STI Technologie	**Fiche d'activité n°1 de la séquence n°8**	**Durée : 1h30**	CYCLE 4 — 5 4 3
	Intitulée : Données entrées et produites par le programme du volet à lames orientables		
Thématique abordée :	**T12-La programmation d'une nouvelle fonctionnalité**		

Compétences et connaissances travaillées dans l'activité

Thème abordé : Structure, fonctionnement, comportement : des objets et des systèmes techniques à comprendre

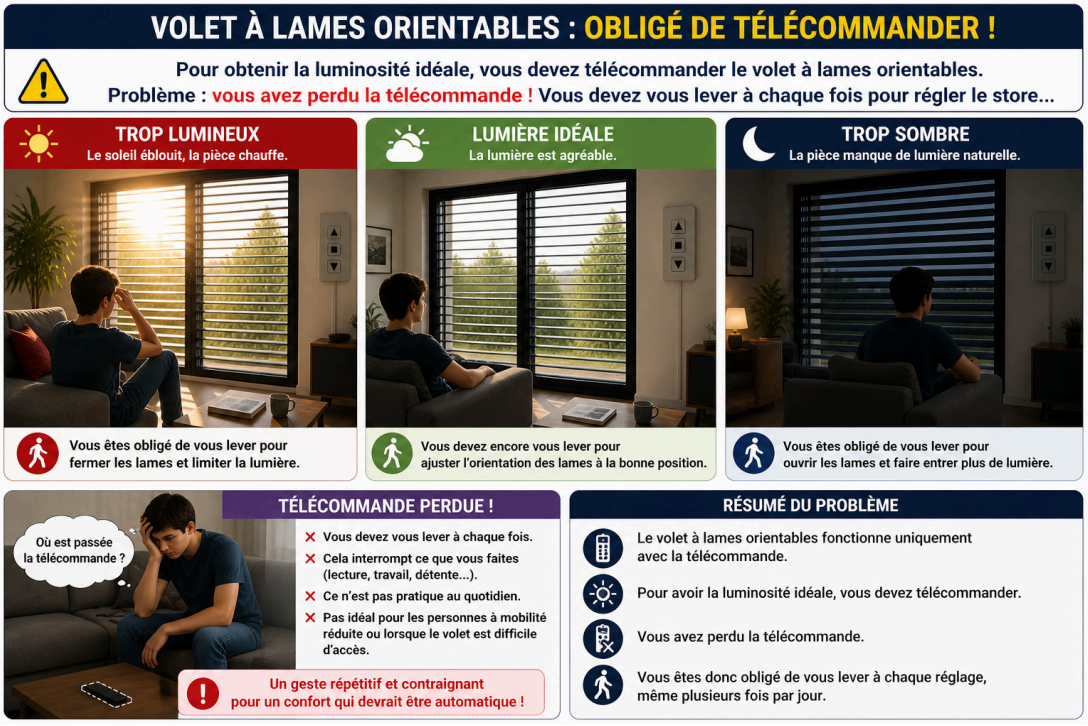
Attendu de fin de cycle : SFC3-Comprendre et modifier un programme associé à une fonctionnalité d'un objet ou d'un système technique

Thématique : T12-La programmation d'une nouvelle fonctionnalité

Objectifs d'apprentissages (Compétences détaillées)	**Connaissance**
SFC 31-Déterminer les données utilisées et produites par un programme associé à une fonctionnalité en vue de le modifier SFC 32- Programmer un algorithme lié à une nouvelle fonctionnalité	SFC3g événement SFC3h déclenchement d'une séquence d'instructions par un événement SFC3i entrées ou sorties d'un programme (données issues par exemple de capteurs IHM et sorties pouvant être en lien avec un actionneur, fichiers) ;

Critères d'apprentissages	**N1** – je sais ce qu'est un événement dans un programme et une donnée d'un programme, **N2** – et je sais caractériser l'événement et les entrées liés à un programme , **N3** – et je sais déterminer les données utilisées et produites par la partie du programme associé à une fonctionnalité d'un OST et Programmer un algorithme lié à une nouvelle fonctionnalité **N4** – et je sais proposer les données produites pour une nouvelle fonctionnalité.

Situation déclenchante de l'activité (2')



Mes constats, mes observations (5')

Mon problème à résoudre (3')

Mes idées pour le résoudre (5')

Activités *(niveaux 1 et 2)* **(20')**

NOM : Groupe : Classe :
Prénom : Rôle dans le groupe : Date :

Dans un premier temps, il faut comprendre comment un programme fait pour savoir quelle est la bonne action à réaliser au bon moment.

En visionnant les ressources vidéos associées aux fiches connaissances «SFC3g-événement » et “SFC3i-Entrées et sorties d’un programme” ou en scannant le Qrcode ci-contre tu dois :

N1-N2 Identifier et caractériser les événements et les Entrées / Sorties d’un programme

N1a-Définir un événement dans un programme :

N1b-Définir une entrée et une sortie dans un programme:

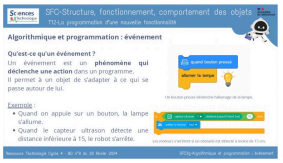
N1c- Quel est le rôle des Entrées / Sorties dans un programme ?

N2a-Quels sont les 3 principaux types d’évènements, en indiquant pour chacun, le déclencheur entre parenthèses ?

N2b-Quels sont les 3 principaux types d’entrées et de sorties dans un programme ?

Types d’entrées	Types de sorties

SFC3g-événement



SFC3i-Entrées et sorties d’un programme



Activités *(niveaux 3 et 4)* **(40’)**

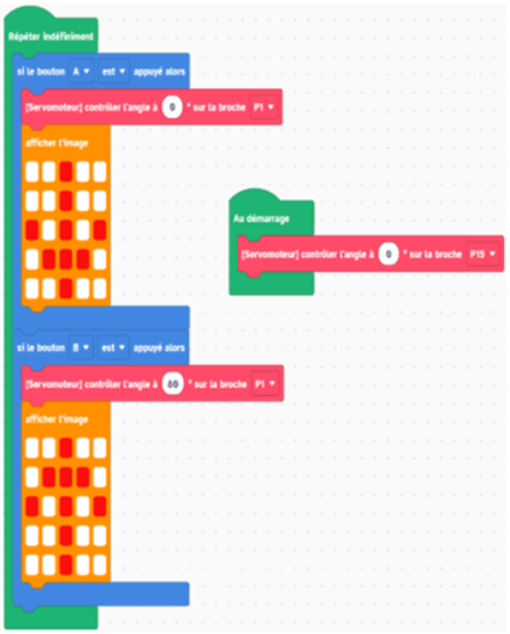
NOM : Groupe : Classe :
Prénom : Rôle dans le groupe : Date :

Maintenant que tu connais les notions d'événements et de données entrant et sortant d'un programme, tu vas identifier les données fournies et produites par le programme de la maquette de volet à disposition dans son fonctionnement classique et dans son automatisation en fonction de la luminosité dans la pièce.

N3-Identifier les données fournies et produites par le programme du volet.

N3a- Indique dans le tableau ci-dessous les 2 entrées et les 2 sorties du programme en précisant le type de données. Ensuite, relie-les à l'instruction correspondante dans la représentation du programme.

Entrées	.....
	.....
	.....
Sorties	.....
	.....
	.....
	.....



N3b-Nous voulons automatiser la maquette mise à disposition : nous souhaitons que les lames s'orientent automatiquement en fonction de la luminosité dans la pièce.



Quand la pièce est trop sombre, les lames s'ouvrent pour que la luminosité de la pièce soit idéale (valeur définie par l'utilisateur)

Quand la pièce est trop lumineuse, les lames se ferment pour que la luminosité de la pièce soit idéale (valeur définie par l'utilisateur)



A tout moment, le système indiquera à l'utilisateur sur un écran la valeur de la luminosité

A partir de la vidéo ressources youtu.be/2aZmVb2nQBw , présentant l'automatisation de lames orientables sur une pergola bioclimatique, tu dois proposer une solution pour automatiser la maquette de volet en fonction de la luminosité.

Dans le cadre de l'ouverture/ fermeture automatique du volet en fonction de la luminosité,

Réponds aux questions ci-dessous

-Quel capteur faut-il utiliser ?

Quel est l'événement ?

Vidéo ressource :
-Pergola connectée



NOM :
Prénom :

Groupe :
Rôle dans le groupe :

Classe :
Date :

indique la donnée fournie (Entrée) en précisant son type :

Indique les données produites (Sorties) en indiquant leur type Quelle est l'entrée du programme ?

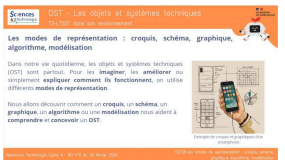
N3c-Rédaction de l'algorithme

Après avoir revu si nécessaire la vidéo ressources de la connaissance OST2B- les différents modes de représentation (à partir de 2 min 09), et à partir des indications ci-dessous, complète l'algorithme

luminosité > 600 Luminosité entre 300 et 600 luminosité <300



OST2B- les différents modes de représentation



Si luminosité inférieure à 300

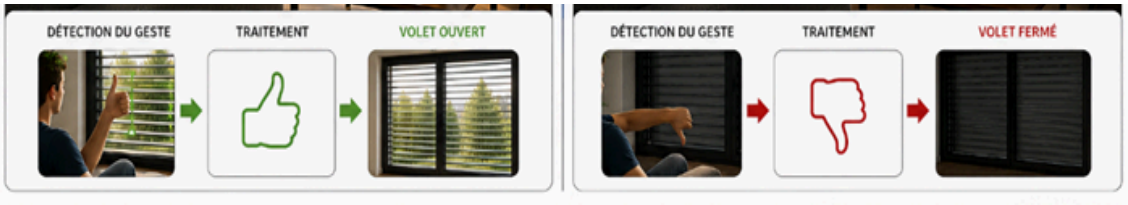
Alors ouvrir lames orientables

Si luminosité

Alors mettre lames orientables en position intermédiaire

N4-Comment piloter le volet par un geste ?

situation déclenchante : On veut pouvoir actionner à distance le volet (fermeture ou ouverture) grâce à une caméra avec intelligence artificielle intégrée qui pourra détecter un geste préprogrammé par l'utilisateur.



N4a- Indique les entrées et sorties correspondantes liées au programme à concevoir.

NOM :
Prénom :

Groupe :
Rôle dans le groupe :

Classe :
Date :

Entrée : Sortie **:** **N4+** propose l'algorithme correspondant	

Ma synthèse (5')

<u>Questions :</u>

Dans les activités réalisées, qu'est-ce qui permet au programme du volet de savoir qu'il doit agir ?

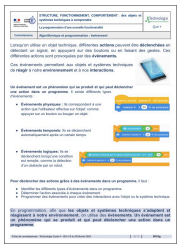
Quelles sont les données d'entrée et de sortie utilisées pour automatiser le volet en fonction de la luminosité ?

Comment peut-on ajouter une nouvelle fonctionnalité à un objet technique déjà programmé ?

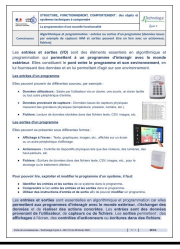
<u>Synthèse générale</u>

Fiches de connaissances (5')

SFC3g événement



SFC3i entrées ou sorties d'un programme



SFC3h déclenchement d'une séquence d'instructions par un événement



<u>NOM</u> :
<u>Prénom</u> :

<u>Groupe</u> :
<u>Rôle dans le groupe</u> :

<u>Classe</u> :
<u>Date</u> :

Quiz auto-correctifs (5')

SFC3g événement	SFC3i entrées ou sorties d'un programme
SFC3h déclenchement d'une séquence d'instructions par un événement	

➢ ***Je développe mes compétences CPS :*** **S3 – Savoir demander de l'aide : " Je travaille en autonomie"**

☐ ***N1*** **—.Je ne demande pas même en cas de difficulté**	☐ ***N2*** **— .Je demande de l'aide sans clarté sur la demande** .	☐ ***N3*** **— .Je demande clairement de l'aide en expliquant ma difficulté**	☐ ***N4*** **— Je demande de l'aide au bon moment et efficacement**

NOM :
Prénom :

Groupe :
Rôle dans le groupe :

Classe :
Date :

STI Technologie	**Fiche d'activité n°1 de la séquence n°8**	**Durée : 1h30**	CYCLE 4 5 4 3
	Intitulée : Données entrées et produites par le programme du volet à lames orientables		
Thématique abordée :	**T12-La programmation d'une nouvelle fonctionnalité**		

Compétences et connaissances travaillées dans l'activité

Thème abordé : Structure, fonctionnement, comportement : des objets et des systèmes techniques à comprendre

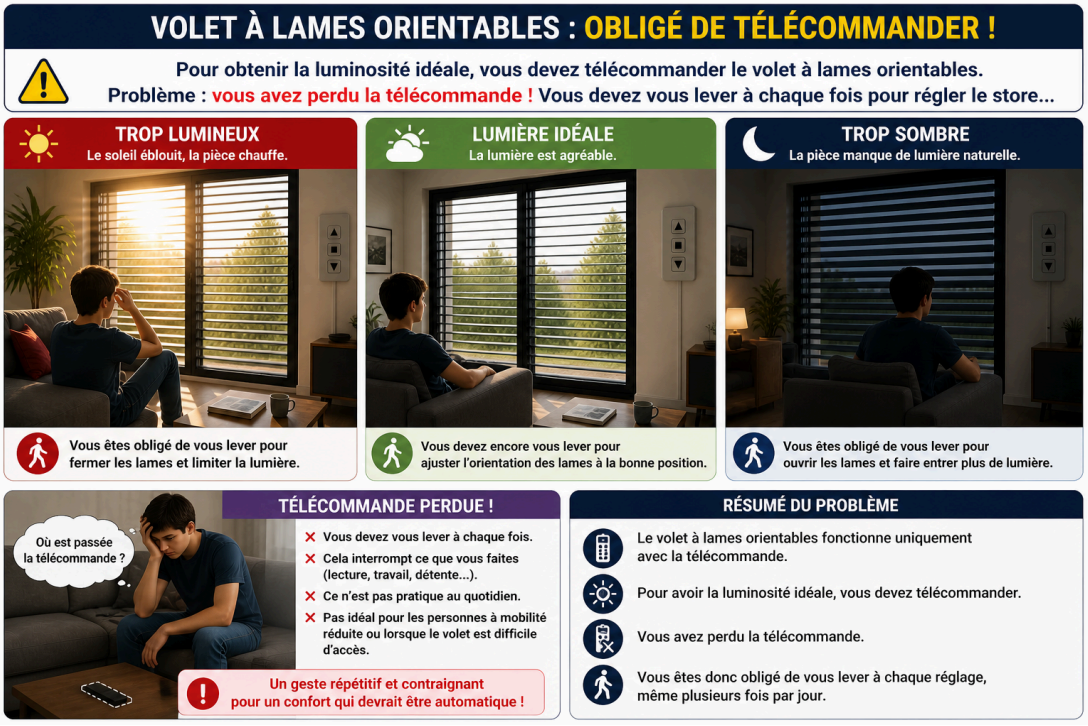
Attendu de fin de cycle : SFC3-Comprendre et modifier un programme associé à une fonctionnalité d'un objet ou d'un système technique

Thématique : T12-La programmation d'une nouvelle fonctionnalité

Objectifs d'apprentissages (Compétences détaillées)	**Connaissance**
SFC 31-Déterminer les données utilisées et produites par un programme associé à une fonctionnalité en vue de le modifier SFC 32- Programmer un algorithme lié à une nouvelle fonctionnalité	SFC3g événement SFC3h déclenchement d'une séquence d'instructions par un événement SFC3i entrées ou sorties d'un programme (données issues par exemple de capteurs IHM et sorties pouvant être en lien avec un actionneur, fichiers) ;

Critères d'apprentissages	**N1** – je sais ce qu'est un événement dans un programme et une donnée d'un programme, **N2** – et je sais caractériser l'événement et les entrées liés à un programme , **N3** – et je sais déterminer les données utilisées et produites par la partie du programme associé à une fonctionnalité d'un OST et Programmer un algorithme lié à une nouvelle fonctionnalité **N4** – et je sais proposer les données produites pour une nouvelle fonctionnalité.

Situation déclenchante de l'activité (2')



Mes constats, mes observations (5')

Une personne est obligée de se lever plusieurs fois par jour pour ouvrir ou fermer son volet car il a perdu la télécommande

Mon problème à résoudre (3')

Comment avoir la bonne luminosité dans la pièce sans avoir à se déplacer ?

Mes idées pour le résoudre (5')

Automatiser le volet pour que les lames s'inclinent automatiquement
Créer un nouveau système pour remplacer la télécommande

Activités *(niveaux 1 et 2)* **(20')**

NOM : Groupe : Classe :
Prénom : Rôle dans le groupe : Date :

Dans un premier temps, il faut comprendre comment un programme fait pour savoir quelle est la bonne action à réaliser au bon moment.

En visionnant les ressources vidéos associées aux fiches connaissances «SFC3g-événement » et “SFC3i-Entrées et sorties d’un programme” ou en scannant le Qrcode ci-contre tu dois :

N1-N2 Identifier et caractériser les événements et les Entrées / Sorties d’un programme

N1a-Définir un événement dans un programme :

Un événement est un phénomène qui déclenche une action dans un programme

N1b-Définir une entrée et une sortie dans un programme:

Entrée : information ou donnée reçue par un programme pour fonctionner

Sortie : Action ou information envoyée par un programme

N1c- Quel est le rôle des Entrées / Sorties dans un programme ?

Les E/S permettent au programme d’interagir avec son monde extérieur

N2a-Quels sont les 3 principaux types d’évènements, en indiquant pour chacun, le déclencheur entre parenthèses ?

-Physique (action physique)

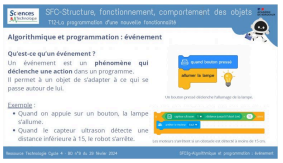
-Temporel (temps écoulé)

- Logique (condition remplie dans le programme)

N2b-Quels sont les 3 principaux types d’entrées et de sorties dans un programme ?

Types d’entrées	Types de sorties
Données utilisateur	Affichage à l’écran
Données des capteurs	Contrôle d’actionneur
Lecture de Données dans un fichier	Ecriture de données dans un fichier

SFC3g-événement



SFC3i-Entrées et sorties d’un programme



Activités *(niveaux 3 et 4)* **(40’)**

Maintenant que tu connais les notions d'événements et de données entrant et sortant d’un programme, tu vas identifier les données fournies et produites par le programme de la maquette de volet à disposition dans son fonctionnement classique et dans son automatisation en fonction de la luminosité dans la pièce.

N3-Identifier les données fournies et produites par le programme du volet.

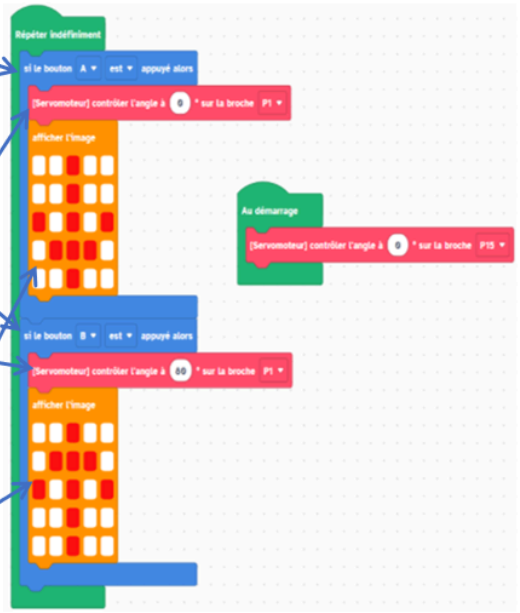
NOM :
Prénom :

Groupe :
Rôle dans le groupe :

Classe :
Date :

N3a- Indique dans le tableau ci-dessous les 2 entrées et les 2 sorties du programme en précisant le type de données. Ensuite, relie-les à l'instruction correspondante dans la représentation du programme.

Entrées	Appui sur bouton A (Donnée utilisateur)
	Appui sur bouton B (Donnée utilisateur)
Sorties	Action moteur (Contrôle actionneur)
	Indication sens moteur (Affichage sur écran)



N3b-Nous voulons automatiser la maquette mise à disposition : nous souhaitons que les lames s'orientent automatiquement en fonction de la luminosité dans la pièce.



Quand la pièce est trop sombre, les lames s'ouvrent pour que la luminosité de la pièce soit idéale (valeur définie par l'utilisateur)

Quand la pièce est trop lumineuse, les lames se ferment pour que la luminosité de la pièce soit idéale (valeur définie par l'utilisateur)



A tout moment, le système indiquera à l'utilisateur sur un écran la valeur de la luminosité

A partir de la vidéo ressources youtu.be/2aZmVb2nQBw , présentant l'automatisation de lames orientables sur une pergola bioclimatique, tu dois proposer une solution pour automatiser la maquette de volet en fonction de la luminosité.

Dans le cadre de l'ouverture/ fermeture automatique du volet en fonction de la luminosité,

Réponds aux questions ci-dessous

-Quel capteur faut-il utiliser ? un capteur de luminosité ou capteur d'ensoleillement

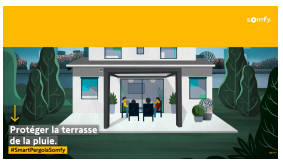
Quel est l'événement ? luminosité dans la pièce

indique la donnée fournie (Entrée) en précisant son type :

Quantité de lumière (données des capteurs)

Indique les données produites (Sorties) en indiquant leur type Quelle est l'entrée du programme ?

Vidéo ressource :
-Pergola connectée



NOM :
Prénom :

Groupe :
Rôle dans le groupe :

Classe :
Date :

Ouverture et fermeture (contrôle d'actionneur)

Indication de la valeur de luminosité (Affichage à l'écran)

N3c-Rédaction de l'algorithme

Après avoir revu si nécessaire la vidéo ressources de la connaissance OST2B- les différents modes de représentation (à partir de 2 min 09), et à partir des indications ci-dessous, complète l'algorithme

luminosité > 600 Luminosité entre 300 et 600 luminosité <300



Si luminosité inférieure à 300

Alors ouvrir lames orientables

Si luminosité supérieure à 300 ou inférieure à 600

Alors mettre lames orientables en position intermédiaire

Si luminosité supérieure à 600

Alors fermer lames orientables

N4-Comment piloter le volet par un geste ?

situation déclenchante : On veut pouvoir actionner à distance le volet (fermeture ou ouverture) grâce à une caméra avec intelligence artificielle intégrée qui pourra détecter un geste préprogrammé par l'utilisateur.



N4a- Indique les entrées et sorties correspondantes liées au programme à concevoir.

Entrée : Geste de commande (données utilisateur)

Sortie **:** Ouverture et fermeture (contrôle d'actionneur)

N4+ propose l'algorithme correspondant

OST2B- les différents modes de représentation



NOM :
Prénom :

Groupe :
Rôle dans le groupe :

Classe :
Date :

Si luminosité inférieure à 300 ou geste “ouvrir” alors ouvrir lames orientables Si luminosité supérieure à 600 ou geste “fermer” alors fermer lames orientables	

Ma synthèse (5’)

Questions :

Dans les activités réalisées, qu'est-ce qui permet au programme du volet de savoir qu'il doit agir ?

Un événement est un phénomène qui se produit et qui déclenche une action dans un programme. Il peut être provoqué par l'utilisateur, par un capteur ou par une condition détectée par le système.

Quelles sont les données d'entrée et de sortie utilisées pour automatiser le volet en fonction de la luminosité ?

Les entrées sont les informations reçues par le programme (boutons, gestes, capteurs...). Les sorties sont les actions réalisées par le programme (commande d'un moteur, affichage sur un écran, activation d'un actionneur...).

Comment peut-on ajouter une nouvelle fonctionnalité à un objet technique déjà programmé ?

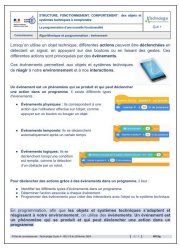
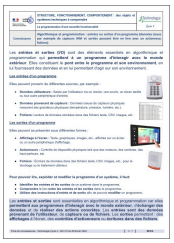
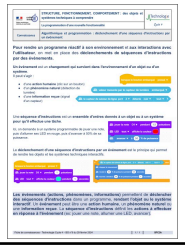
Pour ajouter une nouvelle fonctionnalité, il faut identifier les nouvelles données nécessaires, ajouter les capteurs ou dispositifs adaptés puis modifier le programme afin qu'il traite ces nouvelles informations et commande les actionneurs de manière appropriée.

Synthèse générale

Un programme permet à un objet technique d'interagir avec son environnement grâce à des événements, des données d'entrée et de sortie. Les capteurs (ex : détecteur de lumière) permettent d’identifier un événement en envoyant des données d’entrée au programme. Le programme va alors envoyer des données de sorties qui vont commander une action (ex : fermer le volet).

Pour faire évoluer un système programmé et lui ajouter une nouvelle fonctionnalité, il est nécessaire d'ajouter les données utiles puis de modifier le programme afin qu'il réagisse correctement aux nouvelles informations reçues.

Fiches de connaissances (5’)

SFC3g événement 	SFC3i entrées ou sorties d’un programme 
SFC3h déclenchement d’une séquence d’instructions par un événement 	

Quiz auto-correctifs (5’)

SFC3g événement	SFC3i entrées ou sorties d’un programme

NOM :
Prénom :

Groupe :
Rôle dans le groupe :

Classe :
Date :

SFC3h déclenchement d'une séquence d'instructions par un événement	

➢ ***Je développe mes compétences CPS :*** **S3 – Savoir demander de l'aide : " Je travaille en autonomie"**

☐ ***N1*** **—.Je ne demande pas même en cas de difficulté**	☐ ***N2*** **— .Je demande de l'aide sans clarté sur la demande** .	☐ ***N3*** **— .Je demande clairement de l'aide en expliquant ma difficulté**	☐ ***N4*** **— Je demande de l'aide au bon moment et efficacement**

NOM :
Prénom :

Groupe :
Rôle dans le groupe :

Classe :
Date :

STI Technologie	**Fiche d'activité n°2 de la séquence n°8**	**Durée : 2h30**	CYCLE 4 5 4 3
	Intitulée : Automatiser le volet		
Thématique abordée :	**T12-La programmation d'une nouvelle fonctionnalité**		

Compétences et connaissances travaillées dans l'activité

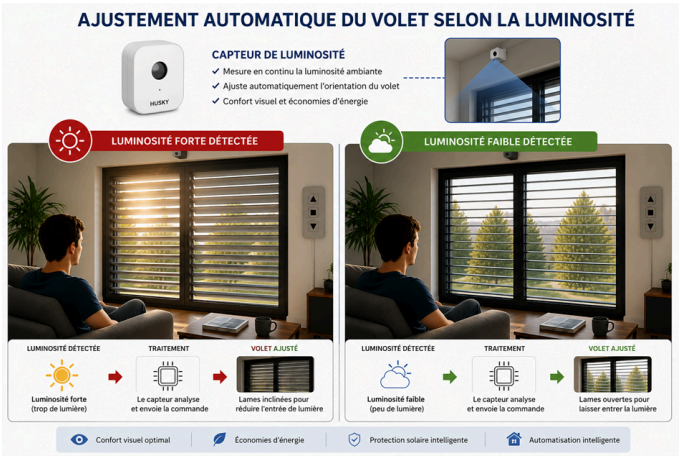
Thème abordé : Structure, fonctionnement, comportement : des objets et des systèmes techniques à comprendre
Attendu de fin de cycle :SFC3-Comprendre et modifier un programme associé à une fonctionnalité d'un objet ou d'un système technique
Thématique : T12-La programmation d'une nouvelle fonctionnalité

Objectifs d'apprentissages (Compétences détaillées)	**Connaissance**
SFC 33- Modifier et tester le programme associé à une nouvelle fonctionnalité d'un OST	SFC3a instruction d'affectation, variable (type mot, nombre et booléen) SFC3k Lien entre la programmation graphique par bloc et la programmation textuelle (fin de 3e)

Critères d'apprentissages	**N1** – je sais identifier les éléments constituant un programme, **N2** – je sais compléter la structure interne de l'OST et connecter les composants **N3** – Je sais modifier et tester le programme associé à une nouvelle fonctionnalité d'un OST **N4** – Je sais associer les blocs de texte en langage Python aux blocs du programme initial qui remplissent les mêmes rôles.

Situation déclenchante de l'activité (2')

Nous voulons passer du fonctionnement précédent observé à ceux présentés ci-dessous :





Mes constats, mes observations (5')

Mon problème à résoudre (3')

Mes idées pour le résoudre (5')

NOM :
Prénom :

Groupe :
Rôle dans le groupe :

Classe :
Date :

Activités *(niveaux 1 et 2)* **(20')**

Avant de modifier le comportement du volet, il est nécessaire de comprendre comment est construit un programme et comment les différents composants communiquent avec la carte de programmation. Dans cette première partie, tu vas identifier les principaux blocs utilisés dans un programme puis préparer le système en réalisant les connexions nécessaires entre les capteurs, les actionneurs et la carte micro:bit.

N1-Commandes et fonction de programmation

N1a-Indique le bloc correspondant à sa définition

Pour réaliser le programme, tu vas utiliser les différents blocs de programmation

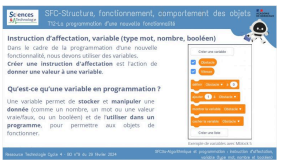
variable / opérateur mathématique / opérateur logique / instruction conditionnelle / instruction itérative(ou boucle) / séquences d'instructions simples

Complète le tableau ci-dessous
(si nécessaire, relis les fiches connaissances SFC 3a, SFC3b, SFC3c SFC3d)

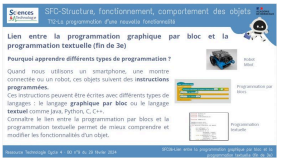
Rep	Nom du bloc	Définition
1	instruction itérative (ou boucle)	Permet de répéter un bloc d'instructions un nombre défini de fois ou jusqu'à ce qu'une condition soit remplie.
2		Permet de stocker et de manipuler des données (provenant de capteurs par exemple).
3		Réalise des calculs mathématiques et permet d'obtenir des valeurs numériques.
4		Combine ou modifie des conditions, produisant ainsi un résultat qui est soit « Vrai » soit « Faux » (ET / OU / NON).
5		Permet de contrôler le déroulement d'un programme en fonction de conditions (vrai/faux).
6		Suite d'instructions exécutées les unes après les autres sans conditions particulières.

Ressources vidéo et QR Code

SFC3a instruction d'affectation, variable (type mot, nombre et booléen)



SFC3k Lien entre la programmation graphique par bloc et la programmation textuelle



Fiches connaissances déjà vues en 5ème et 4ème :

SFC3a instruction d'affectation, variable (type mot, nombre et booléen)

SFC3b opérateurs arithmétiques et logiques (ET, OU, NON)

SFC3c instruction conditionnelle

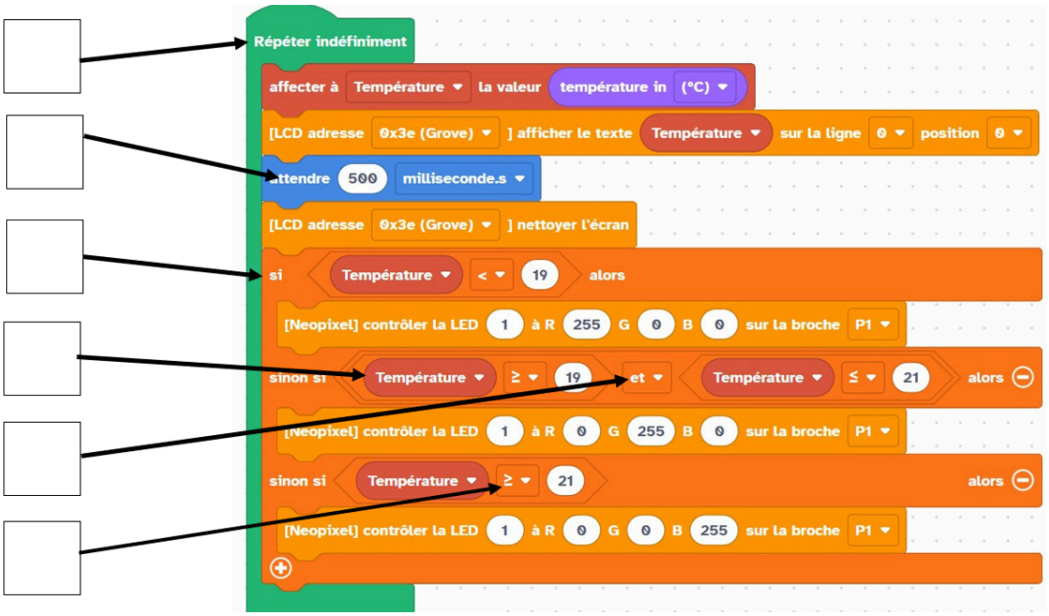
SFC3d instructions itératives ;

NOM :
Prénom :

Groupe :
Rôle dans le groupe :

Classe :
Date :

N1b-Identifie les blocs sur l'exemple de programme ci-dessous en indiquant le repère correspondant



N2-Associer les composants à la carte de programmation.

Il faut maintenant connecter les différents composants à ajouter et permettre au programme de recevoir les données entrantes. Pour cela :

- Il faut câbler chaque composant sur le bon port de l'interface,
- Il faut associer le port au bloc de programmation

N2a-Identification du port de connexion à choisir

Ouvre le fichier **"connexion composants"** à partir du site "VITTASCIENCE" :

https://fr.vittascience.com/microbit/?mode=mixed&console=bottom&toolbox=vittascience&localId=loc654737b9de09f0



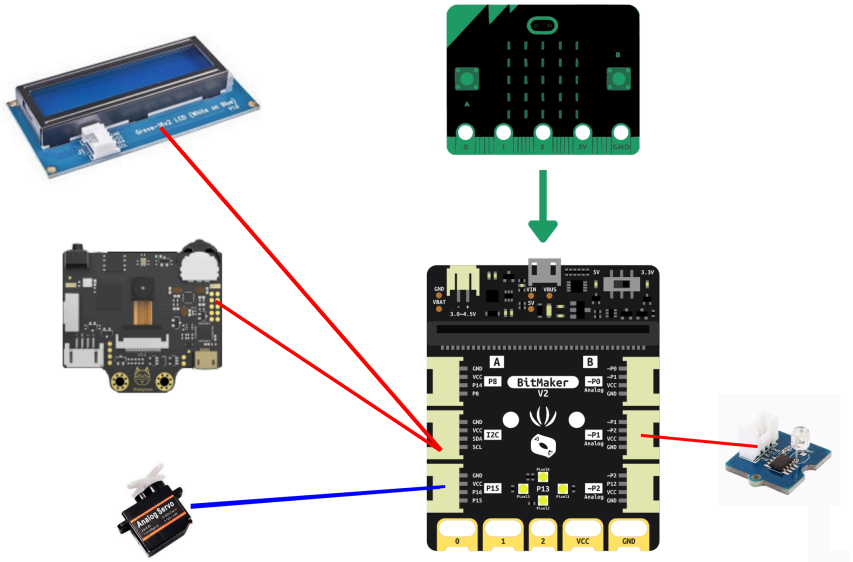
Travail à faire :

- Laisse le pointeur sur le bloc souhaité pour faire apparaître les paramètres de connexion
- Indique par un trait les câblages à faire sur la carte de connexion BITMAKER

NOM :
Prénom :

Groupe :
Rôle dans le groupe :

Classe :
Date :



N2b - Paramétrage des blocs de programmation

Sur l'application de programmation VITTASCIENCE, que faut-il modifier au niveau du bloc de programmation pour que le programme reçoive la bonne donnée (ex : valeur de luminosité) ?

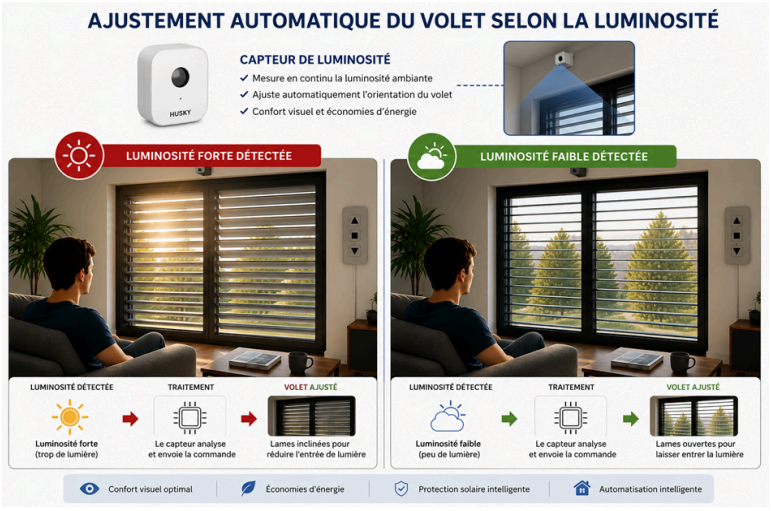
connexion composants.py **(à télécharger dossier programme)**

Activités *(niveaux 3 et 4)* **(35')**

Les composants sont maintenant connectés et prêts à être utilisés par le programme. Il est temps de passer à la phase de développement. Dans cette partie, tu vas modifier le programme afin que le volet réagisse automatiquement à son environnement grâce au capteur de luminosité ou à des commandes gestuelles détectées par la caméra intelligente. Tu découvriras également comment un programme réalisé avec des blocs graphiques peut être traduit en langage Python.

Ressources

N3-PIloter automatiquement le volet.



La luminosité idéale de la pièce est d'environ 500 lumens. De façon automatique en fonction de la luminosité, le programme devra soit fermer les lames, soit les ouvrir pour obtenir une luminosité optimale dans la pièce.

N3a-Connaître la luminosité de la pièce

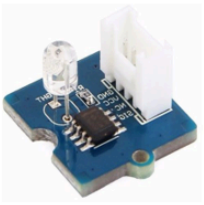
NOM :
Prénom :

Groupe :
Rôle dans le groupe :

Classe :
Date :

Nous allons modifier le programme existant pour connaître la luminosité dans la pièce.

Pour cela, le programme fera afficher sur l'écran LCD la luminosité captée par le détecteur de lumière

Détecteur de lumière	Ecran LCD
	
Bloc : 	Bloc : 

Travail à faire :

- Connecte les composants sur le shield
- Ouvrir le programme **" Luminosité_Pièce_Elève "** sur VITTASCIENCE
- Créer la variable **"luminosité"** et associez-la à la luminosité mesurée sur le capteur de luminosité sur la broche P0
- Modifier le programme pour afficher sur l'écran LCD la valeur de luminosité mesurée suivie de l'unité (ex : xxx lux)
- Vérifier le résultat et modifier si nécessaire.
- Faire contrôler par le professeur

N3b-Automatisation de la Fermeture / Ouverture des lames orientables en fonction de la luminosité

L'algorithme correspondant à l'automatisation du volet en fonction de la luminosité est le suivant :

Répéter indéfiniment :

Affecter la variable luminosité à la valeur du capteur de luminosité

Afficher la variable luminosité sur l'écran LCD sur la ligne 0 en position 0

Afficher Lux sur l'écran LCD sur la ligne 0 en position 5

Attendre 500ms

Nettoyer l'écran

Si luminosité strictement inférieure à 300 lumens

Alors activer les lames à une valeur de servomoteur de (0°)

Si luminosité supérieure ou égale à 300 lumens et strictement inférieure à 400 lumens

Alors activer les lames à une valeur de servomoteur de (20°)

Si luminosité supérieure ou égale à 300 lumens et strictement inférieure à 400 lumens

Alors activer les lames à une valeur de servomoteur de (40°)

Programme :

Luminosité_Pièce_Elève.py (à télécharger dossier programme)

CORRECTION : " Luminosité Pièce CORR " (à télécharger dossier programme)

NOM :
Prénom :

Groupe :
Rôle dans le groupe :

Classe :
Date :

Si luminosité supérieure ou égale à 300 lumens et strictement inférieure à 400 lumens

Alors activer les lames à une valeur de servomoteur de (60°)

Si luminosité supérieure à 600 lumens

Alors activer les lames à une valeur de servomoteur de (80°)

Travail à faire :

- Ouvrir le programme **" Luminosité_Incrément_Eleve"**
- Connecter les composants
- Replacer et modifier les blocs en fonction de l'algorithme ci-dessus
- Tester et modifier si nécessaire
- Faire contrôler par le professeur

Programme :

Luminosité_Incrément_Elève.py (**à télécharger dossier programme**)

CORRECTION : " Luminosité_Incrément_CORR " (**à télécharger dossier programme**)

N3c - Commander le volet par caméra IA

Pour remplacer la télécommande perdue, nous voulons commander le volet à l'aide d'un geste :



pouce levé : ouverture du volet

pouce baissé : fermeture du volet.

Nous allons utiliser pour cela une caméra intelligente à reconnaissance de geste

-Tutoriel Huskylens.pdf

-Huskylens_images_pouces.jpg

Programme :

Huskylens_Pouces_Elève.py (**à télécharger dossier programme**)

CORRECTION : " Husky lens Pouces CORR " (**à télécharger dossier programme**)

Travail à faire

- Connecter la caméra HUSKY LENS
- Suivre le document ressources afin de réaliser l'apprentissage des images par la caméra intelligente (utiliser également l'image des deux pouces)
- Ouvrir le programme "**Huskylens_Pouces_Eleve**"
- Modifier le programme pour réaliser la commande attendue
- Tester et modifier si nécessaire
- Faire vérifier par le professeur

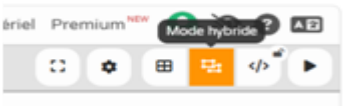
N4-Du programme par blocs au programme Python

Afin d'être plus précis dans l'inclinaison des lames du volet, nous décidons de programmer la maquette en langage Python.

N4a- Associer les lignes de programme Python aux blocs du programme initial.

Tu as programmé le volet à l'aide de blocs graphiques sur Vittascience. Pourtant, la carte micro:bit exécute en réalité un programme écrit dans un langage textuel appelé **Python**.

Il est possible de visualiser les lignes python que l'application envoie à la carte micro bit en cliquant sur l'icône "mode hybride".

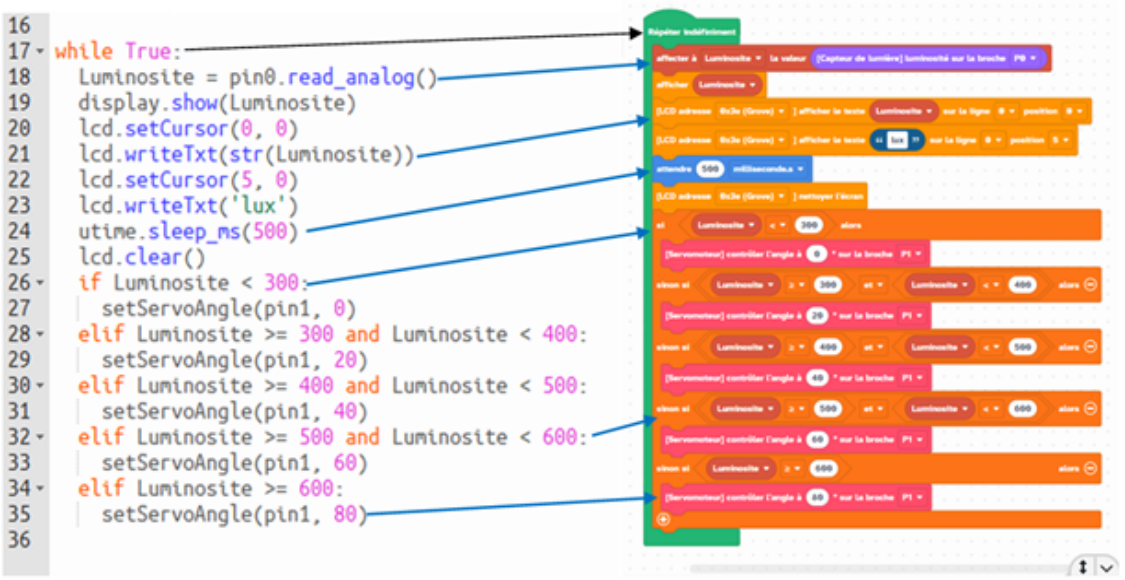


Ouvre le fichier **Luminosité Incrément CORR** et passe en mode hybride pour visualiser le programme python. Tu obtiens la présentation ci-dessous

NOM :
Prénom :

Groupe :
Rôle dans le groupe :

Classe :
Date :



```
16
17 while True:
18   Luminosite = pin0.read_analog()
19   display.show(Luminosite)
20   lcd.setCursor(0, 0)
21   lcd.writeTxt(str(Luminosite))
22   lcd.setCursor(5, 0)
23   lcd.writeTxt('lux')
24   utime.sleep_ms(500)
25   lcd.clear()
26   if Luminosite < 300:
27     setServoAngle(pin1, 0)
28   elif Luminosite >= 300 and Luminosite < 400:
29     setServoAngle(pin1, 20)
30   elif Luminosite >= 400 and Luminosite < 500:
31     setServoAngle(pin1, 40)
32   elif Luminosite >= 500 and Luminosite < 600:
33     setServoAngle(pin1, 60)
34   elif Luminosite >= 600:
35     setServoAngle(pin1, 80)
36
```

Travail à faire : Dans le tableau ci-dessous, indique la ligne du programme python qui correspond à chacun des blocs :

Bloc de programmation	Repère Ligne Python
Répéter indéfiniment	**17**
Attendre 500 ms	
Affecter la variable luminosité au capteur	
Afficher la luminosité sur l'écran LCD	
Si luminosité < 300	
Positionner servomoteur à 80°	
Si luminosité > 500 et < 600	

N4b- Modification programme en python

Pour accéder au programme en langage Python, il faut cliquer sur le bouton “ mode code”



Pour limiter le risque de luminosité trop forte, tu dois ajouter une nouvelle condition directement au programme python :

Lorsque la luminosité dépasse **700 lux**, écrire sur l’écran LCD (ligne 2) : “ Surexposition”

-Complète les 3 lignes de programme à rajouter :

elif

lcd.setCursor(,)

lcd.

- Modifier le programme Python et vérifier le résultat

- Fais contrôler par le professeur

Ma synthèse (5’)

NOM : Groupe : Classe :
Prénom : Rôle dans le groupe : Date :

Questions

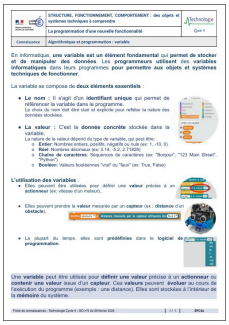
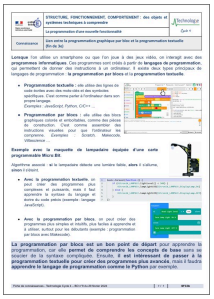
1. Pour ajouter une nouvelle fonctionnalité à un objet technique comme le volet orientable, quelles modifications faut-il réaliser ?
….

2. Comment le programme utilise-t-il les informations fournies par les capteurs pour prendre une décision ?
….

3. Quel est le lien entre les blocs graphiques de Vittascience et le programme en langage Python ?
…

Synthèse de la classe:

Fiches de connaissances (5')

SFC3a instruction d'affectation, variable (type mot, nombre et booléen) 	SFC3k Lien entre la programmation graphique par bloc et la programmation textuelle (fin de 3e) 

Quiz auto-correctifs (5')

SFC3a instruction d'affectation, variable (type mot, nombre et booléen)	SFC3k Lien entre la programmation graphique par bloc et la programmation textuelle (fin de 3e)

➢ ***Je développe mes compétences CPS : C3 – Capacité à résoudre des problèmes de façon créative : "Je résous un problème technique"***

☐ ***N1*** **— Je ne sais pas par où commencer.**	☐ ***N2*** **—Je propose une idée avec aide**	☐ ***N3*** **— Je teste et améliore une solution** .	☐ ***N4*** **— Je propose plusieurs solutions et les améliore**

NOM :
Prénom :

Groupe :
Rôle dans le groupe :

Classe :
Date :

STI Technologie	**Fiche d'activité n°2 de la séquence n°8**	**Durée : 2h30**	CYCLE 4 5 4 3
	Intitulée : Automatiser le volet		
Thématique abordée :	**T12-La programmation d'une nouvelle fonctionnalité**		

Compétences et connaissances travaillées dans l'activité

Thème abordé : Structure, fonctionnement, comportement : des objets et des systèmes techniques à comprendre **Attendu de fin de cycle** :SFC3-Comprendre et modifier un programme associé à une fonctionnalité d'un objet ou d'un système technique **Thématique :** T12-La programmation d'une nouvelle fonctionnalité	
Objectifs d'apprentissages (Compétences détaillées) SFC 33- Modifier et tester le programme associé à une nouvelle fonctionnalité d'un OST	**Connaissance** SFC3a instruction d'affectation, variable (type mot, nombre et booléen) SFC3k Lien entre la programmation graphique par bloc et la programmation textuelle (fin de 3e)
Critères d'apprentissages	**N1** – je sais identifier les éléments constituant un programme, **N2** – je sais compléter la structure interne de l'OST et connecter les composants **N3** – Je sais modifier et tester le programme associé à une nouvelle fonctionnalité d'un OST **N4** – Je sais associer les blocs de texte en langage Python aux blocs du programme initial qui remplissent les mêmes rôles.

Situation déclenchante de l'activité (2')

Nous voulons passer du fonctionnement précédent observé à ceux présentés ci-dessous :



Mes constats, mes observations (5')

On doit maintenant modifier le programme du volet pour que les lames orientables se ferment ou s'ouvrent automatiquement en fonction de la luminosité ou par la commande de geste repérée par la camérainélligente

Mon problème à résoudre (3')

Comment modifier le système et le programme pour qu'il soit automatisé

Mes idées pour le résoudre (5')

- **ajouter les composants nécessaires**
- **modifier le programme**

NOM : Groupe : Classe :
Prénom : Rôle dans le groupe : Date :

Activités *(niveaux 1 et 2)* **(20')**

Avant de modifier le comportement du volet, il est nécessaire de comprendre comment est construit un programme et comment les différents composants communiquent avec la carte de programmation. Dans cette première partie, tu vas identifier les principaux blocs utilisés dans un programme puis préparer le système en réalisant les connexions nécessaires entre les capteurs, les actionneurs et la carte micro:bit.

N1-Commandes et fonction de programmation

N1a-Indique le bloc correspondant à sa définition

Pour réaliser le programme, tu vas utiliser les différents blocs de programmation

variable / opérateur mathématique / opérateur logique / instruction conditionnelle / instruction itérative(ou boucle) / séquences d'instructions simples

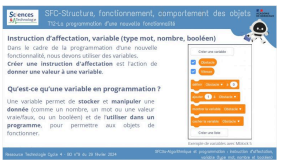
Complète le tableau ci-dessous
(si nécessaire, relis les fiches connaissances SFC 3a, SFC3b, SFC3c SFC3d)

Rep	Nom du bloc	Définition
1	**Boucle**	Permet de répéter un bloc d'instructions un nombre défini de fois ou jusqu'à ce qu'une condition soit remplie
2	**Variable**	permet de stocker et de manipuler des données. (provenant de capteurs par exemple)
3	**Opérateur arithmétiques**	réaliser des calculs mathématiques et d'obtenir des valeurs numériques
4	**Opérateur logique**	combiner ou modifier des conditions, produisant ainsi un résultat qui est soit "Vrai" soit "Faux" (ET / OU/ NON)
5	**Instruction conditionnelle**	permettent de contrôler le déroulement d'un programme en fonction de conditions (vrai/faux)
6	**Séquence d'instruction simple**	Suite d'instructions exécutées les unes après les autres sans conditions particulières

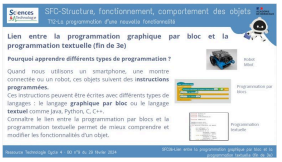
N1b-Identifie les blocs sur l'exemple de programme ci-dessous en indiquant le repère correspondant

Ressources vidéo et QR Code

SFC3a instruction d'affectation, variable (type mot, nombre et booléen)



SFC3k Lien entre la programmation graphique par bloc et la programmation textuelle



Fiches connaissances déjà vues en 5ème et 4ème :

SFC3a instruction d'affectation, variable (type mot, nombre et booléen)

SFC3b opérateurs arithmétiques et logiques (ET, OU, NON)

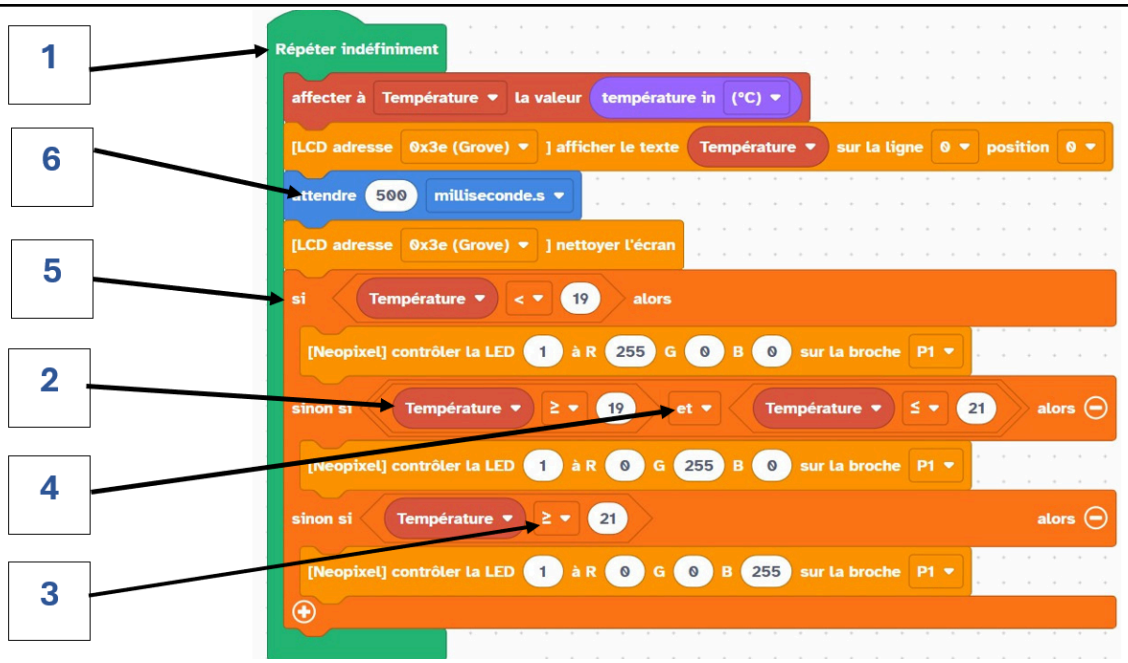
SFC3c instruction conditionnelle

SFC3d instructions itératives ;

NOM :
Prénom :

Groupe :
Rôle dans le groupe :

Classe :
Date :



N2-Associer les composants à la carte de programmation.

Il faut maintenant connecter les différents composants à ajouter et permettre au programme de recevoir les données entrantes. Pour cela :

- Il faut câbler chaque composant sur le bon port de l'interface,
- Il faut associer le port au bloc de programmation

N2a-Identification du port de connexion à choisir

Ouvre le fichier **"connexion composants"** à partir du site "VITTASCIENCE" :

https://fr.vittascience.com/microbit/?mode=mixed&console=bottom&toolbox=vittascience&localId=loc654737b9de09f0



Travail à faire :

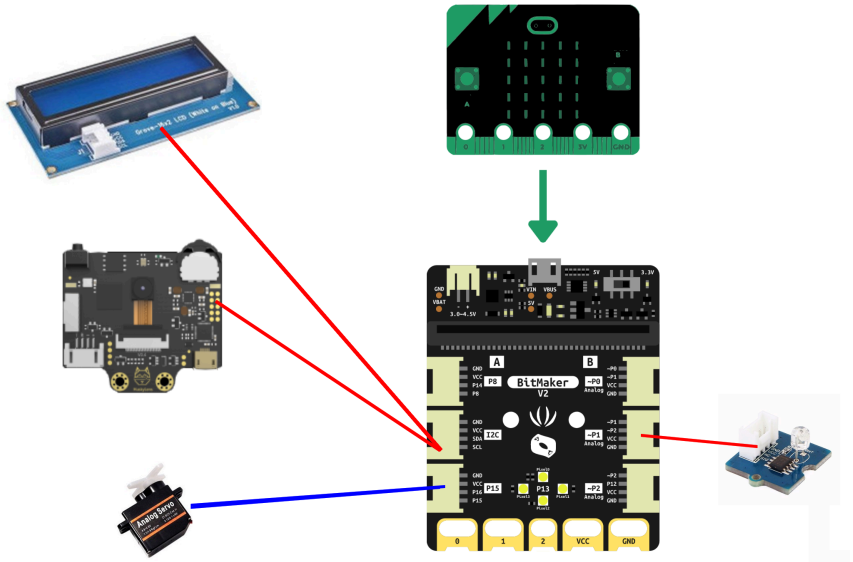
- Laisse le pointeur sur le bloc souhaité pour faire apparaître les paramètres de connexion
- Indique par un trait les câblages à faire sur la carte de connexion BITMAKER

connexion composants.py

NOM :
Prénom :

Groupe :
Rôle dans le groupe :

Classe :
Date :



(à télécharger dossier programme)

N2b - Paramétrage des blocs de programmation

Sur l'application de programmation VITTASCIENCE, que faut-il modifier au niveau du bloc de programmation pour que le programme reçoive la bonne donnée (ex : valeur de luminosité) ?

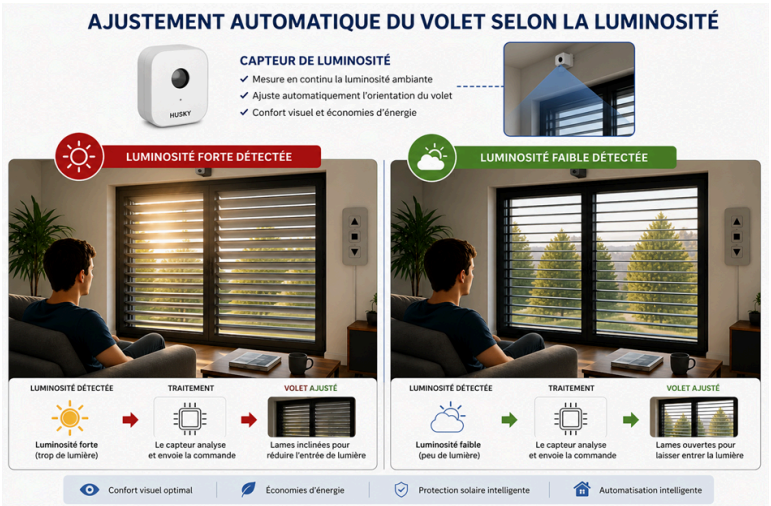
Il faut indiquer le port de connexion sur lequel est branché le composant.

Activités *(niveaux 3 et 4)* **(35')**

Les composants sont maintenant connectés et prêts à être utilisés par le programme. Il est temps de passer à la phase de développement. Dans cette partie, tu vas modifier le programme afin que le volet réagisse automatiquement à son environnement grâce au capteur de luminosité ou à des commandes gestuelles détectées par la caméra intelligente. Tu découvriras également comment un programme réalisé avec des blocs graphiques peut être traduit en langage Python.

Ressources

<u>N3-PIloter automatiquement le volet.</u>



La luminosité idéale de la pièce est d'environ 500 lumens. De façon automatique en fonction de la luminosité, le programme devra soit fermer les lames, soit les ouvrir pour obtenir une luminosité optimale dans la pièce.

N3a-Connaître la luminosité de la pièce

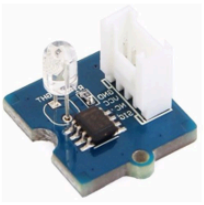
<u>NOM</u> :
<u>Prénom</u> :

<u>Groupe</u> :
<u>Rôle dans le groupe</u> :

<u>Classe</u> :
<u>Date</u> :

Nous allons modifier le programme existant pour connaître la luminosité dans la pièce.

Pour cela, le programme fera afficher sur l'écran LCD la luminosité captée par le détecteur de lumière

Détecteur de lumière	Ecran LCD
	
Bloc : 	Bloc : 

Travail à faire :

- Connecte les composants sur le shield
- Ouvrir le programme **" Luminosité_Pièce_Elève "** sur VITTASCIENCE
- Créer la variable **"luminosité"** et associez-la à la luminosité mesurée sur le capteur de luminosité sur la broche P0
- Modifier le programme pour afficher sur l'écran LCD la valeur de luminosité mesurée suivie de l'unité (ex : xxx lux)
- Vérifier le résultat et modifier si nécessaire.
- Faire contrôler par le professeur

N3b-Automatisation de la Fermeture / Ouverture des lames orientables en fonction de la luminosité

L'algorithme correspondant à l'automatisation du volet en fonction de la luminosité est le suivant :

Répéter indéfiniment :

Affecter la variable luminosité à la valeur du capteur de luminosité

Afficher la variable luminosité sur l'écran LCD sur la ligne 0 en position 0

Afficher Lux sur l'écran LCD sur la ligne 0 en position 5

Attendre 500ms

Nettoyer l'écran

Si luminosité strictement inférieure à 300 lumens

Alors activer les lames à une valeur de servomoteur de (0°)

Si luminosité supérieure ou égale à 300 lumens et strictement inférieure à 400 lumens

Alors activer les lames à une valeur de servomoteur de (20°)

Si luminosité supérieure ou égale à 300 lumens et strictement inférieure à 400 lumens

Alors activer les lames à une valeur de servomoteur de (40°)

Programme :

Luminosité_Pièce_Elève.py (à télécharger dossier programme)

CORRECTION : " Luminosité Pièce CORR " (à télécharger dossier programme)

NOM :
Prénom :

Groupe :
Rôle dans le groupe :

Classe :
Date :

Si luminosité supérieure ou égale à 300 lumens et strictement inférieure à 400 lumens

Alors activer les lames à une valeur de servomoteur de (60°)

Si luminosité supérieure à 600 lumens

Alors activer les lames à une valeur de servomoteur de (80°)

Travail à faire :

- Ouvrir le programme **" Luminosité_Incrément_Eleve"**
- Connecter les composants
- Replacer et modifier les blocs en fonction de l'algorithme ci-dessus
- Tester et modifier si nécessaire
- Faire contrôler par le professeur

Programme :

Luminosité_Incrément_Elève.py (**à télécharger dossier programme**)

CORRECTION : " Luminosité_Incrément_CORR " (**à télécharger dossier programme**)

N3c - Commander le volet par caméra IA

Pour remplacer la télécommande perdue, nous voulons commander le volet à l'aide d'un geste :



pouce levé : ouverture du volet

pouce baissé : fermeture du volet.

Nous allons utiliser pour cela une caméra intelligente à reconnaissance de geste

-Tutoriel Huskylens.pdf

-Huskylens_images_pouces.jpg

Programme :

Huskylens_Pouces_Elève.py (**à télécharger dossier programme**)

CORRECTION : " Husky lens Pouces CORR " (**à télécharger dossier programme**)

Travail à faire

- Connecter la caméra HUSKY LENS
- Suivre le document ressources afin de réaliser l'apprentissage des images par la caméra intelligente (utiliser également l'image des deux pouces)
- Ouvrir le programme "**Huskylens_Pouces_Eleve**"
- Modifier le programme pour réaliser la commande attendue
- Tester et modifier si nécessaire
- Faire vérifier par le professeur

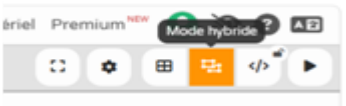
N4-Du programme par blocs au programme Python

Afin d'être plus précis dans l'inclinaison des lames du volet, nous décidons de programmer la maquette en langage Python.

N4a- Associer les lignes de programme Python aux blocs du programme initial.

Tu as programmé le volet à l'aide de blocs graphiques sur Vittascience. Pourtant, la carte micro:bit exécute en réalité un programme écrit dans un langage textuel appelé **Python**.

Il est possible de visualiser les lignes python que l'application envoie à la carte micro bit en cliquant sur l'icône "mode hybride".



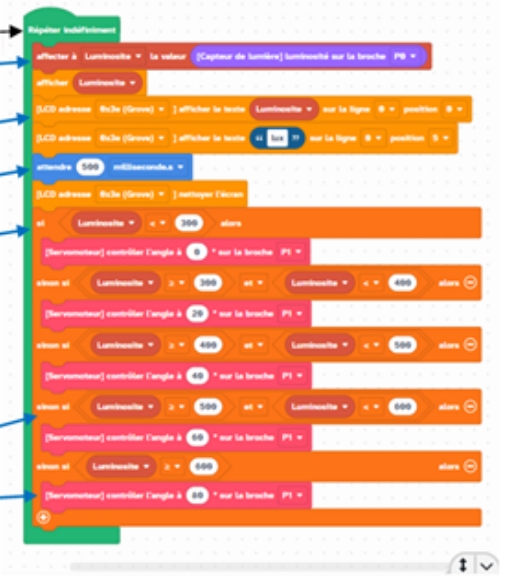
Ouvre le fichier **Luminosité Incrément CORR** et passe en mode hybride pour visualiser le programme python. Tu obtiens la présentation ci-dessous

NOM :
Prénom :

Groupe :
Rôle dans le groupe :

Classe :
Date :

```
16
17 while True:
18   Luminosite = pin0.read_analog()
19   display.show(Luminosite)
20   lcd.setCursor(0, 0)
21   lcd.writeTxt(str(Luminosite))
22   lcd.setCursor(5, 0)
23   lcd.writeTxt('lux')
24   utime.sleep_ms(500)
25   lcd.clear()
26   if Luminosite < 300:
27     setServoAngle(pin1, 0)
28   elif Luminosite >= 300 and Luminosite < 400:
29     setServoAngle(pin1, 20)
30   elif Luminosite >= 400 and Luminosite < 500:
31     setServoAngle(pin1, 40)
32   elif Luminosite >= 500 and Luminosite < 600:
33     setServoAngle(pin1, 60)
34   elif Luminosite >= 600:
35     setServoAngle(pin1, 80)
36
```



Travail à faire : Dans le tableau ci-dessous, indique la ligne du programme python qui correspond à chacun des blocs :

Bloc de programation	Repère Ligne Python
Répéter indéfiniment	17
Attendre 500 ms	24
Affecter la variable luminosité au capteur	18
Afficher la luminosité sur l'écran LCD	21
Si luminosité < 300	26
Positionner sevomoteur à 80 °	35
Si luminosité >500 et <600	32

N4b- Modification programme en python

Pour accéder au programme en langage Python, il faut cliquer sur le bouton “ mode code”



Pour limiter le risque de luminosité trop forte, tu dois ajouter une nouvelle condition directement au programme python :

Lorsque la luminosité dépasse **700 lux**, écrire sur l’écran LCD (ligne 2) : “ Surexposition”

-Complète les 3 lignes de programme à rajouter :

```
elif Luminosite >= 700:

 lcd.setCursor(0, 1)

lcd.writeTxt(str('surexposition '))
```

- Modifier le programme Python et vérifier le résultat

- Fais contrôler par le professeur

Ma synthèse (5’)

NOM : Groupe : Classe :
Prénom : Rôle dans le groupe : Date :

Questions

1. Pour ajouter une nouvelle fonctionnalité à un objet technique comme le volet orientable, quelles modifications faut-il réaliser ?
….

2. Comment le programme utilise-t-il les informations fournies par les capteurs pour prendre une décision ?
….

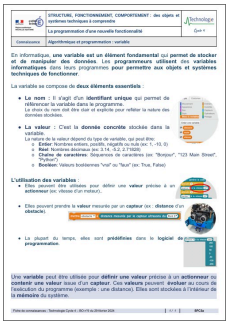
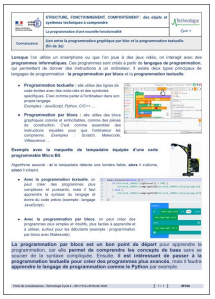
3. Quel est le lien entre les blocs graphiques de Vittascience et le programme en langage Python ?
…

Synthèse de la classe:
Pour modifier le comportement d'un système automatisé, il faut :

- ajouter et connecter les composants nécessaires (capteurs et actionneurs) ;
- paramétrer correctement leurs ports de connexion dans le programme ;
- utiliser des variables pour stocker les données mesurées ;
- utiliser des instructions conditionnelles pour adapter le comportement du système ;
- tester et corriger le programme jusqu'à obtenir le fonctionnement attendu.

Un programme réalisé avec des blocs graphiques est automatiquement traduit en langage Python avant d'être transféré dans la carte programmable. Les blocs et le Python permettent d'obtenir le même résultat, mais les blocs sont plus faciles à utiliser pour apprendre à programmer.

Fiches de connaissances (5')

SFC3a instruction d'affectation, variable (type mot, nombre et booléen) 	SFC3k Lien entre la programmation graphique par bloc et la programmation textuelle (fin de 3e) 

Quiz auto-correctifs (5')

SFC3a instruction d'affectation, variable (type mot, nombre et booléen)	SFC3k Lien entre la programmation graphique par bloc et la programmation textuelle (fin de 3e)

➢ ***Je développe mes compétences CPS : C3 – Capacité à résoudre des problèmes de façon créative : "Je résous un problème technique"***

☐ ***N1*** **— Je ne sais pas par où commencer.**	☐ ***N2*** **—Je propose une idée avec aide**	☐ ***N3*** **— Je teste et améliore une solution** .	☐ ***N4*** **— Je propose plusieurs solutions et les améliore**

NOM :
Prénom :

Groupe :
Rôle dans le groupe :

Classe :
Date :

3 Technologie	**EVALUATION DE COMPÉTENCE**	Insuffisante Non acquis · Fragile Non acquis · Satisfaisante Niveau d'acquisition · Très bonne
NOM : **Prénom :**	**Classe :** **Date :**	

Observations et conseils de progrès

Référence aux compétences et connaissances évaluées

Thème abordé : Structure, fonctionnement, comportement : des objets et des systèmes techniques à comprendre
Attendu de fin de cycle :SFC3-Comprendre et modifier un programme associé à une fonctionnalité d'un objet ou d'un système technique
Thématique : T12-La programmation d'une nouvelle fonctionnalité

Objectifs d'apprentissages (Compétences détaillées)	**Connaissance**	**Niveau atteint**
SFC 31-Déterminer les données utilisées et produites par un programme associé à une fonctionnalité en vue de le modifier SFC 32- Programmer un algorithme lié à une nouvelle fonctionnalité SFC 33- Modifier et tester le programme associé à une nouvelle fonctionnalité d'un OST	SFC3g événement SFC3h déclenchement d'une séquence d'instructions par un événement SFC3i entrées ou sorties d'un programme (données issues par exemple de capteurs IHM et sorties pouvant être en lien avec un actionneur, fichiers) ; SFC3a instruction d'affectation, variable (type mot, nombre et booléen) ; SFC3k Lien entre la programmation graphique par bloc et la programmation textuelle (fin de 3e)	

Critères d'évaluation	N1 – je sais ce qu'est un événement dans un programme et une donnée d'un programme, et je sais identifier les éléments constituant un programme.	
	N2 – je sais caractériser l'événement et les entrées liés à un programme, et je sais compléter la structure interne de l'OST et connecter les composants	
	N3 – je sais déterminer les données utilisées et produites par la partie du programme associé à une fonctionnalité d'un OST , et Je sais programmer un algorithme lié à une nouvelle fonctionnalité, et je sais modifier et tester le programme associé à une nouvelle fonctionnalité d'un OST	
	N4 –je sais proposer les données produites pour une nouvelle fonctionnalité, et Je sais associer les blocs de texte en langage Python aux blocs du programme initial qui remplissent les mêmes rôles.	

Sur le <u>document 1</u> ci-dessous est expliqué comment fonctionne un projecteur de terrasse :



L'allumage du projecteur se fait :

- Soit lorsque le capteur de luminosité détecte un manque de lumière(inférieur à 250 lux) ;
- Soit à distance à partir d'une application sur smartphone qui indique l'état du projecteur (allumé ou éteint).

N1 – Connaissance

N1 – Je sais ce qu'est un événement dans un programme et une donnée d'un programme, et je sais identifier les éléments constituant un programme.

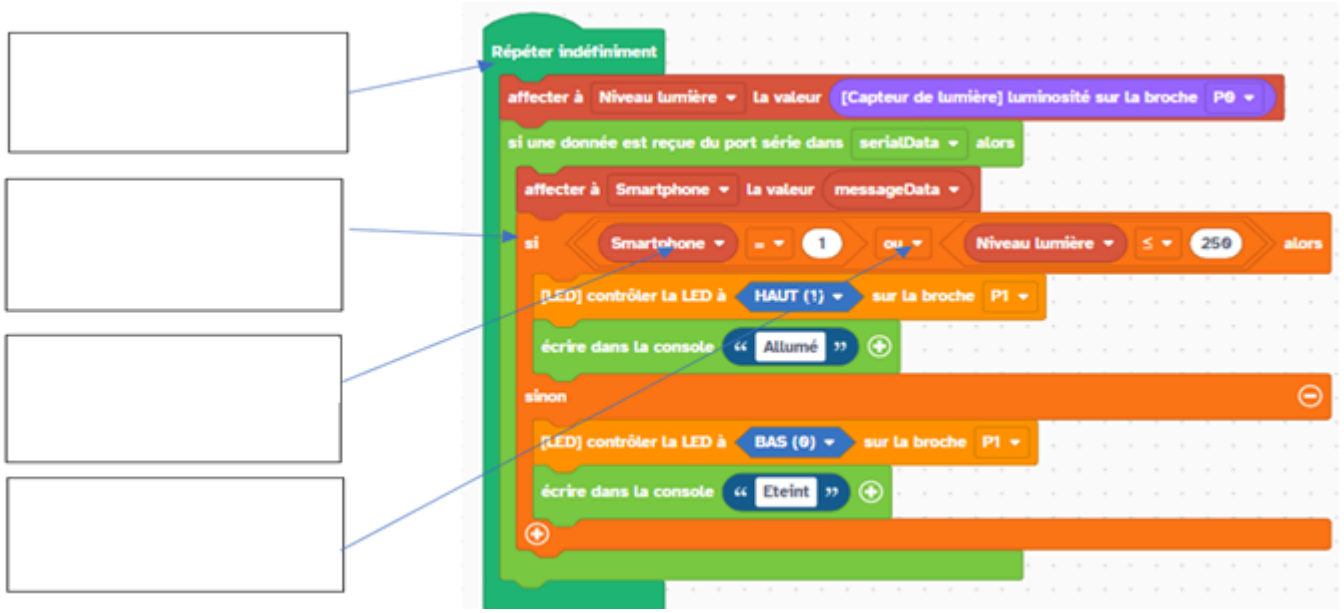
N1a-Ecris les termes en face de leur définition : Évènement / Entrée / Sortie :

	Élément servant à stocker une donnée dans la mémoire du programme
	Phénomène qui déclenche une action dans un programme
	Action ou information envoyée par le programme vers l'extérieur
	Suite d'instructions exécutées les unes après les autres dans un programme.
	Information reçue par le programme provenant d'un utilisateur, d'un capteur ou d'un fichier

N1b- Dans la programmation, que permettent les entrées/ sorties au niveau du programme (3 réponses attendues)?

N1c- Identifie des instructions du programme en utilisant des termes dans la liste ci-dessous :

variable / opérateur mathématique / opérateur logique / instruction conditionnelle / instruction itérative(ou boucle) / séquences d'instructions simples



N1d –Définis la notion de variable dans un programme :

N2 – Compréhension

N2-Je sais caractériser l'événement et les entrées liés à un programme, et je sais compléter la structure interne de l'OST et connecter les composants

N2a- Quels sont les 2 événements qui entraînent l'allumage du projecteur

N2b- A l'aide du document 1, retrouve le composant qui correspond à chacun des types de données ci-dessous :

Données utilisateurs :			**Affichage à l'écran :** Texte,	
Données provenant de capteurs :			**Contrôle des Actionneurs :**	

En observant le programme qui pilote le projecteur (question N1c) , réponds aux question suivantes

N2c- Quelles sont les 2 variables créées pour gérer le système ?

N2d- L'ampoule à Led du projecteur a été connectée sur le port I2C de la carte programmable. Le projecteur fonctionne-t-il correctement ?

N3 – Application

N3 – Je sais déterminer les données utilisées et produites par la partie du programme associé à une fonctionnalité d'un OST, et je sais programmer une nouvelle fonctionnalité en langage par blocs

En observant le programme qui pilote le projecteur (question N1c) , réponds aux question suivantes :

N3a – Quel est le seuil de luminosité pour lequel le projecteur s'allume ?

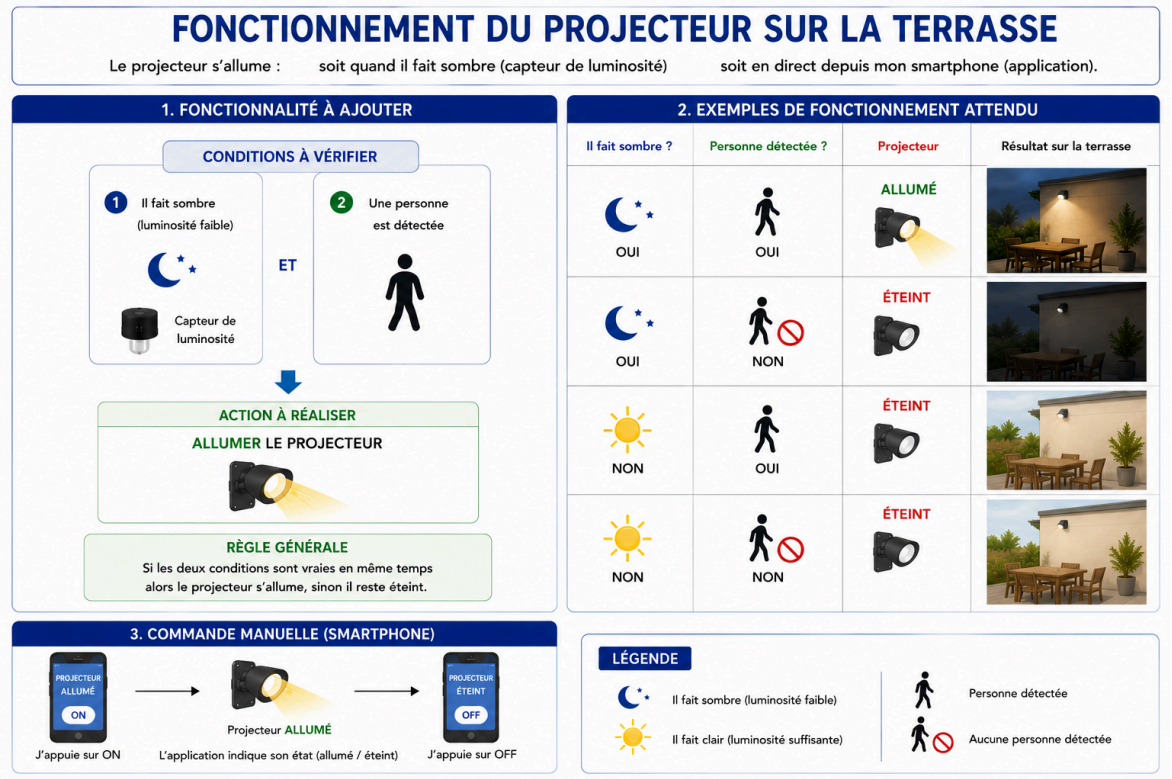
N3b – Complète l'algorithme lié au fonctionnement du projecteur (fonctionnement lié au document 1) :

Répéter indéfiniment

Si

Alors Allumer projecteur et Indiquer « état allumé »

Sur le document 2 ci-dessous est présentée une amélioration à apporter au système d'automatisation du projecteur :

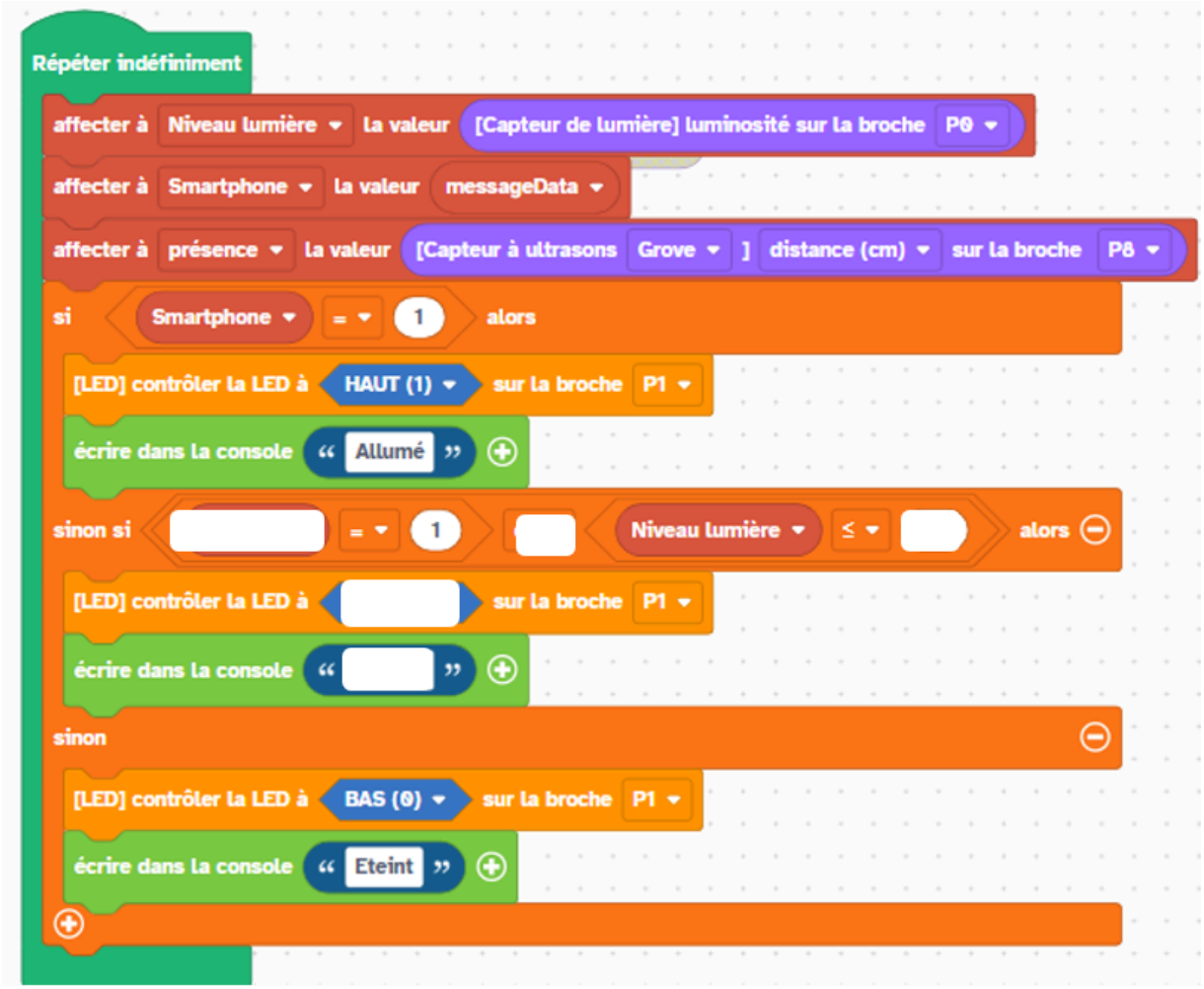


N3c-Quel composant faut-il ajouter? Justifie.

Un

Que faudra-t-il créer au niveau du programme ?

N3d : Complète le programme correspondant



N4 – Maîtrise

N4 – je sais proposer les données produites pour une nouvelle fonctionnalité, et Je sais associer les blocs de texte en langage Python aux blocs du programme initial qui remplissent les mêmes rôles.

N4a – Justifie l'intérêt de rajouter la fonctionnalité décrite dans le document 2 :

Sans cette fonctionnalité, le projecteur s'allume la nuit même si la terrasse est vide.

Ci dessous se trouve le programme en mode textuel envoyé à la carte programmable

```
27
28  while True:
29    Niveau_lumi_C3_A8re = pin1.read_analog()
30    Smartphone = messageData
31    presence = getUltrasonicData(pin1, pin1, 'distance')
32    if Smartphone == 1:
33      pin1.write_digital(1)
34      print('Allumé')
35    elif presence == 1 and Niveau_lumi_C3_A8re <= 250:
36      pin1.write_digital(1)
37      print('Allumé')
38    else:
39      pin1.write_digital(0)
40      print('Eteint')
```

N4b-Comment s'appelle ce langage de programmation ?

N4c – Indique à quelle ligne de programme correspondent les blocs de programmation ou actions ci-dessous

Bloc ou action	ligne de programme correspondante
Répéter indéfiniment	
Affecter la valeur du capteur de présence à la variable “présence”	
Si appui sur le bouton On de l’application installée	
Eteindre le projecteur	

N4d- Ecris les 3 lignes de programme à rajouter correspondant à la partie de l’algorithme :

Si aucune présence détectée ou luminosité supérieure à 250 lux

Alors Éteindre projecteur et indiquer “éteint”

3 Technologie	**EVALUATION DE COMPÉTENCE**	Insuffisante Non acquis / Fragile Non acquis / Satisfaisante Niveau d'acquisition / Très bonne
NOM : **Prénom :**	**Classe :** **Date :**	

Observations et conseils de progrès

Référence aux compétences et connaissances évaluées

Thème abordé : Structure, fonctionnement, comportement : des objets et des systèmes techniques à comprendre **Attendu de fin de cycle** :SFC3-Comprendre et modifier un programme associé à une fonctionnalité d'un objet ou d'un système technique **Thématique :** T12-La programmation d'une nouvelle fonctionnalité		
Objectifs d'apprentissages (Compétences détaillées) SFC 31-Déterminer les données utilisées et produites par un programme associé à une fonctionnalité en vue de le modifier SFC 32- Programmer un algorithme lié à une nouvelle fonctionnalité SFC 33- Modifier et tester le programme associé à une nouvelle fonctionnalité d'un OST	**Connaissance** SFC3g événement SFC3h déclenchement d'une séquence d'instructions par un événement SFC3i entrées ou sorties d'un programme (données issues par exemple de capteurs IHM et sorties pouvant être en lien avec un actionneur, fichiers) ; SFC3a instruction d'affectation, variable (type mot, nombre et booléen) ; SFC3k Lien entre la programmation graphique par bloc et la programmation textuelle (fin de 3e)	**Niveau atteint**
Critères d'évaluation	N1 – je sais ce qu'est un événement dans un programme et une donnée d'un programme, et je sais identifier les éléments constituant un programme.	
	N2 – je sais caractériser l'événement et les entrées liés à un programme, et je sais compléter la structure interne de l'OST et connecter les composants	
	N3 – je sais déterminer les données utilisées et produites par la partie du programme associé à une fonctionnalité d'un OST , et Je sais programmer un algorithme lié à une nouvelle fonctionnalité, et je sais modifier et tester le programme associé à une nouvelle fonctionnalité d'un OST	
	N4 –je sais proposer les données produites pour une nouvelle fonctionnalité, et Je sais associer les blocs de texte en langage Python aux blocs du programme initial qui remplissent les mêmes rôles.	

Sur le <u>document 1</u> ci-dessous est expliqué comment fonctionne un projecteur de terrasse :



L'allumage du projecteur se fait :

- Soit lorsque le capteur de luminosité détecte un manque de lumière(inférieur à 250 lux),
- Soit à distance à partir d'une application sur smartphone qui indique l'état du projecteur (allumé ou éteint).

N1 – Connaissance

N1 – Je sais ce qu'est un événement dans un programme et une donnée d'un programme, et je sais identifier les éléments constituant un programme.

N1a-Ecris les termes en face de leur définition : Évènement / Entrée / Sortie :

	Élément servant à stocker une donnée dans la mémoire du programme
Évènement	**Phénomène qui déclenche une action dans un programme**
Sortie	**Action ou information envoyée par le programme vers l'extérieur**
	Suite d'instructions exécutées les unes après les autres dans un programme.
Entrée	**Information reçue par le programme provenant d'un utilisateur, d'un capteur ou d'un fichier**

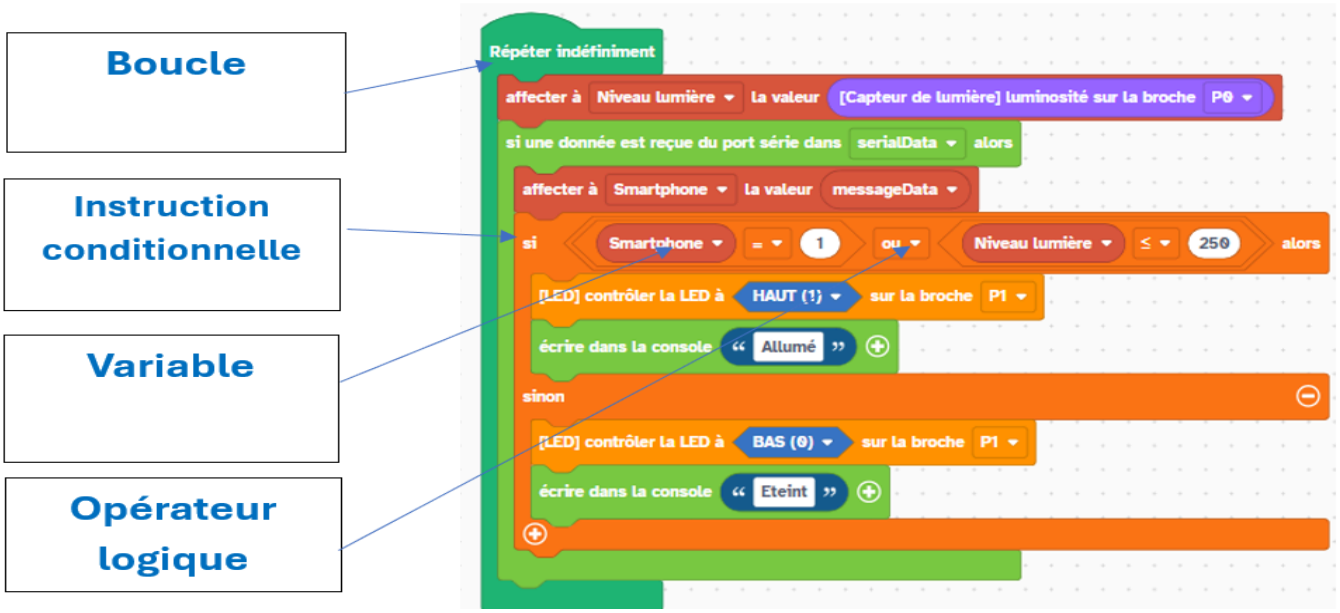
N1b- Dans la programmation, que permettent les entrées/ sorties au niveau du programme (3 réponses attendues)?

- D'interagir avec le monde extérieur
- D'échanger des données
- De réaliser des actions concrètes.

N1c- Identifie des instructions du programme en utilisant des termes dans la liste ci-dessous :

variable / opérateur mathématique / opérateur logique / instruction conditionnelle /

instruction itérative(ou boucle) / séquences d'instructions simples



N1d –Définis la notion de variable dans un programme :

Valeur qui peut changer et stockée dans la mémoire du système. Elle permet de définir une valeur précise à un actionneur ou contenir une valeur envoyée par un capteur.

N2 – Compréhension

N2-Je sais caractériser l'événement et les entrées liés à un programme, et je sais compléter la structure interne de l'OST et connecter les composants

N2a- Quels sont les 2 événements qui entraînent l'allumage du projecteur

- La détection d'une luminosité faible
- L'appui sur le bouton On sur le smartphone

N2b- A l'aide du document 1, retrouve le composant qui correspond à chacun des types de données ci-dessous :

Données utilisateurs :	- Appui sur On - Appui sur OFF		**Affichage à l'écran :** Texte,	- Etat « allumé ou éteint »
Données provenant de capteurs :	- Luminosité		**Contrôle des Actionneurs :**	-.Allumage du projecteur

En observant le programme qui pilote le projecteur (question N1c) , réponds aux question suivantes

N2c- Quelles sont les 2 variables créées pour gérer le système ?

- Niveau lumière
- Smartphone

N2d- L'ampoule à Led du projecteur a été connectée sur le port I2C de la carte programmable. Le projecteur fonctionne-t-il correctement ?

non , car le programme indique que l'ampoule doit être connectée sur le port P1

N3 – Application

N3 – Je sais déterminer les données utilisées et produites par la partie du programme associé à une fonctionnalité d'un OST, et je sais programmer une nouvelle fonctionnalité en langage par blocs

En observant le programme qui pilote le projecteur (question N1c) , réponds aux question suivantes :

N3a – Quel est le seuil de luminosité pour lequel le projecteur s'allume ? inférieur à 250 lux

N3b – Complète l'algorithme lié au fonctionnement du projecteur (fonctionnement lié au document 1) :

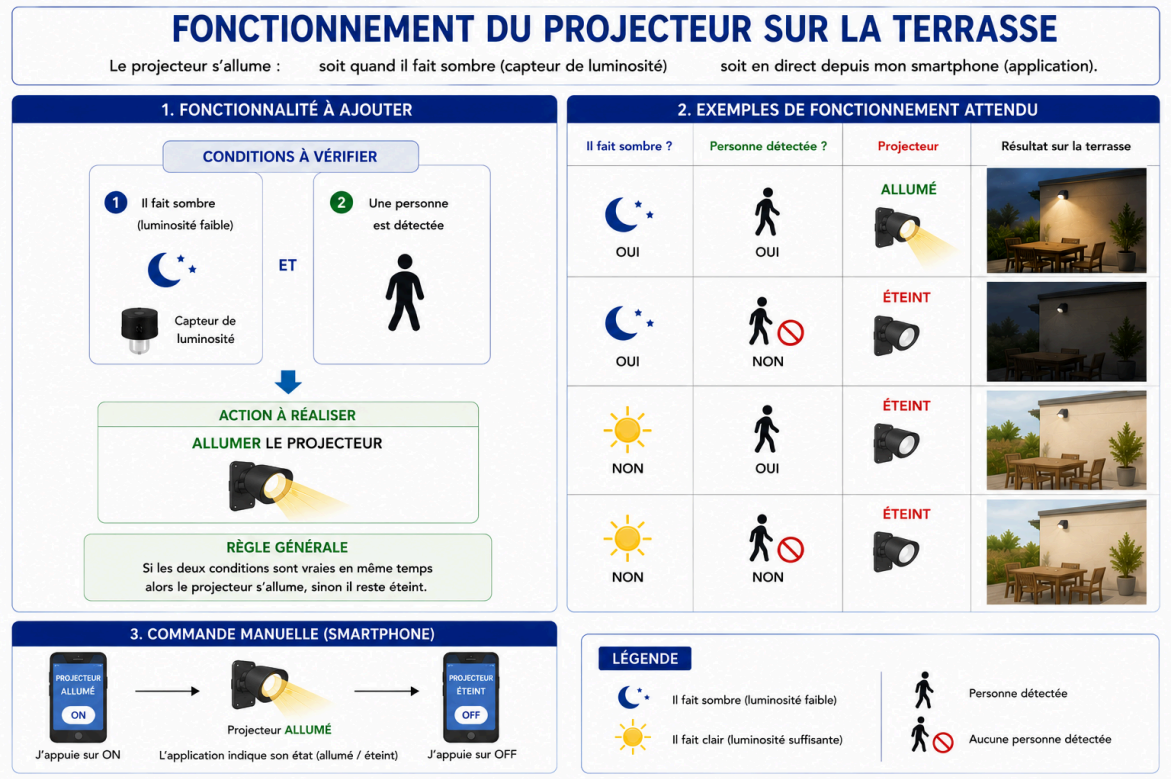
Répéter indéfiniment

Si luminosité inférieure à 250 lux ou appui ON sur smartphone

Alors Allumer projecteur et Indiquer « état allumé »

Sinon Éteindre projecteur et indiquer « état éteint »

Sur le document 2 ci-dessous est présentée une amélioration à apporter au système d'automatisation du projecteur :



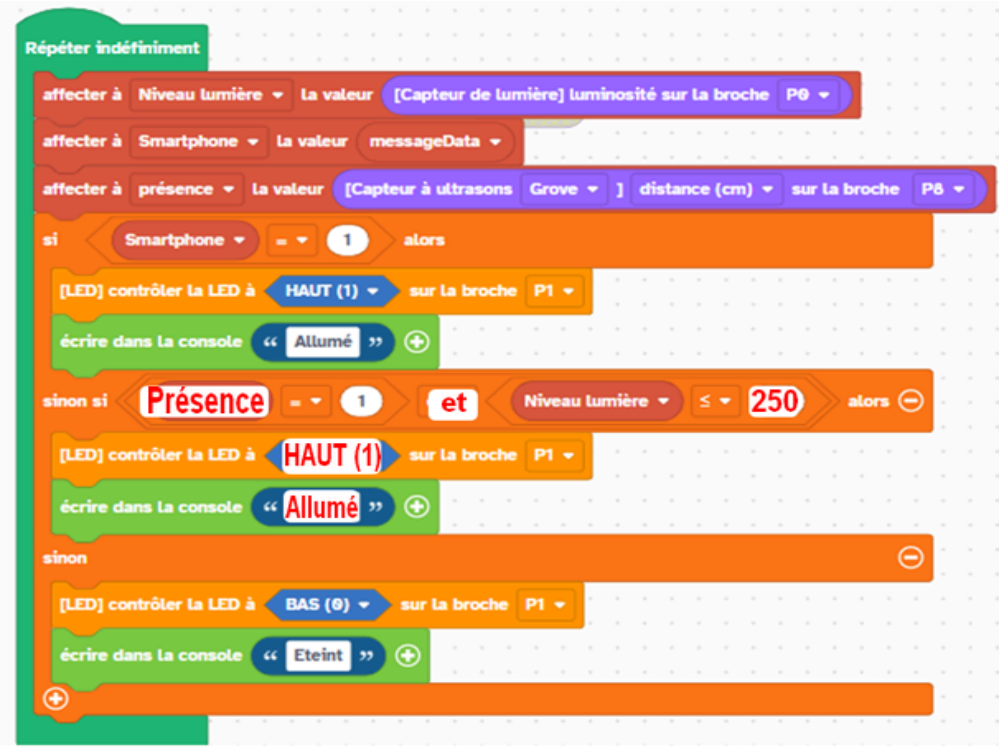
N3c-Quel composant faut-il ajouter? Justifie.

Un détecteur de mouvement qui permet de savoir s'il y a une présence

Que faudra-t-il créer au niveau du programme ?

Une variable liée à la présence d'une personne, exemple "présence"

N3d : Complète le programme correspondant



N4 – Maîtrise

N4 – je sais proposer les données produites pour une nouvelle fonctionnalité, et Je sais associer les blocs de texte en langage Python aux blocs du programme initial qui remplissent les mêmes rôles.

N4a – Justifie l'intérêt de rajouter la fonctionnalité décrite dans le document 2 :

Sans cette fonctionnalité, le projecteur s'allume la nuit même si la terrasse est vide.

Ci dessous se trouve le programme en mode textuel envoyé à la carte programmable

```
27
28 while True:
29   Niveau_lumi_C3_A8re = pin1.read_analog()
30   Smartphone = messageData
31   presence = getUltrasonicData(pin1, pin1, 'distance')
32   if Smartphone == 1:
33     pin1.write_digital(1)
34     print('Allumé')
35   elif presence == 1 and Niveau_lumi_C3_A8re <= 250:
36     pin1.write_digital(1)
37     print('Allumé')
38   else:
39     pin1.write_digital(0)
40     print('Eteint')
```

N4b-Comment s'appelle ce langage de programmation ? le Python

N4c – Indique à quelle ligne de programme correspondent les blocs de programmation ou actions ci-dessous

Bloc ou action	ligne de programme correspondante
Répéter indéfiniment	28
Affecter la valeur du capteur de présence à la variable "présence"	31
Si appui sur le bouton On de l'application installée	32
Eteindre le projecteur	39

N4d- Ecris les 3 lignes de programme à rajouter correspondant à la partie de l'algorithme :

Si aucune présence détectée ou luminosité supérieure à 250 lux

Alors Éteindre projecteur et indiquer "éteint"

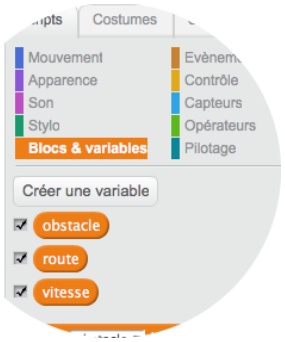
```
if Presence == 0 or Niveau_lumi_C3_A8re >= 250:
  pin1.write_digital(0)
  print('Eteint')
```

	STRUCTURE, FONCTIONNEMENT, COMPORTEMENT : des objets et systèmes techniques à comprendre	Technologie SFC 3 a
	La programmation d'une nouvelle fonctionnalité	
Connaissance	**Algorithmique et programmation : variable**	

En informatique, **une variable est un élément fondamental** qui **permet de stocker et de manipuler des données**. Les **programmeurs utilisent** des **variables informatiques** dans leurs programmes **pour permettre aux objets et systèmes techniques de fonctionner**.

La variable se compose de **deux éléments essentiels** :

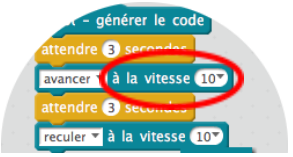
- **Le nom :** Il s'agit d'un **identifiant unique** qui permet de référencer la variable dans le programme.
  Le choix du nom doit être clair et explicite pour refléter la nature des données stockées.



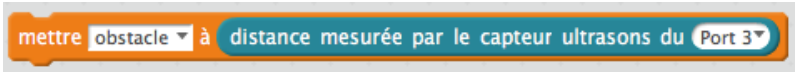
- **La valeur :** C'est la **donnée concrète** stockée dans la variable.
  La nature de la valeur dépend du type de variable, qui peut être:
  - **Entier**: Nombres entiers, positifs, négatifs ou nuls (ex: 1, -10, 0)
  - **Réel**: Nombres décimaux (ex: 3.14, -5.2, 2.71828)
  - **Chaîne de caractères**: Séquences de caractères (ex: "Bonjour", "123 Main Street", "Python")
  - **Booléen**: Valeurs booléennes "vrai" ou "faux" (ex: True, False)

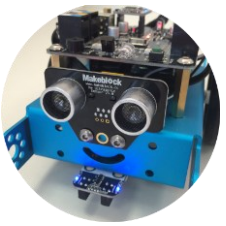
L'utilisation des variables :

- Elles peuvent être utilisées pour définir une **valeur** précise à un **actionneur** (ex: vitesse d'un moteur)..

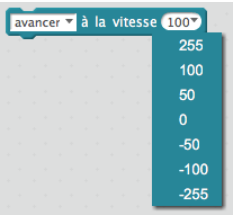


- Elles peuvent prendre la **valeur** mesurée par un **capteur** (ex : **distance** d'un **obstacle**).





- La plupart du temps, elles sont **prédéfinies** dans le **logiciel de programmation**.



Une **variable** peut être utilisée pour **définir une valeur** précise à un **actionneur** ou **contenir une valeur** issue d'un **capteur**. Ces **valeurs** peuvent **évoluer** au cours de l'exécution du programme (exemple : une distance). Elles sont stockées à l'intérieur de la **mémoire** du système.

	STRUCTURE, FONCTIONNEMENT, COMPORTEMENT : des objets et systèmes techniques à comprendre	Technologie
	La programmation d'une nouvelle fonctionnalité	**SFC 3 b**
Connaissance	**Algorithmique et programmation : opérateurs arithmétiques et logiques (ET, OU, NON)**	

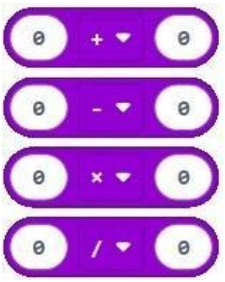
Lorsqu'on utilise un ordinateur, un smartphone, un robot ou même une calculatrice, **on exécute des programmes qui suivent des instructions précises**, appelées **algorithmes**.
Ces algorithmes sont constitués d'**opérateurs arithmétiques** et **logiques** qui permettent de **réaliser des calculs** et de **prendre des décisions**.

Pour comprendre et modifier un programme, il est donc nécessaire de **comprendre ces opérateurs**.

- Un **opérateur arithmétique** permet de **réaliser des calculs mathématiques et d'obtenir des valeurs numériques**.

  Les **opérateurs arithmétiques** correspondent aux opérations mathématiques courantes :

  - **Addition (+) :** additionner deux nombres.
  - **Soustraction (-) :** soustraire un nombre à un autre.
  - **Multiplication (x) :** multiplier deux nombres.
  - **Division (/) :** diviser un nombre par un autre.



- Un **opérateur logique** permet de **combiner ou modifier des conditions**, produisant ainsi un résultat qui est soit "**Vrai**" soit "**Faux**".

  Les **opérateurs logiques** correspondent aux fonctions binaires :

  - **ET :** Permet de tester si **deux conditions sont vraies** en même temps.
  - **OU :** Permet de tester **si l'une ou l'autre des conditions est vraie**.
  - **NON :** Permet d'**inverser le résultat** d'une condition.





Les **programmes informatiques suivent des instructions précises** appelées algorithmes, composés d'opérateurs **arithmétiques (calculs)** et **logiques (décisions).**
Les **opérateurs arithmétiques** (+, -, x, /) **servent aux calculs mathématiques** et les **opérateurs logiques** (ET, OU, NON) **servent à combiner des conditions** (vrai/faux).

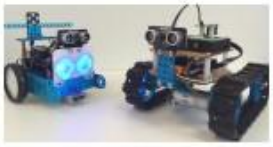
	STRUCTURE, FONCTIONNEMENT, COMPORTEMENT : des objets et systèmes techniques à comprendre **La programmation d'une nouvelle fonctionnalité**	Technologie **SFC 3 c**
Connaissance	**Algorithmique et programmation : instruction conditionnelle**	

Pour adapter le comportement d'un programme en fonction de différentes situations, on utilise des **instructions conditionnelle**s pour réaliser des actions.

- **Les instructions conditionnelles** sont des éléments essentiels de la programmation qui permettent de contrôler le flux d'exécution d'un programme en fonction de conditions booléennes (vrai ou faux).

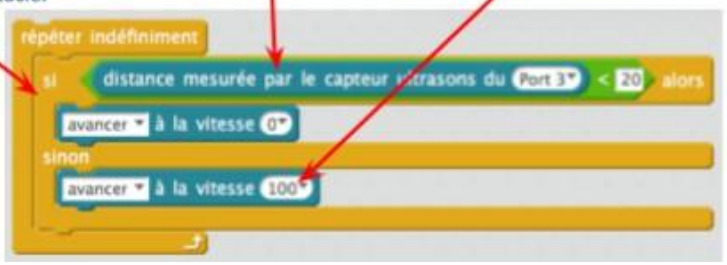
  Elles permettent de prendre des décisions et d'exécuter des actions en fonction de ces décisions.

Exemple : le robot mBot doit détecter un obstacle.



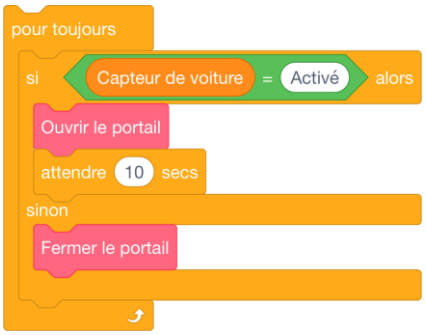
Si obstacle Alors
arrêt
Sinon avancer
Si obstacle Alors arrêt…

Le robot mBot doit laisser la priorité, en détectant l'apparition d'un autre véhicule. Il s'arrête quand il détecte un obstacle devant lui et se remet en route quand il n'y a plus d'obstacle.



- Pour **créer une instruction conditionnelle,** il faut:
  - **Définir la condition** : c'est une question à laquelle on peut répondre par "vrai" ou "faux".

    Exemple : **Si** le capteur de voiture est activé (la condition est vraie).
  - **Déterminer les deux actions possibles** : Une action sera effectuée si la condition est vraie, et l'autre si la condition est fausse.
    Exemple : **Alors** ouvrir le portail (la condition est vraie), **sinon** fermer le portail (la condition est fausse).



Les **instructions conditionnelle sont présentent** dans les programmes des **objets et systèmes techniques** qui **interagissent avec leur environnement** grâce à des **capteurs** : une porte de supermarché, un robot de nettoyage,… .

Les **instructions conditionnelles** permettent de **contrôler le déroulement** d'un programme **en fonction de conditions (vrai/faux)**. Elles permettent de **prendre des décisions et d'exécuter des actions** en conséquence. Les instructions conditionnelles ont la forme suivante : **SI la condition est vrai**, **ALORS** l'instruction s'exécute, **SINON** c'est une autre qui s'exécute.

	STRUCTURE, FONCTIONNEMENT, COMPORTEMENT : des objets et systèmes techniques à comprendre	Technologie SFC 3 d
	La programmation d'une nouvelle fonctionnalité	
Connaissance	**Algorithmique et programmation : instructions itératives**	

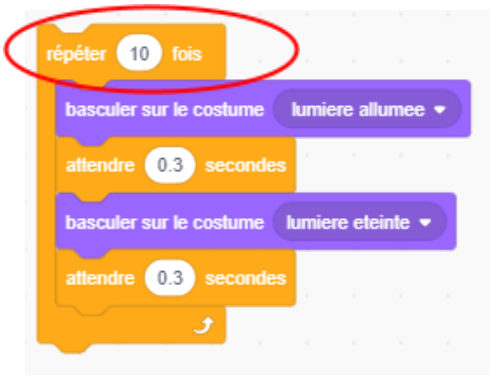
Les **instructions itératives** permettent de **répéter** une **séquence d'instructions plusieurs fois de suite, ou jusqu'à ce qu'une condition soit remplie**. Cela fait gagner du temps et rend le programme plus facile à lire.

Les **instructions itératives** sont **des boucles**. Il existe deux types de boucles :

- **La boucle "répéter…fois" : Elle** répète une séquence d'instructions **un nombre défini de fois.**

  Exemple: On veut faire clignoter une lumière 10 fois:

Avec boucle



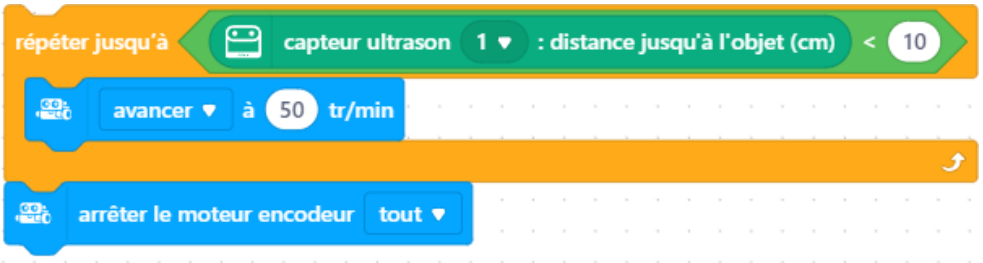
Sans boucle :



il aurait fallu écrire 10 fois les mêmes instructions

- **Les boucles "tant que" ou "répéter jusqu'à ce que"** : Elles répètent une séquence d'instructions tant qu'une condition est **vraie**.

  Exemple : Le robot avance tant qu'il ne trouve pas d'obstacle au-delà de 10 cm, sinon il s'arrête.

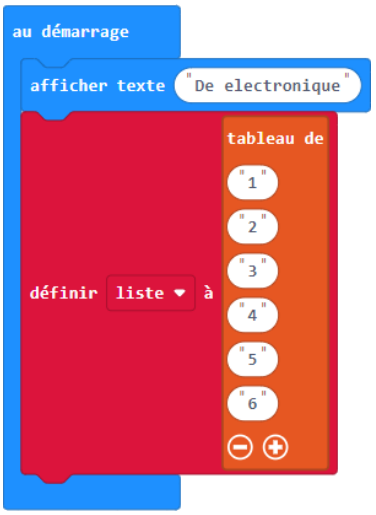


Les instructions itératives permettent de **répéter une séquence d'instructions** dans un programme.

Il existe deux types d'instructions itératives : la **boucle "répéter … fois"** et les **boucles "tant que" ou "répéter jusqu'à ce que"**. Il est important de savoir **choisir le type d'instruction itérative adaptée** à la situation et de savoir **écrire la condition d'arrêt** de la boucle.

	STRUCTURE, FONCTIONNEMENT, COMPORTEMENT : des objets et systèmes techniques à comprendre	Technologie SFC 3 e
	La programmation d'une nouvelle fonctionnalité	
Connaissance	**Algorithmique et programmation : structure de données « listes » afin de stocker des données issues du programme pour les parcourir et les traiter**	

Lorsqu'on veut stocker des informations dans un programme informatique, on peut utiliser différentes **variables.** On peut aussi utiliser **des listes** qui permettent de **ranger des données de manière ordonnée.**



Les listes, également appelées **tableaux dynamiques**, sont des **structures de données** en informatique qui **permettent de stocker et de gérer une collection d'éléments de manière ordonnée**.
Elles offrent un moyen flexible et efficace d'organiser et de manipuler des données dans un programme.

Caractéristiques des listes:

- **Structure ordonnée**: Les éléments d'une liste sont stockés dans un ordre défini, généralement de la première à la dernière position.
- **Accès par index**: Chaque élément de la liste est accessible par un index, qui correspond à sa position dans la liste.
- **Dynamisme**: La taille d'une liste peut évoluer dynamiquement pendant l'exécution du programme, en ajoutant ou supprimant des éléments.

Opérations courantes sur les listes:



- **Ajouter un élément**: Permet d'insérer un nouvel élément à la fin de la liste ou à une position spécifique.
- **Supprimer un élément**: Permet de retirer un élément de la liste en fonction de sa valeur ou de son index.
- **Rechercher un élément**: Permet de trouver un élément spécifique dans la liste et de connaître sa position.
- **Parcourir la liste**: Permet d'itérer sur tous les éléments de la liste successivement.
- **Modifier un élément**: Permet de modifier la valeur d'un élément de la liste en fonction de son index.

Les listes (tableaux dynamiques) permettent de stocker et de gérer des données de manière ordonnée dans un programme informatique. Elles sont **caractérisées par une structure ordonnée, un accès par index et un dynamisme de leur taille**. Les opérations courantes sur les listes **incluent l'ajout, la suppression, la recherche, le parcours et la modification d'éléments**.

	STRUCTURE, FONCTIONNEMENT, COMPORTEMENT : des objets et systèmes techniques à comprendre	Technologie SFC 3 f
	La programmation d'une nouvelle fonctionnalité	
Connaissance	**Algorithmique et programmation : séquences (bloc) d'instructions**	

Lorsqu'on programme un objet ou système technique, on décide **des instructions qu'il doit accomplir**. Ces instructions sont **organisées** en **séquences d'instructions,** appelées aussi **blocs d'instructions.**

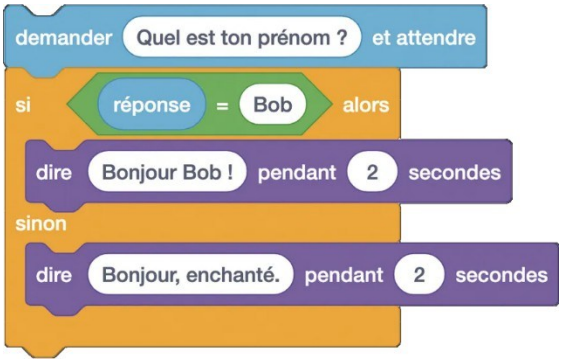
Une séquence d'instructions, aussi appelée bloc d'instructions, est un **ensemble d'instructions regroupées** et **exécutées les unes après les autres dans un ordre défini**. Elle représente la structure de base pour organiser et contrôler le déroulement d'un programme.

Les types de séquences d'instructions

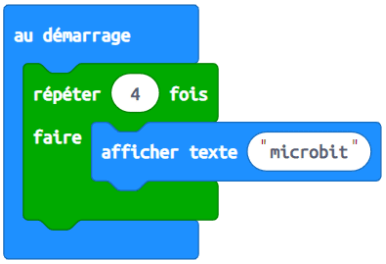
- **Séquence simple**: Suite d'instructions exécutées les unes après les autres sans condition particulière.



- **Séquence avec instructions conditionnelles**: Permet de modifier le flux d'exécution en fonction de conditions booléennes (ex: instructions « si »-« sinon »).



- **Séquence avec boucles** : Permet de répéter un bloc d'instructions un nombre défini de fois ou jusqu'à ce qu'une condition soit remplie (ex: instruction « répéter x fois »).



Lors de la programmation, **les instructions sont organisées en séquences** (blocs d'instructions) **pour contrôler le déroulement du programme**. 3 types de séquences :

- La **séquence simple**: instructions exécutées les unes après les autres.
- La **séquence conditionnelle**: flux d'exécution modifié selon des conditions (ex: "si-sinon").
- La **séquence avec boucles**: répétition d'un bloc d'instructions un nombre défini de fois ou jusqu'à une condition.

	STRUCTURE, FONCTIONNEMENT, COMPORTEMENT : des objets et systèmes techniques à comprendre	Technologie SFC 3 g
	La programmation d'une nouvelle fonctionnalité	
Connaissance	**Algorithmique et programmation : événement**	

Lorsqu'on utilise un objet technique, différentes **actions** peuvent être **déclenchées** en détectant un signal, en appuyant sur des boutons ou en faisant des gestes. Ces différentes actions sont provoquées par des **événements.**

Ces événements permettent aux objets et systèmes techniques de **réagir** à notre **environnement** et à nos **interactions.**



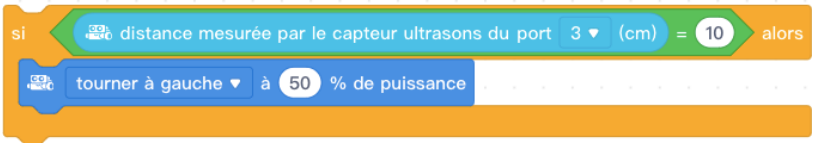
Un événement est un phénomène qui se produit et qui peut déclencher une action dans un programme. Il existe différents types d'événements :



- **Événements physiques :** Ils correspondent à une action que l'utilisateur effectue sur l'objet: comme appuyer sur un bouton ou bouger un objet.



- **Événements temporels:** Ils se déclenchent automatiquement après un certain temps.



- **Événements logiques:** Ils se déclenchent lorsqu'une condition est remplie, comme la détection d'un obstacle par un robot.

Pour déclencher des actions grâce à des événements dans un programme, il faut :

- Identifier les différents types d'événements dans un programme.
- Déterminer l'action associée à chaque événement.
- Programmer des événements pour créer des interactions avec l'objet ou le système technique.

En programmation, afin que **les objets et systèmes techniques s'adaptent et réagissent à notre environnement**, on utilise des **événements**. **Un événement est un phénomène qui se produit et qui peut déclencher une action dans un programme**.

	STRUCTURE, FONCTIONNEMENT, COMPORTEMENT : des objets et systèmes techniques à comprendre	Technologie SFC 3 h
	La programmation d'une nouvelle fonctionnalité	
Connaissance	**Algorithmique et programmation : déclenchement d'une séquence d'instructions par un événement**	

Pour rendre un programme réactif à son environnement et **aux interactions avec l'utilisateur**, on met en place des **déclenchements de séquences d'instructions par des événements.**

Un événement est un **changement qui survient dans l'environnement d'un objet ou d'un système**.
Il peut s'agir :

- d'une **action humaine** (clic sur un bouton)
- d'un **phénomène naturel** (détection de lumière)
- d'une **information reçue** (signal d'un capteur)



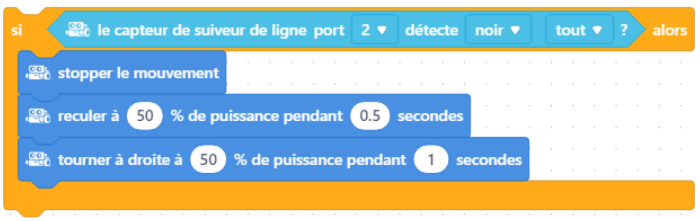
Une **séquence d'instructions** est un **ensemble d'ordres donnés à un objet ou à un système pour qu'il effectue une tâche**.

Ici, on demande à un système programmable de jouer une note, puis d'allumer ses LED en rouge, puis d'avancer à 50% de sa puissance.



Le **déclenchement d'une séquence d'instructions par un événement** est le principe qui permet de rendre les objets et les systèmes techniques interactifs.





Les événements (actions, phénomènes, informations) permettent de **déclencher des séquences d'instructions** dans un programme, **rendant l'objet ou le système interactif**. Un **événement** peut être une **action humaine**, un **phénomène naturel** ou une **information reçue**. La **séquence d'instructions** définit les **actions à effectuer en réponse à l'événement** (ex: jouer une note, allumer une LED, avancer).

	STRUCTURE, FONCTIONNEMENT, COMPORTEMENT : des objets et systèmes techniques à comprendre	Technologie SFC 3 i
	La programmation d'une nouvelle fonctionnalité	
Connaissance	**Algorithmique et programmation : entrées ou sorties d'un programme (données issues par exemple de capteurs IHM et sorties pouvant être en lien avec un actionneur, fichiers)**	

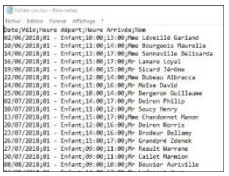
Les **entrées et sorties (I/O)** sont des éléments essentiels en algorithmique et programmation qui **permettent à un programme d'interagir avec le monde extérieur**. Elles constituent le **pont entre le programme et son environnement**, en lui fournissant des données et en lui permettant d'agir sur son environnement.

Les entrées d'un programme



Elles peuvent provenir de différentes sources, par exemple :

- **Données utilisateurs :** Saisie par l'utilisateur via un clavier, une souris, un écran tactile ou tout autre périphérique d'entrée.
- **Données provenant de capteurs :** Données issues de capteurs physiques mesurant des grandeurs physiques (température, pression, lumière, etc.).



- **Fichiers:** Lecture de données stockées dans des fichiers texte, CSV, images, etc.

Les sorties d'un programme



Elles peuvent se présenter sous différentes formes :

- **Affichage à l'écran :** Texte, graphiques, images, etc., affichés sur un écran ou un autre périphérique d'affichage.
- **Actionneurs :** Contrôle de dispositifs physiques tels que des moteurs, des lumières, des haut-parleurs, etc.



- **Fichiers :** Écriture de données dans des fichiers texte, CSV, images, etc., pour le stockage ou le traitement ultérieur.

Pour pouvoir lire, exploiter et modifier le programme d'un système, il faut:

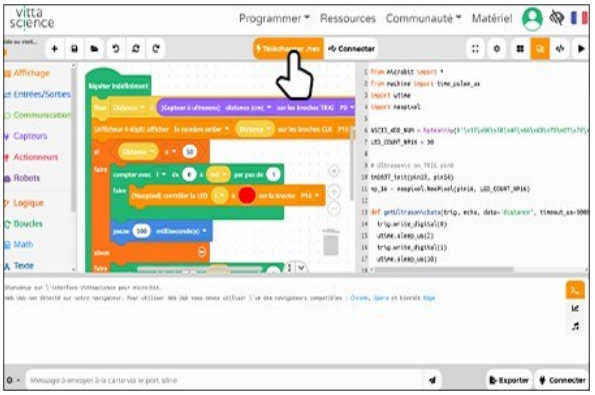
- **Identifier les entrées et les sorties** de ce système dans le programme**.**
- **Comprendre** le lien **entre les entrées et les sorties** dans le programme**.**
- **Utiliser des instructions d'entrée et de sortie** afin de pouvoir **modifier** un programme**.**

Les **entrées et sorties** sont essentielles en algorithmique et programmation car elles **permettent aux programmes d'interagir avec le monde extérieur**, d'**échanger des données** et de **réaliser des actions concrètes**. Les **entrées sont des données provenant de l'utilisateur**, de **capteurs ou de fichiers**. Les **sorties** permettent : des **affichages** à l'écran, des **contrôles d'actionneurs** ou **écritures dans des fichiers**.

	STRUCTURE, FONCTIONNEMENT, COMPORTEMENT : des objets et systèmes techniques à comprendre	Technologie SFC 3 k
	La programmation d'une nouvelle fonctionnalité	
Connaissance	**Lien entre la programmation graphique par bloc et la programmation textuelle (fin de 3e)**	

Lorsque l'on utilise un smartphone ou que l'on joue à des jeux vidéo, on interagit avec des **programmes informatiques**. Ces programmes sont créés à partir de **langages de programmation**, qui permettent de donner des instructions à un ordinateur. Il existe deux types principaux de langages de programmation : **la programmation par blocs** et la **programmation textuelle**.

- **Programmation textuelle :** elle utilise des lignes de code écrites avec des mots-clés et des symboles spécifiques. C'est comme parler à l'ordinateur dans son propre langage.
  Exemples : JavaScript, Python, C/C++ …
- **Programmation par blocs :** elle utilise des blocs graphiques colorés et emboîtables, comme des pièces de construction. C'est comme assembler des instructions visuelles pour que l'ordinateur les comprenne. *Exemples : Scratch, Makecode, Vittascience …*



Exemple avec la maquette de lampadaire équipée d'une carte programmable Micro:Bit.

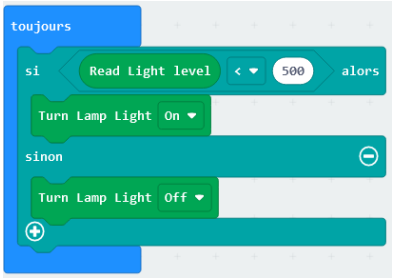


Algorithme associé : **si** le lampadaire détecte une lumière faible, **alors** il s'allume, **sinon** il s'éteint.

- **Avec la programmation textuelle**, on peut créer des programmes plus complexes et puissants, mais il faut apprendre la syntaxe du langage et écrire du code précis (exemple : langage JavaScript).

```
basic.forever(function () {
    if (Kitronik_LAMPbit.lightLevel() < 500) {
        Kitronik_LAMPbit.lampLightLED(Kitronik_LAMPbit.DisplayLamp.On)
    } else {
        Kitronik_LAMPbit.lampLightLED(Kitronik_LAMPbit.DisplayLamp.Off)
    }
})
```

- **Avec la programmation par blocs**, on peut créer des programmes plus simples et intuitifs, plus faciles à apprendre et à utiliser, surtout pour les débutants (exemple : programmation par blocs avec Makecode).



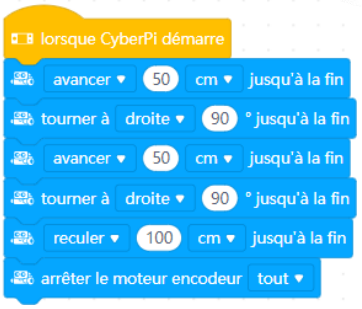
La programmation par blocs est un bon point de départ pour apprendre la programmation, car elle **permet de comprendre les concepts de base** sans se soucier de la syntaxe compliquée. Ensuite, **il est intéressant de passer à la programmation textuelle pour créer des programmes plus avancés**, mais il faudra **apprendre le langage de programmation comme le Python** par exemple.

	STRUCTURE, FONCTIONNEMENT, COMPORTEMENT : des objets et systèmes techniques à comprendre	Technologie SFC 3 j
	La programmation d'une nouvelle fonctionnalité	
Connaissance	**Programmation graphique par blocs**	

Lorsqu'on veut donner des instructions à un robot ou à un jeu vidéo, on peut utiliser **la programmation graphique par blocs**. C'est un moyen **simple et visuel** de **créer des programmes**, sans avoir à écrire du code.



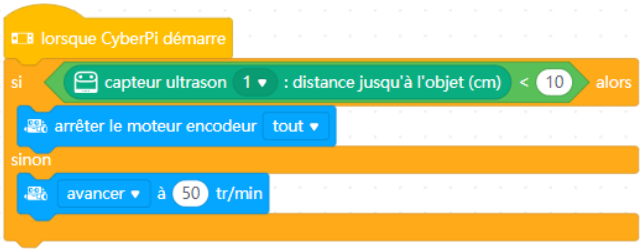
La programmation par blocs **fonctionne avec des blocs de code qui s'emboîtent les uns dans les autres**. Chaque bloc **représente une action ou une condition**. En **assemblant les blocs**, on crée un programme qui **définit le comportement de l'objet ou du système technique**.



Il existe différents types de blocs :

- **Blocs d'action**

Définissent **les actions que l'objet ou le système doit réaliser** (ex: avancer, tourner, afficher un message).



- **Blocs de condition**

Définissent les **conditions qui doivent être remplies** pour que les actions soient exécutées (ex: si l'objet est proche d'un obstacle, alors s'arrêter).



- **Blocs de boucle**

Définissent **la répétition** d'une série d'actions (ex: répéter 4 fois avancer et tourner).

Pour programmer par bloc, il faut savoir

- **Identifier les différents types de blocs**.
- **Assembler les blocs** pour **créer un programme simple**.
- **Tester** un programme.
- **Modifier les paramètres** d'un programme et identifier ses effets.

La **programmation par blocs est un moyen simple et visuel de créer des programmes sans écrire de code**. Elle utilise des **blocs de code emboîtables** qui représentent **des actions, des conditions ou des boucles**. Pour programmer par blocs, il faut **identifier les types de blocs**, **les assembler**, **tester le programme** et **modifier ses paramètres**.