3. Instructions

Bonjour. Dans cette 3ème séquence sur l'algorithmique, nous allons parler des instructions élémentaires. Donc on va vraiment aller dans le cœur de l'algorithmique, à détailler quelles sont les instructions que l'ordinateur est capable d'exécuter.

La première, c'est l'affectation, dont on a parlé à la séquence précédente. On a une variable et on va modifier sa valeur. Donc on peut avoir i qui reçoit une valeur donnée, 0, 17, 42, ou alors on peut avoir une valeur qui est calculée à partir d'autres valeurs et d'autres variables, par exemple i qui reçoit i+1, donc si i valait 3, il vaudra maintenant 4.

L'instruction suivante, c'est le test. Donc le test c'est si une valeur booléenne - une valeur booléenne, c'est une valeur qui est soit vraie, soit fausse - alors on exécute une instruction et si elle est fausse, on exécute une autre instruction. Donc le test sur le booléen peut être "est-ce qu'une valeur est plus grande que 10 ?", donc ça, ça va être le résultat du booléen, donc est-ce que c'est vrai ou est-ce que c'est faux ? A ce moment-là on exécute une instruction ou une autre instruction. Un autre type de test c'est "est-ce qu'on risque de dépasser les valeurs maximales ou minimales pour un type de variables ?". Par exemple si vous avez une valeur entière codée sur un octet entre 0 et 255, on peut avoir envie d'être sûr que l'addition ne va pas dépasser 255. Et un autre cas de test, c'est quand on accède à l'intérieur d'un tableau, on veut être sûr qu'on accède à un endroit licite de la mémoire et donc on va tester qu'on est bien entre les 2 bornes d'un tableau.

L'instruction suivante, c'est la séquence. L'intérêt de l'ordinateur, c'est aussi qui est capable d'exécuter successivement un certain nombre d'opérations, donc là d'instructions. Par exemple, on peut faire une affectation puis un test ou une affectation, une boucle et un test, ou un certain nombre d'affectations successives.

Ensuite, une instruction plus intéressante, c'est la boucle. Donc la boucle, la première boucle la plus commune quand on a des tableaux, c'est le "pour", donc pour une certaine variable qui va itérer d'une valeur à une autre valeur, on exécute une certaine instruction. Donc ça peut servir pour parcourir un tableau. Par exemple quand on veut chercher le maximum d'un tableau, on va le parcourir pour trouver la plus grande valeur. Si on veut calculer la moyenne, ou si on veut remplir un tableau aussi pour l'initialiser, on va faire une boucle "pour". La boucle "tant que" est plus générale donc elle permet de faire plus de choses, et donc c'est "tant que" un certain booléen, on fait une certaine instruction. Donc là, une utilisation classique de la boucle "tant que", c'est une suite convergente, donc quand effectivement on va calculer les itérations d'une certaine suite jusqu'à ce que les valeurs soient suffisamment proches. Alors il faut faire attention à la boucle "tant que", parce qu'il faut être sûr que le programme s'arrête, parce qu'effectivement, on peut avoir des boucles tant que qui ne s'arrêtent pas et donc un programme qui serait susceptible de tourner à l'infini, et donc c'est des problèmes qu'on appelle de terminaisons, donc être sûr qu'effectivement l'algorithme va s'arrêter.

Alors, avec toutes ces instructions, donc la séquence, le test, la boucle et l'affectation de variable, vous pouvez faire tous les algorithmes du monde. Vous prenez toutes ces instructions-là, dans l'ordre que vous voulez, en quantité que vous voulez et vous obtenez n'importe quel algorithme. Donc il suffit de connaître ces instructions-là pour pouvoir faire n'importe quoi en algorithmique.

Maintenant, dans la série "on va faire quelque chose en algorithmique", je vais vous présenter un exemple qui est une recherche dichotomique. L'idée c'est de partir d'un tableau qui va être trié, donc c'est très important qu'il soit trié, et dans lequel on va chercher une certaine valeur. Le tableau c'est T, il a une taille N qui est strictement positive, et on va chercher une valeur V, donc est-ce qu'elle est dans ce tableau ou pas ? La sortie, c'est un booléen, donc un vrai ou un faux, qui va dire si cette valeur-là apparaît dans le tableau.

On va avoir 2 variables qui sont "inf" et "sup", qui en fait représentent la partie du tableau qu'on est en train d'examiner. Au début on regarde dans tout le tableau, et en fait, après on ne va regarder que sur la moitié du tableau, le quart du tableau, le 8ème du tableau, etc. , et pour ça, ce qu'on fait c'est qu'en fait on a le morceau du tableau qu'on est en train de regarder qui est entre "inf" et "sup", et ensuite on calcule le milieu entre les 2, donc qui va être i, et en fait on regarde la valeur à cet endroit-là. Si cette valeur est plus grande que v, ça veut dire que v, s'il y est, il est dans la partie à gauche du tableau, et si par contre cette valeur-là est plus petite que v, ça veut dire qu'on sera dans la partie droite du tableau, et donc on va modifier "inf" et "sup" de façon effectivement à ne garder que la petite portion du tableau, donc cette portion rétrécissant où v est susceptible d'être. Et si jamais à la fin, "inf" et "sup" se sont échangés, ce qui peut sembler bizarre, mais ce qui effectivement va arriver dans le cas où l'élément n'est pas dans le tableau, à ce moment-là, on renvoie "faux".

Un petit exemple : ici vous voyez un tableau, un petit tableau à 9 éléments. Il faut imaginer que ce genre d'algorithmes est exécuté sur des tableaux à plusieurs millions d'éléments, et donc je cherche 5, alors oui je sais que 5 n'est pas dans le tableau, mais ça permet d'avoir un déroulé complet de l'algorithme. Donc imaginons que je cherche 5 pour savoir s'il est dans ce tableau. Donc j'ai mes variables "inf" et "sup" qui sont pour l'instant aux 2 bouts du tableau.

Donc "inf" pointe vers -5 et "sup" pointe vers 25, donc je calcule i qui est le milieu entre "inf" et "sup", donc qui arrive ici, et donc la valeur associée à I c'est 4. Dont 4 d'une part n'est pas 5, donc ça veut dire que pour l'instant je n'ai pas trouvé l'élément, et ensuite, 4 est plus petit que 5, donc ça veut dire que si 5 est dans le tableau, il est dans la partie supérieure du tableau, et donc le nouveau "inf" va être i+1, parce que je sais que le 5 ne peut pas être entre le début et le I du tableau. Donc voilà le nouveau "inf", donc maintenant j'ai un nouveau "inf", un nouveau "sup", je calcule leur milieu qui sera i, donc qui lui pointe vers 6. Donc 6, ce n'est toujours pas 5 donc je n'ai toujours pas trouvé, mais par contre il est plus grand, donc dans ce cas-là, ce que je garde c'est la partie du tableau inférieure.

Donc entre "inf" et i, mais I je sais que ce n'est déjà pas 5, donc entre "inf" et i-1, donc le nouveau "sup", ce sera i-1, donc ce sera également "inf". Donc après je calcule la moyenne entre "inf" et "sup", donc la moyenne ça va faire exactement la même valeur, donc qui pointe vers 4. Donc je me rends bien compte que 4 ce n'est toujours pas 5, donc c'est dans la partie supérieure, donc "sup" et "inf" sont maintenant échangés, ce qui me permet d'arrêter l'algorithme et de dire qu'effectivement 5 n'est pas dans le tableau. Alors juste une petite note, en fait il ne vaut mieux pas calculer (inf+sup)/2 mais cette autre expression pour éviter les dépassements de capacités.