Activité d'introduction à l'algorithmique

Amélie Cordier – 10 septembre 2015

Consignes

- Temps de l'activité : 50mn.
- Lisez les textes suivants (dans l'ordre de votre choix).
- Répondez ensuite aux questions à la fin de ce document (directement sur la feuille).
- Nous discuterons ensuite de ces textes en groupe.

Texte 1. Extraits de la page « Algorithmie » de Wikipedia
https://fr.wikipedia.org/wiki/Algorithmique

L'algorithmique (néologisme algorithmie) est l'étude et la production de règles et techniques qui sont impliquées dans la définition et la conception d'algorithmes, c'est-à-dire de processus systématiques de résolution d'un problème permettant de décrire les étapes vers le résultat. En d'autres termes, un algorithme est une suite finie et non-ambiguë d'instructions permettant de donner la réponse à un problème.

Le mot « algorithme » vient du nom du mathématicien Al-Khwarizmi (latinisé au Moyen Âge en Algoritmi), qui, au IXème siècle écrivit le premier ouvrage systématique donnant des solutions aux équations linéaires et quadratiques. Les premiers algorithmes dont on a retrouvé des descriptions datent des Babyloniens, au IIIe millénaire. Ils décrivent des méthodes de calcul et des résolutions d'équations à l'aide d'exemples.

Un algorithme célèbre est celui qui se trouve dans le livre 7 des Éléments d'Euclide, et appelé algorithme d'Euclide. Il permet de trouver le plus grand diviseur commun, ou PGCD, de deux nombres. Un point particulièrement remarquable est qu'il contient explicitement une itération et que les propositions 1 et 2 démontrent sa correction. C'est Archimède qui proposa le premier un algorithme pour le calcul de π.

Le premier à avoir systématisé des algorithmes est le mathématicien arabophone Al Khuwarizmi, actif entre 813 et 833. Dans son ouvrage Abrégé du calcul par la restauration et la comparaison, il étudie toutes les équations du second degré et en donne la résolution par des algorithmes généraux. Il utilise des méthodes semblables à celles des Babyloniens, mais se différencie par ses explications systématiques là où les Babyloniens donnaient seulement des exemples.

Le savant arabe Averroès (1126-1198) évoque une méthode de raisonnement où la thèse

s'affine étape par étape, itérativement, jusqu'à une certaine convergence et ceci conformément au déroulement d'un algorithme. À la même époque, au XIIe siècle, le moine Adelard de Bath introduit le terme latin de algorismus, par référence au nom de Al Khuwarizmi. Ce mot donne algorithme en français en 1554.

Au XVIIe siècle, on pourrait entrevoir une certaine allusion à la méthode algorithmique chez René Descartes dans la méthode générale proposée par le Discours de la méthode (1637), notamment quand, en sa deuxième partie, le mathématicien français propose de « diviser chacune des difficultés que j'examinerois, en autant de parcelles qu'il se pourroit, et qu'il seroit requis pour les mieux résoudre. » Sans évoquer explicitement les concepts de boucle, d'itération ou de dichotomie, l'approche de Descartes prédispose la logique à accueillir le concept de programme, mot qui naît en français en 1677.

L'utilisation du terme algorithme est remarquable chez Ada Lovelace, fille de Lord Byron et assistante de Charles Babbage (1791-1871).

Avec l'informatique, l'algorithmique s'est beaucoup développée. Donald Knuth (né en 1938), auteur du traité The Art of Computer Programming, décrit de très nombreux algorithmes et posa des fondements mathématiques rigoureux pour l'analyse des algorithmes.

Texte 2. Une introduction (pleine d'humour) à l'algorithmie
http://pise.info/algo/introduction.htm
Christophe Darmangeat

« Un langage de programmation est une convention pour donner des ordres à un ordinateur. Ce n'est pas censé être obscur, bizarre et plein de pièges subtils. Ca, ce sont les caractéristiques de la magie. » - Dave Small

« C'est illogique, Capitaine » - Mr Spock

L'algorithmique est un terme d'origine arabe, comme algèbre, amiral ou zénith. Ce n'est pas une excuse pour massacrer son orthographe, ou sa prononciation. Ainsi, l'algo n'est pas « rythmique », à la différence du bon rock'n roll. L'algo n'est pas non plus « l'agglo ». Alors, ne confondez pas l'algorithmique avec l'agglo rythmique, qui consiste à poser des parpaings en cadence.

1. Qu'est-ce que l'algomachin ?
Avez-vous déjà ouvert un livre de recettes de cuisine ? Avez vous déjà déchiffré un mode

d'emploi traduit directement du coréen pour faire fonctionner un magnétoscope ou un répondeur téléphonique réticent ? Si oui, sans le savoir, vous avez déjà exécuté des algorithmes.

Plus fort : avez-vous déjà indiqué un chemin à un touriste égaré ? Avez vous fait chercher un objet à quelqu'un par téléphone ? Ecrit une lettre anonyme stipulant comment procéder à une remise de rançon ? Si oui, vous avez déjà fabriqué – et fait exécuter – des algorithmes. Comme quoi, l'algorithmique n'est pas un savoir ésotérique réservé à quelques rares initiés touchés par la grâce divine, mais une aptitude partagée par la totalité de l'humanité. Donc, pas d'excuses…

Un algorithme, c'est une suite d'instructions, qui une fois exécutée correctement, conduit à un résultat donné. Si l'algorithme est juste, le résultat est le résultat voulu, et le touriste se retrouve là où il voulait aller. Si l'algorithme est faux, le résultat est, disons, aléatoire, et décidément, cette saloperie de répondeur ne veut rien savoir.

Complétons toutefois cette définition. Après tout, en effet, si l'algorithme, comme on vient de le dire, n'est qu'une suite d'instructions menant celui qui l'exécute à résoudre un problème, pourquoi ne pas donner comme instruction unique : « résous le problème », et laisser l'interlocuteur se débrouiller avec ça ? A ce tarif, n'importe qui serait champion d'algorithmique sans faire aucun effort. Pas de ça Lisette, ce serait trop facile.

Le malheur (ou le bonheur, tout dépend du point de vue) est que justement, si le touriste vous demande son chemin, c'est qu'il ne le connaît pas. Donc, si on n'est pas un goujat intégral, il ne sert à rien de lui dire de le trouver tout seul. De même les modes d'emploi contiennent généralement (mais pas toujours) un peu plus d'informations que « débrouillez vous pour que ça marche ».

Pour fonctionner, un algorithme doit donc contenir uniquement des instructions compréhensibles par celui qui devra l'exécuter. C'est d'ailleurs l'un des points délicats pour les rédacteurs de modes d'emploi : les références culturelles, ou lexicales, des utilisateurs, étant variables, un même mode d'emploi peut être très clair pour certains et parfaitement abscons pour d'autres.

En informatique, heureusement, il n'y a pas ce problème : les choses auxquelles ont doit donner des instructions sont les ordinateurs, et ceux-ci ont le bon goût d'être tous strictement aussi idiots les uns que les autres.

2. Faut–il être matheux pour être bon en algorithmique ?

Je consacre quelques lignes à cette question, car cette opinion aussi fortement affirmée que

faiblement fondée sert régulièrement d'excuse : « moi, de toute façon, je suis mauvais(e) en algo, j'ai jamais rien pigé aux maths ». Faut-il être « bon en maths » pour expliquer correctement son chemin à quelqu'un ? Je vous laisse juge.

La maîtrise de l'algorithmique requiert deux qualités, très complémentaires d'ailleurs :

- il faut avoir une certaine intuition, car aucune recette ne permet de savoir a priori quelles instructions permettront d'obtenir le résultat voulu. C'est là, si l'on y tient, qu'intervient la forme « d'intelligence » requise pour l'algorithmique. Alors, c'est certain, il y a des gens qui possèdent au départ davantage cette intuition que les autres. Cependant, et j'insiste sur ce point, les réflexes, cela s'acquiert. Et ce qu'on appelle l'intuition n'est finalement que de l'expérience tellement répétée que le raisonnement, au départ laborieux, finit par devenir « spontané ».
- il faut être méthodique et rigoureux. En effet, chaque fois qu'on écrit une série d'instructions qu'on croit justes, il faut systématiquement se mettre mentalement à la place de la machine qui va les exécuter, armé d'un papier et d'un crayon, afin de vérifier si le résultat obtenu est bien celui que l'on voulait. Cette opération ne requiert pas la moindre once d'intelligence. Mais elle reste néanmoins indispensable, si l'on ne veut pas écrire à l'aveuglette.

Et petit à petit, à force de pratique, vous verrez que vous pourrez faire de plus en plus souvent l'économie de cette dernière étape : l'expérience fera que vous « verrez » le résultat produit par vos instructions, au fur et à mesure que vous les écrirez. Naturellement, cet apprentissage est long, et demande des heures de travail patient. Aussi, dans un premier temps, évitez de sauter les étapes : la vérification méthodique, pas à pas, de chacun de vos algorithmes représente plus de la moitié du travail à accomplir… et le gage de vos progrès.

3. L'ADN, les Shadoks, et les ordinateurs

Quel rapport me direz-vous ? Eh bien le point commun est : quatre mots de vocabulaire. L'univers lexical Shadok, c'est bien connu, se limite aux termes « Ga », « Bu », « Zo », et « Meu ». Ce qui leur a tout de même permis de formuler quelques fortes maximes, telles que : « Mieux vaut pomper et qu'il ne se passe rien, plutôt qu'arrêter de pomper et risquer qu'il se passe quelque chose de pire » (pour d'autres fortes maximes Shadok, n'hésitez pas à visiter leur site Internet, il y en a toute une collection qui vaut le détour).

L'ADN, qui est en quelque sorte le programme génétique, l'algorithme à la base de construction des êtres vivants, est une chaîne construite à partir de quatre éléments invariables. Ce n'est que le nombre de ces éléments, ainsi que l'ordre dans lequel ils sont arrangés, qui vont déterminer si on obtient une puce ou un éléphant. Et tous autant que nous sommes, splendides réussites de la Nature, avons été construits par un « programme » constitué uniquement de ces quatre briques, ce qui devrait nous inciter à la modestie.

Enfin, les ordinateurs, quels qu'ils soient, ne sont fondamentalement capables de comprendre que quatre catégories d'ordres (en programmation, on n'emploiera pas le terme d'ordre, mais plutôt celui d'instructions). Ces quatre familles d'instructions sont :

- l'affectation de variables
- la lecture / écriture
- les tests
- les boucles

Un algorithme informatique se ramène donc toujours au bout du compte à la combinaison de ces quatre petites briques de base. Il peut y en avoir quelques unes, quelques dizaines, et jusqu'à plusieurs centaines de milliers dans certains programmes de gestion. Rassurez-vous, dans le cadre de ce cours, nous n'irons pas jusque là (cependant, la taille d'un algorithme ne conditionne pas en soi sa complexité : de longs algorithmes peuvent être finalement assez simples, et de petits très compliqués).

4. Algorithmique et programmation

Pourquoi apprendre l'algorithmique pour apprendre à programmer ? En quoi a-t-on besoin d'un langage spécial, distinct des langages de programmation compréhensibles par les ordinateurs ?

Parce que l'algorithmique exprime les instructions résolvant un problème donné indépendamment des particularités de tel ou tel langage. Pour prendre une image, si un programme était une dissertation, l'algorithmique serait le plan, une fois mis de côté la rédaction et l'orthographe. Or, vous savez qu'il vaut mieux faire d'abord le plan et rédiger ensuite que l'inverse...

Apprendre l'algorithmique, c'est apprendre à manier la structure logique d'un programme informatique. Cette dimension est présente quelle que soit le langage de programmation ; mais lorsqu'on programme dans un langage (en C, en Visual Basic, etc.) on doit en plus se colleter les problèmes de syntaxe, ou de types d'instructions, propres à ce langage. Apprendre l'algorithmique de manière séparée, c'est donc sérier les difficultés pour mieux les vaincre.

A cela, il faut ajouter que des générations de programmeurs, souvent autodidactes (mais pas toujours, hélas !), ayant directement appris à programmer dans tel ou tel langage, ne font pas mentalement clairement la différence entre ce qui relève de la structure logique générale de toute programmation (les règles fondamentales de l'algorithmique) et ce qui relève du langage particulier qu'ils ont appris. Ces programmeurs, non seulement ont beaucoup plus de mal à passer ensuite à un langage différent, mais encore écrivent bien souvent des programmes qui même s'ils sont justes, restent laborieux. Car on n'ignore pas impunément les règles fondamentales de l'algorithmique... Alors, autant l'apprendre en tant que telle !

5. Avec quelles conventions écrit–on un algorithme ?

Historiquement, plusieurs types de notations ont représenté des algorithmes. Il y a eu notamment une représentation graphique, avec des carrés, des losanges, etc. qu'on appelait des organigrammes. Aujourd'hui, cette représentation est quasiment abandonnée, pour deux raisons. D'abord, parce que dès que l'algorithme commence à grossir un peu, ce n'est plus pratique du tout du tout. Ensuite parce que cette représentation favorise le glissement vers un certain type de programmation, dite non structurée (nous définirons ce terme plus tard), que l'on tente au contraire d'éviter.

C'est pourquoi on utilise généralement une série de conventions appelée « pseudo-code », qui ressemble à un langage de programmation authentique dont on aurait évacué la plupart des problèmes de syntaxe. Ce pseudo-code est susceptible de varier légèrement d'un livre (ou d'un enseignant) à un autre. C'est bien normal : le pseudo-code, encore une fois, est purement conventionnel ; aucune machine n'est censée le reconnaître. Donc, chaque cuisinier peut faire sa sauce à sa guise, avec ses petites épices bien à lui, sans que cela prête à conséquence.

[...]

Texte 3. Une autre introduction à l'algo par Fabien Torre

http://www.grappa.univ-lille3.fr/~torre/Enseignement/Cours/algo.php

Présentation informelle

Considérons les étapes qui interviennent dans la résolution problème quelconque :

1. concevoir une procédure qui une à fois appliquée amènera à une solution du problème ;
2. résoudre effectivement le problème en appliquant cette méthode.

Le résultat du premier point sera nommé un *algorithme*. Quant au deuxième point, c'est-à-dire la mise en pratique de l'algorithme, nous l'appellerons un *processus*.

Ces notions sont très répandues dans la vie courante. Un algorithme peut par exemple y prendre la forme :

- d'une recette de cuisine,
- d'un mode d'emploi,
- d'une notice de montage,
- d'une partition musicale,
- d'un texte de loi,
- d'un itinéraire routier.

Dans le cas particulier de l'informatique, une étape supplémentaire vient se glisser entre la conception de l'algorithme et sa réalisation à travers un processus : l'algorithme doit être

rendu compréhensible par la machine que nous allons utiliser pour résoudre effectivement le problème. Le résultat de la traduction de l'algorithme dans un langage connu de la machine est appelé un *programme*.

Rapide historique

C'est un mathématicien perse du 8ème siècle, Al-Khawarizmi, qui a donné son nom à la notion d'algorithme. Son besoin était de traduire un livre de mathématiques venu d'Inde pour que les résultats et les méthodes exposés dans ce livre se répandent dans le monde arabe puis en Europe. Les résultats devaient donc être compréhensibles par tout autre mathématicien et les méthodes applicables, sans ambiguïté.

En particulier, ce livre utilisait une numérotation de position, ainsi que le chiffre zéro que ce type de représentation des nombres rend nécessaire. Par ailleurs, le titre de ce livre devait être repris pour désigner une branche des mathématiques, l'algèbre.

Si l'origine du mot *algorithme* est très ancienne, la notion même d'algorithme l'est plus encore : on la sait présente chez les babyloniens, 1800 ans avant JC.

Une tablette assyrienne provenant de la bibliothèque de Assourbanipal et datée de -640 expose une recette pour obtenir des pigments bleus (notons que pour avoir un algorithme satisfaisant, il faudrait préciser ici les temps de cuisson !) :

- tu ajouteras à un *mana* de bon *tersitu* un tiers de *mana* de verre *sirsu* broyé, un tiers de *mana* de sable, cinq *kisal* de craie,
- tu broieras encore,
- tu le recueilleras dans un moule, en le fermant avec un moule réplique,
- tu le placeras dans les ouvreaux du four,
- il rougeoira et *uknû merku* en sortira.

En résumé, il doit être bien clair que cette notion d'algorithme dépasse, de loin, l'informatique et les ordinateurs. Il nécessite un vocabulaire partagé, des opérations de base maîtrisées par tous et de la précision.

Rappel : dans la suite, répondez aux questions directement sur le document. Vous pouvez détacher les feuilles.

Activité d'introduction à l'algorithmique – Questions

Amélie Cordier – 10 septembre 2015

Prénom :
Nom :

1. Avez-vous déjà écrit des algorithmes ?
Oui Non

2. Que signifie le terme algorithme (autrement dit, quelle est son origine éthymologique) ?

3. À quand remontent les premiers algorithmes ?

4. De quand date le terme « algorithme » ?

5. Qu'est-ce qu'une itération ?

6. Quel est le lien entre René Descartes et l'algorithmie ?

7. Quelles qualités doit-on avoir, ou travailler, pour réaliser de bons algorithmes ?

8. Quel est l'intérêt, selon vous, d'apprendre l'algorithmique avant la programmation ?

9. Nous sommes confrontés à des algorithmes tous les jours : par exemple, quand la boulangère vous rend votre monnaie, elle suit un algorithme, ou bien, lorsque vous vous dites « si j'ai cours à 8h, je mets mon réveil à 7h, mais si j'ai cours à 10h, alors je mets mon réveil à 9h et je suis plus content », vous pensez à un algorithme. Quand vous utilisez votre machine à laver, elle suit un algorithme. Donnez ci-dessous 5 exemples d'algorithme que l'on « croise » probablement sans s'en rendre compte dans notre vie quotidienne.

10. Pourquoi la notion de *vocabulaire* et celle de *langage* sont importantes pour réaliser des algorithmes ?

11. Écrivez ci-dessous, en français, l'algorithme que vous avez suivi pour réaliser cette activité.

12. Résumez ci-contre ce que vous avez retenu des textes précédents.