

- Initialisation.
- Traitement.
- Retour 1 et 1 seul. (il peut cependant être vide)

Un algorithme est composé de différents traitements:

- Attribution de variables { $PI=3.14$ | couleurs = {1:ROUGE, 2:BLEU} }
- Mathématique simple { rayon=10 | circonference= $PI * (rayon + rayon)$ }
- Traitement de string { rouge='rouges' | Remplacer('s',' ', rouge)}
- Traitement de liste / tableau { t=[0,2,3,4] | t[2]=3 }

Structures de contrôle

IF - CONDITIONNEL

- **SI** condition1 est vraie **ALORS**
 - exécution 1
- **SINON SI** condition 2 est vraie **ALORS**
 - exécution 2
- **SINON**
 - exécution 3
- **FIN SI**

FOR - BOUCLE

- **POUR TOUS** tableaux d'entités **DANS** entité **FAIRE**
 - exécution avec entité accessible
- **FIN POUR**

WHILE - BOUCLE

- **TANT QUE** condition est vraie **FAIRE**
 - exécution
- **FIN TANT QUE**

Un **algorithme récursif** est un algorithme qui résout un problème en calculant des solutions d'instances plus petites du même problème.

En informatique, la **programmation dynamique** est une méthode algorithmique pour résoudre des problèmes d'optimisation, qui consiste :

- A commencer par résoudre les plus petits des sous-problèmes, puis de combiner leurs solutions pour résoudre des sous-problèmes de plus en plus grands.
- Mémoriser **les calculs déjà effectués** évitant ainsi de les réaliser plusieurs fois.
- Dérécursiver* l'algorithme pour qu'il devienne itératif.
- **Gérer l'espace mémoire du mieux possible** en fonction des données qui ne sont plus nécessaires.

Un algorithme est **randomisé** si son comportement dépend non seulement des **données en entrée** (INPUT) mais aussi de valeurs produites par un **générateur de nombres** pseudo-aléatoires. Ils sont connus également sous le nom d'**algorithmes probabilistes**.

CE QU'IL FAUT RETENIR

Un Algorithme peut être écrit en langage naturel, en «pseudo-code» mais il est construit avec des structures de contrôle qui permettront de l'adapter dans tous les langages de programmation.

Un Algorithme doit surtout répondre à un problème, mais il faut élaborer la «manière» dont-il est créé afin de garantir des temps de traitements raisonnables et des résultats corrects.

Un algorithme est codé en un langage clair mais il ne faut pas hésiter à ajouter des commentaires pour décrire ce que fait le code.