ATL-1209 Organigramme pour représenter des étapes

Halim Djerroud, Denis Bureau

révision 2.0

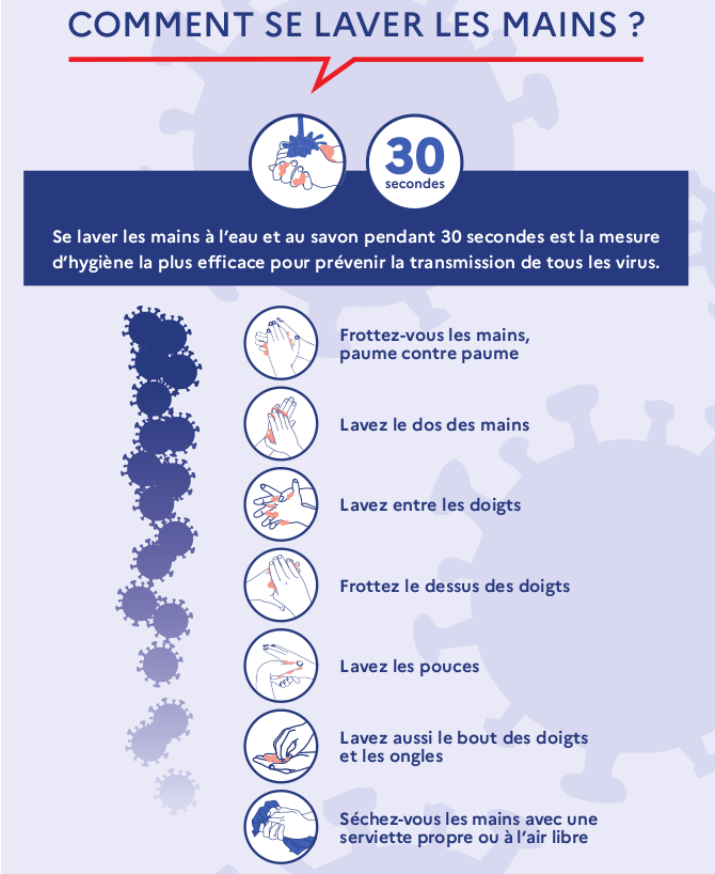
Ce que vous devez rendre est spécifié à la section 6, en dernière page de ce document.

1 Introduction

Un algorithme c'est quoi ?

Un algorithme est une suite d'instructions claires permettant de résoudre un problème donné. Cela revient par exemple à expliquer à quelqu'un **pas à pas** les étapes à faire pour résoudre un problème.

Exemple : Expliquer à quelqu'un comment se laver les mains correctement (*Algorithme proposé par le ministère de la santé pour limiter la propagation du virus Covid-19*) :



— Le lavage de mains est décrit en indiquant sept opérations simples. On utilise le mot "instructions".
— Un algorithme est une suite d'instructions (étapes). Il peut être considéré comme une recette, une manière de faire décrite pas à pas.

2 Organigramme

Représenter les étapes sous forme d'un schéma

À l'instar de l'algorithme pour le lavage des mains proposé par le ministère de la Santé, les étapes sont représentées sous forme d'un schéma permettant de faciliter la lecture des instructions, ce schéma appelé communément un **organigramme** (représentation visuelle d'un algorithme).

Les algorithmes sont le plus souvent utilisés lors de programmation informatique pour expliquer à la machine les étapes à suivre pour atteindre un objectif. Les *Organigrammes* sont utilisés par les programmeurs pour visualiser les étapes avant d'écrire le programme informatique. Donc les **organigrammes** sont des outils de conception de programmes informatiques.

Les avantages des organigrammes

En informatique, on utilise généralement les organigrammes pour modéliser les programmes informatiques avant leur implémentation dans un langage de programmation, par exemple *Python*.

Les avantages de dessiner un organigramme avant de coder sur la machine directement :
- Visualiser les étapes et détecter facilement des cas qui ne sont pas couverts par votre programme
- La visualisation des étapes vous aide à avoir une meilleure optimisation de votre programme
- Les organigrammes sont indépendants des langages de programmation

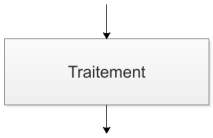
Règles et symboles

En informatique, on utilise généralement des symboles normalisés (ISO 5807) pour modéliser les programmes informatiques avant leur implémentation. Les symboles utilisés sont :

Début ou fin

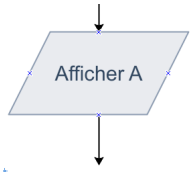
Début ou fin d'un algorithme

Les traitements



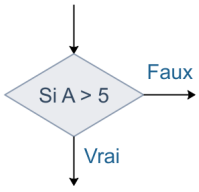
Opération sur des données, instructions, ... ou opération pour laquelle il n'existe aucun symbole normalisé

Entrées/sorties



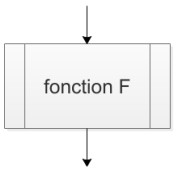
Instructions de communication avec le monde extérieur : Lire et afficher des valeurs

Les conditions



Test ou Branchement Conditionnel : Décision d'un choix parmi d'autres en fonction des conditions

Les appels de sous-programme



Appel d'un sous-programme

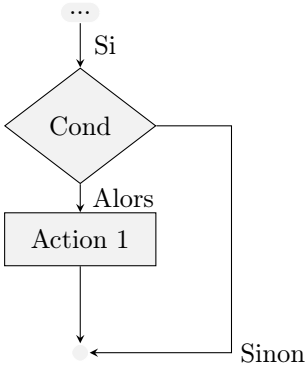
Règles pour construire un organigramme

Il existe différentes règles pour construire un organigramme :
— La lecture de l'organigramme se fait verticalement
— Les lignes de liaisons entre les symboles ne doivent pas se couper
— Une ligne de liaison doit toujours arriver sur le haut et au centre d'un symbole
— Les commentaires sont à placer de préférence à droite ou au-dessus
— Les flèches qui descendent à droite et celles qui remontent à gauche

3 Structures de l'organigramme de programmation

SI... ALORS...[IF... THEN...]

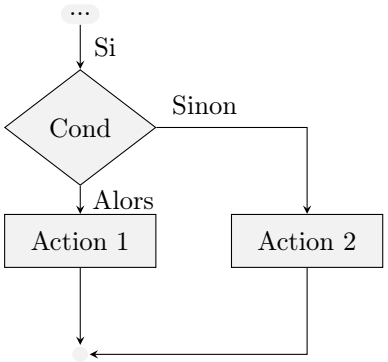
Si la condition est vraie alors `l'action` est exécutée, sinon elle ne l'est pas.




```
SI (condition vraie )
ALORS
    BLOC D'INSTRUCTIONS (Action 1)
FIN SI
```

SI... ALORS... SINON...[IF... THEN... ELSE...]

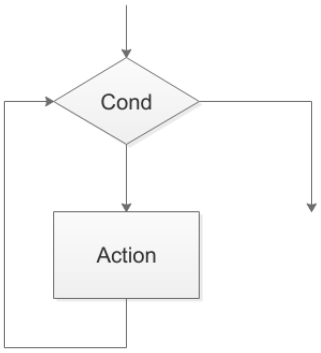
Exécution d'un test, selon le résultat, le programme se poursuit vers une branche ou vers l'autre.
Si la condition est vraie alors `l'action1` est exécutée, sinon c'est `l'action 2` qui sera exécutée.




```
SI ( condition vraie )
ALORS
    BLOC D'INSTRUCTIONS (Action 1)
SINON
    BLOC D'INSTRUCTIONS (Action 2)
FINSI
```

TANTQUE... FAIRE...[WHILE... DO...]

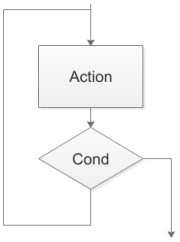
Exécution du bloc `Action` tant-que la condition est vrai




```
TANT QUE ( condition est vraie )
FAIRE
    BLOC D'INSTRUCTIONS
FIN TANT QUE
```

REPETER... TANT QUE... [DO... While...]

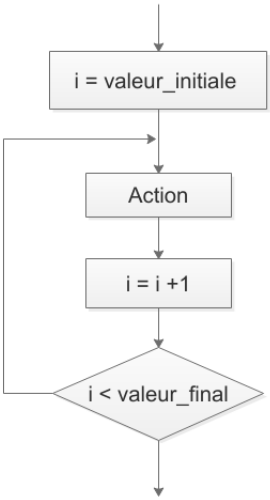
Contrairement à la boucle tant que, `l'action` est ici exécutée au moins une fois.




```
REPETER
    BLOC D'INSTRUCTIONS (Action)
TANT QUE (condition est vraie)
```

POUR... FAIRE...[FOR... DO...]

Cette boucle est utilisée quand le nombre de boucles à exécuter est connu d'avance.




```
POUR i de valeur_initiale À valeur_finale PAS n
ALORS
    BLOC D'INSTRUCTIONS (Action)
FIN POUR
```

4 Exemples d'organigrammes

Algorithme d'Euclide

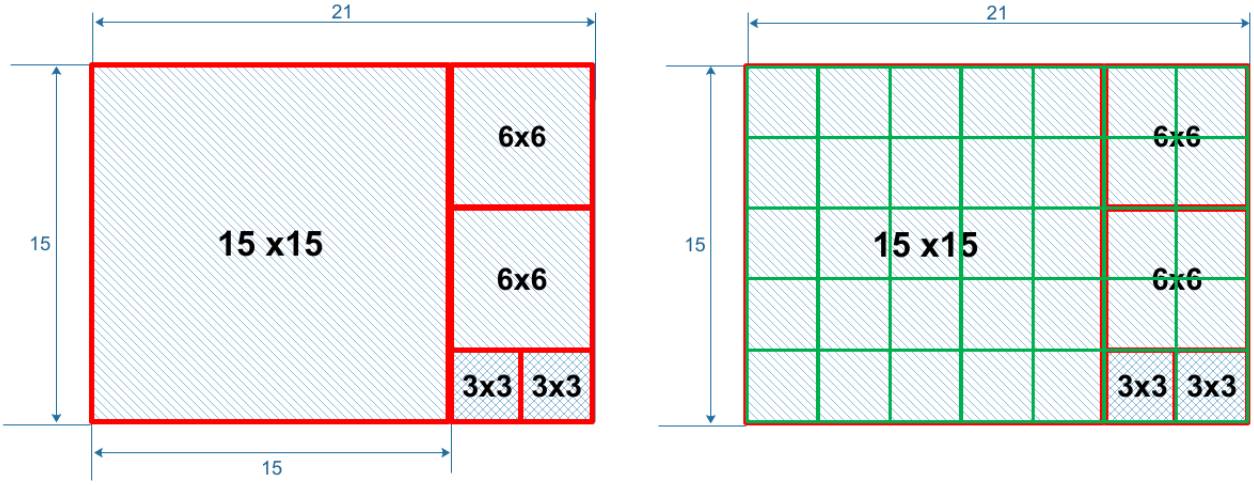
En mathématiques, l'algorithme d'Euclide permet de calculer le plus grand commun diviseur (PGCD).

On peut représenter un nombre entier comme une longueur sur une ligne droite. Deux entiers peuvent être représentés sur un rectangle. Le plus grand des deux entiers représente la longueur du rectangle et le plus petit sa largeur.

Si on souhaite carreler entièrement ce rectangle avec le plus grand carré possible, alors il faut chercher le plus grand diviseur commun (PGCD) de la largeur et la longueur du rectangle.

Euclide propose un algorithme qui permet de calculer le PGCD en décomposant ce rectangle en carrés de plus en plus petits, par divisions successives, de la longueur par la largeur, jusqu'à obtenir une largeur qui est égale à la longueur.

Considérons une feuille de dimensions $Longueur = 21\ cm$ par $largeur = 15\ cm$ (voir image ci-dessous). Le plus grand carré qu'on peut former a un côté de 15 cm, il reste un rectangle de 15 $cm \times 6\ cm$. Dans le carré restant on peut former deux carrés de 6 cm mais il reste un rectangle de côtés $6\ cm \times 3\ cm$ que l'on peut carreler entièrement de carrés de 3 cm. Le carré de 3 cm est le PGCD, car il permet de carreler entièrement la feuille.



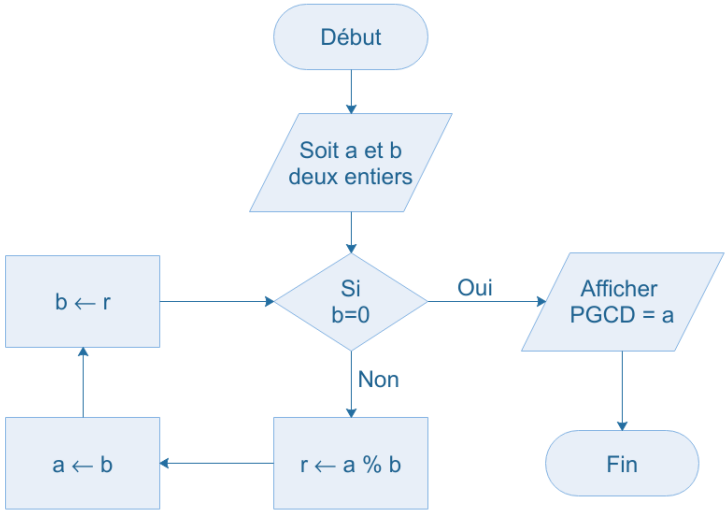
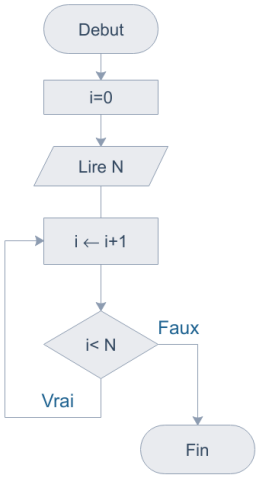
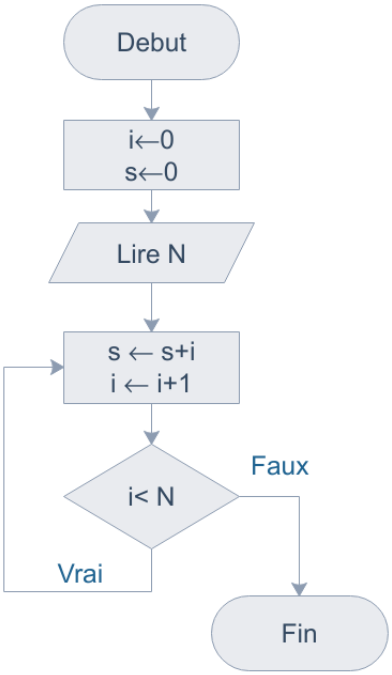


FIGURE 1 – Organigramme pour calculer le PGCD

Organigramme des N premiers nombres entiers :

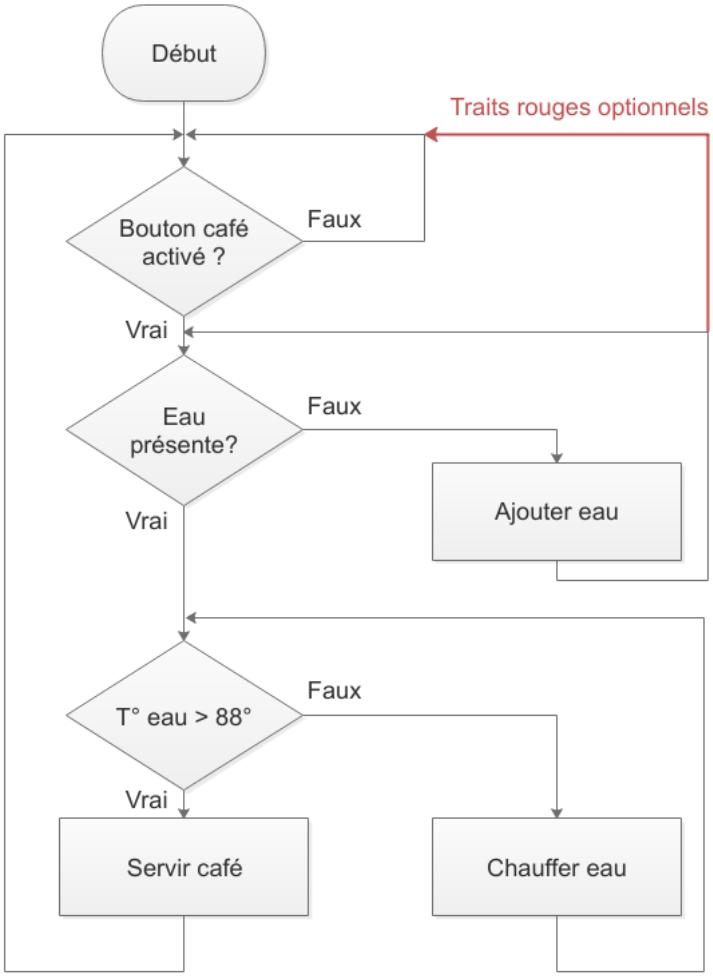


Organigramme de la somme des N premiers nombres entiers :



Organigramme décrivant le fonctionnement simple d'une cafetière

Le café ne peut couler que si le bouton est activé et s'il y a de l'eau présente dans le réservoir ; sinon **ajouter eau** si l'eau est à une température $> 88°$.



5 Exercices (Proposés par A. Djebali)

Exercice 1

Premiers exercices simples. Dessiner les organigrammes représentant les traitements suivants :

1. Calculer et afficher la factorielle pour un entier positif donné N.
   Rappel : La factorielle de N (notée $N!$) $= 1 \times 2 \times \ldots \times N$.
2. Calculer et afficher la multiplication de 2 entiers positifs a et b par additions successives (en supposant évidemment ne pas disposer de la multiplication). Optionnellement :
   (a) Voyez-vous comment optimiser votre algorithme pour qu'il fasse le moins d'additions possible ?
   (b) Ajouter le traitement du signe pour autoriser a et/ou b à être négatif(s).
3. Calculer et afficher de la division de 2 entiers positifs a et b par soustractions successives (en supposant que $a > b$). Optionnellement :
   Ajouter le traitement du cas $a < b$.
4. Recherche séquentielle d'un entier X dans un tableau de N entiers. Si x trouvé, alors afficher son indice, -1 sinon.
5. Rechercher et afficher la plus petite valeur dans un tableau de N entiers.

Exercice 2 (Proposé par A. Djabali)

Méthode de dichotomie

Méthode de la dichotomie ou méthode de la bissection est en mathématiques, un algorithme de recherche d'un zéro d'une fonction qui consiste à répéter des partages d'un intervalle en deux parties puis à sélectionner le sous-intervalle dans lequel existe un zéro de la fonction.

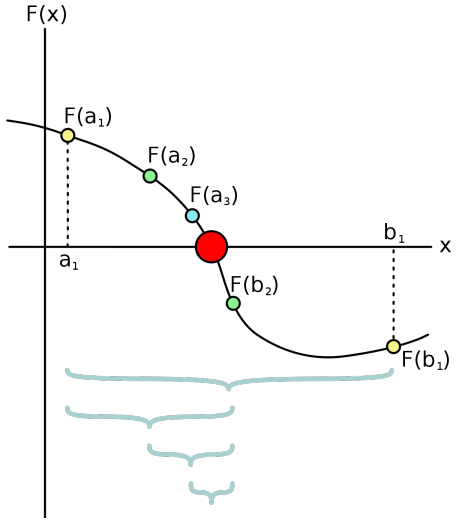


FIGURE 2 – Méthode de dichotomie (image Wikipédia)

Principe : On considère deux nombres réels a et b et une fonction réelle f continue sur l'intervalle $[a, b]$ telle que $f(a)$ et $f(b)$ soient de signes opposés. Supposons que nous voulions résoudre l'équation $f(x) = 0$. D'après le théorème des valeurs intermédiaires, f a au moins un zéro dans l'intervalle $[a, b]$. La méthode de dichotomie consiste à diviser l'intervalle en deux en calculant $m = (a + b)/2$. Il y a maintenant deux possibilités : ou $f(a)$ et $f(m)$ sont de signes contraires, ou $f(m)$ et $f(b)$ sont de signes contraires.

Recherche dichotomique : dans un tableau trié

Le principe est le suivant : comparer l'élément avec la valeur de la case au milieu du tableau ; si les valeurs sont égales, la tâche est accomplie, sinon on recommence dans la moitié du tableau pertinente.

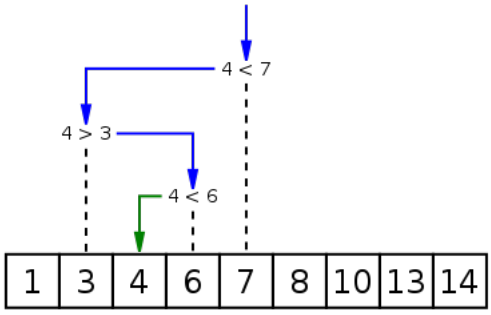


FIGURE 3 – Méthode de recherche dichotomique, recherche de la valeur 4 dans un tableau tiré (image Wikipédia)

L'algorithme est le suivant :
- Trouver la position la plus centrale du tableau.
- Comparer la valeur de cette case à l'élément recherché.
- Si la valeur est égale à l'élément, alors retourner la position,
- sinon reprendre la procédure dans la moitié de tableau pertinente.

La dichotomie, quel est intérêt ?

On suppose que le tableau contient $n = 2^k$ éléments (où k : *entier* ≥ 0).

Question : Combien d'itérations l'algorithme effectuera-t-il au maximum, dans les deux recherches séquentielle et dichotomique ?

- Recherche séquentielle : Pour $k = 100$ (et dans le pire des cas) l'algorithme séquentiel effectuera $2^{100} \sim 10^{30}$ itérations.
- La recherche dichotomique : n'en effectuera que 100.

Durée d'exécution Le temps de réponse de votre programme en dépend.

- Vous disposez d'une machine capable d'effectuer 1 million d'itérations en une seconde,
- L'algorithme séquentiel : prendra 3×10^{16} années de calcul ! (Rappel : $10^{12} = 1000$ *milliards*)
- L'algorithme dichotomique fournira le résultat en moins d'une milliseconde !

Travail demandé

Donnez un algorithme et un organigramme décrivant la recherche dichotomique.

En entrée :

1. X : élément recherché
2. Tab : Tableau trié par ordre croissant
3. N : nombre d'éléments dans le tableau

6 Ce qu'il faut rendre

Un compte rendu comprenant essentiellement :

1. Les 5 organigrammes de l'exercice 1.
2. L'algorithme et l'organigramme décrivant la recherche dichotomique.
3. Compte rendu à envoyer par mail à votre intervenant de TD (Chakib.Benaliouad pour le gr.3, Christian.Thomas pour les gr.2/9/11/12, Denis.Bureau pour le gr.1, et Halim.Djerroud pour les gr.4/5/6/7/8/10)@esiee.fr **avant la fin des 4h de l'atelier.**
4. Le sujet du mail devra obligatoirement commencer par ATL-1209.
5. Le corps du mail devra obligatoirement comporter les NOM Prénom des 2 membres du binôme.
6. Un seul fichier devra être joint/attaché à ce message : un fichier `.zip` contenant l'algorithme de la recherche dichotomique et les 6 organigrammes demandés (au format image ou PDF).

Conseils

Les organigrammes seront dessinés sur ordinateur grâce à une des possibilités indiquées en annexe.

- - collaborer à deux pour le 1.1
- - vérifier que vous pouvez bien enregistrer ce premier organigramme dans un fichier pdf ou jpg
- - se répartir les 1.2 à 1.5 (2 chacun)
- - collaborer à deux pour l'exercice 2

Astuce : la flèche d'affectation ($\leftarrow$) peut être obtenue en tapant *ALT-27*.

Annexe

Logiciels de création d'organigrammes :

- - Le logiciel *DIA* : un logiciel libre (projet GNOME) de création de diagrammes et de schémas divers.
- - *LibreOffice Draw* : est un outil spécial qui permet de créer des graphiques, des diagrammes et d'autres types d'éléments de dessin avec du texte.
- - Solutions de dessin en-ligne : Creately et Gliffy sont des sites web pour créer des schémas.