|  |  |
| --- | --- |
| **Chapitre** | **2. Outils et méthodes d’analyse et de description des systèmes** |
| **Objectif général de formation** | * identifier les éléments influents d’un système, * décoder son organisation, * utiliser un modèle de comportement pour prédire ou valider ses performances. |
| **Paragraphe** | 2.3 Approche comportementale |
| **Sous paragraphe** | 2.3.6 Comportements informationnels des systèmes |
| **Connaissances** | Modèles algorithmiques : structures algorithmiques élémentaires (boucles, conditions, transitions conditionnelles). Variables |
| **Niveau d’enseignement** | Première Terminale |
| **Niveau taxonomique** | **3.** Le contenu est relatif à la **maîtrise d’outils d’étude ou d’action** : utiliser, manipuler des règles ou des ensembles de règles (algorithme), des principes, des démarches formalisées en vue d’un résultat à atteindre. |
| **Commentaire** | *Activités pratiques liées à la mise en œuvre d’un produit industriel ou d’un système permettant l’application des différents modèles de description de l’information (en statique et en dynamique) et la caractérisation des entrées-sorties de ses différents constituants.*  *Les modèles de comportement sont étudiés autour d’un point de fonctionnement. Au niveau de l’expression de l’information on se limite aux grandeurs statistiques usuelles (moyenne et écart type)* |
| **Liens** |  |

## Remarque préliminaire

Cet item « nécessite une étroite coordination avec la progression pédagogique en mathématiques ».  
Voir [1] page 7/7.

## Structures algorithmiques

D’après [2].

Les codages en langage C sont donnés à titre d’exemple. L’étude de ce langage ne figurant pas dans le tronc commun Sti2D.

La notation algorithmique utilise un **pseudo-code**. Il n’existe pas de normalisation du pseudo-code, mais des **conventions d’usage**.

### Structure linéaire (séquence) .

On exécute successivement une suite d'action dans l'ordre de leur énoncé.

Algorigramme

Début

Action 1

Action 2

Fin

**Exemple en langage C**

**{**

*Action 1* ;

*Action 2* ;

**}**

**Notation algorithmique**

**Début**

*Action 1*

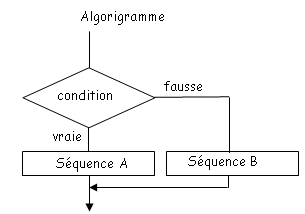
*Action 2*

**Fin**

### Structures alternatives

#### Structure SI…ALORS…SINON…

Cette structure offre le choix entre deux séquences s'excluant mutuellement.

****

**Exemple en langage C**

**if (** condition **)**

**{** Séquence A **; }**

**else**

**{** Séquence B **; }**

**Notation algorithmique**

**Si** condition **Alors**

Séquence A

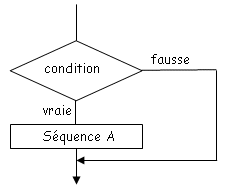
**Sinon**

Séquence B

**Fin Si**

**Remarque :**

La structure peut se limiter à SI…ALORS, si la condition est vrai on exécute la séquence A si elle est fausse on quitte la structure sans exécuter de séquence.

****

**Notation algorithmique**

**Si** *condition* **Alors**

*Séquence A*

**Fin Si**

**Exemple en langage C**

**if** ( *condition* )

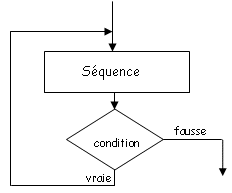
**{** *Séquence A* ;**}**

### Structures répétitives (ou itératives).

#### Structure FAIRE…TANT QUE

La séquence est exécutée au moins une fois, elle est répétée tant qu'elle est vraie.

**Algorigramme :**

****

**Exemple en langage C**

**do**

**{** *Séquence* **; }**

**while (**condition vraie**) ;**

**Notation algorithmique**

**Faire**

*Séquence*

**Tant que** *condition vraie*

### Structure TANT QUE…FAIRE

On teste d'abord la condition. La séquence est exécutée tant que la condition est vraie.

******

**Exemple en langage C**

**while (***condition***)**

**{** Séquence **; }**

**Notation algorithmique**

**Tant que** *condition vraie*

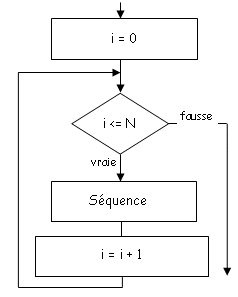
*Séquence*

**Fin tant que**

### Structure POUR…FAIRE

On effectue un nombre d'itérations donné

**Algorigramme :**



**Exemple en langage C**

**for(** *i=0*; *i<=N*; *i++* **)**

**{** *Séquence* ; **}**

**Notation algorithmique**

**Pour** i = 0 **à** N

**Faire** Séquence

**Fin Pour**

## Variables

En informatique, les variables associent un nom (le symbole) à une valeur ou un objet [4].

Les variables peuvent être **typées**, mais ceci est très dépendant de l’environnement de programmation utilisé.

En **programmation orientée objet**, le type d’une variable est une **classe**, et la variable désigne un **objet** qui est une **instance** de cette **classe**.

## Etudier l’algorithmique sans programmer

Exécuter des algorithmes sans langage de programmation peut sembler une gageure.

C’est pourtant ce que permet Algobox [3].

En réalité, la programmation s’effectue à l’aide d’un pseudo-code dont les instructions sont entrées en cliquant dans l’interface graphique.

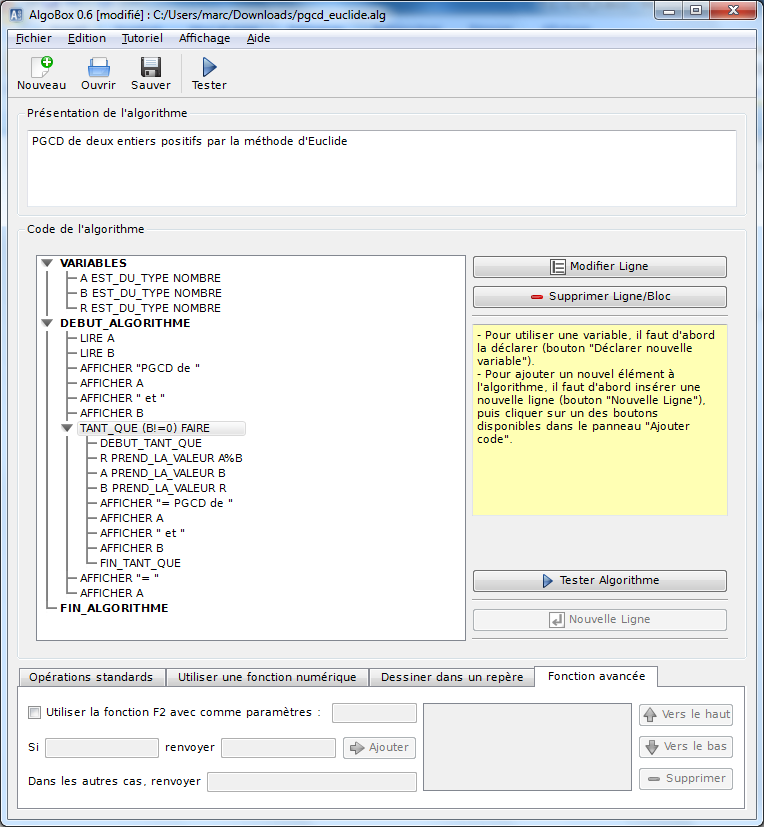


Figure : algorithme du PGCD dans Algobox

## Références

1. Extrait du B.O. spécial n°3 du 17 mars 2011 : Mathématiques - classe de 1ère des séries STI2D et STL :  
   <http://euler.ac-versailles.fr/webMathematica/textes_officiels/officiel_2012/maths_STI2D_STL_171037.pdf>
2. Cours d’algorithmique S-SI – Hoareau - Lycée Louis Payen  
   <http://louispayen.apinc.org/cours/algorithmiques1.doc>
3. Algobox : Logiciel pédagogique d'aide à la création et à l'exécution d'algorithmes  
   <http://www.xm1math.net/algobox/>
4. Wikipedia – variables (informatique) : <http://fr.wikipedia.org/wiki/Variable_%28informatique%29>