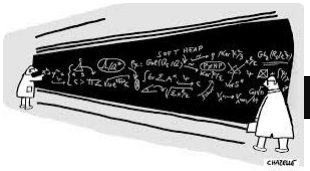


Structure Algorithmique

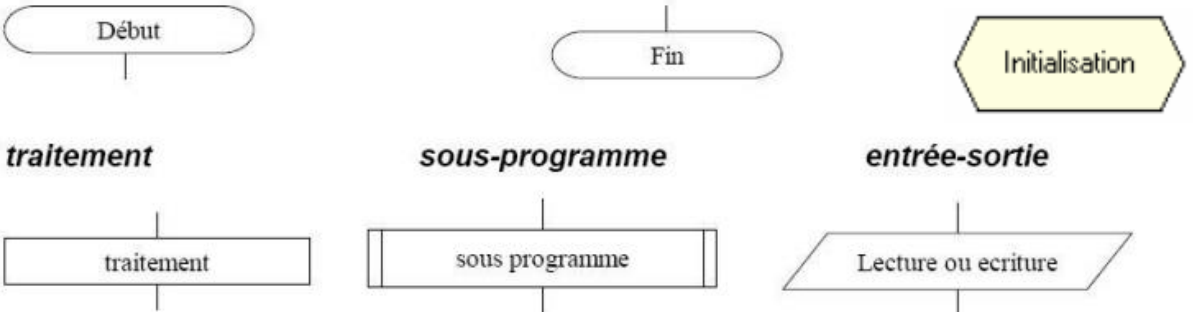
1. Qu'est-ce qu'un Algorithme ?

Un Algorithme est une suite ordonnée de directives composées d'actions et de décisions qu'il faut exécuter en séquence suivant un enchaînement strict pour accomplir une tâche quelconque. Il ne tient pas compte des technologies mises en œuvre (indépendant d'un langage informatique).

2. Qu'est-ce qu'un Algorigramme ?

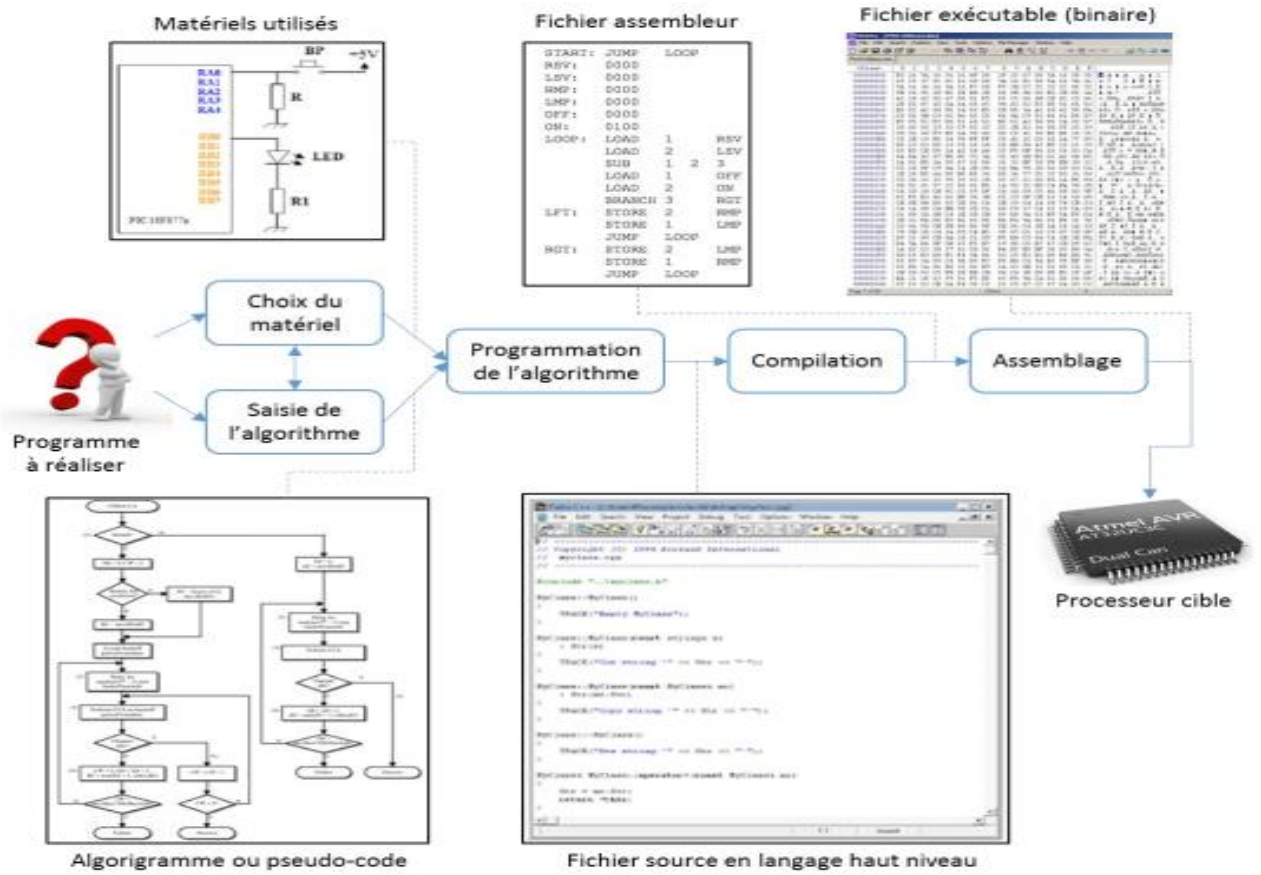
L'organigramme est une représentation graphique normalisée utilisée pour analyser ou décoder un problème..

Ils comportent un début et une fin, des liaisons orientés, des blocs "symbole" : traitement (affectations de variables, calculs..), gestions des entrées et des sorties, sous-programmes (fonctions).



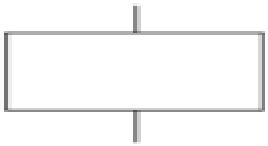
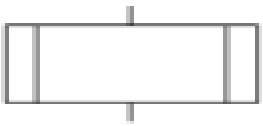
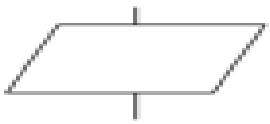
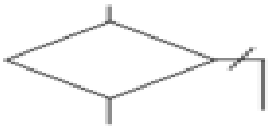
3. La chaîne de développement d'un programme

Cette chaîne désigne l'ensemble des étapes nécessaires à la construction d'un fichier exécutable (binaire).



4. Les principaux symboles d'un algorigramme

Les principaux symboles normalisés pour la représentation graphique d'un algorithme sont les suivants :

Symbole	Désignation	Symbole	Désignation
Symboles de traitement		**Symboles auxiliaires**	
	**Symbole général** Opération ou groupe d'opérations sur des données, instructions pour laquelle il n'existe aucune symbole normalisé		**Renvoi** Symbole utilisé deux fois pour assurer la continuité lorsqu'une partie de ligne de liaison n'est pas représentée.
	**Sous-programme** Portion de programme considérée comme une simple opération		**Début, fin, interruption** Début, fin ou interruption d'un algorigramme
	**Entrée-Sortie** Mise à disposition d'une information à traiter ou enregistrement d'une information à traitée		**Commentaire** Symbole utilisé pour donner des indications sur les opérations effectuées
Symbole de test			Les différents symboles sont reliés entre eux par des lignes de liaisons
	**Branchement** Exploitation de conditions variables impliquant un choix parmi plusieurs		

Sens conventionnel des liaisons

Le sens général des lignes de liaison doit être :

- De haut en bas
- De gauche à droite

Lorsque le sens général ne peut pas être respecté, des pointes de flèches à cheval sur la ligne indiquent le sens utilisé.

Règles de construction

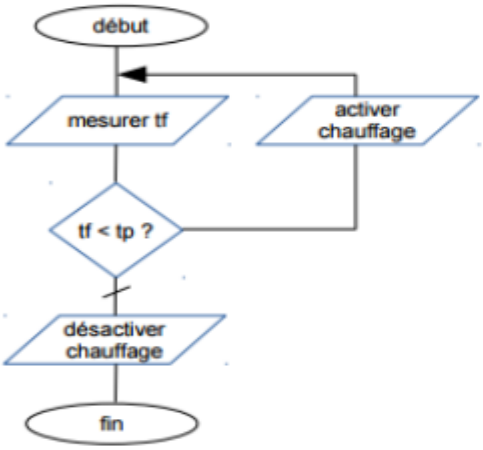
- Centrer l'algorigramme sur une feuille
- Respecter le sens conventionnel des liaisons
- Les lignes de liaison entre symboles ne doivent pas en principe se couper (utiliser un symbole de renvoi)
- Une ligne de liaison doit toujours arriver sur le haut et au centre d'un symbole.
- Les commentaires sont à placer de préférence à droite, et les renvois de branchement à gauche.

5. Le pseudo-code

Le pseudo-code est une écriture conventionnelle permettant de représenter un algorithme sous forme textuel. Ce mode de représentation consiste à exprimer en langage naturel, mais selon une disposition particulière et des mots choisis, les différentes opérations constituant l'algorithme, conformément au code donné dans le tableau qui suit :

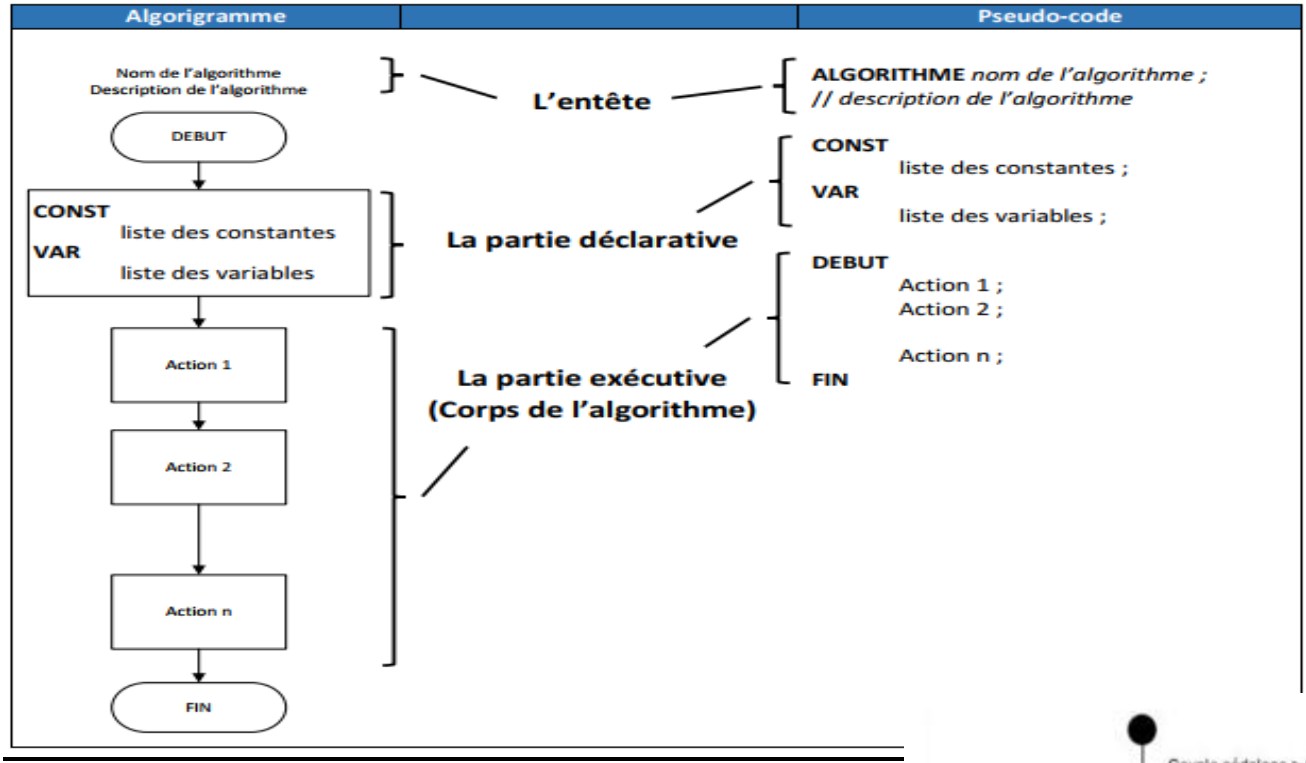
Mots et symboles du pseudo-code	Opérations réalisées
ALGORITHME	Nomme l'algorithme
DEBUT	Début de l'algorithme
FIN	Fin de l'algorithme
FAIRE	Exécution d'une opération
LIRE	Acquisition ou chargement d'une donnée
ECRIRE	Edition ou sauvegarde d'un résultat
AFFICHER	Affiche sur l'écran
ALLER A	Branchement inconditionnel
SI…ALORS…SINON	Branchement inconditionnel
SELON CAS…AUTREMENT	Branchement conditionnel généralisé
TANT QUE…FAIRE…	} Répétition conditionnelle
RÉPÉTER…JUSQU'À…	
POUR…DE…À…	Répétition contrôlée
//	Commentaire

Exemple d'algorigramme et de pseudo-code équivalent :

Algorigramme	Pseudo-code
	*début* *mesurer tf* *tant que tf < tp* activer chauffage *mesurer tf* *fin tant que* *désactiver chauffage* *fin*

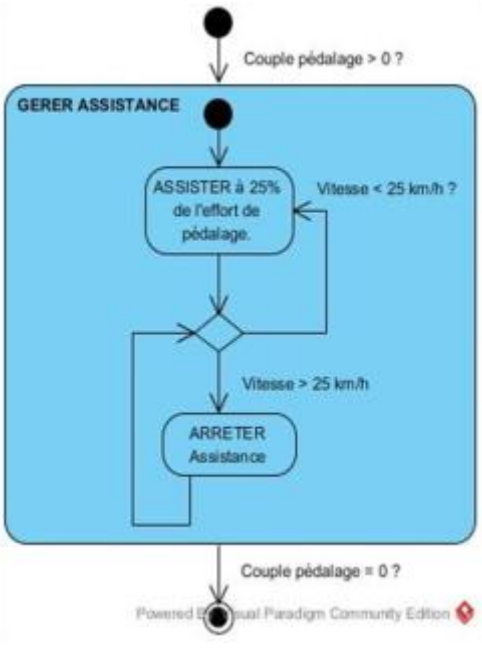
6. L'organisation d'un algorithme

Un algorithme est organisé de la manière suivante :



7. DIAGRAMME D'ETAT SysML

Nom	Symbole	Description
Etape initiale		
Etats		Mettre à jour les sorties de la carte. Verbe d'action à l'infinitif. Exemple : ASSISTER pédalage
Nœuds de décisions		Doit comporter deux sorties : une condition et son complément.
Transitions		Doivent s'écrire sous forme d'une condition binaire, qui peut prendre la valeur 0 ou 1. *Exemple : Vitesse > 25 km.h^{-1} ?*
Etape fin		Marque la fin d'une activité

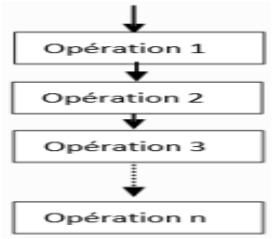


8. Les structures algorithmiques de base

8.1 La structure linéaire

La structure linéaire se caractérise par une suite d'actions à exécuter successivement dans l'ordre énoncé.

Algorigramme



Pseudo-code

```
FAIRE
        Opération 1 ;
        Opération 2 ;
        Opération 3 ;
        |
        |
        Opération n ;
FIN FAIRE
```

Exemples de structure linéaire dans différents langages :

Flowcode

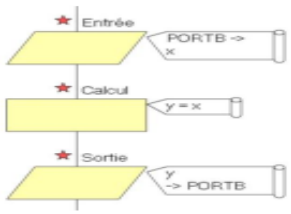
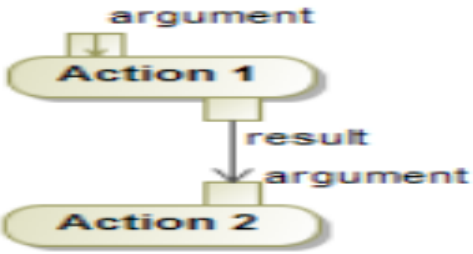


Diagramme d'états



Arduino

```
digitalWrite(13, HIGH);   // set the LED on
delay(1000);              // wait for a second
digitalWrite(13, LOW);    // set the LED off
delay(1000);              // wait for a second
```

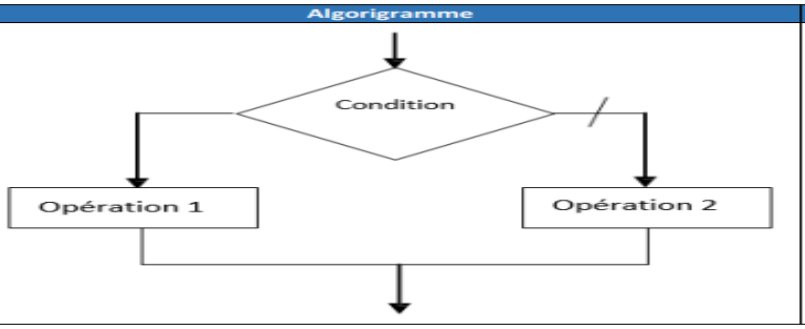
Python

```
a = 3
print(a)
a = a + 3
b = a - 2
print("a =", a, "et b =", b)
```

7.2 La structure alternative « SI…ALORS…SINON »

La structure alternative n'offre que deux issues possibles à la poursuite de l'algorithme en s'excluant mutuellement.

Algorigramme



Pseudo-code

```
SI Condition ALORS
        FAIRE Opération 1 ;
SINON
        FAIRE Opération 2 ;
FIN SI
```

Exemples de structure « SI…ALORS…SINON » dans différents langages :

Flowcode

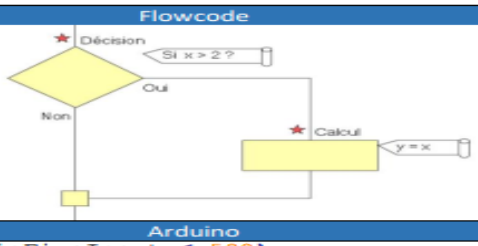
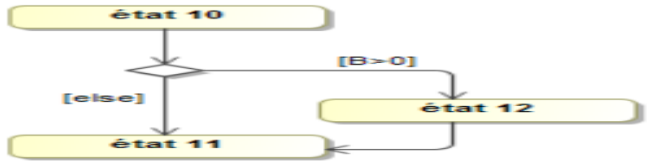


Diagramme d'états



Arduino

```
if (pinFiveInput < 500)
{
  // action A
}
else
{
  // action B
}
```

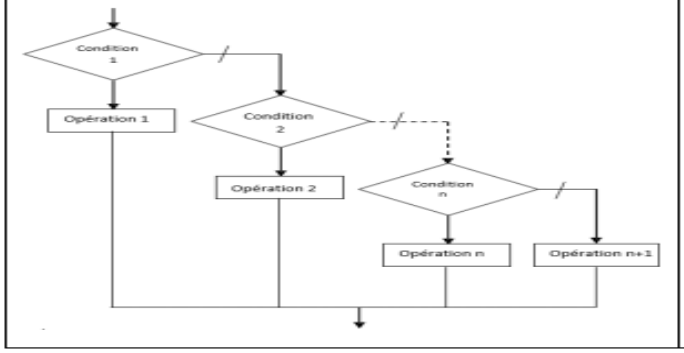
Python

```
a = 5
if a > 0:
    print("a est supérieur à 0.")
else:
    print("a est inférieur ou égal à 0.")
```

7.3 La structure de choix « SELON »

La Structure de choix permet, en fonction de plusieurs conditions de type booléen, d'effectuer des actions différentes suivant les valeurs que peut prendre une même variable.

Algorigramme

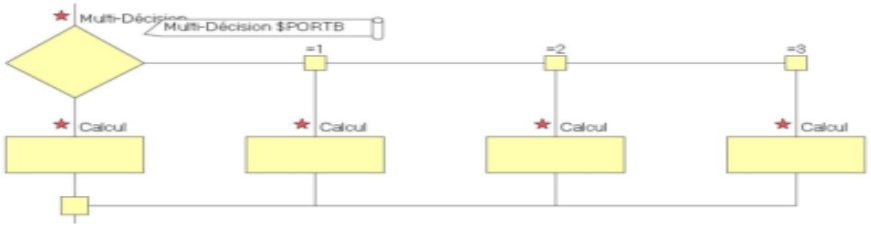


Pseudo-code

```
SELON cas
        Cas 1 : FAIRE Opération 1 ;
        Cas 2 : FAIRE Opération 2 ;
        |
        |
        Cas n : FAIRE Opération n ;
AUTREMENT
        FAIRE Opération n+1 ;
FIN SELON ;
```

Exemples de structure « SELON » dans différents langages :

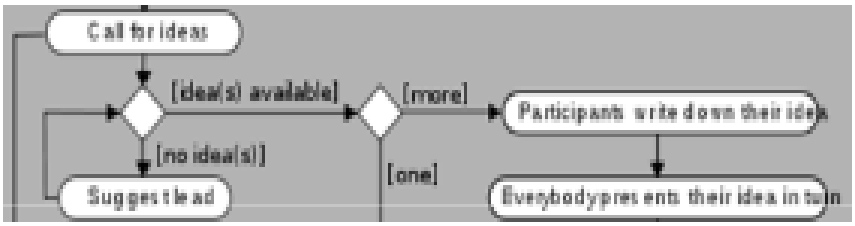
Flowcode



Arduino

```
switch (var) {
    case 1:
      //do something 1
      break;
    case 2:
      //do something 2
      break;
    default:
      // if nothing else
matches, do the default
      // default is
optional
    break;
  }
```

Diagramme d'états



Python

```
switch(n) {
  case 0:
    printf("0");
    break;
  case 1:
    printf("1");
    break;
  default:
    printf("other");
  break;
}
```

7.4 Les structures itératives

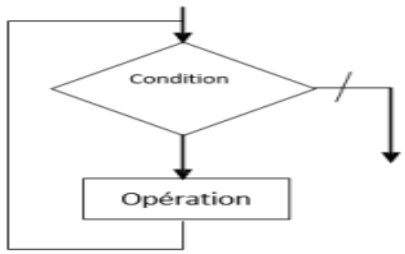
Les structures itératives répètent l'exécution d'une opération ou d'un traitement.

7.4.1 La structure « TANT QUE…FAIRE »

Dans cette structure, on commence par tester la condition ; si elle est vraie, le traitement est exécuté.

L'ACTION PEUT NE JAMAIS ETRE EXECUTEE

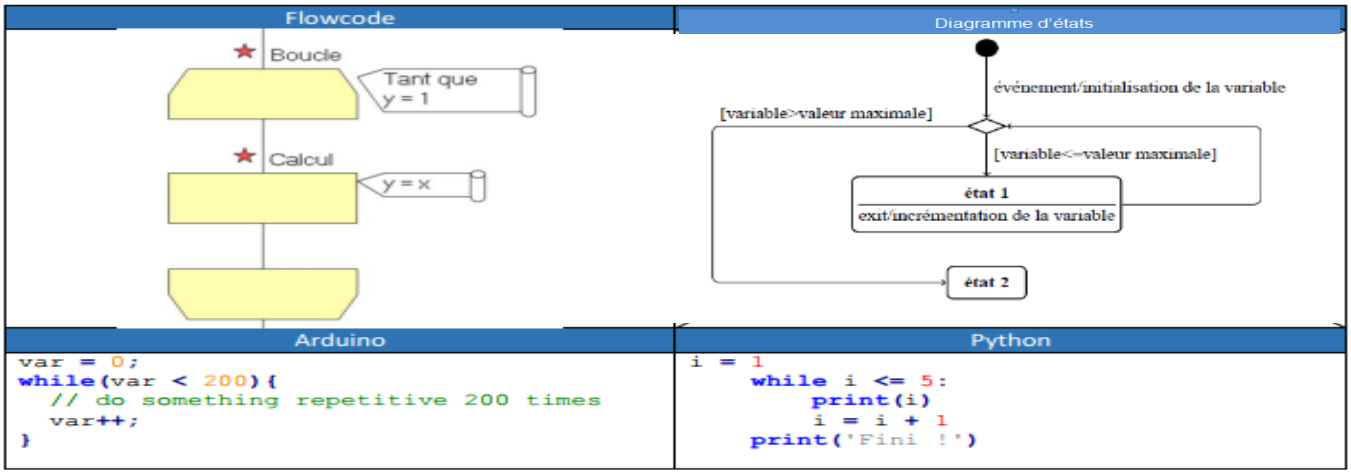
Algorigramme



Pseudo-code

```
TANT QUE condition
        FAIRE Opération ;
FIN TANT QUE
```

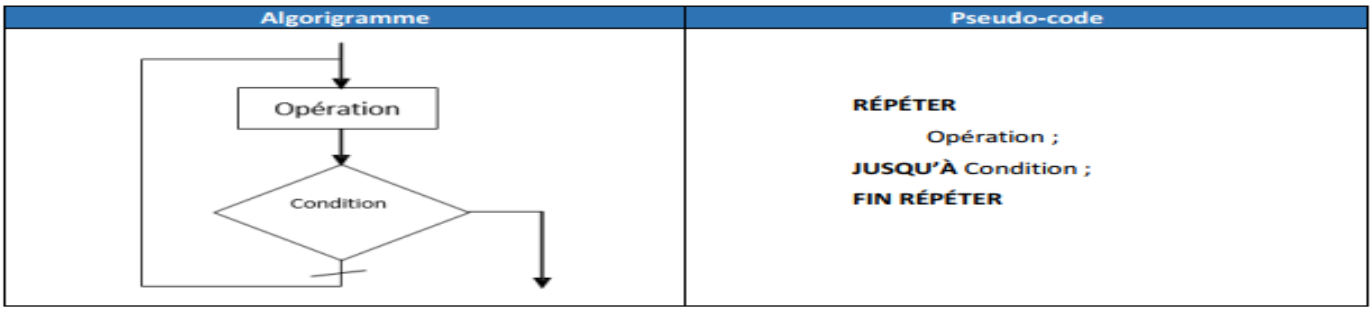
Exemples de structure « TANT QUE…FAIRE » dans différents langages :



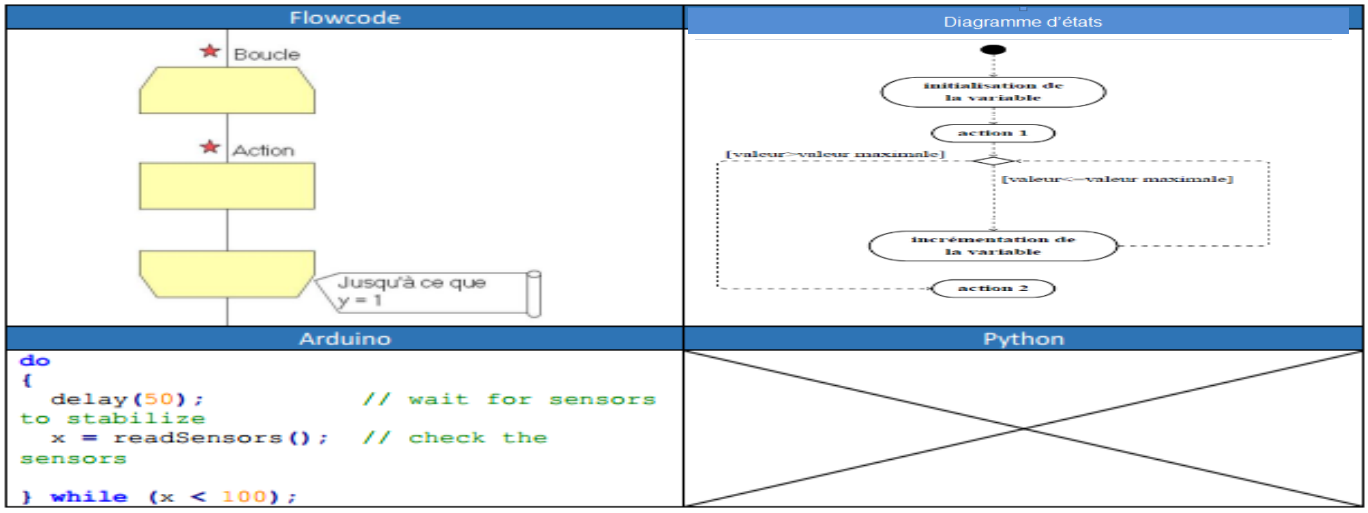
7.4.2 La structure « RÉPÉTER…JUSQU'À »

Dans cette structure, le traitement est exécuté une première fois puis sa répétition se poursuit jusqu'à ce que la condition soit vérifiée.

L'ACTION EST EXECUTEE AU MOINS UNE FOIS

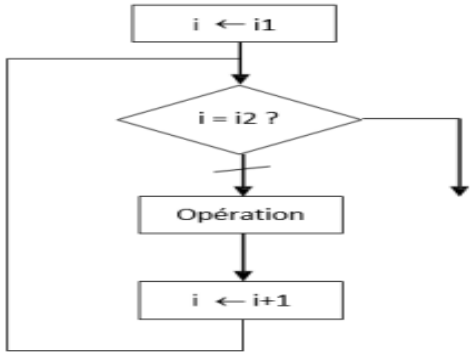


Exemples de structure « RÉPÉTER…JUSQU'À » dans différents langages :

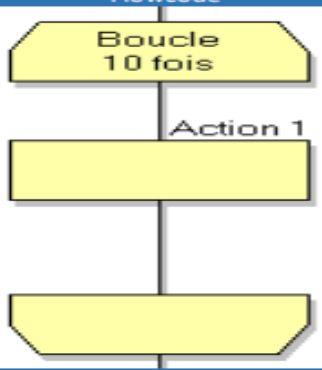
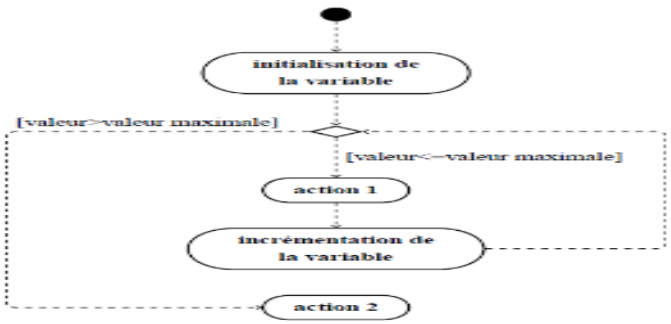


7.4.3 La structure « POUR…DE…À… »

Dans cette structure, le traitement est exécuté un nombre limité de fois, défini au préalable.

Algorigramme	Pseudo-code
	**POUR** i **DE** i1 **À** i2 **FAIRE** Opération ; **FIN POUR**

Exemples de structure « POUR…DE…A » dans différents langages :

Flowcode	Diagramme d'états
	

Arduino

```
for (int i=0; i <= 255; i++){
    analogWrite(PWMpin, i);
    delay(10);
}
```

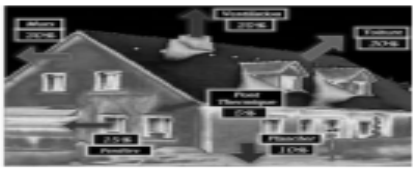
Python

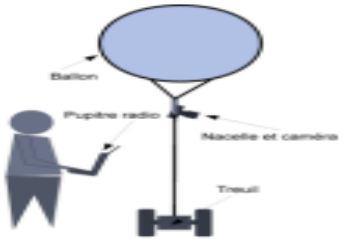
```
S = 0
for i in range(1, 31):    # pour i allant de 1 à 30
      S = S + i
print(S)
```

9. 8. EXERCICES

8.1 THERMOGRAPHIE PAR BALLON CAPTIF

Un ballon captif embarque une caméra thermique permettant d'effectuer des bilans thermiques.
Le pilotage peut être manuel (voir ci-contre), ou automatique.





- le treuil est actionné le sens de la montée. La sortie associée au treuil (**MotTreuil = 1**) ; une lumière Orange s'allume. (**LedOrange = 1**).
- un codeur mesure le nombre de tours déroulés et stocke cette valeur dans **NbTours** ;
- lorsque un nombre de tours **NbToursHaut** de câble est déroulée, le treuil s'arrête. (**MotTreuil = 0**).
- la Led orange s'éteint (**LedOrange = 0**) ; une led verte s'allume (**Ledverte = 1**) ; la caméra filme : (**Cam = 1**);
- au bout de 5 mn, la caméra arrête de filmer : (**Cam = 0=** ; La Led verte s'éteint et la led orange se rallume.
- le treuil enroule le câble **MotTreuil** est réglé à **-1**.
- lorsque le nombre de tours déroulés redevient nulle, le treuil s'arrête ; la led orange s'éteint.

Question 1 : Rédiger l'algorithme du fonctionnement.
Question 2 : Décrire le fonctionnement à l'aide d'un diagramme d'état.

8.2 PISTE DE SKI INTERIEURE.

Le snowhall permet la pratique des sports de glisse sur neige artificielle, en intérieur et toute l'année.
La température dans le bâtiment est contrôlée par 5 frigorifères qui ventilent et refroidissent l'air. À chaque canon à neige est associé un frigorifère.

Les caractéristiques globales peuvent conduire le système à absorber une intensité électrique supérieure à celle disponible.
- si l'intensité s'approche de l'intensité limite, l'augmentation de puissance du groupe froid est interdite.
- si l'intensité dépasse la valeur limite on force la diminution de la puissance.

Un sous-programme de contrôle définit deux variables booléennes :
- **interdictionAugmentation**, vaut **VRAI** si la puissance du groupe froid ne doit plus être augmentée
- **forçageDiminution**, vaut **VRAI** si la puissance consommée par le groupe doit être diminuée.

Tconsigne est la température de consigne ; **commandeGroupe** est le % de commande pour un frigorifère.

Question 1 : Compléter l'algorigramme ci-dessous.
Question 2 : Décrire cet algorigramme au moyen d'un diagramme d'activité (d'état).

```
Toutes les 30 s Faire
        T ← acquerirTempératureEau()
        Si T < (Tconsigne – 0,2) .....
                Si commandeGroupe ≥ 10 Alors
                        commandeGroupe ← commandeGroupe - 10
                Sinon
                        commandeGroupe ← 0
                Fin Si
        Si T > (Tconsigne + 0,2) .....
                Si commandeGroupe ≤ 90 Alors
                        commandeGroupe ← commandeGroupe + 10
                Sinon
                        commandeGroupe ← 100
                Fin Si
Fin
```

8.3 BALISE MARITIME

La balise maritime est équipée à son sommet d'un système d'éclairage qui sert à guider les bateaux dans la nuit. Le signal lumineux émis par la balise est intermittent et possède un rythme propre qui permet de l'identifier. Le rythme est donné par la répartition des temps de lumière (L) et d'obscurité (O) :

Une cellule photoélectrique permet de réaliser la détection du jour et de la nuit : la balise va s'allumer automatiquement la nuit et s'éteindre le jour.

La cellule produit une information logique sur C0. C0 = '1'lorsqu'il fait nuit. On utilisera une variable NUIT pour mémoriser cette information.

La lampe qui s'allume et s'éteint selon le rythme défini sera reliée à D0.
Dessiner l'algorigramme et le diagramme d'état qui traduit ce fonctionnement.