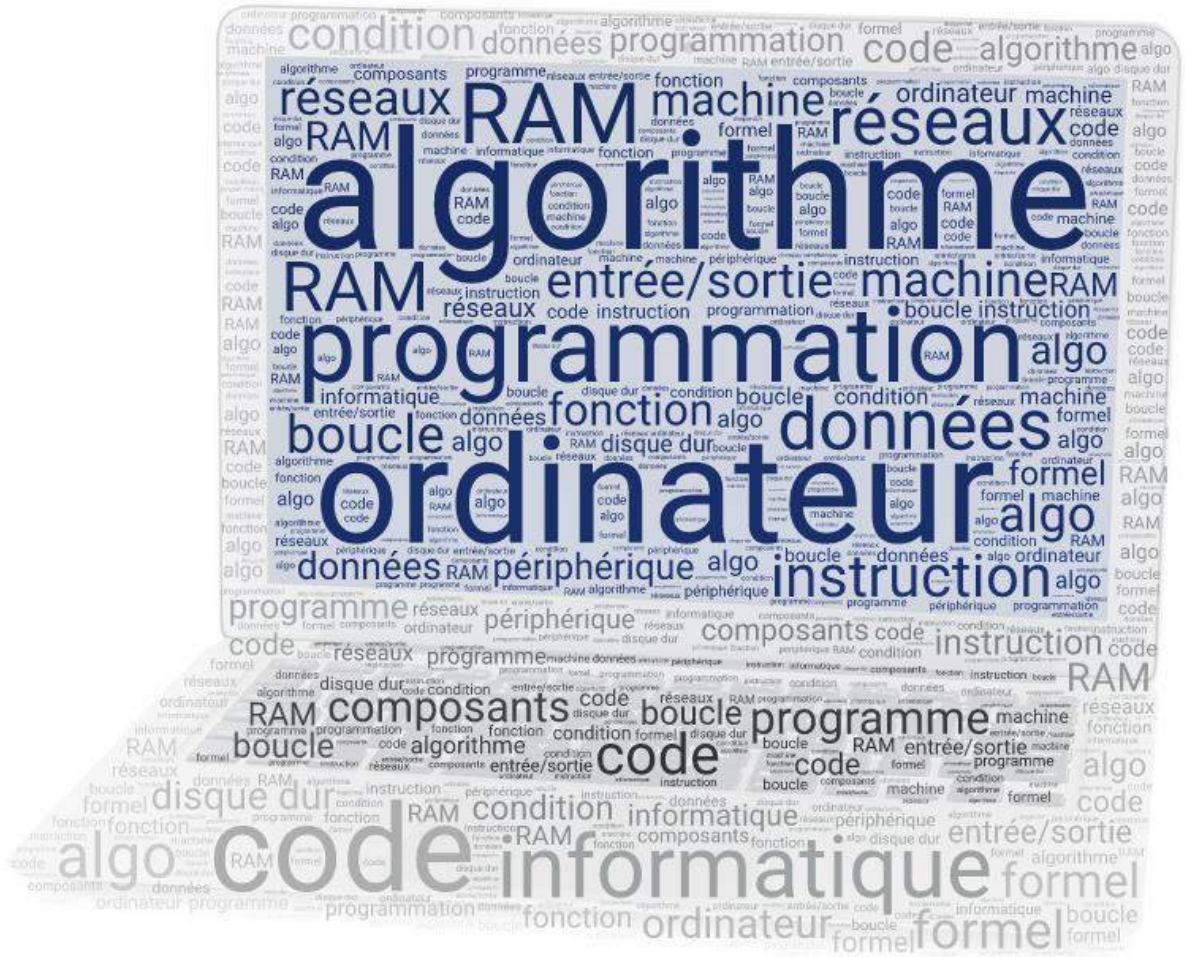
Initiation à la programmation



Nom :
Prénom :
Classe :

#wallcode | digital wallonia .be

Une partie de ces notes sont inspirées de plusieurs resources libres : http://visatice.ulg.ac.be, https://pixees.fr/classcode-v2/, https://code.org, https://docs.cs50.net/, http://csunplugged.org/

Retrouvez toutes les informations sur https://sicarre.be/ ou contactez info@sicarre.be

1 Un ordinateur, cette machine stupide

1.1 Les objectifs

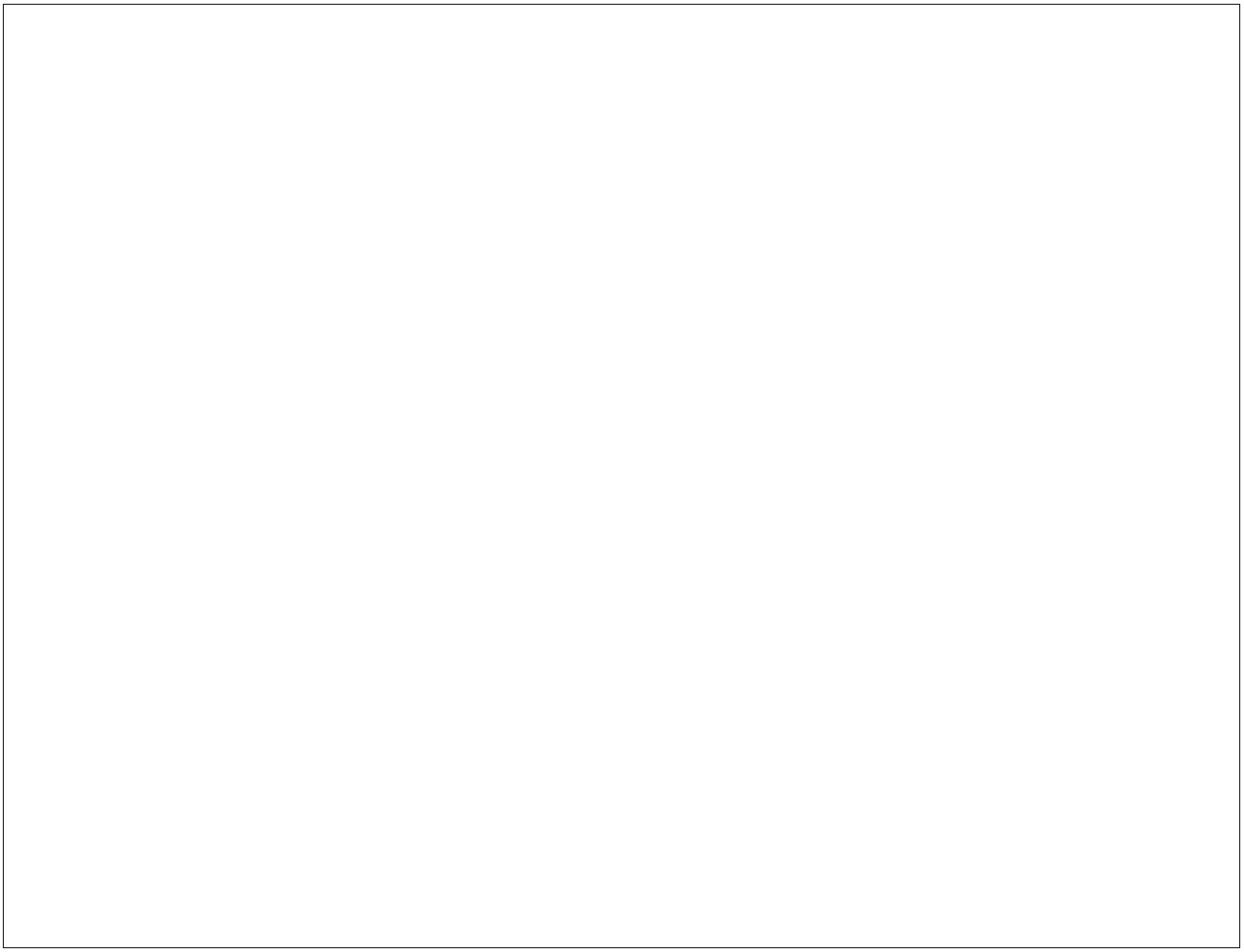
À la fin de la séance, tu seras capable de :

Comprendre le rôle des différents composants d'un ordinateur dans l'exécution d'un programme

- Lister les composants de base d'un ordinateur
- Expliquer la fonction de chaque composant de base d'un ordinateur

1.2 Activité 1 **:** Les composants de l'ordinateur

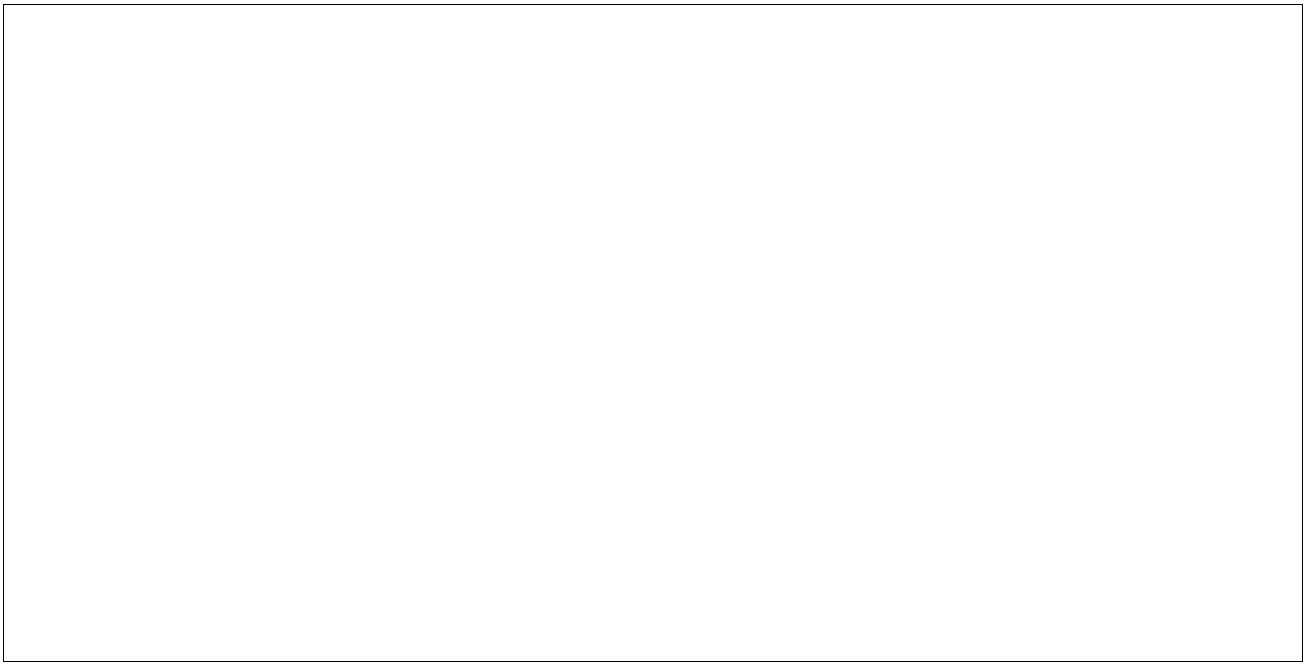
A. Mes représentations: Pour moi, qu'est-ce qu'un ordinateur?



B. Est-ce que ceci est un ordinateur ?

Entoure les images qui correspondent à des ordinateurs.

- De quoi ces objets sont-ils constitués ?
- Quels sont les points communs ?
- Quel critère de classification as-tu utilisé ?
- Quelles sont les parties essentielles et celles dont on peut se passer ?

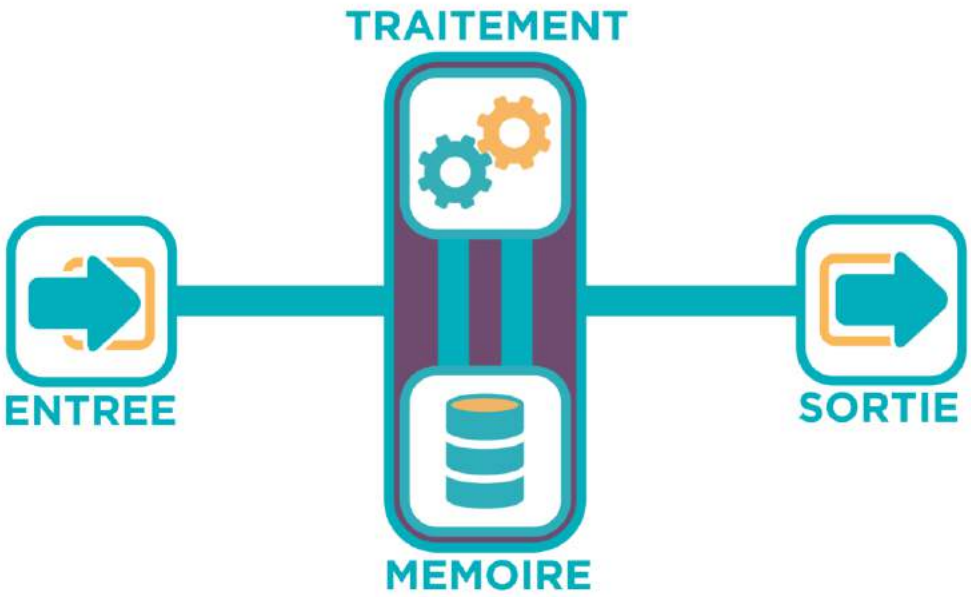


C. Observation d'une machine.

Objectifs:

- Découvrir les composants d'un ordinateur
- Expliquer la fonction de chaque composant de base d'un ordinateur

Essayons de classer les différents composants de l'ordinateur dans ce schéma :



1.3 Activité 2 : Les limites de l'ordinateur.

Voici un texte écrit en polonais.

Komputery, a inne maszyny liczące

Komputer od tradycyjnego kalkulatora odróżnia zdolność wykonywania wielokrotnie, automatycznie powtarzanych obliczeń, wg algorytmicznego wzorca zwanego programem, gdy tymczasem kalkulator może zwykle wykonywać tylko pojedyncze działania. Granica jest tu umowna, ponieważ taką definicję komputera spełniają też kalkulatory programowalne (naukowe, inżynierskie), jednak kalkulatory służą tylko do obliczeń matematycznych, podczas gdy nazwa komputer najczęściej dotyczy urządzeń wielofunkcyjnych.
Jakkolwiek istnieją mechaniczne urządzenia liczące, które potrafią realizować całkiem złożone programy, zazwyczaj nie zalicza się ich do komputerów. Warto jednak pamiętać, że prawzorem komputera była maszyna Turinga, którą można by wykonać w całości z części mechanicznych, a pierwsze urządzenia ułatwiające obliczenia były znane w starożytności, np. abakus z 440 p.n.e..

À moins que vous ne connaissiez cette langue, vous êtes incapables de lui attribuer le moindre sens.

Vous êtes donc comme un ordinateur face à de l'information numérisée.

Cependant, cela ne signifie pas que vous êtes incapables d'effectuer certains traitements sur ce texte.

Parmi les épreuves qui suivent :

- quelles sont celles que vous allez réussir sans aucun problème ? **(A)**
- quelles sont celles que vous êtes incapables de réussir ou seulement difficilement ? **(B)**

	A	B
1. Dire combien il compte de lettres		
2. Écrire tout le texte en lettres capitales		
3. Y trouver le mot "maszyna"		
4. Dire combien il compte de noms communs		
5. Surligner le deuxième paragraphe		
6. Dire combien la première phrase de ce paragraphe compte de mots		
7. Synthétiser le texte en deux lignes maximum		
8. Souligner les mots qui désignent des oiseaux		
9. Dire quelle est la lettre la plus fréquente		
10. Dire combien de fois la chaîne (suite) de caractères « części » apparaît dans le texte		
11. Mettre le texte au futur simple		

1.4 **Structuration**

Lors de ces deux activités, nous avons mis en place un vocabulaire précis.
Essaie d'associer les définitions suivantes aux notions étudiées.

Définitions:

1	La mémoire vive est généralement appelée RAM pour Random Access Memory ce qui signifie mémoire à accès aléatoire, entendez "accès direct". La mémoire vive ou RAM accueille, de manière temporaire toutes les informations (données et programmes) qui viennent progressivement enrichir les capacités du système.
2	Le processeur ou CPU pour "Central Processing Unit" a essentiellement pour tâche de : 1. lire des données en mémoire ou dans les I/O (Entrées/Sorties en anglais) 2. traiter les données 3. écrire des données en mémoire ou dans les entrées/sorties Les performances des CPU évoluent très rapidement et dépendent du nombre d'instructions qu'il peut exécuter par cycle d'horloge et du nombre de bits qu'il peut traiter en parallèle.
3	On y enregistre les données et les programmes que l'ordinateur est susceptible de traiter.
4	Un dispositif connecté à un système de traitement de l'information central (ordinateur, console de jeux, etc.1) et qui ajoute à ce dernier des fonctionnalités.
5	Servent à fournir des informations (ou données) au système informatique tel que clavier (frappe de texte), souris (pointage), scanner (numérisation de documents papier), micro, webcam, etc.
6	Servent à faire sortir des informations du système informatique tel que écran, imprimante, haut-parleur, etc
7	Une méthode, décrite de manière assez claire et rigoureuse pour qu'elle puisse être appliquée sans ambiguïté par un humain, ou par une machine, ou même par un extra-terrestre (s'il peut lire le Français… et encore… on n'a pas besoin de tout le dictionnaire, mais juste de quelques mots).
8	Algorithme qui a été traduit en instructions compréhensibles par la machine, pour que la machine puisse l'exécuter.
9	Traitements que nous pouvons appliquer à une information dépourvue de sens.

Notions:

A	Périphérique
B	Périphérique de sortie
C	Processeur
D	RAM
E	Algorithme
F	Traitement formel
G	Périphérique d'entrée
H	Disque dur
I	Programme

1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**

Donnons maintenant une définition d'ordinateur :

Un ordinateur, c'est …

……

……

……

……

……

……

1.5 Retour à tes representations

Retourne à tes representations et à tes choix de classification pour les images de l'exercice 1.2. Sur base de ce que tu as appris et de la définition sur laquelle la classe s'est accordée, est-ce que tu changerais des images de classification ? Est-ce que ton idée de ce qu'est un ordinateur est maintenant plus précise ?

2 Non ce n'est pas de la magie.

2.1 Les objectifs.

À la fin de la séance, l'apprenant sera capable de :

Lire et comprendre un algorithme simple permettant de résoudre un problème donné.

- Expliquer un algorithme simple
- Simuler l'exécution d'un algorithme simple

2.2 Activité : Un tour de magie.

A. Consignes du tour de magie.

- L'activité se déroule par groupe de 3 élèves.
- Chaque groupe reçoit un jeu de cartes (21 cartes)
- Le groupe réalise le tour de magie de la façon suivante :
    - Un élève joue la "mémoire" et lit les instructions ci-dessous une par une.
    - Un élève joue le "processeur" et exécute les instructions données par la mémoire.
    - Un élève est le "participant".

Instructions :

Mélanger 21 cartes à jouer
Prendre un participant au hasard
Donner les cartes au participant
Demander au participant de retenir une des cartes dans sa tête
Fermer les yeux
Demander au participant de la montrer aux autres
Ouvrir les yeux
Reprendre les cartes au participant
Dire au participant de regarder les cartes quand elles sont distribuées
Distribuer trois cartes côte à côte, face visible
Empiler les cartes restantes de la même façon sur les tas déjà existant
Demander au participant de dire dans quel tas se trouve sa carte
Empiler les trois tas en mettant le tas indiqué au milieu
Distribuer trois cartes côte à côte, face visible
Empiler les cartes restantes de la même façon sur les tas déjà existant
Demander au participant de dire dans quel tas se trouve sa carte
Empiler les trois tas en mettant le tas indiqué au milieu
Distribuer trois cartes côte à côte, face visible
Empiler les cartes restantes de la même façon sur les tas déjà existant
Demander au participant de dire dans quel tas se trouve sa carte
Récupérer la carte au milieu de ce tas
Dire : "Tadaa : C'est cette carte-là."

B. Mise en commun et structuration.

3 L'algorithme de la tartine.

3.1 Les objectifs.

À la fin de la séance, l'apprenant sera capable de :
Concevoir un algorithme simple permettant de résoudre un problème donné

- Écrire un algorithme comme une suite d'instructions
- Identifier les instructions de base à disposition
- Définir des instructions non ambiguës

3.2 Activité : Algo de la tartine.

A. Consignes

L'activité se déroule de manière individuelle.

Votre professeur se transforme pour une durée déterminée en robot.

Il va devoir réaliser la tâche suivante : "**Faire une tartine beurrée à la confiture**".

Pour cela, il dispose du matériel suivant:

- une tartine
- beurre
- confiture en pot
- un couteau
- une assiette

Quelles sont les instructions que tu dois lui donner afin qu'il puisse réaliser cette tâche sans difficulté?

Note ces instructions sur ta feuille.

B. Mise en commun et structuration.

4 Cherchons le trésor.

4.1 Les objectifs.

À la fin de la séance, l'apprenant sera capable de :

Lire et comprendre un algorithme simple permettant de résoudre un problème donné
- Expliquer un algorithme simple
- Simuler l'exécution d'un algorithme simple
- Adapter un algorithme simple existant

Concevoir un algorithme simple permettant de résoudre un problème donné
- Écrire un algorithme comme une suite d'instructions
- Identifier les instructions de base à disposition
- Définir des instructions non ambiguës
- Utiliser des structures de contrôles (instructions conditionnelles ou boucles) de manière appropriée dans un algorithme
- Combiner des instructions, des structures de contrôle pour écrire un algorithme

Utiliser un algorithme simple pour résoudre un problème donné
- Identifier une suite d'instructions qui constituent un tout réutilisable

4.2 Activité : Chercher le trésor

A. Consignes

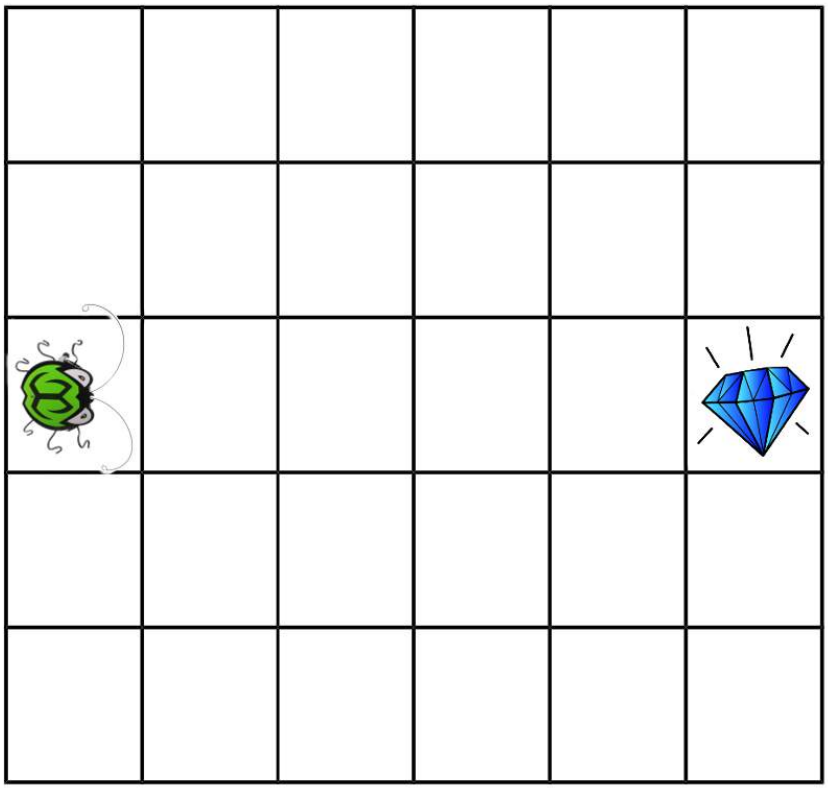
L'activité se déroule de manière individuelle.

Pour chacune des grilles suivantes, rédige des instructions claires au travers d'un alogrithme pour que *beetlebot le robot atteigne le diamant*.

= beetlebot

= le diamant

Grille n°1 (5X6)

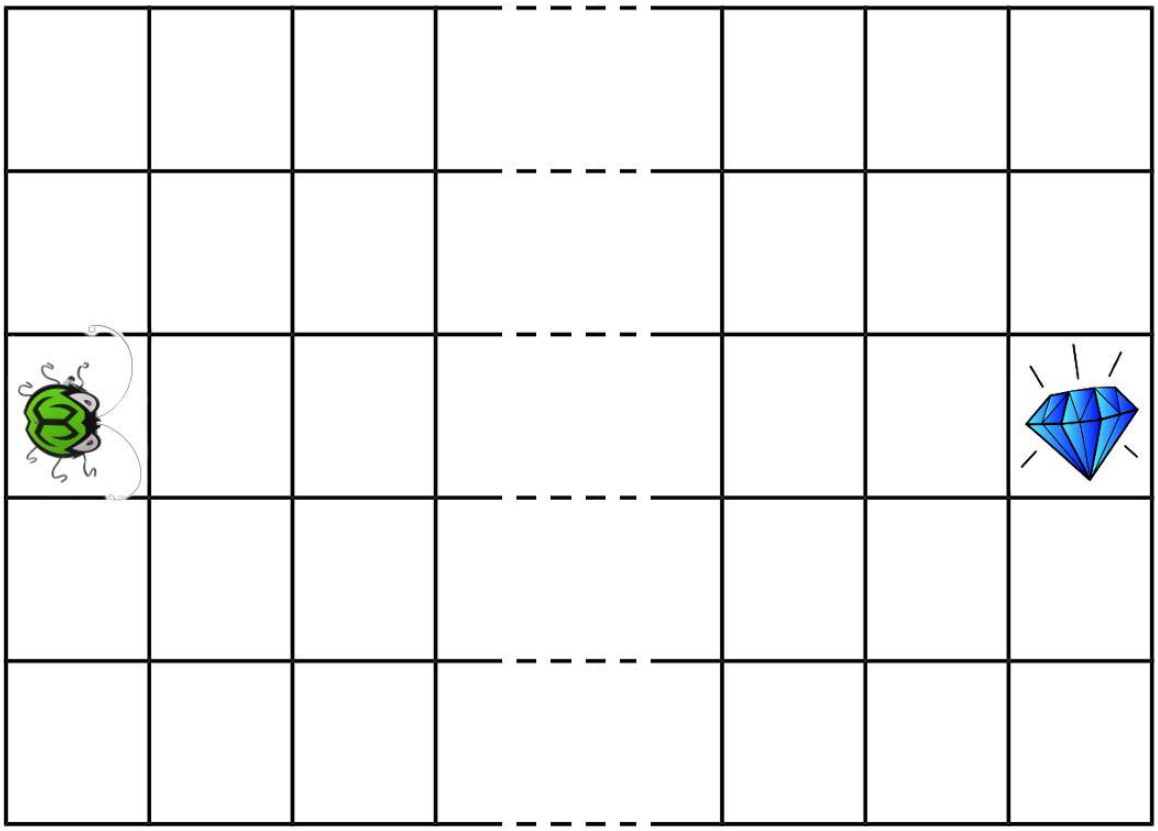


Mets-toi d'accord avec ton voisin sur les instructions de base que vous allez utiliser

Instructions:

Mise en commun:

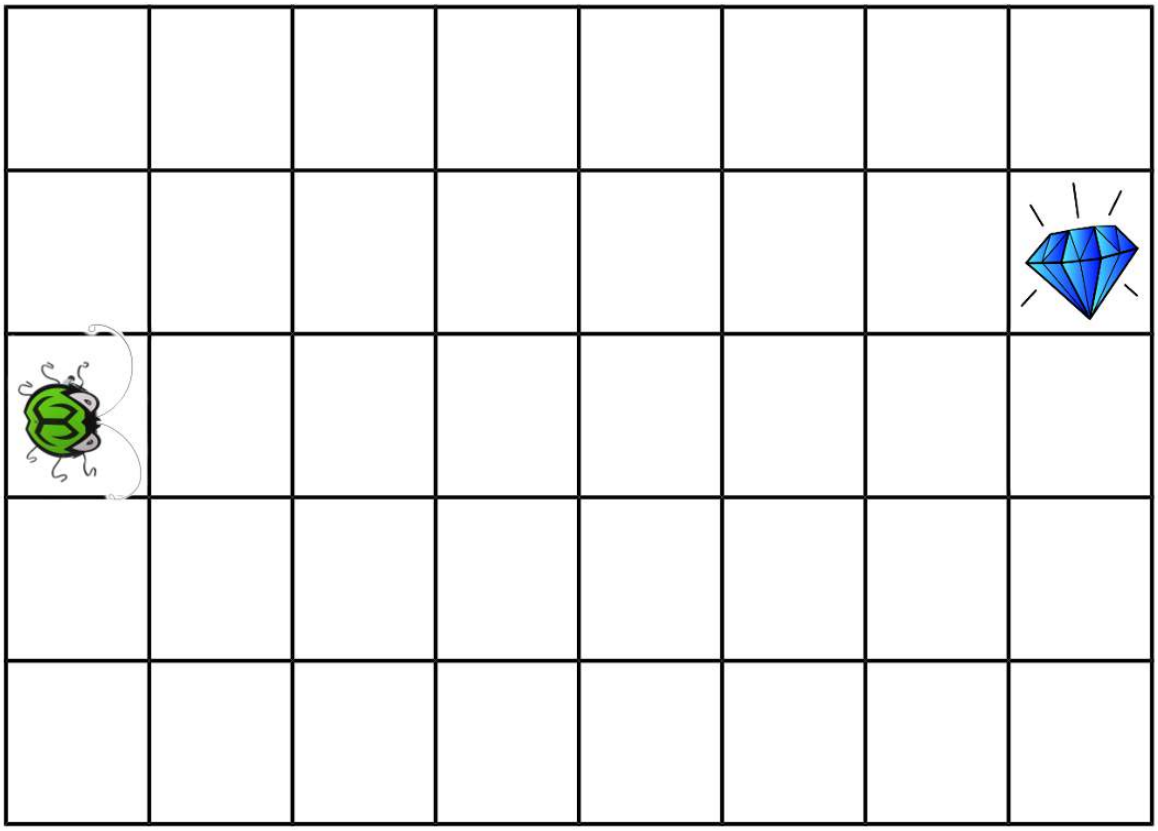
Grille n°2 (5X?)



Instructions:

Mise en commun:

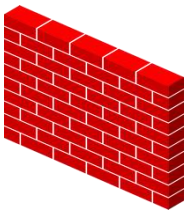
Grille n°4 (5X8)



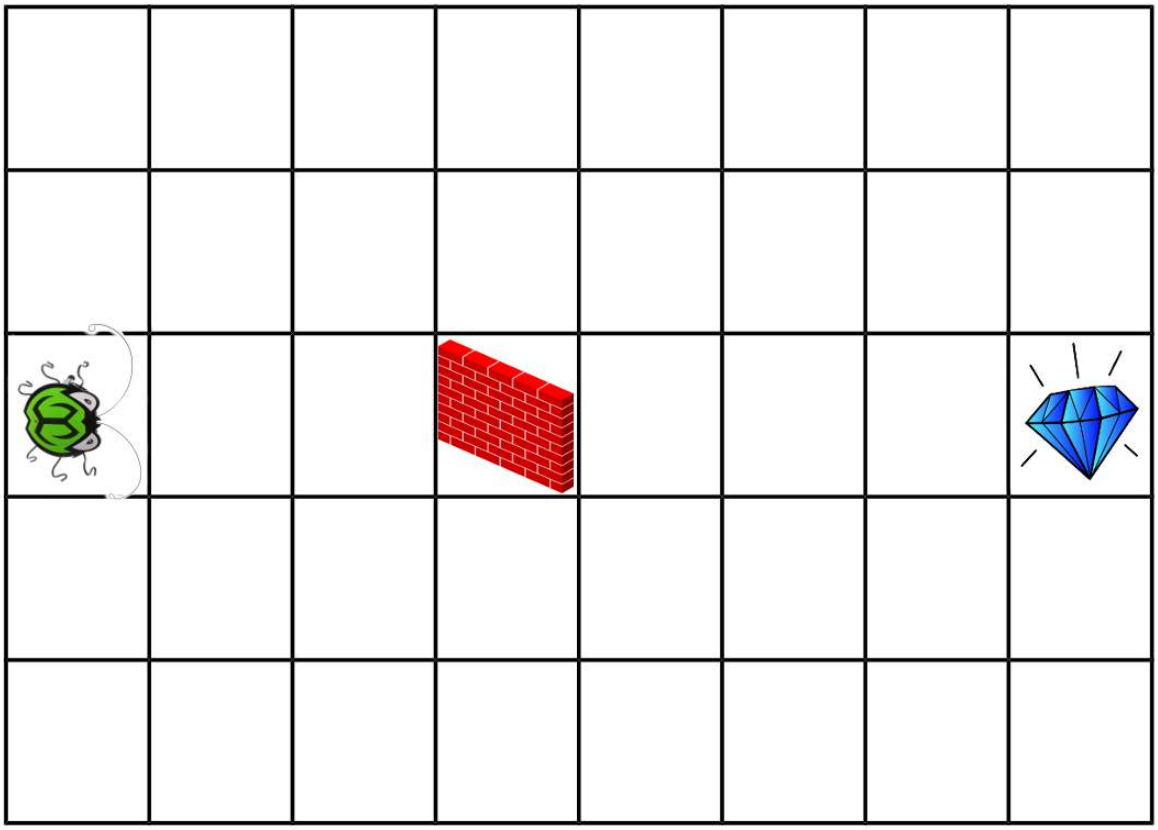
Instructions:

Mise en commun:

Grille n°5 (5X8)



= Un obstacle infranchissable pour une souris!



Instructions:

B. Mise en commun et structuration.

5 Programmation par blocs

5.1 Les objectifs.

À la fin de la séance, l'apprenant sera capable de :

Lire et comprendre un algorithme simple permettant de résoudre un problème donné

- Expliquer un algorithme simple
- Simuler l'exécution d'un algorithme simple
- Adapter un algorithme simple existant

Concevoir un algorithme simple permettant de résoudre un problème donné

- Écrire un algorithme comme une suite d'instructions
- Identifier les instructions de base à disposition
- Définir des instructions non ambiguës
- Définir et utiliser des expressions de manière appropriée dans un algorithme
- Définir et utiliser des variables de manière appropriée dans un algorithme
- Utiliser des structures de contrôles (instructions conditionnelles ou boucles) de manière appropriée dans un algorithme
- Combiner des instructions, des structures de contrôle, des variables pour écrire un algorithme

Utiliser un algorithme simple pour résoudre un problème donné

- Identifier les entrées/sorties d'un algorithme simple
- Discuter des conditions d'utilisation d'un algorithme simple
- Comparer l'efficacité de deux algorithmes
- Identifier une suite d'instructions qui constituent un tout réutilisable
- Réutiliser une suite d'instructions en définissant une fonction

Comprendre qu'un langage de programmation est une façon de décrire un algorithme de sorte qu'il soit compréhensible par une machine

- Identifier la correspondance entre les instructions de base d'un algorithme et celles du langage de programmation à utiliser
- Identifier et utiliser les types de données adéquats
- Traduire un algorithme simple dans un langage de programmation
- Rédiger un programme simple dans un langage de programmation visuel
- Exécuter sur machine un programme simple

5.2 Déroulement de la séquence :

1. Créer un compte et se connecter avec le code donné par l'enseignant
2. Explication de l'interface :
    - où est l'énoncé (attention, il faut le lire à chaque fois),
    - comment les blocs s'emboitent,
    - la signification logique des couleurs (se construira au fur et à mesure),
    - la scène à gauche et le bouton démarrer/réinitialiser,
    - le bouton étape,
    - Comment passer à l'exercice suivant, à la leçon suivante.
3. Travailler par deux. Chacun son tour fait un exercice.

5.3 Planning

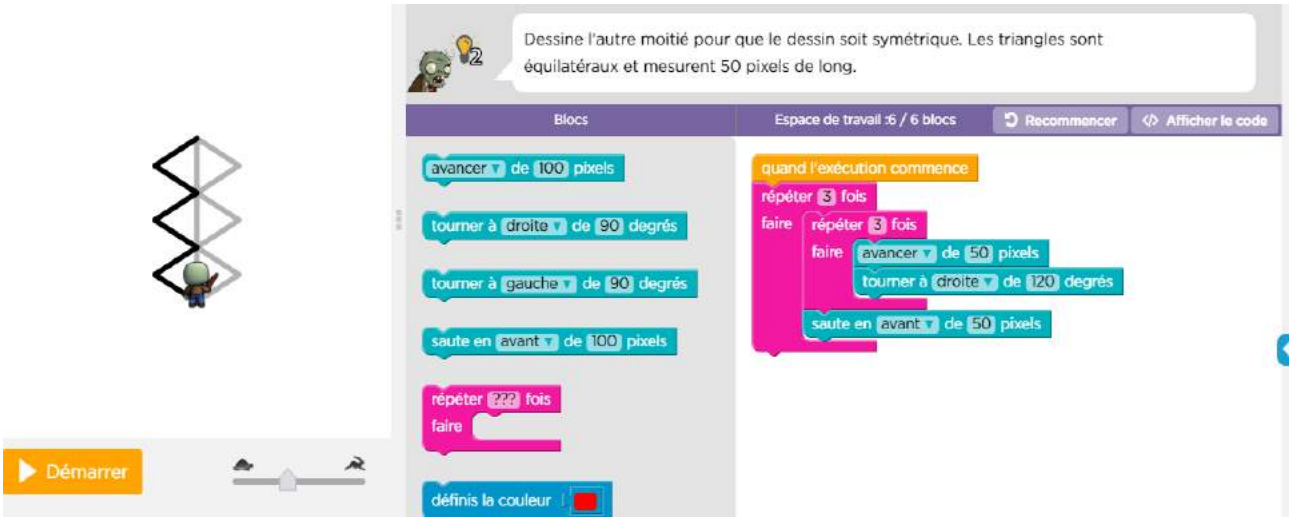
Cours 3 (https://studio.code.org/s/course3)

Leçons 2, 3, 5, 6, 8, 11, 13, 15

5.4 Synthèse : Programmation par blocs

Leçon 2 : *Labyrinthe zombie*
Instructions de bases et utilisation de boucles.

Leçon 3 : *Artiste*
Instructions de bases géométrie (angles, périmètres, ...)
Instructions avec paramètre, boucle (répéter X fois), boucles imbriquées



Certaines instructions s'exécutent à la suite les unes des autres : on parle de programmation séquentielle.

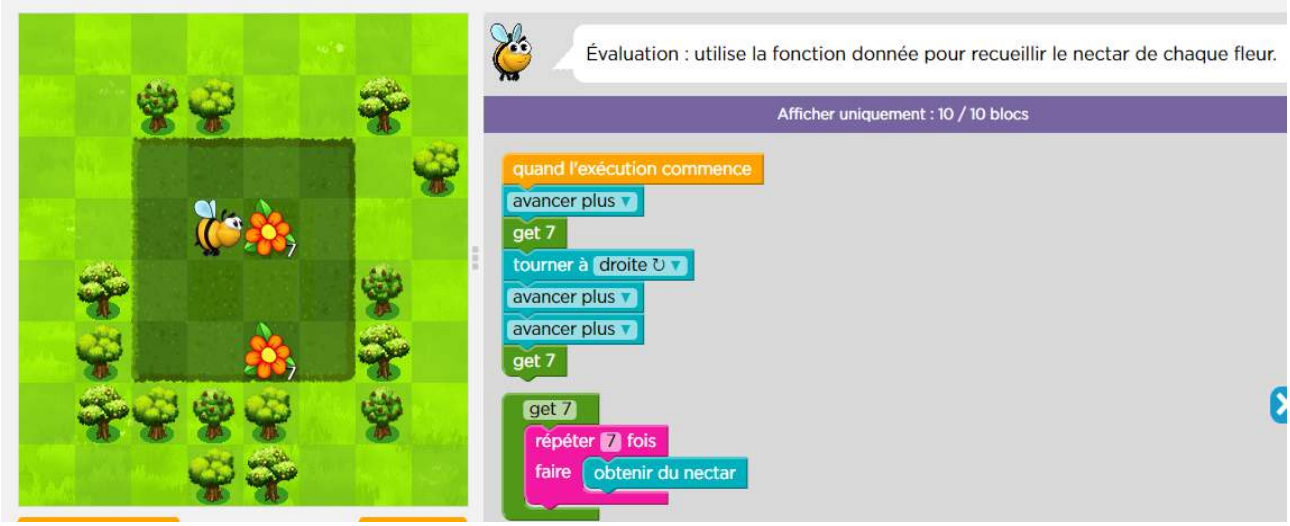
L'exécution d'un programme est reproductible (si les instructions ne changent pas, ni les données à manipuler, le programme donne toujours le même résultat)

L'ordinateur ne fait qu'exécuter les instructions qu'on lui donne : ni plus, ni moins.

Une boucle permet de répéter plusieurs fois la même action

Leçon 6 : *Abeille*

Fonctions (utilisation et édition et rédaction)



Une **fonction** sert à étendre le langage en créant de nouveaux blocs, de nouvelles instructions. Cela évite également de dupliquer du code. Ainsi, si une erreur se glisse dans une fonction ou si on veut la modifier, il ne faut modifier qu'à un endroit.

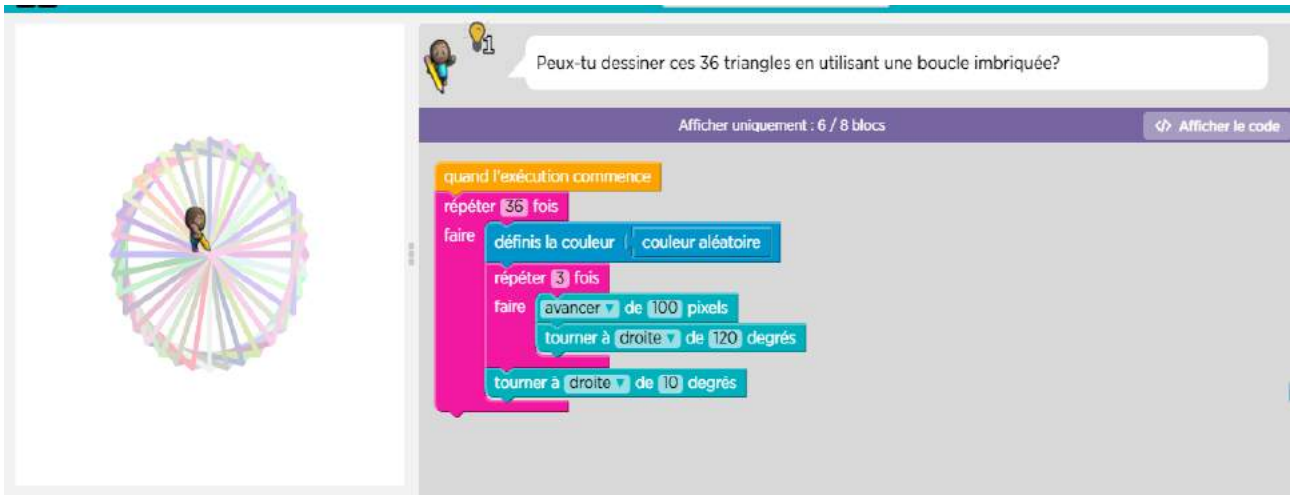
Leçon 8 : *Labyrinthe zombie*

Conditions

Leçon 11 : *Artiste*

Boucles imbriquées

Boucler en réutilisant du code donné et le bon angle



On travaille ici la lecture de code. Il faut comprendre le code donné avant de l'utiliser. Il est aussi important de bien se rendre compte dans les boucles imbriquées, à quelle partie du code (dans quelle boucle) correspond quel mouvement de l'artiste.

Leçon 13 : *Abeille*

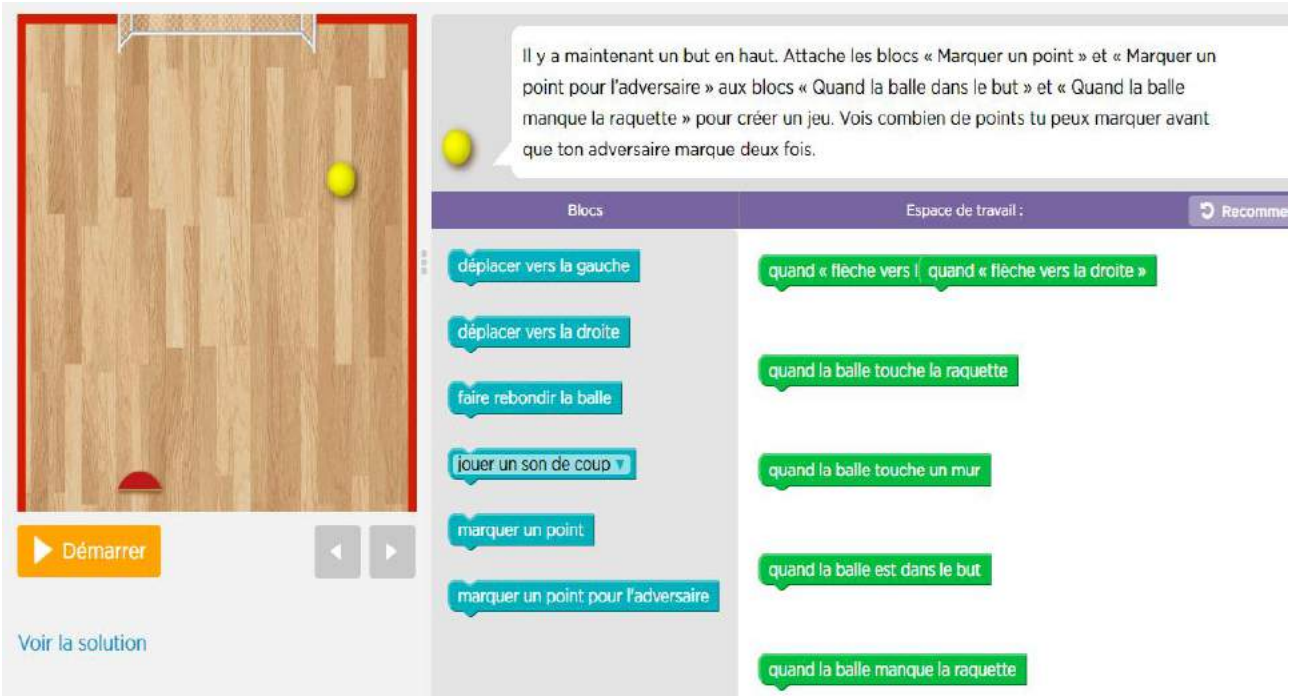
Boucles imbriquées, tant que

Certaines boucles, dites « conditionnelles », sont répétées jusqu'à ce qu'une condition soit remplie.

Leçon 15 : *Casse brique*

Cette leçon est importante à faire pour faciliter la transition vers Scratch qui est un mélange séquentiel/évènementiel.

Programmation évènementielle



Certaines instructions ne s'exécutent qu'au déclenchement d'un évènement : on parle de **programmation évènementielle**.