

Qu'est-ce qu'un organigramme ?(Flowcharts)

Un organigramme est un type de diagramme qui représente visuellement un flux de travail (workflow) ou un processus à l'aide d'un ensemble de symboles ou d'icônes pour indiquer différentes actions/décisions/étapes au sein du processus avec des flèches indiquant la direction du flux. Les organigrammes sont utilisés dans de nombreuses professions différentes pour aider à analyser, concevoir, documenter et/ou gérer un flux de travail ou un processus.

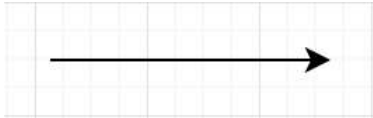
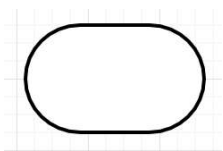
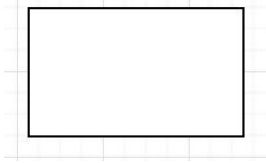
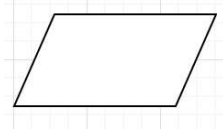
Pourquoi utiliser un organigramme ?

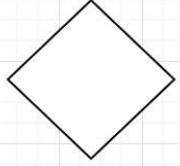
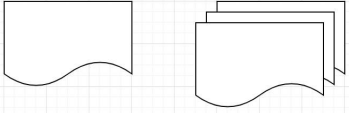
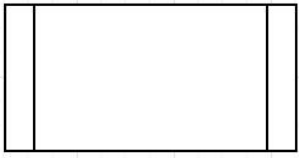
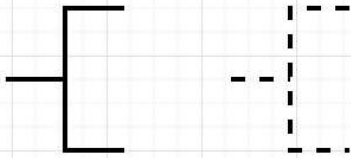
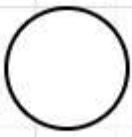
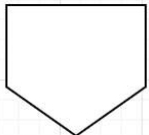
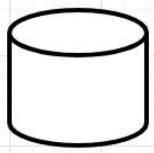
De la même manière que le pseudo-code, les organigrammes offrent une autre façon d'aborder et/ou de comprendre un flux de travail. Un autre avantage est que, lorsqu'ils sont bien faits, les organigrammes peuvent être compris par des non-développeurs et par des développeurs qui ne connaissent pas le langage dans lequel le processus est écrit.

Symboles d'organigramme

Ce type de diagramme est extrêmement polyvalent et est utilisé dans de nombreux domaines professionnels. En raison du nombre d'industries utilisant des organigrammes, de nombreux ensembles de symboles et de styles ont été créés avec une industrie spécifique à l'esprit. Un bon jeu de symboles à utiliser est le jeu de symboles ISO 5807, car ils sont bien connus et souvent utilisés dans l'industrie technologique.

## Quelques symboles ISO 5807 couramment utilisés

Symbol	Name	Description
	Flowline / Arrowhead	Shows the order. Comes from one symbol to another
	Terminal / Terminator	Shows where a process (or sub-process) starts or ends; text is usually "Start", "Begin", or "End"
	Process / Rectangle	Process, action step, or operation; text is usually a verb, examples include: "Edit Video", "Choose Plan", "Set variable to 'hello'"
	Data (I/O)	Inputs to (entering data) and Outputs from (displaying results) a process

	Decision / Conditional	Used to ask a question whose answer determines the route taken from the question. Arrow from bottom is Yes / True, Arrow from side(s) is No / False, (Always label the arrows to make it clear)
	Document (s)	Represents a document or report
	Subroutine / Predefined Process	A process that can be used in the current process but defined elsewhere (should have a separate flowchart for it)
	Annotation (Comment)	Additional info about a step; the line connecting to the step can be solid or dashed
	On-page Connector	Replaces long or confusing lines, use a letter to reference
	Off-page Connector	To connect this process to the next step, which is on another page; use page number for reference
	Data File / Database	Data represented by a cylinder (originally representative of disk drive)

# An Example

Pensons à la tâche Foundation et incluons un exemple d'organigramme :

Les instructions de tâche sont les suivantes :

Écrivez une fonction qui trace une ligne droite dans le terminal.

Prototype : `void print_line(int n);`

Vous ne pouvez utiliser que la fonction `_putchar` pour imprimer

Où `n` est le nombre de fois que le caractère `_` doit être imprimé

La ligne doit se terminer par un `\n`

Si `n` est égal ou inférieur à 0, la fonction ne doit imprimer que `\n`

Prenez un moment pour écrire une solution en pseudo-code.

D'accord. Voici ce que j'ai trouvé.

Définir une variable égale à `n` (`int nCopy`)

Mettre en place une boucle `while` (condition : `nCopy` est supérieur à 0)

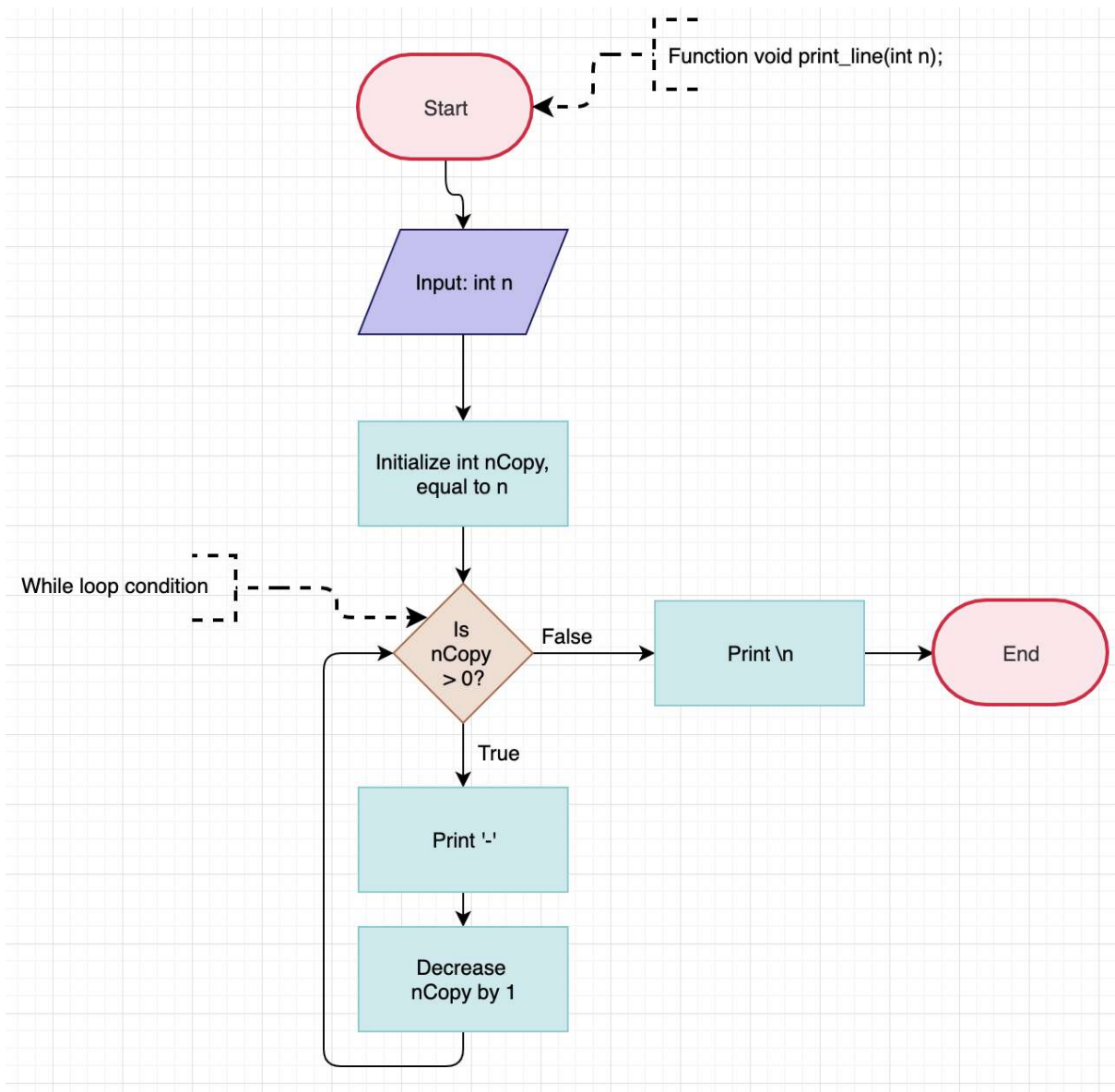
Imprimer -

Diminuer `nCopy` de 1

Imprimer `\n`

Maintenant, en utilisant les symboles ISO 5807 du tableau ci-dessus, dessinez un organigramme pour cette fonction.

Alright, does your flowchart look similar to the diagram below?



Quelques bonnes utilisations des organigrammes :

Créez un organigramme avant de coder une tâche pour aider à créer ou à valider un pseudo-code

Ajoutez à un fichier README.md pour montrer visuellement le fonctionnement de votre projet/processus

Utiliser dans une présentation pour aider le public à comprendre

Votre fonction ou processus ne fonctionne pas comme prévu ? Créez un organigramme de la façon dont il est ACTUELLEMENT. Ensuite, parcourez le graphique avec quelques exemples d'entrées.

Un organigramme peut être de niveau aussi élevé ou aussi détaillé que vous le souhaitez.

Par exemple, vous pouvez décomposer davantage l'organigramme ci-dessus pour inclure ce qui se passe en mémoire. Vous pouvez même créer des organigrammes à partir de vos

expériences quotidiennes non techniques. Le choix d'un film ou d'un endroit où manger pourrait être affiché sous forme d'organigramme !

Un outil utile pour créer des organigrammes est [draw.io](https://draw.io). Vous pouvez utiliser des symboles prédéfinis et vos diagrammes peuvent être enregistrés sur votre Google Drive.

Pour plus d'informations sur les organigrammes et les diagrammes associés, une recherche Google avec simplement "organigramme" peut vous aider à démarrer. ;)

## Embarquement blanc (white boarding)

Dans les projets du monde réel, votre cerveau devrait être capable d'écrire une solution rapide et efficace pour des choses compliquées et vous ne pouvez le faire que lorsque vous pratiquez beaucoup de questions de codage. Comprenez que le langage et les frameworks ne sont que des outils, ils ne vous apprendront pas à résoudre des problèmes. Vous développez des compétences en résolution de problèmes lorsque vous pratiquez beaucoup de questions de codage.

Chaque développeur a ses propres astuces et suit son propre modèle pour résoudre les problèmes de codage, mais lorsqu'il s'agit de nouveaux développeurs, ils ne savent toujours pas par où commencer. Beaucoup d'entre eux comprennent les problèmes, la logique et les bases de la syntaxe, ils comprennent aussi les codes de quelqu'un d'autre et ils peuvent suivre avec eux, mais quand il s'agit de résoudre les questions par eux-mêmes, ils se retrouvent bloqués. Ils ne comprennent pas comment transformer leurs pensées en code même s'ils comprennent la syntaxe ou la logique. Cet article est le résultat d'une recherche de masse sur la manière dont les programmeurs abordent efficacement un problème de codage.

### Étape 1 : comprendre et analyser le problème

Peu importe si vous avez vu la question dans le passé ou non, lisez-la plusieurs fois et comprenez-la complètement. Maintenant, réfléchissez à la question et analysez-la attentivement. Parfois, nous lisons quelques lignes et assumons le reste des choses par nous-mêmes, mais un léger changement dans votre question peut changer beaucoup de choses dans votre code, alors faites attention à cela. Maintenant, prenez un papier et écrivez tout. Qu'est-ce qui est donné (input) et qu'est-ce que vous devez savoir (output) ? En parcourant le problème, vous devez vous poser quelques questions...

Avez-vous bien compris le problème ? Seriez-vous capable d'expliquer cette question à quelqu'un d'autre ? Quelles entrées et combien sont nécessaires ? Quelle serait la sortie pour ces entrées Avez-vous besoin de séparer certains modules ou pièces du problème ? Avez-vous suffisamment d'informations pour résoudre cette question ?

### Étape 2 : Parcourez attentivement les exemples de données et les exemples

Lorsque vous essayez de comprendre le problème, prenez quelques exemples d'entrées et essayez d'analyser la sortie. Les exemples d'entrées vous aideront à mieux comprendre le problème. Vous obtiendrez également des éclaircissements sur le nombre de cas que votre code peut gérer et sur la sortie ou la plage de sortie possible. Considérez quelques entrées ou données simples et analysez la sortie. Considérez certaines entrées complexes et plus importantes et identifiez ce qui sera la sortie et le nombre de cas que vous devez prendre pour le problème. Considérez également les cas extrêmes. Analysez ce que serait la sortie s'il n'y a pas d'entrée ou si vous donnez une entrée invalide.

Étape 3 : Décomposer le problème lorsque vous voyez une question de codage complexe ou importante, au lieu d'avoir peur et de ne pas savoir comment résoudre cette question, décomposer le problème en plus petits morceaux, puis essayez de résoudre chaque partie du problème. Vous trouverez ci-dessous quelques étapes à suivre pour résoudre les questions de codage complexes...