TECHNOLOGIE *Ce que je dois retenir*	L'ALGORITHME ORGANIGRAMME OU LOGIGRAMME	CYCLE 4

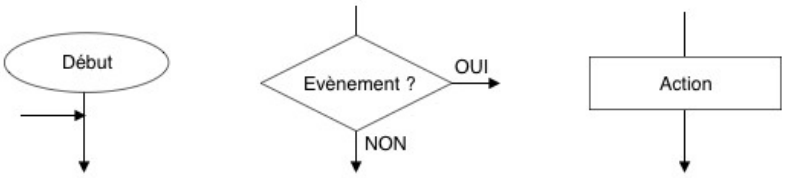
CT 1.3 – CT 2.5 – CT 2.7 DIC 1.5	Imaginer des solutions pour produire des objets et des éléments de programmes informatiques en réponse au besoin.
CT 3.1 OTSCIS 2.1	Exprimer sa pensée à l'aide d'outils de description adaptés : croquis, schémas, graphes, diagrammes, tableaux.
CT 4.2 – CT 5.5 IP 2.3	Écrire un programme dans lequel des actions sont déclenchées par des événements extérieurs.

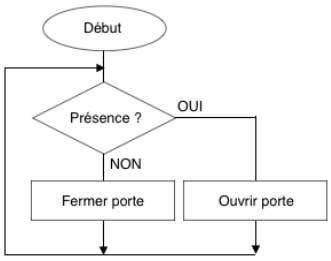
Symboles de base

Un algorithme est une suite d'instructions précises et structurées qui décrit la manière dont on résout un problème.
Cette description peut être textuelle (si, alors, sinon, tant que …) ou graphique (appelé également organigramme ou logigramme).

Début
Si Présence
Alors ouvrir porte
Sinon fermer porte
Retour au début

Dans ce cas des normes d'écritures sont à respecter :

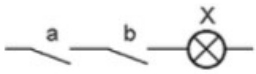


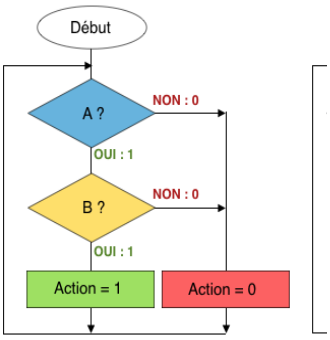


Fonctions ET et OU

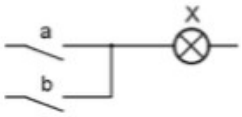
L'utilisation des fonctions ET et OU sont essentielles pour présenter correctement une solution.

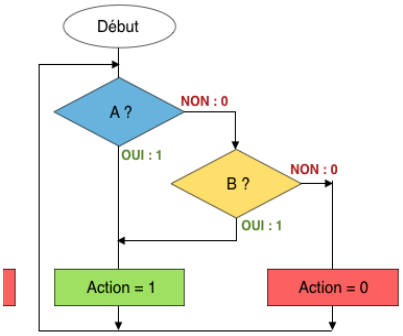
Fonction ET		
A ?	B ?	Sortie
0	0	0
0	1	0
1	0	0
1	1	1





Fonction OU		
A ?	B ?	Sortie
0	0	0
0	1	1
1	0	1
1	1	1



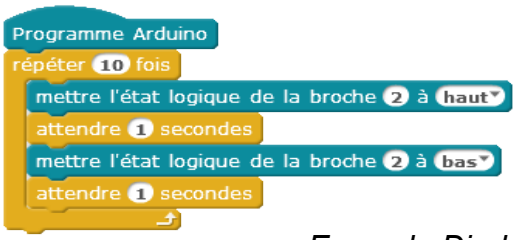


Boucles *TANT QUE, JUSQU'À, REPETER …*

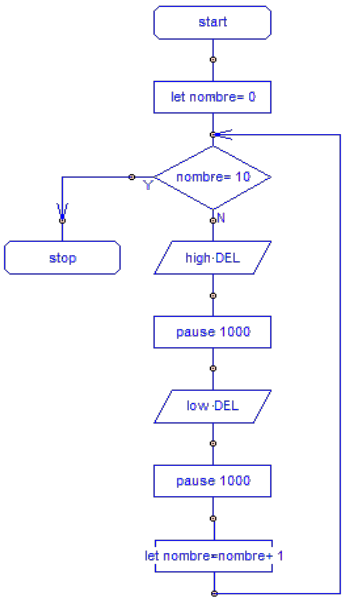
Lorsque des instructions sont répétées, on utilise des **boucles** pour optimiser le programme.

Exemple de boucles : TANT QUE, JUSQU'À, REPETER …

Il est possible d'imbriquer plusieurs boucles les unes dans les autres pour répondre au problème.



Exemple Diode clignote 10 fois



Déclenchement d'une action par un événement, instructions conditionnelles

L'enchaînement des opérations et le **déclenchement d'actions** se fait toujours par un **événement** :

- interne au programme (début programme, variable, ...)

- externe au programme (capteur, touche du clavier, ...)

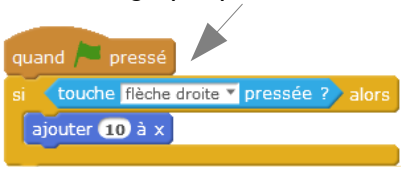
Condition dans un Algorithme

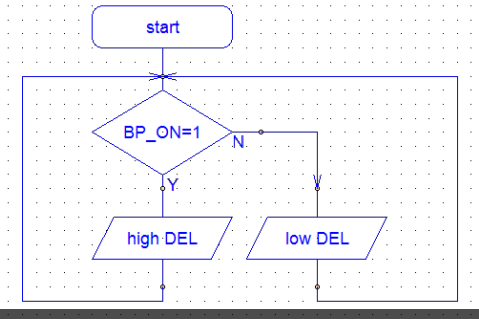
SI ...

ALORS ...

SINON ...

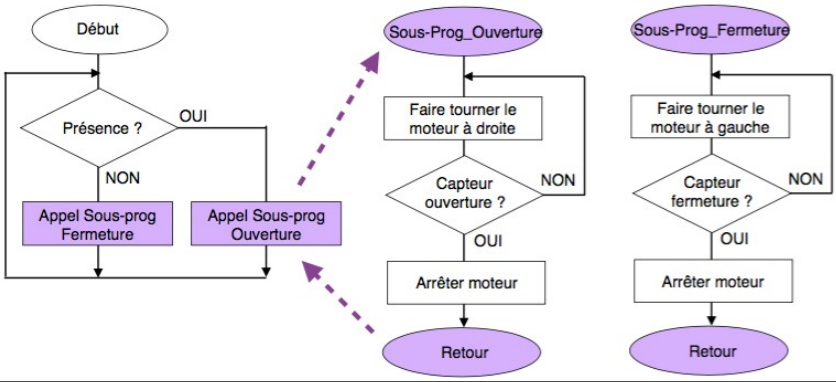
Condition en langage graphique





Algorithme et gestion des sous-problèmes

L'utilisation des sous-problèmes est idéale pour une meilleure lisibilité, pour alléger l'algorithme lors de succession d'actions identiques, pour faciliter le travail en collaboration, pour facilité une recherche d'erreur (test individuel des sous-problèmes).



Sous-Programme

Les **sous-programmes** sont des modules de programmation indépendants répondant à des **sous-problèmes** du programme principal.

Exemple 1: Dialogue entre 2 personnages

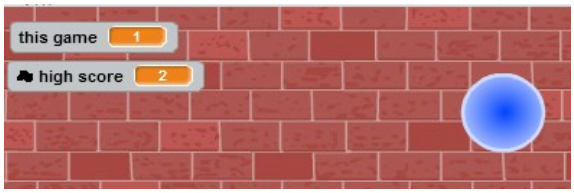
Sous-problème 1 : faire parler Chat

Sous-problème 2 : faire parler Pico

Variable informatique

Une **variable** est une donnée (information) associée à un nom. Elle est mémorisée et elle peut changer dans le temps, lors de l'exécution du programme.

Exemple : timer



Exemple : score et meilleur score pour un jeu