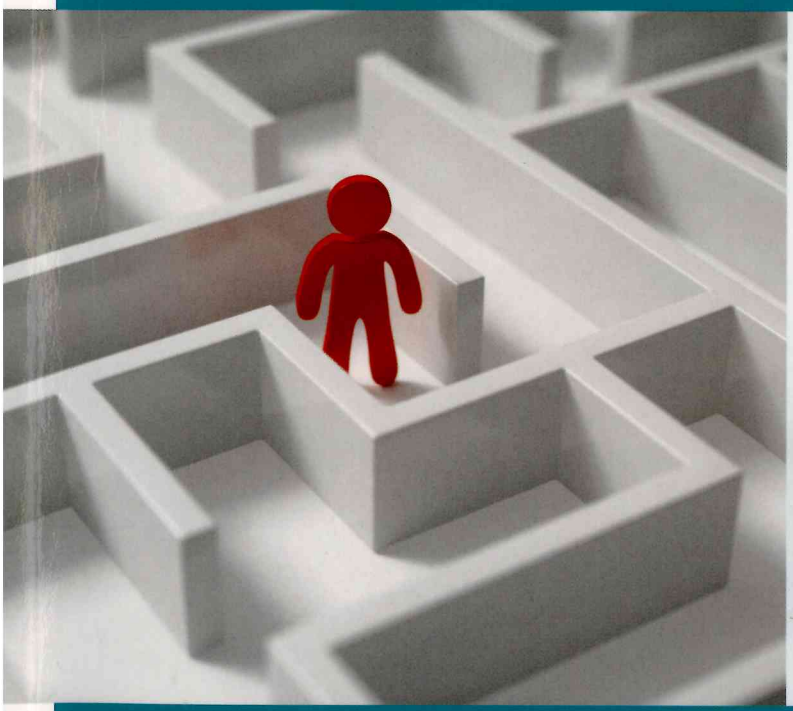
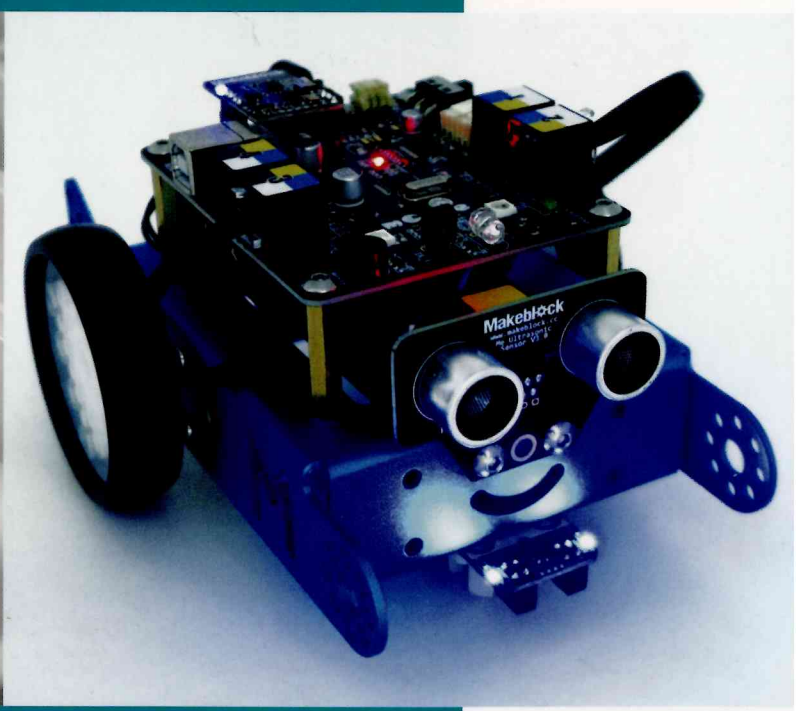
Cycle 4

Cahier d'algorithmique et de programmation

- Technologie
- Mathématiques





- Des activités d'apprentissage, des projets pour les EPI.
- Un entraînement progressif en vue du brevet.
- Des logiciels gratuits : Scratch et son extension robotique mBlock.
- Un robot simple et fiable sous Arduino.

Manuel numérique

DELAGRAVE

Cycle 4

Cahier d'algorithmique et de programmation

- Technologie
- Mathématiques

Sous la direction de Dominique Sauzeau

Grégory Anguenot, Julien Launay, professeurs de technologie

Robert Corne, Olivier Vogt, professeurs de mathématiques

DELAGRAVE

Matériel nécessaire pour l'exploitation du cahier :

- Logiciel Scratch : https://scratch.mit.edu/scratch2download/
- Logiciel mBlock (extension robotique pour Scratch) : http://learn.makeblock.cc/learning-scratch/
- Robot mBot sous Arduino (projets 3, 6, 9 et 12) : disponible chez différents fournisseurs tels a4 et Technologie services.

Maquette intérieure et couverture : Marion Clément/Domino
Mise en page et infographies : STDI
Iconographie : Sidonie Reboul

Vous utilisez un **Manuel connecté Delagrave**, qui propose des QR codes et/ou des liens hypertextes permettant d'accéder en ligne à des ressources numériques complémentaires.

Les éditions Delagrave font leurs meilleurs efforts pour sécuriser la consultation et l'utilisation des ressources en ligne qu'elles éditent, produisent ou hébergent, conformément aux règles d'usages de l'internet. Delagrave ne saurait être tenu responsable des interruptions de services dues aux caractéristiques et limites du réseau Internet, notamment dans le cas d'interruption quelle qu'en soit la cause, des performances techniques et des temps de réponse pour consulter ou interroger les ressources proposées. L'accès aux ressources associées au Manuel connecté Delagrave est garanti pour une période maximum de deux ans, à compter de la date de parution de cet ouvrage indiquée ci-dessous (copyright). Au terme de cette période, l'utilisateur du Manuel connecté Delagrave ne saurait exiger le maintien du service proposé.

Dans le cas où les QR codes et liens hypertextes permettent d'accéder à des sites internet tiers, la responsabilité des éditions Delagrave n'est pas engagée, notamment quant à leur éventuel dysfonctionnement ou à leur indisponibilité d'accès.

ISBN : 978-2-206-10152-1
Éditions Delagrave - 5, allée de la 2e DB - 75015 Paris
www.editions-delagrave.fr

SOMMAIRE

DÉCOUVERTE

ACTIVITÉ 1 L'algorithmique et la programmation 4
ACTIVITÉ 2 L'écriture d'un programme : les entrées et les sorties 6
PROJET 1 Programmer mon premier jeu 8
PROJET 2 Dessiner des figures géométriques 11
PROJET 3 Programmer le déplacement d'un robot 14
BILAN 17

NIVEAU 1

ACTIVITÉ 3 Les variables : affectation de valeurs 18
ACTIVITÉ 4 La structure « Si Alors » 20
PROJET 4 Jouer au jeu de Tic Tac Toe 22
PROJET 5 Coder un message en morse 25
PROJET 6 Guider un robot à distance 28
BILAN 31

NIVEAU 2

ACTIVITÉ 5 La structure « Répéter » 32
ACTIVITÉ 6 La structure « Si Alors Sinon » 34
PROJET 7 Jouer au jeu des allumettes (jeu de Nim) 36
PROJET 8 Décaler les lettres d'un message (chiffre de César) 39
PROJET 9 Faire suivre une ligne à un robot 42
BILAN 45

NIVEAU 3

ACTIVITÉ 7 La structure « Répéter jusqu'à » 46
ACTIVITÉ 8 La structure « Si Alors Sinon » imbriquée 48
PROJET 10 Sortir d'un labyrinthe 50
PROJET 11 Jouer avec une raquette et une balle (jeu de Pong) 53
PROJET 12 Faire surveiller un espace par un robot 56
BILAN 59

COMPLÉMENTS

Entraînement au brevet 60
Guide de programmation Scratch 62
Grille de suivi et d'évaluation 64

ACTIVITÉ 1 L'algorithmique et la programmation

Je découvre

PARTIE 1 Écrire un algorithme

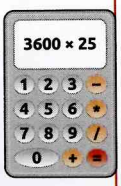
Une biscuiterie industrielle fabrique en grande quantité des cookies. Le lancement de la production se fait par tranche de 3 600 unités (3 600, 7 200, 10 800, 14 400, etc.). Les préparateurs disposent d'une fiche technique pour lancer les approvisionnements et la fabrication. Les cookies sont emballés par 8 dans un paquet.

Fiche technique N° C8 - Cookies

Ingrédients pour 8 cookies	Beurre : 25 g – Sucre : 25 g - 1/2 œuf - Levure : 3 g - Farine : 40 g - Chocolat : 30 g
 © H. de Oliveira/Expansion-Rea	**Étapes de fabrication** **1.** Faire ramollir le beurre. **2.** Ajouter le sucre, les œufs, la levure et mélanger. **3.** Répandre la farine tout en pétrissant afin que la pâte soit bien homogène. **4.** Incorporer les pépites de chocolat. **5.** Former sur une plaque de petites boules avec la pâte à cookies. **6.** Enfourner pendant 10 minutes.

1 **À partir de la fiche technique, repérer le nombre d'ingrédients utilisés et le nombre d'étapes de fabrication.**

.....



• INFORMATIONS •

À chaque nouvelle fabrication de cookies, le préparateur applique la même méthode pour calculer l'approvisionnement des ingrédients en grammes (en unités pour les œufs).

C8 - Cookies
Quantités à commander (en g)

Étape 1	Quantité de beurre = (Nombre de cookies * 25) / 8
Étape 2	Quantité de sucre = (Nombre de cookies * 25) / 8
	Nombre d'œufs = (Nombre de cookies * 0,5) / 8
	Quantité de levure = (Nombre de cookies * 3) / 8
Étape 3	Quantité de farine = (Nombre de cookies * 40) / 8
Étape 4	Quantité de chocolat = (Nombre de cookies * 30) / 8

2 **Calculer pour chaque ingrédient la quantité nécessaire pour fabriquer 3 600 cookies (450 paquets) puis compléter le tableau.**

Beurre	11 250 g	Sucre	.....	Œuf	.....	Levure	.....	Farine	18 000 g	Chocolat	13 500 g

3 **Justifier l'intérêt pour le préparateur d'avoir écrit la méthode de calcul (algorithme) sur un document.**

.....

4 **Déterminer à quoi servent les algorithmes d'un moteur de recherche ou d'un système de navigation (GPS).**

.....

.....

Je comprends

• Un **algorithme** est une suite d'**instructions** à appliquer dans un ordre logique pour résoudre un problème et obtenir rapidement un résultat. Il est écrit à la main ou à l'aide d'un logiciel dans un langage compréhensible par tous.

Je retiens

Un **algorithme** sert à préparer l'écriture d'un **programme** informatique.

PARTIE 2 Utiliser un algorithme de calcul dans un programme informatique

1 Lancer le logiciel Scratch et ouvrir le fichier **Cookies**. lienmini.fr/a152-cookies

2 Tester ce programme en cliquant sur le drapeau vert, puis sur les quantités à fabriquer : 3 600, 7 200, 10 800.

3 Cliquer sur le lutin 10800 et sur l'onglet **Scripts**.

• INFORMATIONS •

Quatre lutins sont présents dans ce programme. À chaque lutin est associé un **script** ou **séquence d'instructions**. Un lutin permet d'activer une séquence d'instructions.

Programme informatique (Scratch)	Langage de programmation graphique (par assemblage de blocs d'instruction)
	

4 Préciser l'avantage que procure l'utilisation de ce programme lors du lancement d'une nouvelle fabrication.

.....

.....

5 Repérer le nombre de blocs utilisés dans la séquence d'instructions (script) du lutin 10800.

.....

.....

6 Comparer la méthode de calcul du programme à celle de l'algorithme écrit à la main en page précédente.

.....

.....

Je comprends

• Un **programme** permet à son utilisateur, par le biais d'un écran informatique, de traiter très rapidement de nombreuses informations (textes, dessins, sons, etc.). Il est développé pour un domaine d'utilisation (le calcul, le dessin, le jeu, le guidage, la musique, etc.) à l'aide d'un logiciel de programmation.

Je retiens

Un programme est composé d'une ou plusieurs **séquences d'instructions (script)**. Il est écrit à l'aide d'un **langage de programmation**. On distingue :
– les langages de programmation graphique qui reposent sur l'assemblage de blocs ;
– les langages de programmation textuelle qui reposent sur l'écriture d'une suite de commandes spécifiques (code).

ACTIVITÉ 2 L'écriture d'un programme : les entrées et les sorties

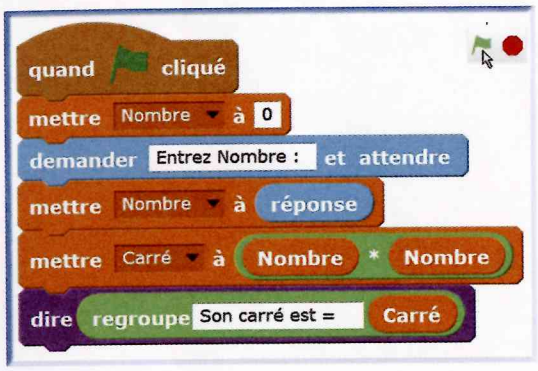
Je découvre

Le carré d'un nombre est la multiplication d'un nombre par lui-même. Le carré d'un nombre est toujours un nombre positif. Sa notation mathématique correspond à un nombre suivi de l'exposant « ^{2} ». Exemple : $10^2 = 10 \times 10 = 100$.

x^2

1 **Lancer l'application Scratch et ouvrir le fichier Carré.** lienmini.fr/a152-carre

2 **Cliquer sur le drapeau vert et tester ce programme en faisant varier la valeur du nombre à saisir.**



• INFORMATIONS •

Le programme contient une variable « Nombre ».
Cette variable stocke le nombre saisi.
En début de programme, une valeur de départ (ici 0) est donnée à la variable « Nombre » : on l'initialise.

3 **Identifier le nom du bloc d'instruction utilisé pour saisir la valeur d'entrée d'un nombre.**

.....

.....

4 **Préciser quel traitement est réalisé pour obtenir le carré du nombre saisi.**

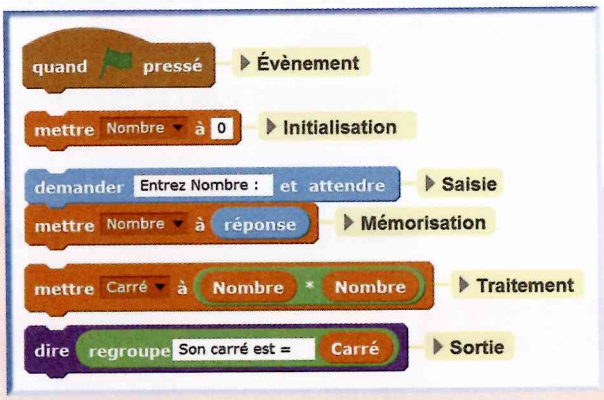
.....

5 **Indiquer les éléments affichés à la fin du programme et le nom du bloc d'instruction utilisé.**

.....

.....

.....



Je comprends

- Avant de commencer à écrire un programme, on analyse le problème et on écrit un algorithme.
- On réfléchit ensuite à la structure du programme. Généralement, elle suit l'ordre suivant :
  - déclaration des variables ;
  - initialisation des variables ;
  - saisie et mémorisation des **entrées** ;
  - **traitement** des données ;
  - **sortie** des résultats.

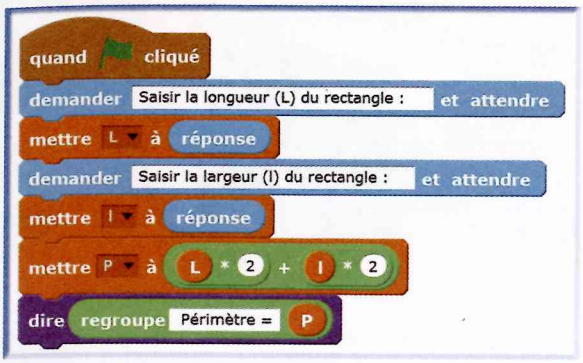
Je retiens

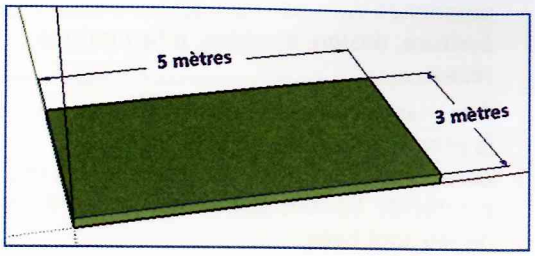
- Le bloc d'instruction **demander** permet de saisir une valeur (Entrée).
- Le bloc d'instruction **dire** affiche la valeur contenue dans une variable (Sortie).

J'applique

20'

APPLICATION 1 Le périmètre d'un rectangle





1. **À partir de l'écran affiché ci-dessus, identifier le nom du bloc d'instruction qui permet de saisir la longueur et la largeur d'un rectangle.** .....

2. **Calculer et noter la valeur obtenue en sortie si l'on saisit 5 et 3.** .....

3. **Lancer l'application Scratch et ouvrir le fichier Périmètre.** ✓lienmini.fr/a152-perimetre

4. **Cliquer sur le drapeau vert. Vérifier le résultat du programme en saisissant les valeurs 5 et 3 pour la longueur et la largeur du rectangle.**

30'

APPLICATION 2 Le temps de parcours d'un robot aspirateur

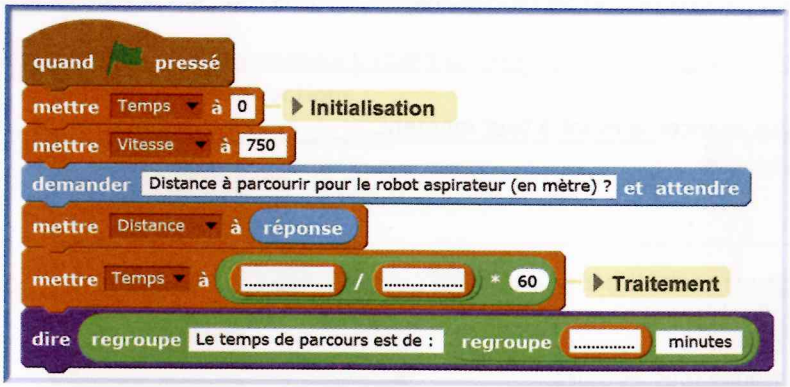
Un programme interne au robot aspirateur calcule son temps de parcours (en **minutes**). Sa vitesse moyenne est de 750 m/h. La formule de calcul employée est la suivante :
Temps = (Distance/Vitesse) × 60.

© S-E, Fotolia

1. **Calculer le temps passé par le robot aspirateur pour se déplacer sur une distance de 100 mètres.**
Temps = ...

2. **Préciser ce que demande le programme ci-dessous.**
.....

3. **Compléter ci-dessous le bloc d'instruction qui réalise le traitement des données et compléter le bloc d'instruction** dire **permettant d'afficher le résultat.**



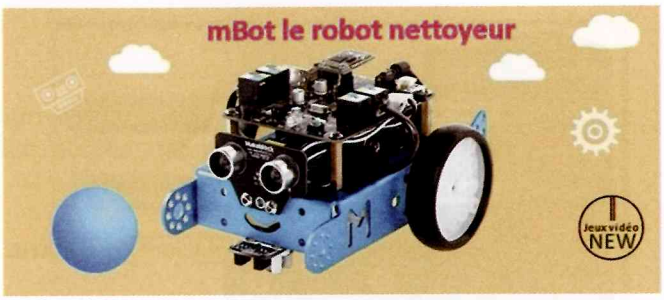
4. **Lancer l'application Scratch et ouvrir le fichier Aspirateur.** ✓lienmini.fr/a152-aspirateur

5. **Tester le programme en faisant varier les distances saisies (ex. : 50 m, 100 m, 225 m, etc.).**

PROJET 1 Programmer mon premier jeu

Un jeu vidéo est une œuvre dont la création implique la coopération de plusieurs personnes dans des domaines variés : **écriture**, **design**, **musique**, **informatique**, **réseaux**...

À la manière d'un studio de cinéma, la production démarre : le choix des personnages, des décors, des niveaux, etc. est décidé. Le fonctionnement et les règles du jeu sont fixés.



Au cours du développement d'un prototype du jeu, les programmeurs doivent pouvoir faire prendre aux personnages et aux objets créés des formes, des trajectoires différentes en fonction des **évènements** qui surviennent. Au final, une phase de tests et de mise au point des animations et du programme permet au jeu d'être distribué.

Nous allons réaliser un jeu autour d'un robot qui se déplace dans un circuit. Il faudra éliminer des balles pour libérer le circuit.

- **Comment programmer un jeu ?**

Étape 1 Analyser le principe de création d'un jeu vidéo

15'

1 **Lancer le logiciel Scratch et ouvrir le fichier Évènement 1.** lienmini.fr/a152-evenement1

2 **Décrire l'objet (lutin) qui a été créé dans ce programme.**

.....

3 **Cliquer sur le lutin « Robot mBot » et sur l'onglet Scripts, puis sur le drapeau vert et utiliser la touche « flèche vers le haut » du clavier.**

4 **Noter dans le tableau ci-dessous les actions qui ont été déclenchées par chaque évènement.**

Type d'évènement	Évènement et action associée
quand pressé	Quand le joueur clique sur le drapeau vert : → .....
↑	Quand le joueur appuie sur la touche « flèche vers le haut » : → .....

5 **Préciser s'il est possible de faire avancer le robot à tout moment.**

.....

• INFORMATIONS •

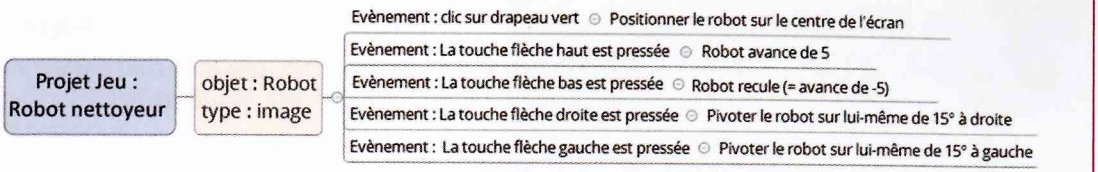
Le déroulement du programme est contrôlé par un ou plusieurs **évènements**. Un même objet (lutin) peut être associé à plusieurs évènements.
La rubrique Évènements regroupe tous les blocs d'instruction qui permettent de les programmer.
On distingue différents types d'évènements programmables : **appui sur une touche, clic sur une icône ou un objet, envoi ou réception d'un message**.

Étape 2 Modifier, compléter, écrire un algorithme

20'

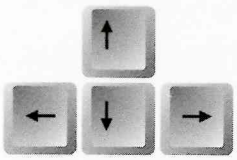
• INFORMATIONS •

Le jeu consiste à piloter un robot et à nettoyer le circuit sur lequel trainent des balles.
La **carte mentale** suivante détaille les différents évènements et actions associés à l'objet « Robot ».



En vous aidant des informations disponibles ci-dessus :

1 **Noter la solution retenue pour piloter le robot dans toutes les directions.**



.....

.....

2 **Compléter les algorithmes associés aux évènements 2, 3, 4 et noter sur la carte mentale leur numéro respectif (1,2,3,4).**

Évènement n° 1	Algorithme 1
Appui sur la touche ↑	Avancer sur une distance de 5

Évènement n° 2	Algorithme 2
Appui sur la touche ↓	.....

Évènement n° 3	Algorithme 3
Appui sur la touche ←	.....

Évènement n° 4	Algorithme 4
Appui sur la touche →	.....

Étape 3 Écrire un programme

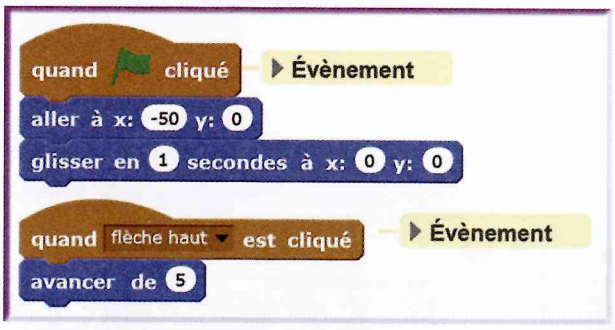
20'

Phase 1 : Piloter le robot

1 **Lancer le logiciel Scratch et ouvrir le fichier Évènement 1.** ✔ lienmini.fr/a152-evenement1

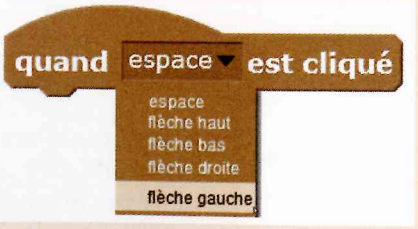
2 **Cliquer sur le lutin « Robot mBot » et sur l'onglet Scripts.**

3 **Compléter le programme (créer 3 scripts) pour que le joueur puisse déplacer le robot dans les quatre directions.**



Aide

Programmation par blocs



4 **Tester le bon fonctionnement du programme.**

20'

Étape 4 Mettre au point et exécuter un programme

Phase 2 : Nettoyer le circuit

1. Lancer le logiciel **Scratch** et ouvrir le fichier Évènement 2.
   lienmini.fr/a152-evenement2

2. Désigner les objets (lutin) qui ont été créés dans ce programme.

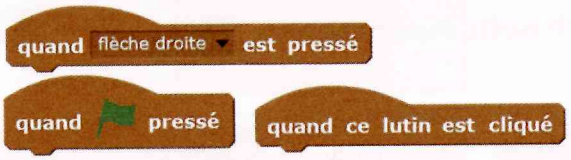
   .....

   .....

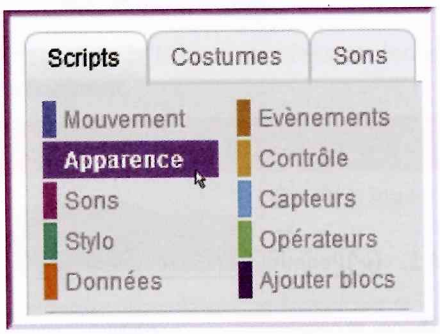
• INFORMATIONS •

Une balle est sur le circuit. Le joueur doit nettoyer la piste et faire disparaître la balle.

3. Entourer le type d'évènement à sélectionner parmi la liste suivante pour faire disparaître la balle et décrire l'action associée à l'évènement.

Type d'évènement	Évènement et action associée
	Quand le joueur clique sur le lutin « Balle Jaune » : → ....

4. Entourer la commande de la rubrique Apparence qui permet de faire disparaître la balle lorsque l'on clique dessus.

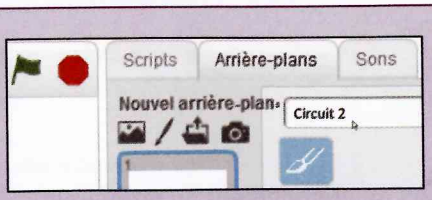


5. Modifier et tester le programme (script) du lutin « **Balle Jaune** » pour que le ballon disparaisse lorsque l'on clique dessus.

6. Tester et corriger votre programme.

Aller plus loin

Améliorer le premier jeu pour qu'il y ait plus de balles et plusieurs circuits à nettoyer avec le robot.



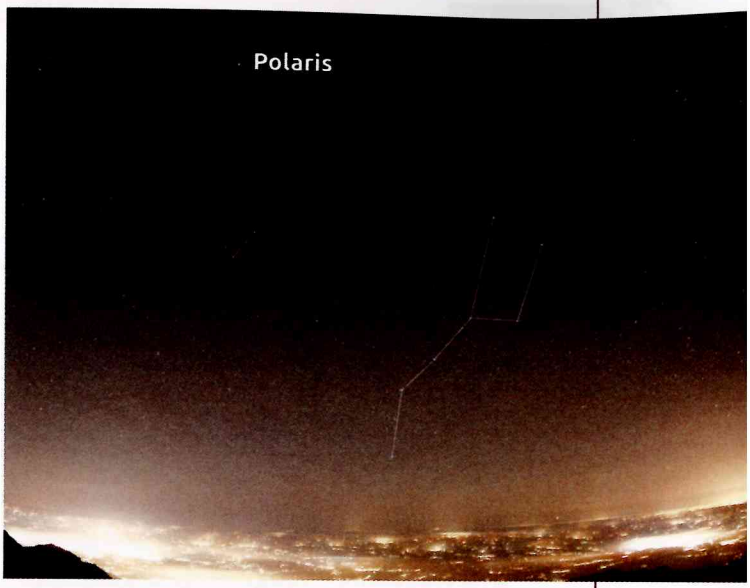
PROJET 2 Dessiner des figures géométriques

De nombreuses **constellations** sont identifiables dans le ciel. Une constellation est la représentation d'un groupe d'étoiles voisines présentant une figure particulière. Parmi les plus connues, on peut citer la **Grande Ourse** et la **Petite Ourse**, identifiables par leur forme de casserole.

La Grande Ourse est la troisième constellation du ciel par son étendue. Elle contient la « **grande casserole** ».

La Petite Ourse est une constellation assez petite et peu lumineuse. Elle contient la « **petite casserole** ». Elle permet de situer le **pôle nord** céleste et l'**étoile polaire** appelée Polaris (tout en haut du manche).

- **Comment dessiner sur un écran les constellations de la Grande Ourse et de la Petite Ourse ?**



© C. Lehenaff/Photononstop

Étape 1 Analyser l'organisation d'une constellation

20'

1 **Indiquer à quelle forme s'apparentent les constellations de la Grande Ourse et de la Petite Ourse.**

.....

.....

• INFORMATIONS •

Sur le schéma ci-dessous, les étoiles sont indiquées par un point nommé par une lettre (A, B, C, etc.). Les longueurs séparant chaque étoile sont indiquées en pas. Ainsi pour la constellation de la Grande Ourse, 60 pas séparent l'étoile A de l'étoile B.

2 **Déterminer le nombre d'étoiles qui composent la Grande Ourse et la Petite Ourse.**

.....

3 **Rechercher à quelle étoile connue correspond la lettre H.**

.....

.....

4 **Déterminer le nombre de segments à tracer pour la Grande Ourse et la Petite Ourse.**

.....

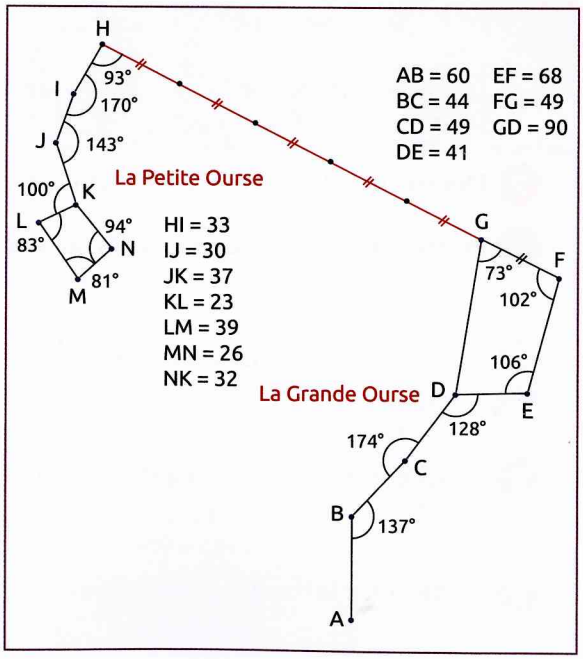
.....

5 **Préciser en nombre de pas, la distance qui sépare les étoiles G et H.**

.....

6 **Noter la valeur de l'angle formé par les trois étoiles I, J, K.**

.....



Étape 2 Modifier, compléter, écrire un algorithme

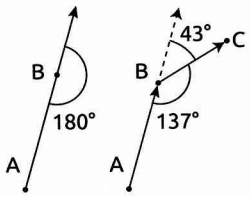
20'

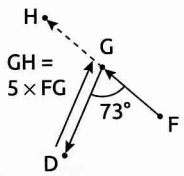
1 **Compléter l'algorithme de la situation 1 dans le tableau afin d'afficher les étoiles A, B, C et leurs segments.**

2 **Compléter l'algorithme de la situation 2 dans le tableau afin d'afficher les étoiles F, G, H, leurs segments et celui de GD.**

INFORMATIONS *(Situation 1)*

Lorsque le lutin va de A vers B, il doit tourner à droite vers C. L'angle indiqué au niveau de B est de 137° donc le lutin va tourner à droite de **180° – 137° = 43°**. En effet, si le lutin devait aller tout droit, l'angle au niveau de B serait de 180°.

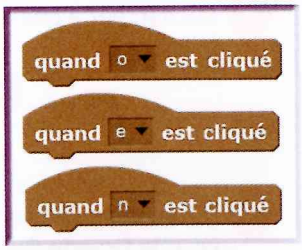
Situation 1 Dessiner les étoiles A, B, C et leurs segments	Algorithme 1
Aller de A vers B Tourner vers C Aller à C 	Dessiner Étoile A Avancer de 60 pas Dessiner Étoile B Tourner à droite de ..... Avancer de ..... pas Dessiner Étoile .....

Situation 2 Dessiner les étoiles F, G, H et leurs segments	Algorithme 2
Aller de F vers G Tourner vers D Aller à D Reculer pour revenir à G Se replacer en direction de H Aller à H 	Dessiner Étoile F Avancer de ..... pas ..... Tourner à ..... de 107° Avancer de ..... pas ..... Tourner à ..... de 107° ..... Dessiner Étoile H

Étape 3 Écrire un programme

40'

Phase 1 : Dessiner la Grande Ourse



1 **Lancer le logiciel de programmation Scratch.**
Ouvrir le fichier Grande Ourse. lienmini.fr/a152-grande-ourse

2 **Cliquer sur le lutin « Étoile » et sur l'onglet Scripts.**

3 **Préciser le nombre d'évènements qui sont programmés.**

.....

4 **Appuyer sur la touche « o » (comme ourse) et préciser ce qu'il se passe.**

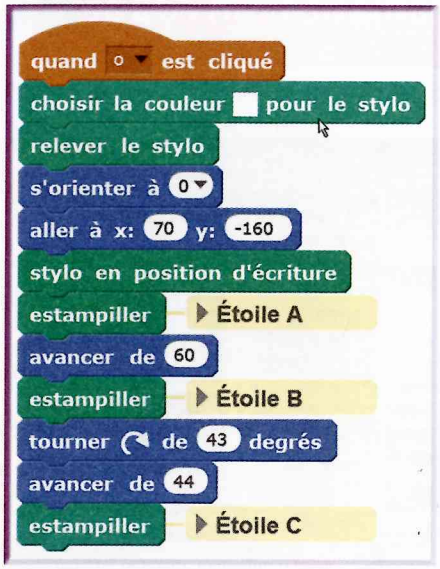
.....

5 **Appuyer sur la touche « e » (comme étoile) plusieurs fois et préciser ce qu'il se passe.**

.....

6 **Appuyer sur la touche « n » (comme noir) et préciser ce qu'il se passe.**

.....



• INFORMATIONS •

Pour afficher une constellation avec ses différentes étoiles, il faut dessiner (**estampiller**) l'étoile, tourner d'un angle, avancer ou reculer.

Aide

Programmation par blocs

`choisir la couleur pour le stylo` permet de choisir la couleur du segment.

`aller à x:0 Y:0` permet de déplacer le lutin vers un point précis sur l'écran.

`stylo en position d'écriture` permet de laisser une trace.

`avancer de` permet de tracer le segment de la taille désirée.

`estampiller` permet de dessiner une image du lutin à l'écran.

7 **À partir de la rubrique Aide, compléter le script évènementiel « o », afin de construire le reste de la constellation. La séquence d'instructions devrait s'arrêter par le tracé du segment allant de l'étoile G vers l'étoile D.**

40'

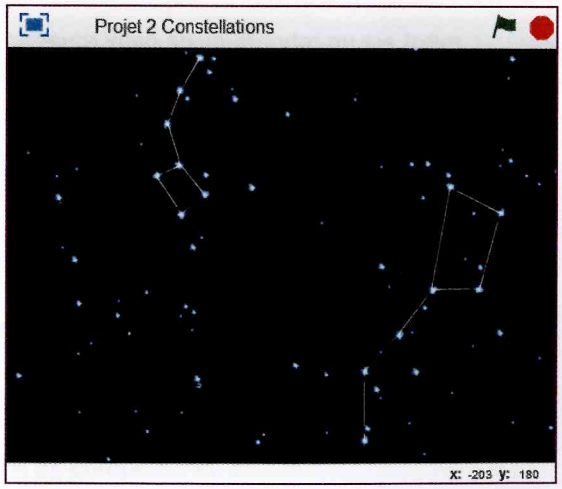
Étape 4 Mettre au point et exécuter un programme

Phase 2 : Dessiner la Petite Ourse

1 **Lancer le logiciel de programmation Scratch.**
Ouvrir le fichier Petite Ourse. lienmini.fr/a152-petite-ourse

• INFORMATIONS •

Afin de trouver le début de la Petite Ourse, il va falloir se rendre à l'étoile H. Il n'y a pas de segment à tracer entre les étoiles G et H, donc il faut penser à utiliser successivement le bloc d'instruction `relever le stylo` puis de nouveau le bloc d'instruction `stylo en position d'écriture` une fois arrivé à l'étoile H.



2 **Programmer dans le script « o » le passage de la fin de la construction de la Grande Ourse (tracé du segment [GD]) jusqu'au début de la construction de la Petite Ourse (dessin de l'étoile H).**

3 **Compléter le script « o », afin de construire le reste de la constellation de la Petite Ourse.**

4 **Tester le programme et noter le ou les problèmes rencontrés.**

• INFORMATIONS •

Il est possible de trouver les coordonnées d'un point en positionnant la souris sur l'écran du programme (voir exemple ci-dessus). L'étoile polaire se trouve au point (x : – 75 ; y : 172).

Aller plus loin

L'étoile polaire appelée Polaris se trouve au bout de la « petite casserole ». Comment faire scintiller plus fort l'étoile polaire ?

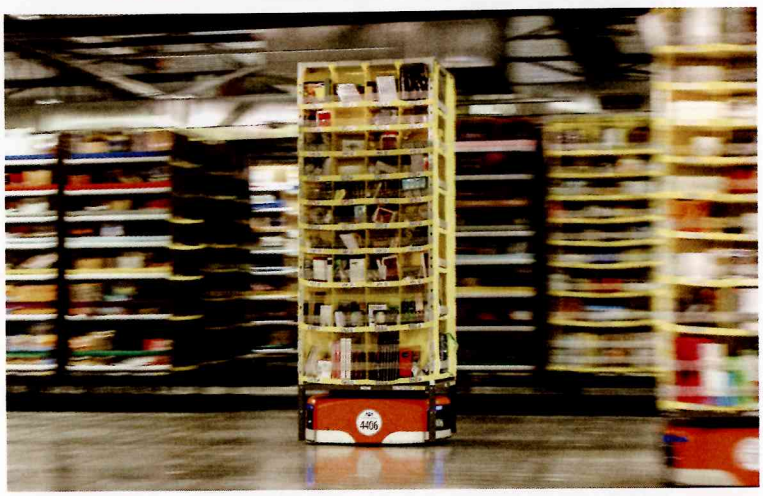
PROJET 3 Programmer le déplacement d'un robot

Il existe des robots qui se déplacent de manière autonome pour transporter des produits d'un espace de stockage à un autre.

Les robots Kiva par exemple, déplacent en permanence les étagères d'un entrepôt où sont stockés les produits à livrer.

Les préparateurs de commandes ne perdent plus de temps à déambuler dans les allées puisque les étagères comme les produits viennent à eux automatiquement.

- **Comment programmer le déplacement d'un robot ?**



© Bloomberg/Getty Images

Étape 1 Analyser le déplacement du robot mBot

30'

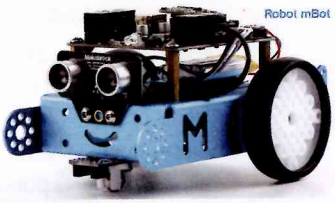
1 **Donner un exemple de situation où l'on programme le déplacement d'un robot et l'avantage qu'en tirent les êtres humains.**

.....

.....

• INFORMATIONS •

mBot est un robot muni de deux roues activées par deux moteurs. Il est programmable à l'aide du logiciel **mBlock**.



2 **Lancer le logiciel mBlock. Ouvrir le fichier Déplacement 1.**
✓ lienmini.fr/a152-deplacement1

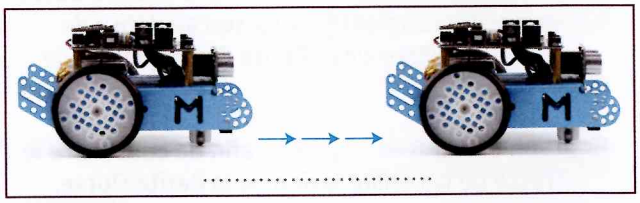
3 **Allumer le robot et implanter le programme (voir document ressource).** ✓ lienmini.fr/a152-robot-implanter

4 **Observer le comportement du robot. Mesurer et noter la distance de déplacement et sa durée.**

.....

.....

.....



.....

5 **Lancer le logiciel de programmation mBlock. Ouvrir le fichier Déplacement 2.** ✓ lienmini.fr/a152-deplacement2

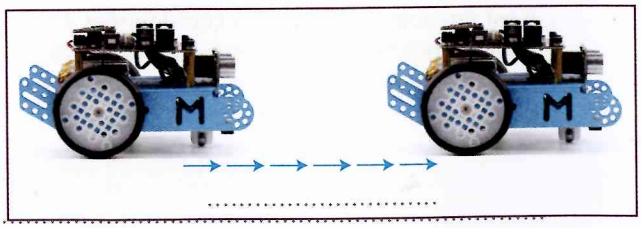
6 **Allumer le robot et implanter le programme (voir document ressource).** ✓ lienmini.fr/a152-robot-implanter

7 **Observer le comportement du robot. Mesurer et noter la distance de déplacement et sa durée.**

.....

.....

.....



.....

Étape 2 Modifier, compléter, écrire un algorithme

20'

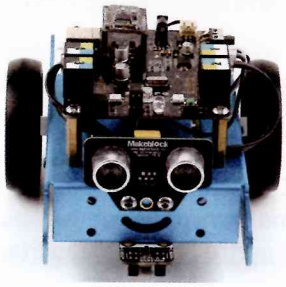
1 **Reporter dans le tableau qui suit les distances que vous avez obtenues dans la 1re étape lors des deux déplacements.**

Distance D parcourue (cm)	.....	.....
Temps T (seconde)	1	2
Vitesse V $= \left(\frac{D}{T}\right)$	.....	.....

2 **Calculer dans les deux cas la vitesse V de déplacement. Noter vos résultats dans le tableau.**

• INFORMATIONS •

Le déplacement du robot mBot se fait grâce à deux moteurs (M1 et M2) qu'il faut activer.
Dans les situations ci-dessous, le robot avance par défaut à la vitesse 100 (vitesse par défaut donnée par le constructeur du robot) pendant une durée fixée en seconde. Pour arrêter le robot, on met sa vitesse à 0.



3 **Préciser le paramètre à modifier si on veut régler la distance de parcours du robot.**

.....

4 **Compléter l'algorithme de la situation 2 et celui de la situation 3.**

Algorithmes

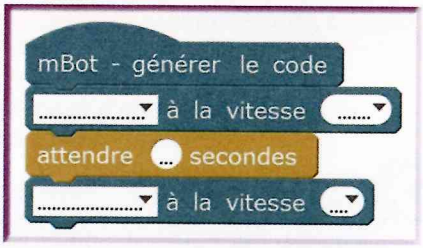
Situation 1 Le robot avance environ de 10 cm	Situation 2 Le robot avance environ de 20 cm	Situation 3 Le robot avance environ de 50 cm
Avancer tout droit à la vitesse 100	.....	.....
Pendant 1 seconde	.....	.....
Arrêter les moteurs	.....	.....

Étape 3 Écrire un programme

30'

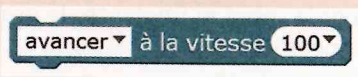
Phase 1 : Programmer le déplacement du robot

1 **En vous aidant de l'algorithme de la situation 3 (Étape 2), compléter ci-dessous le programme pour que le robot avance de 50 cm.**



Aide

Programmation par blocs



Ce bloc d'instruction permet de faire avancer le robot.

Ce bloc d'instruction permet d'arrêter les moteurs.

2 **Lancer le logiciel mBlock. Ouvrir le fichier Déplacement 3.** ✓ lienmini.fr/a152-deplacement3

3 **Sélectionner le lutin « Robot mBot » et cliquer sur l'onglet Scripts.**

4 À l'aide des blocs des rubriques Contrôle et Pilotage, compléter le programme pour que le robot avance de 50 cm.

5 Allumer le robot et implanter le programme (voir document ressource). lienmini.fr/a152-robot-implanter

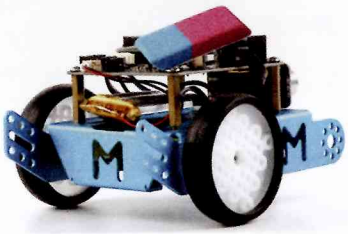
6 Tester votre programme.

Étape 4 Mettre au point et exécuter un programme

30'

Phase 2 : Programmer le retour du robot

Le robot emmène un petit objet (une gomme par exemple) sur une distance de 1 mètre, attend 3 secondes et repart en reculant pour retourner à son point de départ.



1 Calculer le temps qu'il est nécessaire de faire fonctionner les moteurs pour que le robot puisse se déplacer sur une distance de 1 mètre (voir tableau Étape 2).

.....

.....

2 Lancer le logiciel mBlock. Ouvrir le fichier Déplacement 4. lienmini.fr/a152-deplacement4

3 Sélectionner le lutin « Robot mBot » et cliquer sur l'onglet Scripts.

4 À l'aide des blocs des rubriques Contrôle et Pilotage, compléter le programme qui permet de gérer cette nouvelle situation.

.....

.....

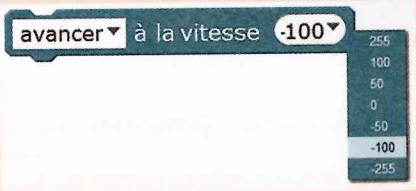
.....

.....

.....

Aide

Programmation par blocs



Ce bloc d'instruction permet de régler la vitesse d'avance ou de recul du robot.

Ce bloc d'instruction permet d'arrêter tous les programmes.

5 Allumer le robot et implanter le programme (voir document ressource).
lienmini.fr/a152-robot-implanter

6 Tester votre programme et observer le comportement du robot. Que constatez-vous ?

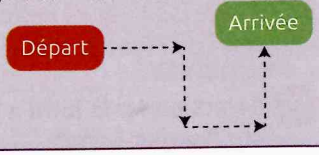
.....

.....

.....

Aller plus loin

Rechercher l'algorithme et le programme associés au trajet ci-dessous. Chaque segment a une longueur de 20 cm.



• BILAN •

L'essentiel

« Algorithme »

→ Un **algorithme** est une **suite d'instructions** qui permet de résoudre un problème et d'obtenir rapidement un résultat.
Il est écrit à la main ou à l'aide d'un logiciel dans un langage compréhensible par tous. Il sert à préparer l'écriture d'un programme informatique.

« Programme »

→ Un **programme** permet à son utilisateur de traiter de nombreuses informations (textes, dessins, sons, etc.).
Il est développé pour un domaine d'utilisation (calcul, dessin, jeu, guidage, musique, etc.).
Il est composé d'une ou plusieurs **séquences d'instructions**.

→ Un programme est écrit à l'aide d'un **langage de programmation**. On distingue :
– les langages de programmation graphique qui reposent sur l'**assemblage de blocs** ;
– les langages de programmation textuelle qui reposent sur l'**écriture de commandes spécifiques** (code).

→ L'écriture d'un programme suit généralement l'ordre suivant : la déclaration et l'initialisation des **variables**, la saisie et la mémorisation des entrées, le traitement des données, la sortie des résultats.

« Évènement »

→ Un **évènement** permet de déclencher une séquence d'instructions (script). On distingue différents types d'évènements : un clic sur une icône ou un objet, l'appui sur une touche, l'envoi ou la réception d'un message...

Aide mémoire

« Algorithme »

Calculer le périmètre d'un cercle

Quand le drapeau vert est activé

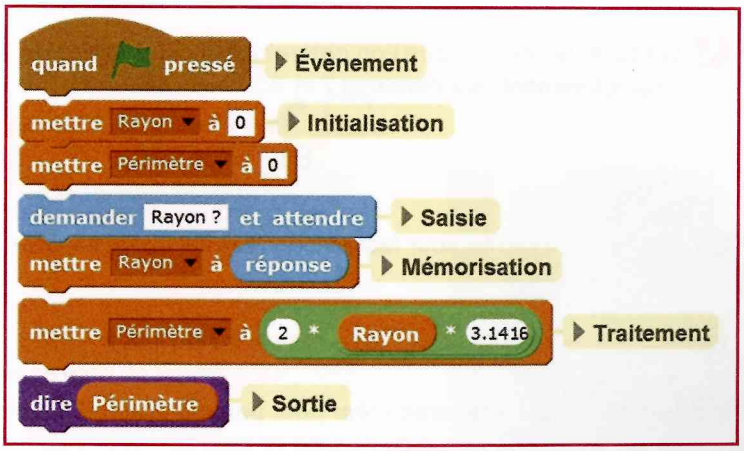
Rayon = 0 Périmètre = 0

Saisir et mémoriser le rayon du cercle

Périmètre = 2 * Rayon * 3.1416 (Pi)

Afficher le périmètre du cercle

« Programme »



QCM Entourer la bonne réponse.

1. Un algorithme permet :
a. d'activer des évènements.
b. de résoudre un problème et d'obtenir un résultat.
c. d'assembler des blocs d'instructions.

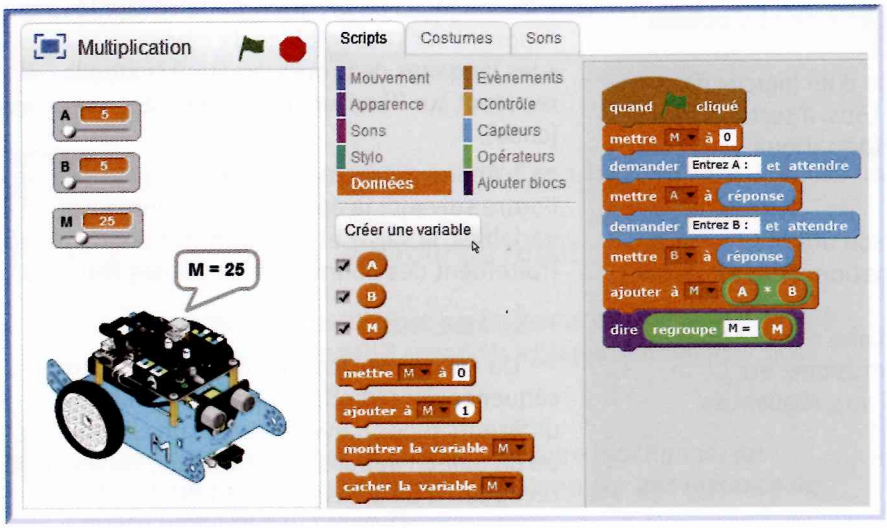
2. Un programme est composé :
a. d'une ou plusieurs séquences d'instructions.
b. de plusieurs langages de programmation.
c. de différents capteurs.

3. L'évènement du programme ci-dessus correspond :
a. à la réception d'un message.
b. à l'appui sur la touche « c ».
c. au clic sur le drapeau vert.

4. La séquence d'instructions du programme ci-dessus permet :
a. de dessiner un cercle.
b. de calculer le périmètre d'un cercle.
c. d'évaluer le volume d'une sphère.

ACTIVITÉ 3 Les variables : affectation de valeurs

Je découvre



1 À partir de l'écran affiché ci-dessus, indiquer les valeurs affectées aux variables A et B.

.....

2 Préciser le contenu de la variable M et sa valeur dans le programme.

.....

3 Entourer le résultat que l'on obtient lorsque les variables A et B sont égales respectivement aux nombres 7 et 8.

81 49 56

4 Expliquer la raison pour laquelle on met à « 0 » la variable M en début de programme.

.....

.....

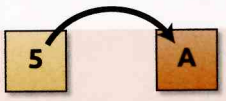
5 Préciser ce qu'il faudrait créer pour pouvoir multiplier trois nombres. Donner le contenu de la formule de calcul de M dans ce cas.

.....

.....

Je comprends

- Une **variable** est désignée par son nom (A, B, etc.). Elle sert à stocker une valeur (un nombre, un texte) ou une liste d'éléments (nombres, textes). Les variables sont définies avant l'écriture du programme.
- L'**affectation** consiste à mémoriser une valeur ou un résultat dans une variable. On dit par exemple « affecter la valeur 5 à la variable A ».



Affectation du nombre 5 dans la variable A

Je retiens

- Le bloc d'instruction `Créer une variable` permet de créer une variable de type nombre ou texte.
- Le bloc d'instruction `mettre A à 0` permet d'affecter à une variable une valeur ou le résultat d'un traitement.
- Le bloc d'instruction `ajouter à A 1` permet d'ajouter à une variable une valeur.

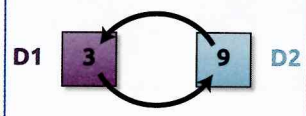
J'applique

20'

APPLICATION 1 Comment échanger la valeur de deux variables ?

La valeur 3 est affectée à la variable D1 et la valeur 9 à la variable D2. On souhaite échanger ces deux valeurs à l'aide d'une troisième variable appelée « TEMP ».

1 **Noter dans le tableau suivant la valeur des variables TEMP, D1 et D2 après l'exécution des trois blocs d'instruction.**



Bloc d'instruction		TEMP	D1	D2
1 →	mettre TEMP à D1	.....	3	9
2 →	mettre D1 à D2	3	.....	9
3 →	mettre D2 à TEMP	3	9	.....

2 **Ouvrir le fichier Échange.** lienmini.fr/a152-echange

3 **Tester le programme et expliquer son rôle.**

.....

.....

4 **Expliquer le rôle de la variable « TEMP » dans ce programme.**

.....

.....

30'

APPLICATION 2 Comment calculer le total des entrées ?

Le prix d'entrée journalier d'un parc de loisirs est 30 euros pour un adulte et 20 euros pour un enfant (– 12 ans). Le caissier saisit le nombre d'entrées adultes et enfants. Le système informatique affiche la somme à payer et imprime un ticket.

© Yauhen 0409, Fotolia

1 **Rechercher la formule de calcul qui permet d'obtenir le coût total pour une famille qui souhaite passer une journée dans ce parc de loisirs.**

Coût total = ...

• INFORMATIONS •

Cinq variables ont été créées pour automatiser ce calcul dans un programme : Coût_total ; Nombre_Adultes ; Nombre_Enfants ; Prix_Adulte ; Prix_Enfant.

2 **Fixer la valeur à laquelle il faut initialiser les variables « Prix adulte » et « Prix enfant ».**

.....

3 **Déterminer les deux variables qui doivent faire l'objet d'une entrée dans le programme.**

.....

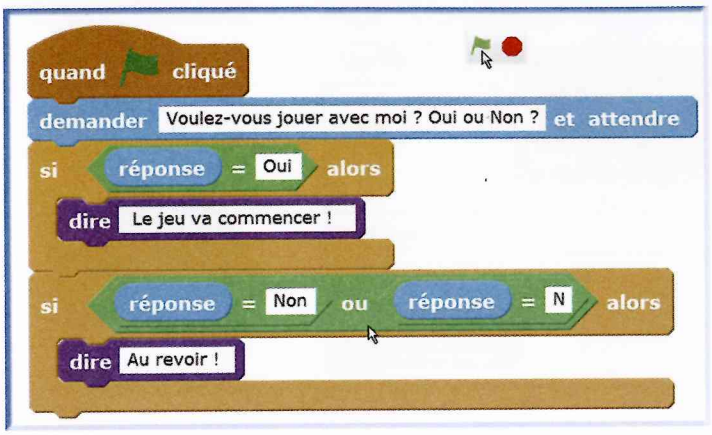
4 **Lancer l'application Scratch et ouvrir le fichier Parc.** lienmini.fr/a152-parc

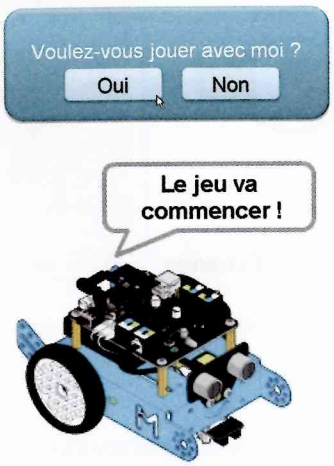
5 **Compléter et tester ce programme. Utiliser les blocs d'instruction** Demander **pour les entrées et** Dire **pour les sorties.**

ACTIVITÉ 4 La structure « Si Alors »

Je découvre

Les logiciels demandent régulièrement à leurs utilisateurs une confirmation : pour continuer à jouer, pour sauvegarder un document, pour se connecter... Une question est alors posée à laquelle il faut répondre en cliquant sur un bouton : « Oui » ou « Non ».





1 **Préciser ce qu'affiche le programme si l'utilisateur saisit le mot « Oui ».**

.....

2 **Indiquer ce qu'affiche le programme si l'utilisateur saisit le mot « Non ».**

.....

3 **Rechercher dans le programme s'il existe un autre moyen de quitter le jeu.**

.....

4 **Détailler le fonctionnement des blocs d'instruction conditionnelle « si Réponse = Non ou Réponse = N alors ».**

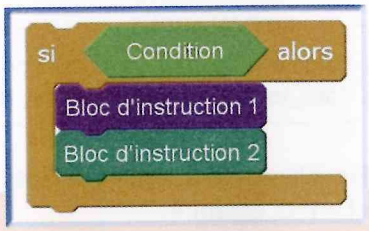
.....

.....

.....

5 **Rechercher ce que va afficher le programme lorsque l'utilisateur saisit la lettre « O ».**

.....



Je comprends

- La structure alternative « **Si Alors** » permet de poser une **condition**. La condition comprend un **opérateur de comparaison** (<, =, >) ou un **opérateur logique** (Et, Ou, Non).
- Par exemple, A < B (A est strictement inférieur à B) est une condition. Si cette condition est vérifiée (Vrai), alors un ou plusieurs blocs d'instruction seront exécutés.

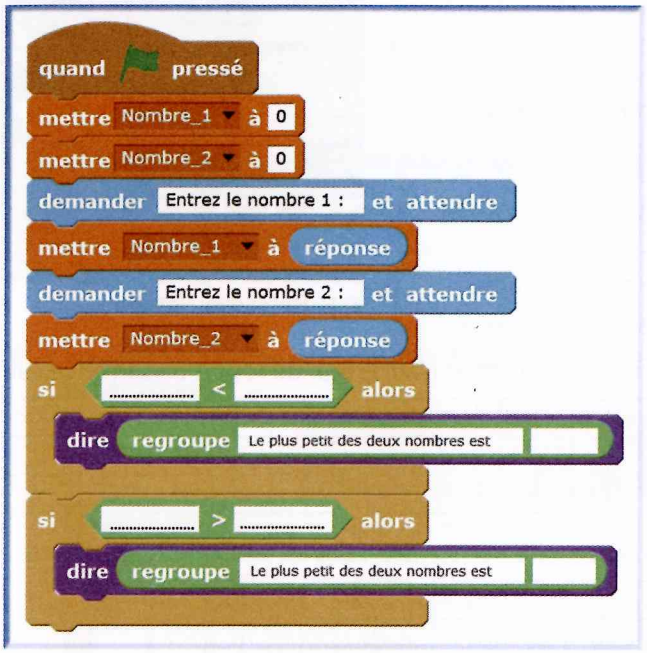
Je retiens

- Le bloc d'instruction **Si (condition) Alors** permet de tester une condition.
- Les blocs d'instruction conditionnelle < = > permettent de comparer deux valeurs.
- Les blocs d'instruction conditionnelle permettent de faire des opérations logiques : et ou non

J'applique

APPLICATION 1 Afficher le plus petit de deux nombres

20'



1. Lancer le logiciel Scratch et ouvrir le fichier **Nombres.** lienmini.fr/a152-nombres

2. Compléter le programme pour qu'il affiche le plus petit de deux nombres quel que soit leur ordre de saisie.

3. Tester plusieurs fois le programme en saisissant deux nombres dans un ordre différent. Noter ce que vous constatez.

.....

.....

.....

4. Préciser ce qui se passera si les deux nombres saisis sont égaux.

.....

.....

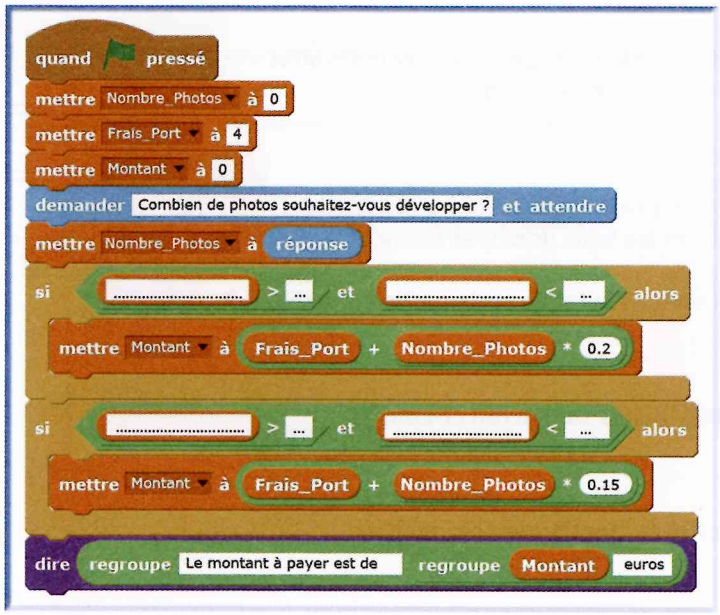
APPLICATION 2 Calculer le prix d'un tirage photos

30'

Une boutique en ligne propose des tarifs dégressifs suivant le nombre de tirages photos commandés.

Format	Entre 1 et 49	Entre 50 et 100
11 x 15 cm	0,20 € par tirage + 4 € de frais de port	0,15 € par tirage + 4 € de frais de port

© D. Tabler, Fotolia



1. Lancer le logiciel Scratch. Ouvrir le fichier **Tirages.** lienmini.fr/a152-tirages

2. Repérer et noter le nom des trois variables utilisées dans ce programme.

.....

.....

.....

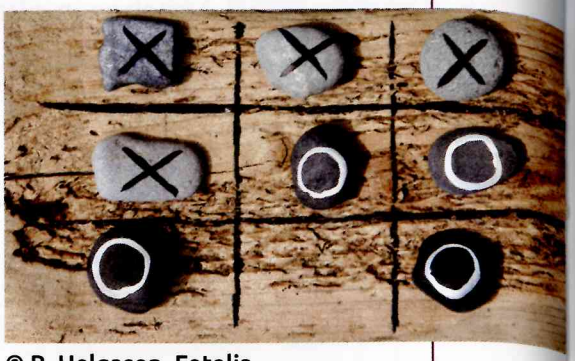
3. Compléter et tester le programme pour qu'il calcule le montant du tirage en fonction du nombre de photos à développer et des frais de port.

4. Tester le programme pour 100 et 101 tirages. Que constatez-vous ?

.....

.....

PROJET 4 Jouer au jeu de Tic Tac Toe



© B. Helgason, Fotolia

Le **Tic Tac Toe** est un jeu de réflexion qui se joue à deux sur une grille de 9 cases (3 lignes et 3 colonnes).

Chaque joueur doit successivement remplir une case avec le symbole qui lui est attribué : une **croix** ou un **rond**. Le gagnant est celui qui aligne en premier trois symboles identiques, horizontalement, verticalement ou en diagonale.

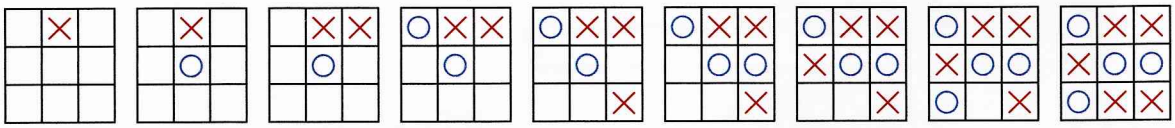
Si aucun joueur n'y parvient il y a **match nul**. À chaque nouvelle partie le premier joueur change.

La stratégie pour ne pas perdre au Tic Tac Toe est la suivante.

Soit × (croix) le joueur 1 et ○ (rond) son adversaire, le joueur 2.

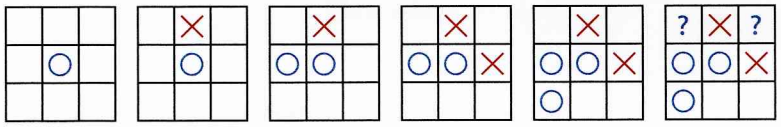
Principe 1 : si × ne joue pas au centre, il est sûr de ne pas gagner (sauf erreur de la part de son adversaire).

Ex.



Principe 2 : Si ○ commence au centre et si × joue ensuite horizontalement ou verticalement, il perd.

Ex.



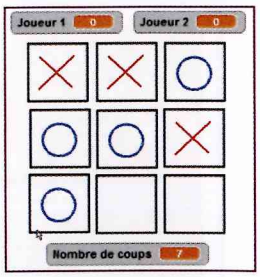
Principe 3 : si × joue dans la diagonale, ○ doit jouer dans la même diagonale.

Ce jeu est présent aujourd'hui sur ordinateur, tablette ou téléphone portable. Sur ces différents supports, on peut jouer seul contre la machine ou à deux joueurs.

- **Comment programmer le jeu de Tic Tac Toe ?**

Étape 1 Analyser le fonctionnement et les principes d'un jeu

20'



1. **À partir de l'exemple ci-dessus qui illustre le principe 1, donner le résultat final de la partie et le nombre de coups maximum jouable.**

.....

2. **Prendre une feuille de papier, tracer un quadrillage. Mener une partie de Tic Tac Toe avec votre voisin en utilisant les trois principes énoncés.**

3. **À partir de l'écran ci-contre, déterminer le résultat de cette partie en 7 coups.**

.....

4. **Repérer les différents objets qui sont affichés sur l'écran ci-contre.**

.....

5. **À l'aide de ces différents objets, décrire le fonctionnement du jeu.**

.....

.....

.....

6. **Préciser les trois états que peut prendre une case.**

.....

Étape 2 Modifier, compléter, écrire un algorithme

20'

• INFORMATIONS •

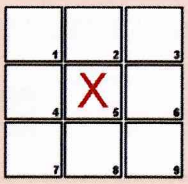
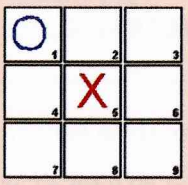
En début de partie, un tirage aléatoire entre deux chiffres (1 ou 2) permet de désigner le joueur qui commence. Au **joueur 1** est attribué la **croix**, au **joueur 2** le **rond**. Lorsqu'un des deux joueurs doit jouer, il clique sur une des cases. Si c'est son tour (Tirage 1 ou 2) et que la case est vide, il affiche une croix (joueur 1) ou un rond (joueur 2).

1 Préciser les deux conditions pour lesquelles un joueur peut afficher dans une case le symbole qui lui est attribué.

.....

.....

2 En vous aidant des informations disponibles ci-dessus, compléter l'algorithme suivant pour la situation 2.

	Situation 1 - Premier coup (Tirage aléatoire = 1)		Situation 2 - Second coup (Tirage = 2)
Évènement Le premier joueur clique sur la case 5. 	Si Tirage = 1 et Case 5 = 0 (Vide) Alors Afficher une croix Si Tirage = 2 et Case 5 = 0 (Vide) Alors Afficher un rond	**Évènement** Le second joueur clique sur la case 1. 	Si Tirage = 1 et Case 1 = 0 (Vide) Alors .... Si Tirage = 2 et Case 1 = 0 (Vide) Alors .....

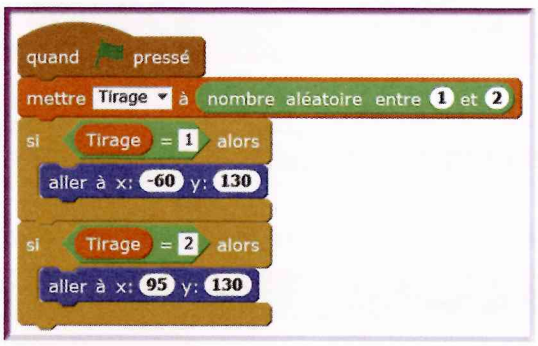
Étape 3 Écrire un programme

60'

Phase 1 : Afficher successivement les symboles « Croix » et « Rond »

1 Lancer le logiciel Scratch et ouvrir le fichier Cases. lienmini.fr/a152-cases

2 Cliquer sur le lutin « À vous » et sur l'onglet Scripts. Expliquer le fonctionnement du premier script.



Aide

Programmation par blocs

nombre aléatoire entre 1 et 10

Ce bloc d'instruction tire un nombre entier au hasard compris entre les deux nombres indiqués.

.....

.....

.....

3 Expliquer le fonctionnement des deux structures alternatives Si alors **de ce premier script.**

Si la variable Tirage = 1 Alors le lutin « À vous » .....

Si la variable Tirage = 2 .....

• INFORMATIONS •

Durant une partie, il est possible de jouer jusqu'à 9 coups : il y a 9 cases à programmer. En début de partie, chaque case est vide.

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

4 **Cliquer sur le lutin « Case 1 » et sur l'onglet Costumes. Noter le nom des trois aspects (costumes) que peut prendre une case dans le jeu.**

→ ..... → ..... → .....

5 **Cliquer sur le lutin « Case 1 » et sur l'onglet Scripts. Préciser l'action qui suit l'évènement dans ces deux scripts.**

Quand l'utilisateur du programme clique sur le drapeau vert :

.....

.....

.....

Quand le script « Case 1 » reçoit le message « Fin de Partie » :

.....

.....

• INFORMATIONS •

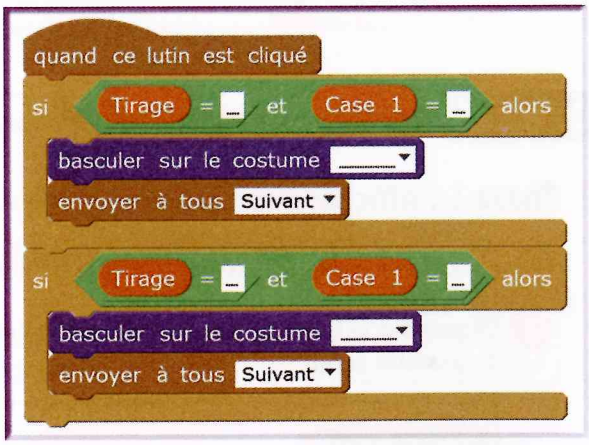
En début de partie, un tirage permet de désigner le joueur qui commence. Ensuite à chaque nouveau coup, le tirage n'est plus aléatoire : il est inversé afin que chaque joueur puisse jouer successivement.

En vous aidant de l'algorithme écrit en situation 2 (Étape 2) :

6 **Compléter dans le script du lutin « Case 1 », les deux structures Si alors pour que le joueur dont c'est le tour de jouer (Tirage 1 ou 2) puisse afficher dans la case vide de son choix une croix (Joueur 1) ou un rond (Joueur 2).**

7 **Après avoir fait vérifier votre programme par votre professeur, compléter les scripts des 8 autres cases.**

8 **Disputer une partie avec un camarade pour tester le bon fonctionnement des 9 scripts que vous avez modifiés.**



Étape 4 Mettre au point et exécuter un programme

30'

Phase 2 : Afficher le nombre de coups joués

1 **Lancer le logiciel Scratch et ouvrir le fichier Tic Tac Toe.** ✓lienmini.fr/a152-tic-tac-toe

2 **Cliquer sur le lutin « Case 1 » et sur l'onglet Scripts. Créer et afficher sur l'écran la variable Nombre de coups.**

3 **Compléter le script pour que le programme compte et affiche le nombre de coups joués. Répéter cette opération pour les lutins « Case 2 », « Case 3 », ... « Case 9 ».**

4 **Engager une partie avec un camarade pour tester le bon fonctionnement de tout le programme.**

Aller plus loin

GAGNÉ

Lorsqu'un des deux joueurs aligne trois symboles, il a gagné. Comment repérer un coup gagnant et afficher le nombre de parties gagnées pour chaque joueur ?

PROJET 5 Coder un message en Morse

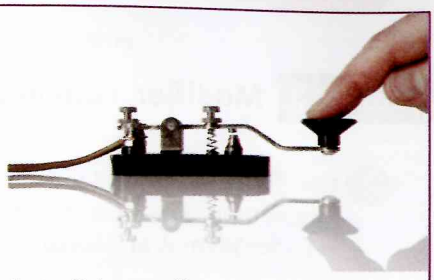
Le **code Morse** a été mis au point par Alfred Lewis Vailen, partenaire de Samuel Morse (inventeur du télégraphe), en 1837 aux États-Unis. Cette codification permet de transmettre un message à l'aide d'**impulsions courtes ou longues** sous la forme de signaux électriques ou lumineux.

Aujourd'hui le codage d'un texte en Morse est encore utilisé par des amateurs. Les signaux électriques reçus sont **convertis en sons** : le décodage final se fait donc à l'oreille.

Les lettres, les chiffres et les signes de ponctuation sont formés d'un ensemble de points et de traits (1 à 6). L'impulsion longue représentée par un **trait** est trois fois plus longue que l'impulsion brève représentée par un **point**.

Le 3 novembre 1860, la convention radiotélégraphique internationale de Berlin crée le signal de détresse « **SOS** » (*Save Our Souls* - Sauvez nos âmes). Ces trois lettres furent choisies pour leur reconnaissance instantanée en code Morse et pour la simplicité de leur codage.

• Comment coder un message en Morse rapidement ?



© Jayfish, Fotolia

Code Morse international

A	• —	S	• • •
B	— • • •	T	—
C	— • — •	U	• • —
D	— • •	V	• • • —
E	•	W	• — —
F	• • — •	X	— • • —
G	— — •	Y	— • — —
H	• • • •	Z	— — • •
I	• •		
J	• — — —	1	• — — — —
K	— • —	2	• • — — —
L	• — • •	3	• • • — —
M	— —	4	• • • • —
N	— •	5	• • • • •
O	— — —	6	— • • • •
P	• — — •	7	— — • • •
Q	— — • —	8	— — — • •
R	• — •	9	— — — — •
		0	— — — — —

Étape 1 Analyser le principe de codage d'un message en Morse

20'

1 **Expliquer comment sont codés les lettres et les chiffres en Morse.**

.....

.....

2 **Préciser comment est transformé le signal électrique codé en Morse lors de sa réception.**

.....

.....

• INFORMATIONS •

On représente le point par le son « **TI** » et le trait par le son « **TA** ».

3 **Dans le tableau suivant, coder chaque lettre en Morse.**

Lettre alphabet	A	B	C	D	E
Code Morse	. _				
Son	TITA	TATITITI	TATITATI	TATITI	TA

4 **Dans le tableau suivant, coder chaque chiffre en Morse et compléter les sons correspondants.**

Chiffre	1	2	3	4	5
Code Morse	. _ _ _ _				
Son	TITATATATA				

5 **Dans le tableau suivant, coder le mot « BAC » en Morse et compléter les sons correspondants.**

Message	B	A	C
Code Morse			
Son			

Étape 2 Modifier, compléter, écrire un algorithme

20'

1 **Comparer les deux algorithmes suivants. Repérer l'instruction supplémentaire utilisée dans le second algorithme « Coder un mot en Morse ».**

.....

.....

• INFORMATIONS •

Pour programmer l'affichage d'un mot en Morse, on extrait chaque lettre du mot. Puis, pour chaque lettre, on affiche le code Morse correspondant.

S	O	S
• • •	— — —	• • •

Algorithmes

Situation 1 Coder une lettre en Morse	Situation 2 Coder un mot en Morse

Créer la variable Lettre
Demander la lettre à coder
Stocker la réponse dans la variable Lettre
Si la variable Lettre est égale à A Alors
  Afficher la lettre A en Morse
Si la variable Lettre est égale à B Alors
  Afficher la lettre B en Morse
Si la variable Lettre est égale à C Alors
  Afficher la lettre C en Morse
etc.
	Créer la variable Mot
	Demander le mot à coder
	Stocker la réponse dans la variable Mot
	Extraire la 1re lettre du mot et la stocker dans la variable Lettre
	Si la variable Lettre est égale à A Alors
	  Afficher la lettre A en Morse
	etc.
	Extraire la 2e lettre du mot et la stocker dans la variable Lettre
	Si la variable Lettre est égale à A Alors
	  Afficher la lettre A en Morse
	etc.

2 **Déterminer la ou les difficultés que l'on rencontre dans les algorithmes des situations 1 et 2.**

.....

.....

Étape 3 Écrire un programme

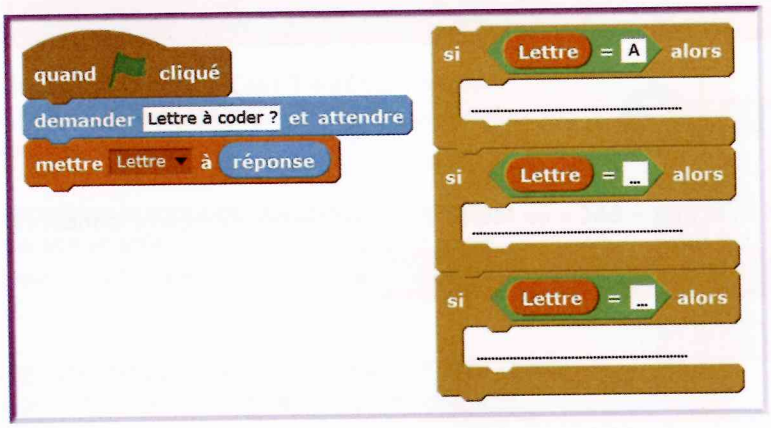
30'

Phase 1 : Coder une lettre en Morse

1 **En vous appuyant sur l'algorithme de la situation 1 (Étape 2), compléter ci-dessous les trois structures alternatives.**

• INFORMATIONS •

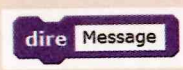
À chaque lettre correspond un code en Morse. Pour la lettre « A » on affiche un point un trait « . _ », pour la lettre « B » un trait trois points «_ . . . », pour la lettre « C » un trait un point un trait un point « _ . _ . ».



Aide

Programmation par blocs

Ce bloc d'instruction permet de comparer deux valeurs.



Ce bloc d'instruction permet d'afficher un message.

2. **Lancer le logiciel Scratch. Ouvrir le fichier Lettre Morse 1.** lienmini.fr/a152-lettre-morse1

3. **Compléter le programme (script) Lettre Morse 1. Tester votre programme.**

Phase 2 : Créer un sous-programme

• INFORMATIONS •

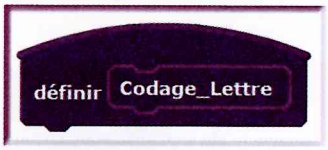
Le premier programme **Lettre Morse 1** ne code qu'une lettre et ne reconnaît que les lettres A, B et C. Si on veut coder un mot de trois lettres, il faut recopier trois fois le codage en morse de toutes les lettres de l'alphabet.
Pour éviter une saisie répétitive des mêmes blocs d'instruction, on peut **créer un bloc** ou **sous-programme**. La rubrique Ajouter Blocs de Scratch permet de créer un nouveau bloc.

4. **Lancer le logiciel Scratch. Ouvrir le fichier Lettre Morse 2.** lienmini.fr/a152-lettre-morse2

5. **Cliquer sur la rubrique Ajouter Blocs et créer un bloc « Codage_Lettre ».**

6. **Accrocher les trois blocs d'instruction « Si Alors » au bloc « définir ».**

7. **Ajouter le bloc d'instruction Codage_Lettre au programme (script) principal et tester votre programme.**



Étape 4 Mettre au point et exécuter un programme

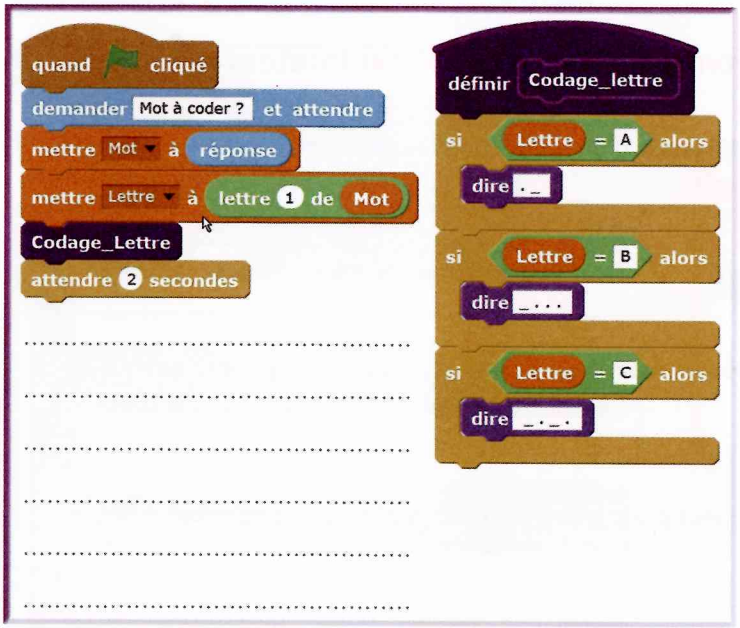
20'

Phase 3 : Coder un mot de trois lettres en Morse

1. **Lancer le logiciel Scratch. Ouvrir le fichier Mot Morse.** lienmini.fr/a152-mot-morse

2. **Cliquer sur le lutin « Robot » et sur l'onglet Scripts.**

3. **En vous appuyant sur l'algorithme de la situation 2 (Étape 2), compléter et tester le programme Mot Morse afin qu'il affiche le code Morse d'un mot de trois lettres (A, B, C).**



Aide

Programmation par blocs

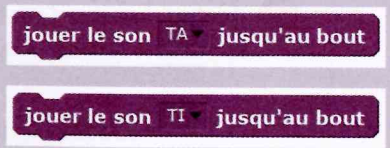
Ce bloc d'instruction permet de connaître le premier caractère d'un mot.

Aller plus loin

Compléter le programme pour qu'il affiche le code Morse du signal de détresse « SOS » et qu'il joue le son « TI » qui représente le point et le son « TA » qui représente le trait.

lienmini.fr/a152-son-ti

lienmini.fr/a152-son-ta



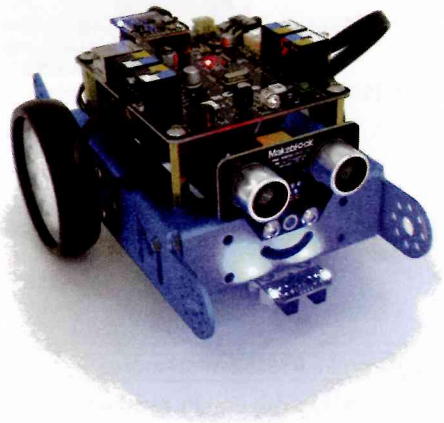
PROJET 6 Guider un robot à distance

© Michael Gottschalk/Getty Images

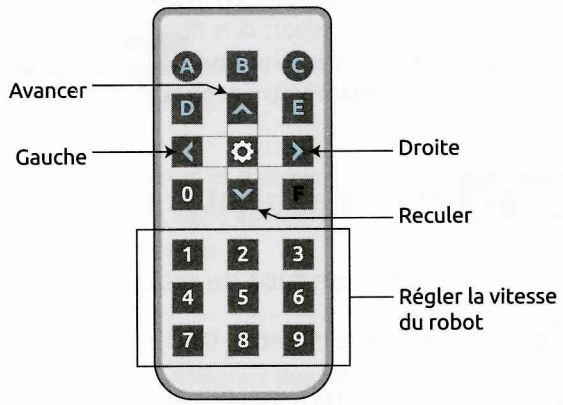
Pour explorer des zones dangereuses ou difficiles d'accès (galeries souterraines, bâtiments en désaffection), réaliser des travaux à risque (déminage, contrôle d'emplacement...) on utilise des robots que l'on peut guider à distance.

Le robot blindé **Teodor** (ci-dessus) est piloté à distance grâce à une console portative. Monté sur chenilles et bardé de caméras haute définition, il donne toutes les informations relatives à un colis suspect : radiographie aux rayons X et détection de produits chimiques ou toxiques. Ces images sont transmises aux démineurs qui peuvent ensuite analyser la situation et agir en conséquence. Son bras articulé, muni d'une pince, peut saisir et déplacer le colis suspect ou le matériel de déminage.

Le robot **mBot** est contrôlable à distance grâce à une **télécommande**. Il est possible de programmer les touches de la télécommande pour leur associer certaines actions. Généralement, on affecte aux flèches le déplacement du robot et aux chiffres une vitesse.



Télécommande - Robot mBot



● **Comment programmer un robot pour qu'il puisse être contrôlé à distance ?**

Étape 1 Analyser le fonctionnement d'un robot explorateur

20'

1 **Donner deux situations où l'utilisation d'un robot contrôlé à distance présente un intérêt.**

.....

.....

.....

2 **Désigner les touches de la télécommande que l'on peut programmer pour orienter à distance le robot mBot.**

.....

3 **Désigner les touches que l'on peut programmer pour régler la vitesse du robot mBot.**

.....

.....

4 **Rechercher une fonction essentielle au contrôle à distance du robot qui n'est pas repérée sur la télécommande.**

.....

.....

Étape 2 Modifier, compléter, écrire un algorithme

20'

• INFORMATIONS •

Le déplacement du robot mBot se fait grâce à deux moteurs (M1 et M2).
La rotation des moteurs est codée dans un sens entre 0 et 255, entre 0 et – 255 dans l'autre sens.
Les valeurs disponibles sont les suivantes : – 255 ; – 100 ; – 50 ; 0 ; 50 ; 100 ; 255. La valeur 0 correspond à l'arrêt du moteur. Les valeurs 100 et – 100 correspondent aux valeurs par défaut.
Pour faire tourner le robot, les deux moteurs doivent fonctionner en sens inverse.

Moteurs	Le robot est arrêté	Le robot avance	Le robot tourne à droite	Le robot tourne à gauche	Le robot recule
Moteur 1 (M1)	Puissance = 0	Puissance = 100	Puissance = 100	Puissance = – 100	Puissance = – 100
Moteur 2 (M2)	Puissance = 0	Puissance = 100	Puissance = – 100	Puissance = 100	Puissance = – 100

En vous aidant des informations disponibles ci-dessus, compléter les algorithmes suivants pour les situations 2, 3 et 4.

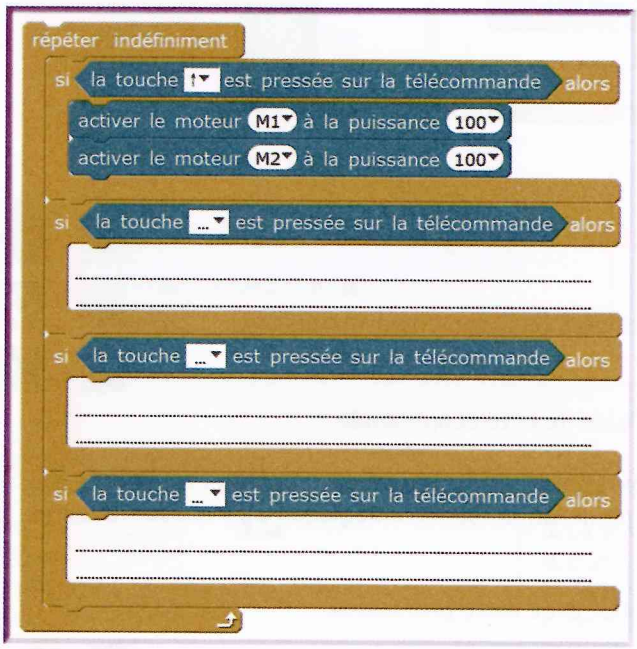
Situation 1	Situation 2	Situation 3	Situation 4
Si la touche ↑ est pressée Alors Faire tourner M1 à 100 Faire tourner M2 à 100	Si la touche → est pressée Alors	Si la touche ← est pressée Alors	Si la touche ↓ est pressée Alors ....

Étape 3 Écrire un programme

50'

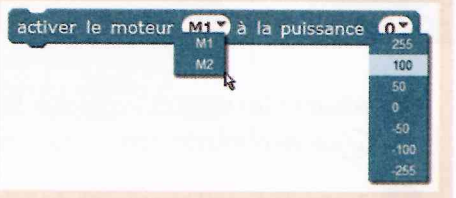
Phase 1 : Contrôler les déplacements du robot à l'aide des touches de direction

1 Compléter le programme pour que l'on puisse, avec les quatre touches de direction de la télécommande, orienter le robot.



Aide

Programmation par blocs



Ce bloc d'instruction permet de choisir le moteur et sa vitesse.

2 Lancer le logiciel mBlock et ouvrir le fichier Explorateur 1. lienmini.fr/a152-explorateur1

3 À l'aide des blocs de la rubrique Pilotage, compléter le programme qui permet de déplacer le robot à l'aide des quatre touches de direction.

Phase 2 : Contrôler la vitesse d'avance et de recul du robot à l'aide des chiffres

Dans le tableau qui suit, chaque chiffre de la télécommande est associé à une puissance (vitesse du robot mBot).

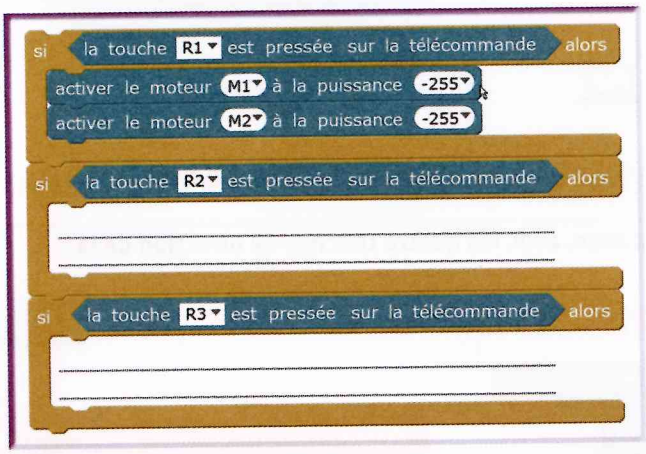
Comportement du robot	Touche	P
Le robot recule au maximum de sa vitesse	1	– 255
Le robot recule à sa vitesse par défaut	2	– 100
Le robot recule lentement	3	.....

Comportement du robot	Touche	P
Le robot avance à sa vitesse maximum	4	.....
Le robot avance à sa vitesse par défaut	5	100
Le robot avance lentement	6	.....

4 **Compléter la colonne « P » (Puissance) du tableau pour chaque comportement du robot.**

5 **Lancer le logiciel mBlock puis ouvrir le fichier Explorateur 2.** ✓lienmini.fr/a152-explorateur2

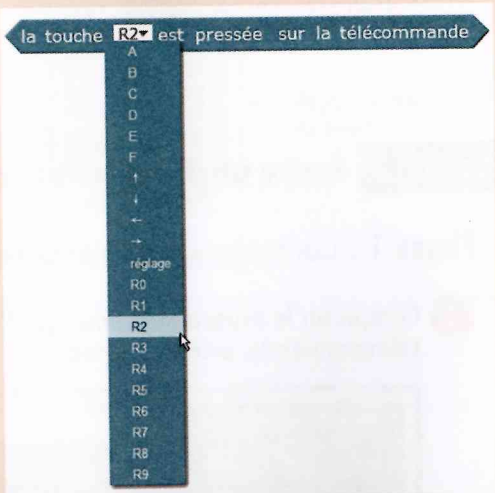
6 **Sachant que les flèches de direction et la touche 1 (R1) sont déjà programmées, compléter le programme pour que les chiffres 2, 3, 4, 5 et 6 de la télécommande fassent varier la vitesse du robot.**



Aide

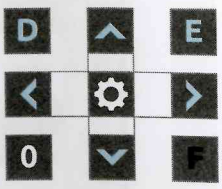
Programmation par blocs

Les touches R0 à R9 du bloc d'instruction correspondent aux touches 0 à 9 de la télécommande.



15'

Étape 4 Mettre au point et exécuter un programme



1 **Allumer le robot et implanter le programme (voir document ressource).** ✓lienmini.fr/a152-robot-implanter

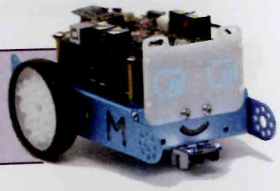
2 **Tester le fonctionnement du robot à l'aide de la télécommande.**

3 **Un problème est apparu : il n'est pas possible d'arrêter le robot. Compléter le programme pour que le robot s'arrête lorsqu'on appuie sur la touche « Réglage » qui se trouve au centre des flèches.**

4 **Tester et corriger le programme du robot avec votre professeur.**

Aller plus loin

Comment afficher le mot « STOP » à destination de personnes en danger en appuyant sur la lettre D de la télécommande ?



• BILAN •

L'essentiel

« Variable »

→ Une **variable** est désignée par son nom (A, B, etc.). Elle sert à stocker une **donnée**. Elle peut être définie comme un **nombre**, une **chaîne de caractères** ou une **liste** d'éléments (nombres, textes). Les variables sont généralement **déclarées** et **initialisées** (association d'une première valeur) au début de l'algorithme ou du programme.

→ L'**affectation** consiste à mémoriser une valeur ou un résultat dans une variable. On dit par exemple « affecter la valeur 5 à la variable A ».

« Si Alors »

→ La **structure alternative** « **Si Alors** » permet de tester une **condition**. La condition comprend un **opérateur de comparaison** (<, =, >) ou un **opérateur logique** (Et, Ou, Non).

→ Par exemple, A < B (A est strictement inférieur à B) est une condition. Si la condition est **vérifiée** (**Vrai**), alors une ou plusieurs instructions seront exécutées.

Aide mémoire

« Variable »

Algorithmique

Afficher le tarif d'une place de cinéma

Créer les variables Age, Tarif Jeune, Tarif Majeur, Tarif Adulte

Initialiser la variable Tarif Jeune à 5

Initialiser la variable Tarif Majeur à 7

Initialiser la variable Tarif Adulte à 10

Programmation

« Si Alors »

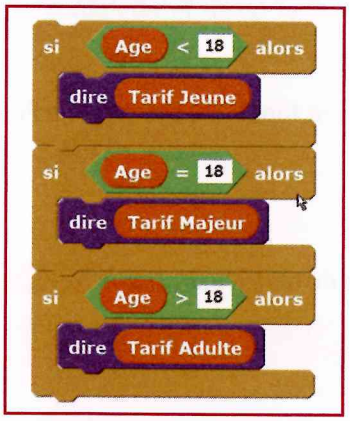
Algorithmique

Si l'âge du client est inférieur à 18 Alors
  Afficher le tarif Jeune

Si l'âge du client est égal à 18 Alors
  Afficher le tarif Majeur

Si l'âge du client est supérieur à 18 Alors
  Afficher le tarif Adulte

Programmation



QCM Entourer la bonne réponse.

1. Une variable sert à :
a. tester une condition.
b. résoudre un problème.
c. stocker une valeur.

2. Dans le programme ci-dessus, à la variable « Tarif Jeune » est affectée la valeur :
a. 7.
b. 10.
c. 5.

3. La structure alternative « Si Alors » permet :
a. d'affecter une valeur à une variable.
b. de tester une condition.
c. de répéter plusieurs fois la même instruction.

4. La condition « Age < 18 » du programme s'applique aux :
a. plus de 12 ans.
b. moins de 18 ans.
c. plus de 18 ans.

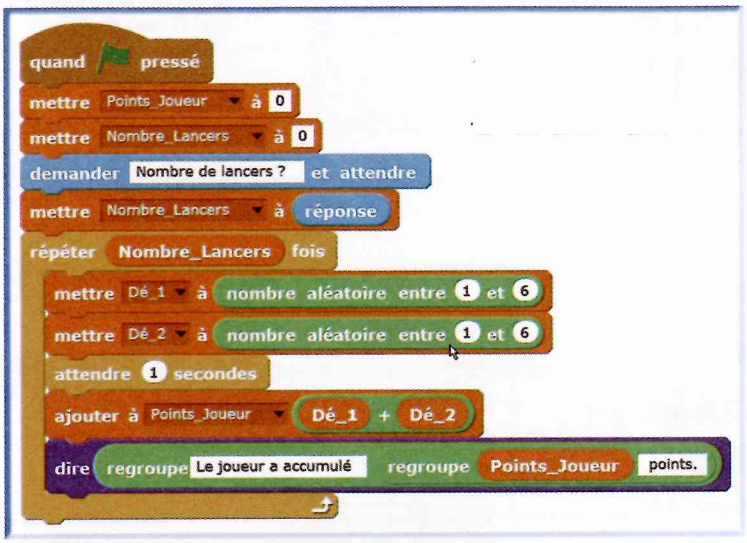
ACTIVITÉ 5 La structure « Répéter »

Je découvre

Le jeu proposé consiste à lancer deux dés à **six faces** en même temps. On effectue la somme des points obtenus. Le vainqueur est celui qui obtient au total le plus de points avec 10 lancers.

• INFORMATIONS •

Dans le programme qui suit, pour simuler le lancer d'un dé à 6 faces, on utilise le bloc d'instruction : nombre aléatoire entre 1 et 6. Il affecte au hasard à chaque lancer de dé un nombre entier compris entre 1 et 6.



1 **À partir de l'écran ci-dessus, noter le nombre de lancers et la somme totale des points obtenus par le joueur.**

Nombre de lancers : ..... Nombre de points obtenus : .....

2 **Indiquer le nom de la variable d'entrée qui permet de faire varier le nombre de lancers.**

.....

3 **Préciser comment fonctionne la variable Points_Joueur à chaque nouveau lancer.**

.....

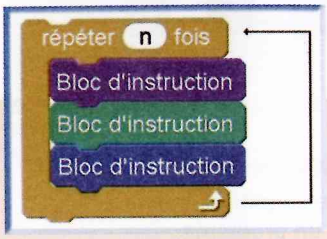
.....

4 **Préciser le rôle du bloc d'instruction** Répéter n fois **dans ce programme.**

.....

.....

.....



Je comprends

La **structure répétitive** « Répéter » permet d'exécuter *n* fois une séquence d'instructions. On parle aussi de « **boucle** ». Le nombre de répétitions peut être fixé par un nombre ou par une variable « Nombre ».

Je retiens

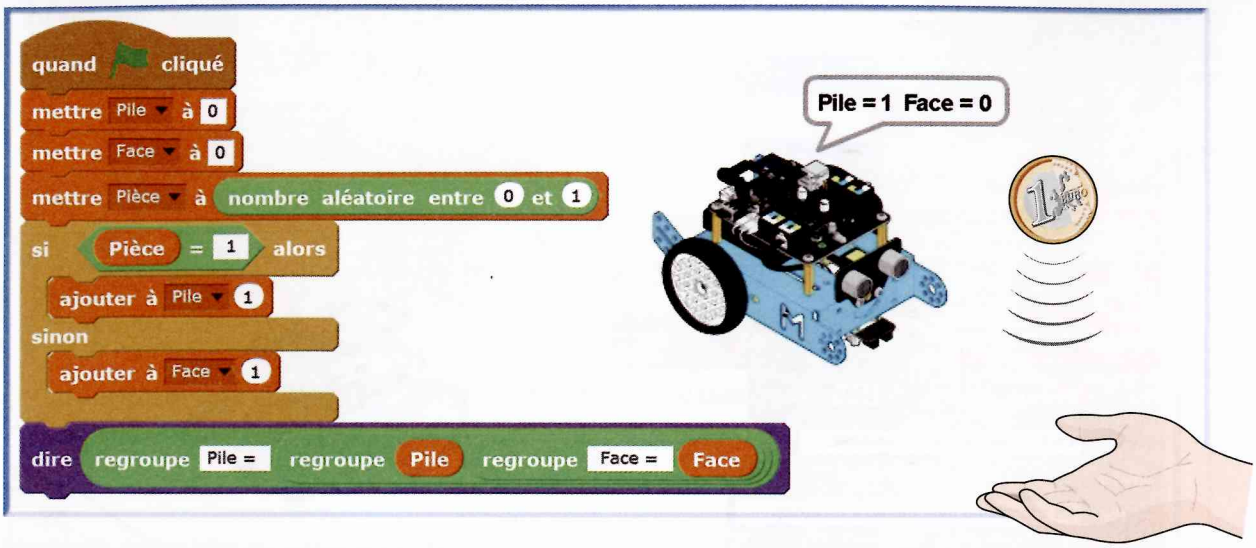
- Le bloc d'instruction Répéter n fois permet d'exécuter *n* fois une séquence d'instructions (traitement).
- Le bloc d'instruction nombre aléatoire entre X et Y permet de générer un nombre entier au hasard dans un intervalle donné.

J'applique

20'

APPLICATION 1 Le jeu de Pile ou Face

Le jeu de **Pile ou Face** est un jeu de hasard. Il consiste à lancer une pièce de monnaie composée d'un côté appelé Pile (**1**) et d'un côté opposé nommé Face (**0**). Ce jeu est souvent utilisé pour départager deux candidats, choisir l'équipe qui engage le match…



1. **Préciser la fonction de ce programme.**

.....

2. **Repérer les deux blocs d'instruction qui permettent de cumuler le nombre de Pile et de Face obtenu.**

Bloc d'instruction 1 : ..... Bloc d'instruction 2 : .....

3. **Lancer le logiciel Scratch et ouvrir le fichier Pièce.** ✓lienmini.fr/a152-piece

4. **En utilisant la structure répétitive (boucle)** répéter 10 fois **, compléter et tester le programme pour qu'il affiche le résultat de 10 lancers.**

20'

APPLICATION 2 Le tirage de 10 lettres

Dans le jeu des **chiffres et des lettres**, on tire 6 chiffres et 10 lettres au hasard. Dans le tirage de 10 lettres ci-contre se cache un mot que vous entendez souvent en informatique. Quel est ce mot ?

R	H	O	L	I	M	E	G	T	A
									E

1. **Lancer le logiciel Scratch et ouvrir le fichier Lettres.** ✓lienmini.fr/a152-lettres

2. **Tester ce programme et expliquer son fonctionnement.**

.....

.....

.....

3. **En utilisant la structure répétitive (boucle)** Répéter n fois **, compléter et tester le programme pour qu'il affiche 10 lettres.**

• INFORMATIONS •

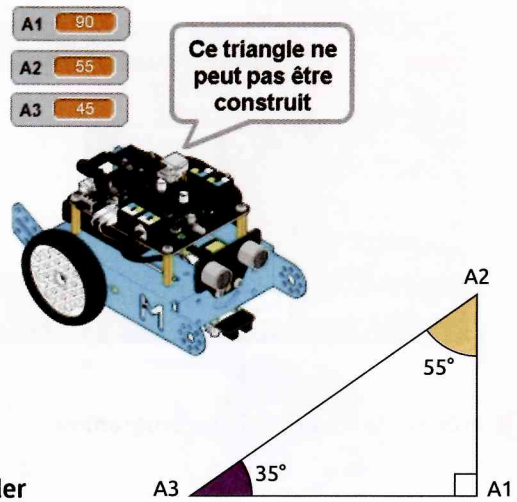
Pour générer un nombre compris entre 1 et 26 (nombre de lettres de l'alphabet), on utilise le bloc d'instruction nombre aléatoire entre a et b et le bloc d'instruction « lettre Nombre de » qui permet d'extraire une lettre.

ACTIVITÉ 6 La structure « Si Alors Sinon »

Je découvre

```
quand pressé
demander [Entrez l'angle A1] et attendre
mettre [A1] à [réponse]
demander [Entrez l'angle A2] et attendre
mettre [A2] à [réponse]
demander [Entrez l'angle A3] et attendre
mettre [A3] à [réponse]
mettre [Somme_Angles] à (A1 + A2 + A3)
si (Somme_Angles = 180) alors
    dire [Ce triangle peut être construit.]
sinon
    dire [Ce triangle ne peut pas être construit.]
```

En géométrie, la somme des angles d'un triangle est égale à **180 degrés**. Le programme suivant permet de vérifier si la somme de trois angles forme un triangle.



1 **À partir des données affichées par le robot ci-dessus, calculer le résultat affecté à la variable Somme_Angles.**

.....

2 **Noter le texte affiché par le programme si la variable Somme_Angles est égale à 180 degrés.**

.....

3 **Noter le texte affiché si la variable Somme_Angles est différente de 180.**

.....

4 **Préciser le rôle de l'instruction** si alors **dans le programme.**

.....

5 **Préciser le rôle de l'instruction** sinon **dans le programme.**

.....

Je comprends

- La **structure alternative** « **Si Alors** » peut être complétée par une seconde instruction : « **Sinon** ». Elle se transforme ainsi en « **Si Alors Sinon** ».
- La première partie de cette structure exécute un traitement si la **condition** est vérifiée. La seconde partie exécute un autre traitement si la condition n'est pas vérifiée.

```
si Condition alors
    Bloc d'instruction
    Bloc d'instruction
sinon
    Bloc d'instruction
```

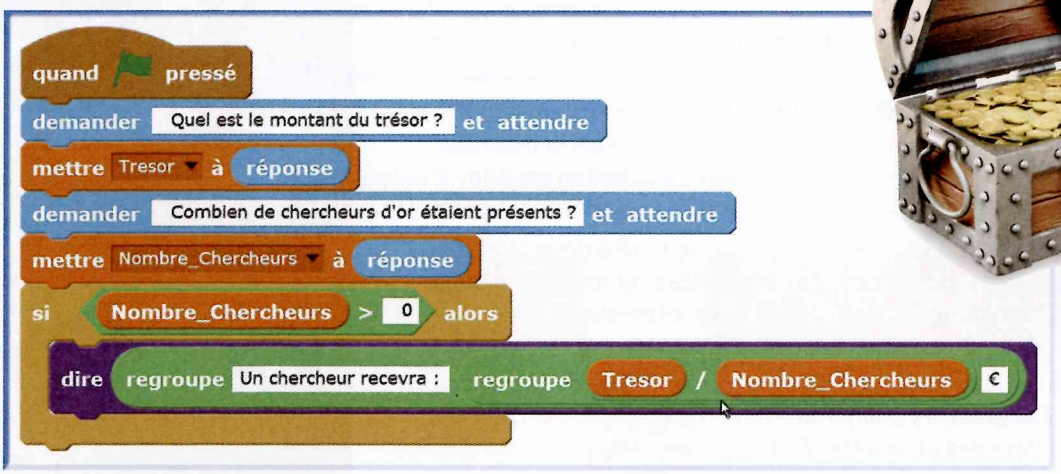
Je retiens

Le bloc d'instruction « Si (condition) Alors... Sinon » permet d'exécuter deux séquences d'instructions :
– une séquence si la condition est vérifiée ;
– une autre séquence si la condition n'est pas vérifiée.

J'applique

20'

APPLICATION 1 Le partage d'un trésor



© Koya 979, Fotolia

1. **Dans le programme affiché ci-dessus, noter le nom du bloc d'instruction qui permet d'obtenir le montant du trésor que chaque chercheur recevra.**

.....

2. **Lancer le logiciel de programmation Scratch et ouvrir le fichier Trésor.** lienmini.fr/a152-tresor

3. **Tester le programme pour différents montants du trésor (1 000, 2 000...) et différents nombres de chercheurs (2, 3, 4...).**

4. **Compléter et tester le programme pour afficher le texte suivant : « Le trésor ne sera pas partagé. » lorsque la condition (Nombre_Chercheurs > 0) n'est pas vérifiée.**

20'

APPLICATION 2 Comment programmer un bouton poussoir ?

Un « **bouton poussoir** » est un interrupteur qui ne change pas d'apparence quand on appuie dessus. Si le courant est coupé, un premier appui rétablit le courant – pour allumer une lumière en général – et un second appui coupe de nouveau le courant.

1. **Lancer le logiciel de programmation Scratch et ouvrir le fichier Bouton-poussoir.** lienmini.fr/a152-bouton-poussoir

2. **Cliquer sur la « Scène Arrière-plans ». Comment s'appellent les deux arrière-plans ?**

.....

.....

3. **Sélectionner le script du lutin « Bouton ».**

	Situation 1 La scène est éteinte	Situation 2 La scène est allumée
Évènement Quand ce lutin est cliqué	Si l'arrière-plan est éteint Alors Basculer sur l'arrière-plan allumé	Sinon Basculer sur l'arrière-plan éteint

4. **En vous aidant du tableau ci-dessus, compléter la structure alternative « Si Alors Sinon » du programme et tester son fonctionnement.**

PROJET 7 Jouer au jeu des allumettes (jeu de Nim)

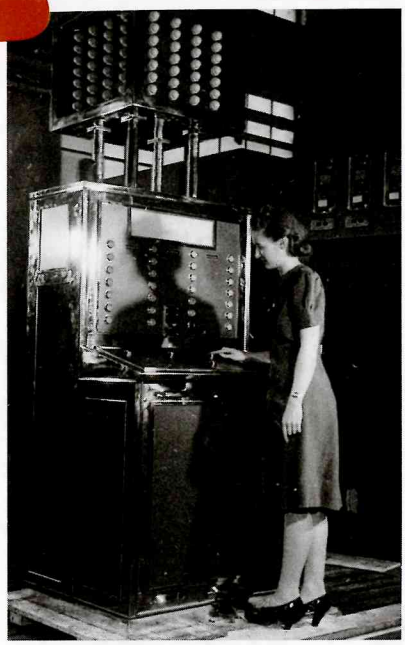
En 1940, Edward Condon a conçu et breveté une machine électromécanique appelée le **Nimatron**, imaginée exclusivement pour jouer au jeu de Nim.

La machine affichait à l'aide de grosses lampes l'état du jeu. Le joueur choisissait le nombre de feux à éteindre. Puis la machine jouait à son tour. Des personnes réussirent à la battre car son concepteur avait volontairement limité son algorithme de calcul.

Le **jeu de Nim** est un jeu de stratégie qui se joue à deux avec des graines, des billes, des jetons, des **allumettes** ou tout autre objet facilement manipulable. Nous allons nous intéresser à la version rendue célèbre par le jeu télévisé Fort Boyard.

On dispose de 20 allumettes. Chaque joueur à tour de rôle retire entre **1 et 3 allumettes** obligatoirement. Le gagnant est celui qui ramasse la dernière allumette. À chaque nouvelle partie, on change le premier joueur.

- **Comment programmer le jeu de Nim avec deux joueurs ?**



© The New York Public Library

Étape 1 Analyser les règles d'un jeu

20'

1. **En quelle année le « Jeu de Nim » ou « Jeu des allumettes » a-t-il été programmé pour la première fois sur le Nimatron ?** .....

2. **Préciser la raison pour laquelle la machine ne gagnait pas toutes les parties.**

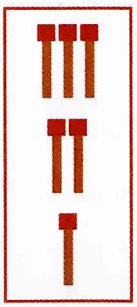
.....

.....

3. **Donner le nombre d'allumettes que chaque joueur peut retirer du jeu.**

.....

4. **Au cours d'une partie du jeu de Nim sur écran, il reste 5 allumettes. C'est au joueur 2 de retirer les allumettes. Expliquer pourquoi le joueur 2 est sûr de perdre s'il retire 3 allumettes parmi les 5.**



.....

.....

5. **Expliquer pourquoi le joueur 2 est sûr de perdre s'il retire 2 allumettes parmi les 5.**

.....

.....

6. **Expliquer pourquoi le joueur 2 est sûr de gagner s'il retire 1 allumette parmi les 5.**

.....

.....

.....

.....

.....

Étape 2 Modifier, compléter, écrire un algorithme

10'

• INFORMATIONS •

– En début de partie, le programme doit d'abord afficher (Estampiller) une allumette (1er costume) puis avancer de 17 pas autant de fois que le nombre d'allumettes à faire apparaître.
– En cours de partie, le programme doit d'abord reculer de 17 pas puis afficher (Estampiller) une allumette blanche (2e costume) autant de fois que d'allumettes à enlever.

Pour la situation 1 et la situation 2, compléter l'algorithme.

Algorithmes

	Situation 1 - Début de partie Évènement : Afficher des allumettes		Situation 2 - En cours de partie Évènement : Enlever des allumettes
Costume 	Répéter la séquence autant de fois qu'il y a d'allumettes à afficher	Costume 	Répéter la séquence autant de fois qu'il y a d'allumettes à enlever ....

Étape 3 Écrire un programme

30'

1. **Lancer le logiciel de programmation Scratch. Ouvrir le fichier Nim 1.** ✓ lienmini.fr/a152-nim1

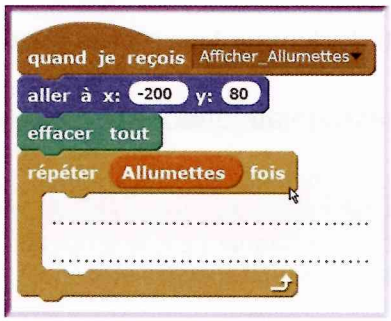
Phase 1 : Afficher les allumettes en début de partie

2. **Cliquer sur le lutin « Allumette » et sur l'onglet Scripts.**

3. **Cliquer sur le drapeau vert et entrer un nombre entre 1 et 20. Quel problème rencontrez-vous ?**

.....

.....



Aide

Programmation par blocs

Permet de dessiner une image du lutin à l'écran.

4. **En vous appuyant sur l'algorithme de la situation 1 (Étape 2), compléter la structure répétitive du programme pour qu'il affiche le nombre d'allumettes demandé. Tester votre programme.**

Phase 2 : Enlever les allumettes en cours de partie

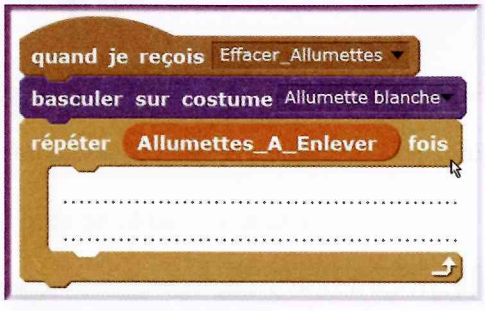
5. **Cliquer sur le drapeau vert et entrer 10 comme nombre d'allumettes.**

6. **Entrer 2 comme nombre d'allumettes à enlever. Quel problème rencontrez-vous ?**

.....

.....

.....



7. **En vous appuyant sur l'algorithme de la situation 2 (Étape 2), compléter la structure répétitive du programme pour qu'il enlève les allumettes. Tester votre programme.**

40'

Étape 4 Mettre au point et exécuter un programme

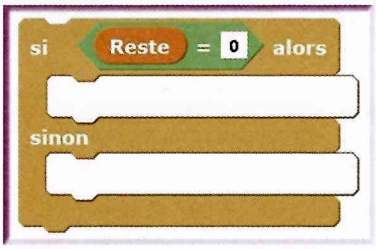
Phase 3 : Vérifier si un des deux joueurs a enlevé les dernières allumettes

1. Lancer le logiciel de programmation Scratch. Ouvrir le fichier Nim 2. lienmini.fr/a152-nim2
2. Cliquer sur le drapeau vert. Démarrer le jeu avec uniquement 3 allumettes.
3. Rechercher sur l'écran du jeu à quelle valeur correspond la variable Reste, si on enlève trois allumettes.

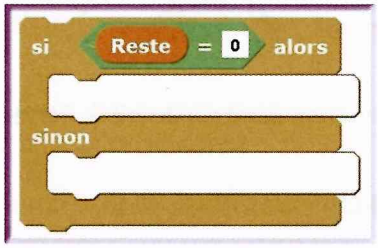
.....

4. Compléter ci-dessous la structure alternative du script des Joueurs 1 et 2 pour que :
– si le Joueur 1 a enlevé les dernières allumettes (Reste = 0), alors il envoie au lutin « Robot » l'ordre d'exécuter le script « Gagnant 1 », sinon il envoie l'ordre d'exécuter le script « Joueur 2 » (le jeu continue) ;
– si le Joueur 2 a enlevé les dernières allumettes (Reste = 0), alors il envoie au lutin « Robot » l'ordre d'exécuter le script « Gagnant 2 », sinon il envoie l'ordre d'exécuter le script « Joueur 1 » (le jeu continue).

Joueur 1



Joueur 2



Aide

Programmation par blocs

5. Modifier les deux scripts des lutins « Joueur 1 » et « Joueur 2 ».

Phase 4 : Afficher un message au joueur qui a gagné la partie

Pour que le robot puisse afficher « Félicitations Joueur 1 » ou « Félicitations Joueur 2 », il faut qu'il reçoive un message du Joueur 1 ou du Joueur 2. Lorsque le robot reçoit le message « Gagnant 1 » ou « Gagnant 2 », le script doit montrer le robot, afficher le message de félicitations pendant 5 secondes et stopper complètement la partie.

6. Compléter les deux programmes du robot lorsqu'il reçoit le message « Gagnant 1 » ou « Gagnant 2 ».

Aide

Programmation par blocs

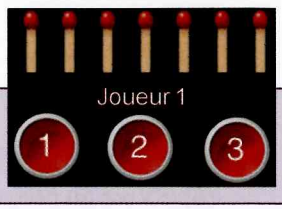
Permet d'afficher le lutin du script en cours.

Permet d'arrêter tous les programmes.

7. Engager une partie avec un camarade pour terminer le programme.

Aller plus loin

Modifier votre programme pour que chaque joueur puisse enlever 1, 2 ou 3 allumettes en cliquant sur un des trois nombres.

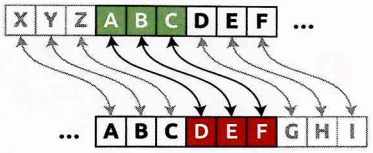


PROJET 8 Décaler les lettres d'un message (chiffre de César)

L'empereur romain César transmettait des messages secrets à ses généraux. Pour cela il utilisait un système de codage appelé depuis le **chiffre de César** ou **code de César**. Le chiffre de César repose sur le chiffrement par décalage. Il consiste à transformer chaque lettre en une autre lettre située à 3 rangs dans l'ordre de l'alphabet. Par exemple la lettre A se transforme en D, la lettre B en E, la lettre C en F, etc.

Le **chiffrement** est une technique qui permet de rendre la compréhension de messages impossible à toute personne qui n'a pas la **clé de chiffrement**.

Le **déchiffrement** est le processus inverse du chiffrement. Il sert à rendre les messages à nouveau intelligibles à condition de connaître la **clé de déchiffrement**. Aujourd'hui le chiffrement et le déchiffrement font appel à des **algorithmes** de plus en plus complexes.



● **Comment programmer un message en utilisant le chiffre de César ?**

Étape 1 Analyser le principe de chiffrement d'un message

20'

1 **Indiquer la valeur du décalage utilisé dans le chiffre de César et le principe de fonctionnement de ce type de chiffrement.**

.....

.....

2 **Coder chaque lettre en utilisant une valeur de décalage de 3 (clé de chiffrement du code de César).**

Lettre	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
Lettre codée	D	E	....	....	....	....	....	....	....	....	....	....	....	....	....	....	....	....	....	....

3 **Coder chaque lettre en utilisant une valeur de décalage de 4 (clé de chiffrement du code KO).**

Lettre	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
Lettre codée	....	....	....	....	....	....	....	....	....	....	O	....	....	....	....	....	....	....	....	....

4 **Coder le mot « STOP » en utilisant la clé du code de César et le mot « BLOC » en utilisant la clé du code KO.**

Lettre	S	T	O	P
Lettre codée	....	....	....	....

Lettre	B	L	O	C
Lettre codée	....	....	....	....

5 **Décoder le message « SLOH » en utilisant la clé du code de César et « JEGI » en utilisant la clé du code KO.**

Lettre codée	S	L	O	H
Lettre décodée	....	....	....	....

Lettre codée	J	E	G	I
Lettre décodée	....	....	....	....

Étape 2 Modifier, compléter, écrire un algorithme

20'

1 **Préciser le résultat que l'on obtient lorsqu'on exécute l'algorithme 1 de la situation 1.**

.....

.....

2 **Détailler sur une feuille de brouillon le fonctionnement de la structure répétitive de l'algorithme 2.**

Algorithmes

Situation 1 Chiffrer une lettre avec le code de César	Situation 2 Chiffrer un mot de quatre lettres avec le code de César

Si la variable Lettre est égale à A Alors
  Affecter à la variable Lettre_Cesar la lettre D
  Afficher la variable Lettre_Cesar
Sinon
  Si la variable Lettre est égale à B Alors
  Affecter à la variable Lettre_Cesar la lettre E
  Afficher la variable Lettre_Cesar
  Sinon
  Si la variable Lettre est égale à C Alors
  Affecter à la variable Lettre_Cesar la lettre F
  Afficher la variable Lettre_Cesar
  Sinon
Afficher « Erreur de saisie. Recommencez ! »
	Créer les variables Lettre, Mot, Position_Lettre, Lettre_Cesar, Mot_Coder
	Demander un mot à coder de 4 lettres
	Stocker la réponse dans la variable Mot
	Initialiser les variables
	Répéter autant de fois qu'il y a de lettres dans la variable Mot
	  Ajouter +1 à la variable Position_Lettre
	  Extraire la lettre
	  Chiffrer la lettre (sous-programme Chiffrement Lettre)
	  Ajouter la lettre codée à la variable Mot_Coder
	  Attendre 1/2 seconde
	Afficher le chiffrement du Mot de 4 lettres

Étape 3 Écrire un programme

40'

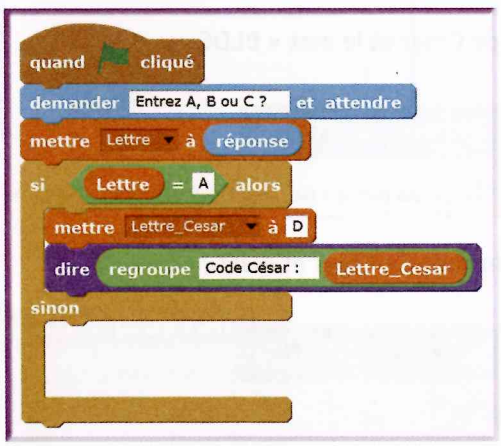
Phase 1 : Chiffrer une lettre avec le code de César

1 **Lancer le logiciel Scratch. Ouvrir le fichier Lettre César 1.** lienmini.fr/a152-lettre-cesar1

2 **En vous appuyant sur l'extrait de l'algorithme présenté en situation 1 (Étape 2), compléter et tester le script pour qu'il affiche le chiffrement des lettres A, B et C selon le code de César.**

• INFORMATIONS •

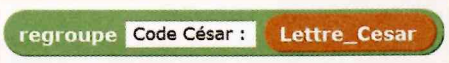
Si la valeur saisie n'est pas égale à A, B ou C, alors on affiche « Erreur de saisie. Recommencez ! ».



Aide

Programmation par blocs

Ce bloc d'instruction permet d'afficher un message.



Ce bloc d'instruction permet de regrouper différents éléments (textes, variables).

Phase 2 : Définir un sous-programme de chiffrage des lettres

3. Déterminer la ou les difficultés que l'on rencontre lorsqu'on a plusieurs lettres à chiffrer.

.....

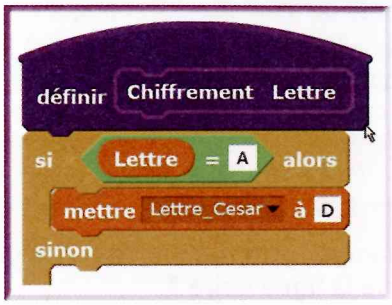
.....

4. Lancer le logiciel **Scratch**. Ouvrir le fichier **Lettre César 2**. lienmini.fr/a152-lettre-cesar2

5. Cliquer sur la rubrique Ajouter Blocs et créer un bloc « Chiffrement Lettre ».

6. Accrocher les trois blocs d'instruction Si Alors Sinon au sous-programme « définir ».

7. Ajouter le bloc d'instruction Chiffrement Lettre au script principal et tester le bon fonctionnement de votre programme.



Étape 4 Mettre au point et exécuter un programme

30'

Phase 3 : Chiffrer un mot de 4 lettres avec le code de César

Vous devez afficher le code de César d'un mot comportant les lettres O, P, S, T.

1. Préciser combien de fois il sera nécessaire d'appeler le bloc d'instruction Chiffrement Lettre.

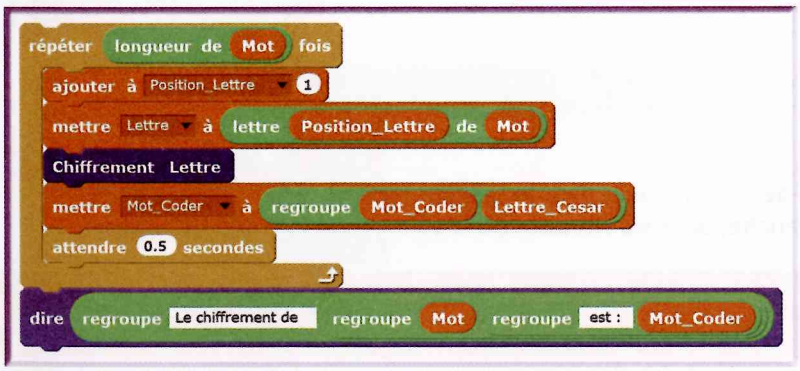
.....

.....

2. Lancer le logiciel **Scratch**. Ouvrir le fichier **Mot César**. lienmini.fr/a152-mot-cesar

3. Cliquer sur le lutin « **Robot** » et sur l'onglet **Scripts**.

4. À partir de l'écran ci-dessous, compléter la structure répétitive du programme **Mot César**.



Aide

Programmation par blocs

longueur de STOP

Ce bloc d'instruction permet d'avoir la longueur d'un mot.

lettre 1 de STOP

Ce bloc d'instruction permet de connaître le premier caractère d'un mot.

5. Tester le fonctionnement de votre programme avec les mots suivants : stop, pots, spot.

6. En vous appuyant sur l'extrait de l'algorithme présenté en situation 2 (Étape 2), expliquer l'intérêt d'utiliser la structure répétitive affichée ci-contre.

.....

.....

.....

.....

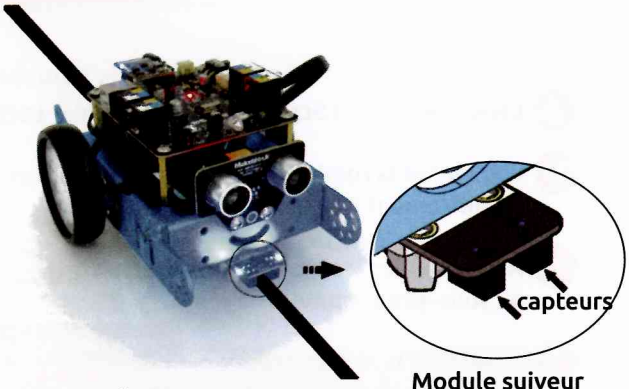
Aller plus loin

Modifier le programme pour que le chiffrement d'un mot puisse se faire avec une valeur de clé donnée (0 < clé < 5).

PROJET 9 Faire suivre une ligne à un robot

Le robot transporteur emmène des objets d'un point à un autre en suivant un marquage au sol (ligne noire). Pour assurer cette fonction, il dispose à l'avant d'un **module suiveur de ligne**, composé de deux capteurs optiques.

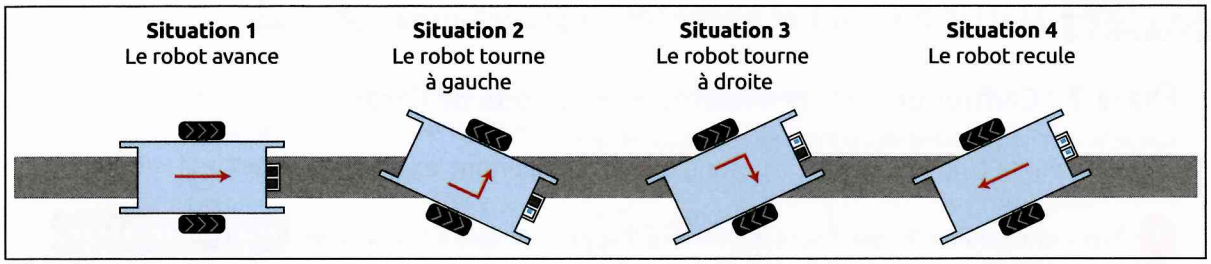
Tant que les deux capteurs détectent la ligne, le robot avance **(situation 1)**. Lorsqu'un des deux capteurs ne détecte plus la ligne, le robot tourne sur lui-même pour se remettre dans l'axe **(situation 2 ou 3)**. Si les deux capteurs sont en dehors de la ligne, le robot recule **(situation 4)**.



Robot transporteur — Module suiveur

- **Comment programmer le robot pour qu'il suive la ligne noire ?**

Comportement du robot



Étape 1 Analyser le comportement d'un robot

20'

1 **Noter l'usage du robot et préciser son comportement dans le cadre de la situation 1.**

.....

.....

.....

.....

2 **En vous aidant du schéma ci-dessus, indiquer dans le tableau suivant la position des deux capteurs optiques (droit et gauche) dans les situations 2, 3 et 4.**

Situation 1 Le robot avance	Situation 2 Le robot tourne à gauche	Situation 3 Le robot tourne à droite	Situation 4 Le robot recule
→ Le capteur optique droit est sur la ligne noire.	→	→	→ .....
→ Le capteur optique gauche est sur la ligne noire.	→	→	→ .....

Étape 2 Modifier, compléter, écrire un algorithme

20'

1 **Pour les situations 2 et 3, compléter chaque test de l'algorithme qui permet au robot de se remettre dans l'axe de la ligne noire.**

• INFORMATIONS •

– Si **les deux capteurs détectent** la ligne noire, l'état du module suiveur de ligne passe à **0**.
– Si **le capteur droit ne détecte plus** la ligne noire, l'état du module suiveur de ligne passe à **1**.
– Si **le capteur gauche ne détecte plus** la ligne noire, l'état du module suiveur de ligne passe à **2**.
En dehors de ces valeurs reçues par le module suiveur de ligne, le robot recule (situation 4).

Situation 1 Les deux capteurs détectent la ligne noire	Situation 2 Le capteur droit ne détecte plus la ligne noire	Situation 3 Le capteur gauche ne détecte plus la ligne noire
Si État suiveur de ligne = **0** Alors Faire avancer le robot Sinon	Si État suiveur de ligne = **1** Alors ..... Sinon	Si État suiveur de ligne = **...** Alors ..... Sinon

2 **Préciser le comportement du robot si l'état du module suiveur de ligne est différent de 0, 1 ou 2.**

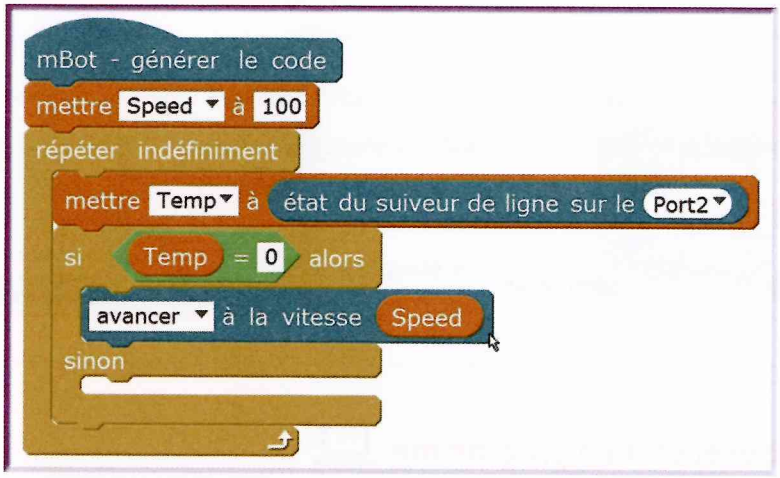
.....

.....

.....

Étape 3 Écrire un programme

45'



• INFORMATIONS •

La vitesse du robot est affectée à la **variable « Speed »**.
L'état du module suiveur de ligne est affecté à la **variable « Temp »**.

À partir de l'écran ci-dessus qui décrit le début du programme du robot transporteur :

1 **Rappeler le rôle du bloc d'instruction** répéter indéfiniment.
Le bloc de programmation répéter indéfiniment permet de .....

.....

2 **Préciser le numéro de la connexion utilisée (port) pour récupérer l'état du module suiveur de ligne.**

.....

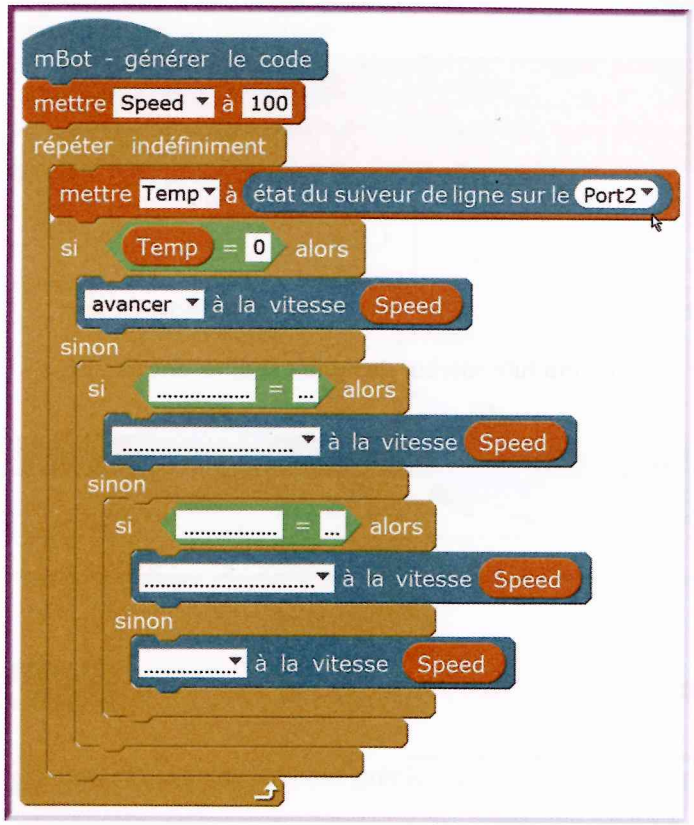
.....

3 **Expliquer le fonctionnement du bloc d'instruction « Si Temp = 0 Alors avancer à la vitesse Speed ».**

.....

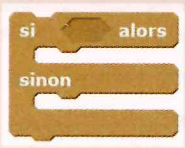
.....

4 **Écrire la fin du programme pour que le robot puisse tourner (situations 2 et 3) et reculer (situation 4).**



Aide

Programmation par blocs



Structure alternative qui permet de tester une condition. La condition compare deux valeurs ou fait des opérations logiques.

Permet de faire avancer le robot à une vitesse déterminée.

5 **Lancer le logiciel de programmation mBlock.**

6 **Ouvrir le fichier Robot suiveur.** lienmini.fr/a152-robot-suiveur

7 **À l'aide des blocs de la rubrique Pilotage, compléter le programme.**

30'

Étape 4 Mettre au point et exécuter un programme

1 **Allumer le robot et implanter le programme (voir document ressource fourni par votre professeur).** lienmini.fr/a152-robot-implanter

2 **Tester et corriger le programme du robot suiveur de ligne avec votre professeur.**

Aller plus loin

Proposer une amélioration du programme pour que le robot s'arrête 20 cm avant de percuter un obstacle.

· BILAN ·

L'essentiel

« Répéter »

→ La **structure répétitive** « **Répéter** » permet d'exécuter un nombre de fois donné une séquence d'instructions. On parle aussi de « **boucle** ».
Le nombre de répétitions peut être fixé par un nombre ou par une variable « Nombre ».

« Si Alors Sinon »

→ La **structure alternative** « **Si Alors** » peut être complétée par une seconde instruction « **Sinon** ».
Elle se transforme alors en « **Si Alors Sinon** ».
La première partie de cette structure exécute un traitement si la **condition est vérifiée**. La seconde partie exécute un autre traitement si la condition n'**est pas vérifiée**.

Aide mémoire

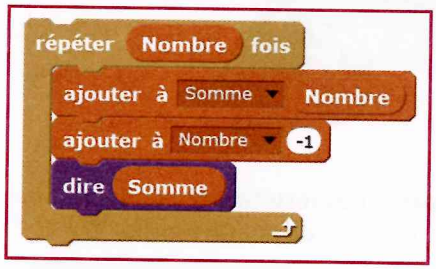
« Répéter »

Algorithmique

Afficher la somme de n entiers

Saisir et mémoriser un nombre
Répéter n fois (n = Nombre)
| Ajouter la valeur du nombre à la variable Somme
| Soustraire 1 à la variable Nombre
| Afficher la somme des nombres

Programmation



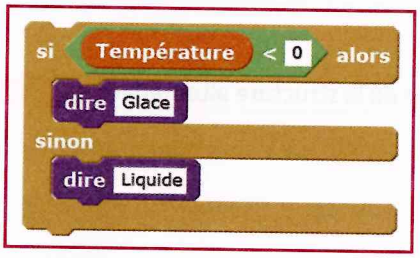
« Si Alors Sinon »

Algorithmique

Afficher l'état de l'eau

Si Température < 0 Alors
| Afficher « Glace »
Sinon
| Afficher « Liquide »

Programmation



QCM Entourer la bonne réponse.

1. La structure répétitive « répéter » facilite :
a. l'affectation d'un nombre à une variable.
b. le test d'une condition.
c. la répétition d'une séquence d'instructions.

2. La structure répétitive du programme ci-dessus permet :
a. d'exécuter un évènement 10 fois.
b. de répéter une séquence d'instructions autant de fois que la valeur saisie dans la variable « Nombre ».
c. de répéter indéfiniment la valeur des 3 premières instructions.

3. La structure alternative « Si Alors Sinon » permet :
a. d'exécuter deux traitements différents en fonction d'une condition.
b. de répéter une condition plusieurs fois.
c. de lancer une séquence d'instructions en fonction d'un évènement.

4. La structure alternative du programme ci-dessus permet d'afficher :
a. trois états possibles de l'eau (glace, liquide, vapeur).
b. la température.
c. l'état de l'eau (glace ou liquide) en fonction de la température.

ACTIVITÉ 7 La structure « Répéter jusqu'à »

Je découvre

a et b sont deux entiers. On dit que b est un des **diviseurs** de a si le **reste** de la division de a par b est égal à 0.

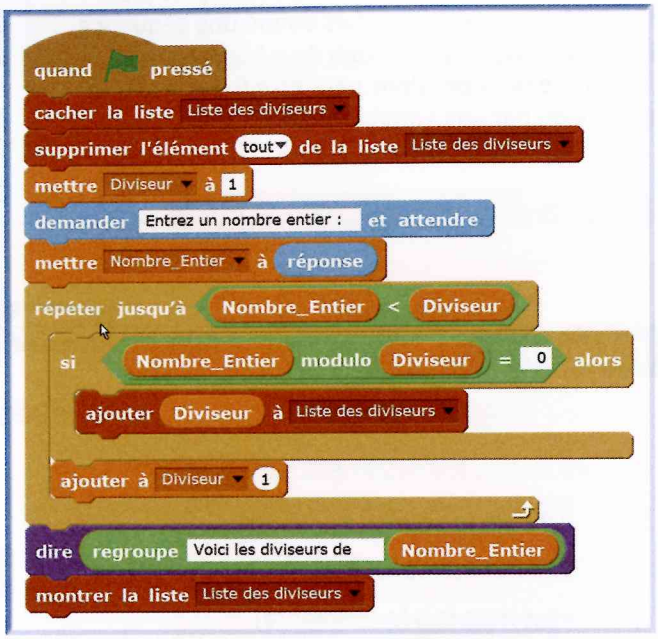
Exemple : 6 est un des diviseurs de 12 car le reste de la division de 12 par 6 est égal à 0.

Pour connaître tous les diviseurs de 12, on répète cette opération pour chaque nombre entier compris entre 1 et 12. On s'arrête à 12 car on ne divise pas le dividende par un nombre plus grand que lui.

Dividende Diviseur

12	6
− 12	2
0	Quotient

Reste



1 **À partir des résultats et du programme affichés ci-dessus, repérer le nombre de diviseurs du nombre entier 12.**

.....

2 **Expliquer pourquoi au début du programme on affecte à la variable « Diviseur » la valeur 1.**

.....

.....

3 **Déterminer le rôle de la structure alternative** Si Alors **dans le programme.**

.....

.....

.....

4 **Préciser le rôle du bloc d'instruction** Répéter jusqu'à condition **dans le programme.**

.....

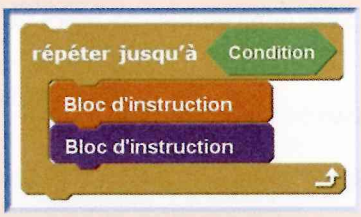
.....

• INFORMATIONS •

Le bloc d'instruction modulo permet de calculer le reste de la division du nombre entier par le diviseur.

Je comprends

La structure répétitive « **Répéter jusqu'à** » permet d'exécuter *n* fois une séquence d'instructions jusqu'à ce que la **condition** soit vérifiée. On parle aussi de « **boucle** ». Lorsque la condition est vérifiée, le programme sort de la boucle.



Je retiens

Le bloc d'instruction « **répéter jusqu'à (condition)** » permet d'exécuter plusieurs fois une séquence d'instructions jusqu'à ce qu'une condition soit vérifiée.

J'applique

APPLICATION 1 Le forage d'un puits

20'

Le forage d'un puits pour atteindre les nappes phréatiques nécessite des machines coûteuses. Une entreprise propose de creuser le premier mètre pour 100 €, le second mètre pour 150 € et ainsi de suite en augmentant le prix de chaque mètre supplémentaire de 50 €.

1 **Sachant qu'on dispose d'un budget de 2 500 €, déterminer à l'aide du tableau suivant le nombre de mètres qu'on peut creuser au maximum.**

Mètres	1	2	3	4	5	6	7	8	9
Prix du mètre	100	150	200	250	.....	.....	.....	.....	.....
Coût total	100	250	450	.....	.....	.....	.....	.....	.....

Avec un budget de 2 500 €, il est possible de creuser à une profondeur maximum de : .....

2 **Lancer le logiciel Scratch et ouvrir le fichier Puits.** ✓lienmini.fr/a152-puits

3 **Tester le programme. Comparer les résultats obtenus à ceux du tableau et justifier l'utilisation de la structure « répéter jusqu'à ».**

© Biosphoto/Albert Montanier

APPLICATION 2 Le plus grand commun diviseur de deux nombres

30'

On utilise régulièrement le Plus Grand Commun Diviseur (PGCD) de deux nombres entiers pour rendre irréductible une fraction.

Exemple :
La liste des diviseurs de 18 est : 1,2,3,6,9,18.
La liste des diviseurs de 24 est : 1,2,3,4,6,8,12,24.
Les diviseurs communs à 18 et 24 sont : 1,2,3,6.
Le PGCD de 18 et de 24 est : 6.

1 **Lancer le logiciel de programmation Scratch et ouvrir le fichier PGCD.**
✓lienmini.fr/a152-PGCD

• INFORMATIONS •

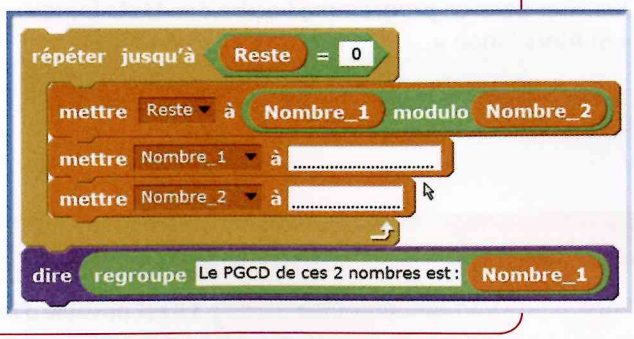
Pour trouver le Plus Grand Commun Diviseur (PGCD) de deux nombres, on peut utiliser l'**algorithme d'Euclide**.
Celui-ci comporte trois étapes. Le bloc d'instruction modulo permet de calculer le reste de la division du nombre entier par le diviseur.

Algorithme d'Euclide

Étape 1 – On calcule le reste de la division euclidienne du dividende par le diviseur.

Étape 2 – On remplace ensuite le dividende par le diviseur et le diviseur par le reste.

Étape 3 – Tant que le reste est différent de 0, on recommence les étapes 1 et 2.
Lorsqu'on obtient un reste égal à 0, le PGCD devient la dernière valeur du diviseur.



2 **En suivant les trois étapes de l'algorithme d'Euclide, compléter les deux derniers blocs d'instruction de la structure « répéter jusqu'à » à la fin du programme pour qu'il affiche le PGCD des deux nombres saisis.**

3 **Tester ce programme avec les deux nombres 13 856 et 896. Le PGCD affiché doit être égal à 32.**

ACTIVITÉ 8 La structure « Si Alors Sinon » imbriquée

Je découvre

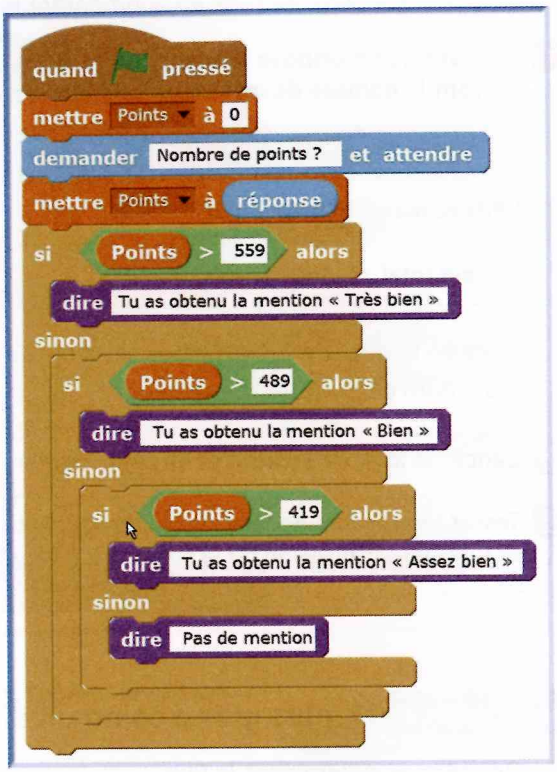
Dans le nouveau **brevet des collèges**, le contrôle continu représente 400 points et le contrôle final 300 points. Le candidat est reçu s'il cumule 350 points sur les 700.

Le diplôme délivré au candidat admis porte :

1. la mention « **Très bien** » quand le candidat a obtenu un total de points au moins égal à 560 sur 700 ;

2. la mention « **Bien** » quand le candidat a obtenu un total de points au moins égal à 490 sur 700 ;

3. la mention « **Assez bien** » quand le candidat a obtenu un total de points au moins égal à 420 sur 700.



1 **Repérer le nombre de tests qu'effectue ce programme.**

.....

.....

2 **Déterminer dans quel cas un élève obtient la mention « Très bien » et le bloc d'instruction correspondant.**

.....

.....

3 **Préciser la condition pour laquelle un élève n'obtient pas de mention et le bloc d'instruction correspondant.**

.....

.....

4 **Justifier dans ce programme l'ordre des tests lorsqu'on imbrique plusieurs structures « Si Alors Sinon ».**

.....

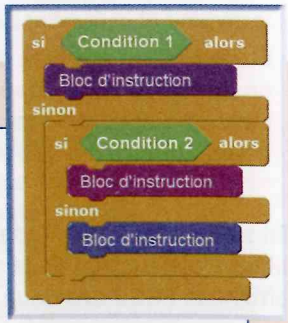
Je comprends

Lorsqu'une **variable** doit être testée en fonction de plusieurs valeurs, on peut utiliser une **structure alternative** « **Si Alors Sinon** » **imbriquée**. Les différents **tests** sont alors introduits dans un ordre précis.

Je retiens

Il est possible d'imbriquer le bloc d'instruction « **si (condition) alors sinon** » dans un autre bloc d'instruction « **si (condition) alors sinon** ».

Si (Condition) Alors
  Instruction
Sinon
  Si (condition) Alors...



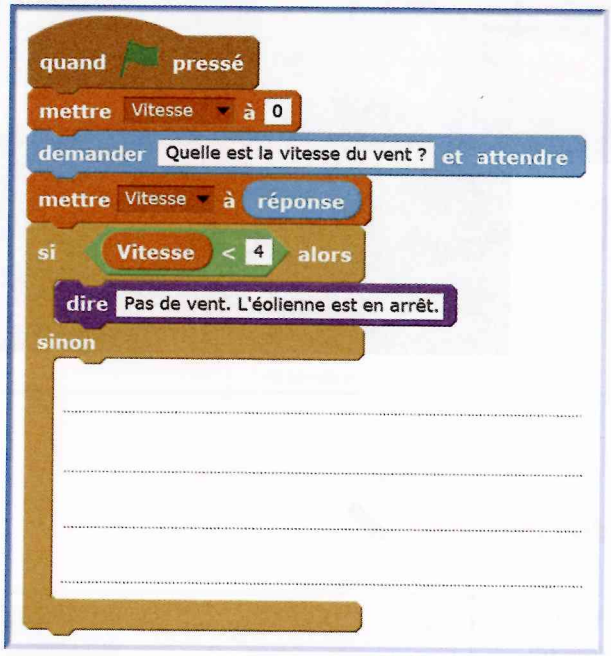
J'applique

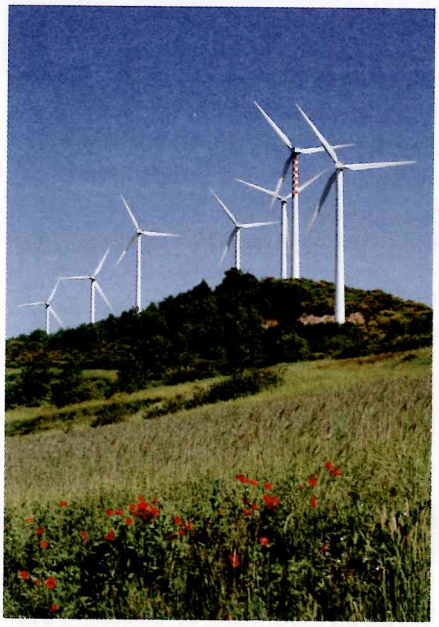
APPLICATION 1 Contrôler le fonctionnement d'une éolienne

20'

L'exploitation des éoliennes est contrôlée à distance par ordinateur. Si la vitesse du vent est strictement inférieure à 4 m/s (soit 14,4 km/h) ou strictement supérieure à 25 m/s (soit 90 km/h), elles doivent s'arrêter automatiquement.

1 **Compléter à l'aide d'une structure alternative imbriquée le programme suivant afin de contrôler l'arrêt ou le fonctionnement de l'éolienne. L'opérateur saisit la vitesse du vent en « m/s ».**





© Kaimanblu, Fotolia

2 **Lancer le logiciel de programmation Scratch et ouvrir le fichier Éolienne.**
lienmini.fr/a152-eolienne

3 **Compléter et tester le programme.**

APPLICATION 2 Calculer rapidement le montant d'une commande

30'

Le collège souhaite commander de nouvelles tablettes pour les élèves de troisième.
Le tableau suivant donne leur prix en fonction de la quantité commandée.

Tablettes	+ de 100	Entre 51 et 100	Entre 11 et 50	Entre 1 et 10
Prix à l'unité (€)	250	270	280	300

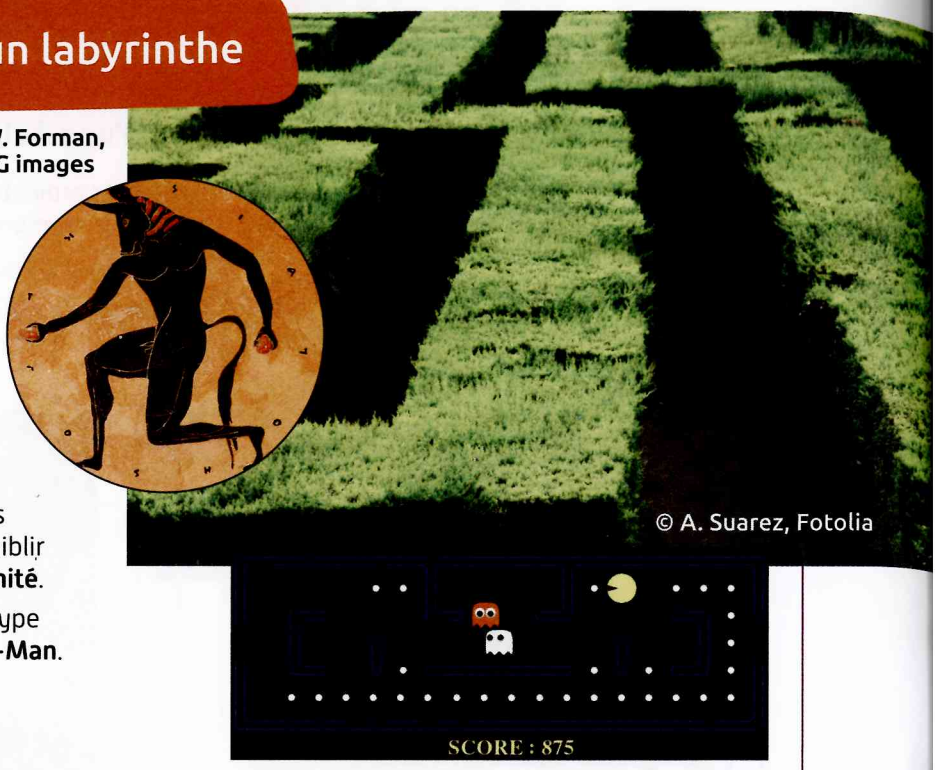
1 **Écrire sur une feuille à l'aide d'une structure alternative imbriquée « Si Alors Sinon » l'algorithme qui permet de connaître le montant total en fonction du nombre de tablettes à commander.**

2 **Lancer le logiciel de programmation Scratch et ouvrir le fichier Tablettes.**
lienmini.fr/a152-tablettes

3 **Compléter le programme pour qu'il calcule et affiche le montant total à payer en fonction du nombre de tablettes à commander.**

4 **Tester ce programme jusqu'à ce que son fonctionnement soit correct.**

PROJET 10 Sortir d'un labyrinthe



© W. Forman, AKG images

© A. Suarez, Fotolia

Le mot « **labyrinthe** » désigne dans la mythologie grecque une suite complexe de galeries construites par **Dédale** pour enfermer le **Minotaure**, une créature féroce à corps d'homme et tête de taureau.

Le jeu vidéo du labyrinthe consiste à se déplacer avec succès dans un décor sinueux, fait d'embranchements, d'impasses et de fausses pistes, à contourner parfois des objets destinés à ralentir ou à affaiblir le personnage. Ceci dans un **temps limité**.

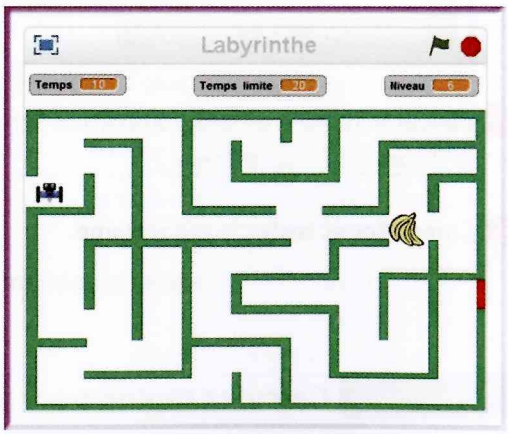
Il existe de multiples variantes de ce type de jeux vidéo : le plus célèbre est **Pac-Man**.

- **Comment programmer le jeu du labyrinthe ?**

Étape 1 Analyser les principes du jeu du labyrinthe

20'

1. **Sur l'écran du labyrinthe ci-contre, tracer en rouge le chemin que le robot doit emprunter pour franchir la ligne d'arrivée rouge.**



2. **Imaginer ce qu'il se passe lorsque le robot (lutin) franchit la ligne d'arrivée rouge.**

.....

.....

3. **Déterminer les 4 types de mouvements que l'on peut programmer afin que le robot puisse se déplacer dans le labyrinthe.**

.....

.....

.....

.....

4. **Repérer dans l'écran ci-dessus l'objet qui permet de savoir combien de labyrinthes ont été parcourus.**

.....

5. **Déterminer le rôle des objets « Temps » et « Temps limite » dans le jeu du labyrinthe.**

.....

.....

.....

.....

Étape 2 Modifier, compléter, écrire un algorithme

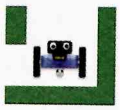
20'

• INFORMATIONS •

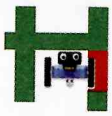
- Lorsque le robot touche un obstacle, il recule de 4 pas (avance de – 4).
- Le joueur dispose d'un temps limité pour arriver à la ligne d'arrivée rouge. Un compteur de temps est activé.
- Lors d'un changement de niveau, le temps est remis à 0, le niveau augmente de 1, le robot est positionné à l'entrée.

Compléter l'algorithme des situations 2 et 4.

Situation 1	Le joueur déplace le robot
	Si j'appuie sur la flèche haute Alors S'orienter vers le haut Avancer de 4 pas

Situation 2	Le robot touche un obstacle
	Si l'obstacle (vert) est touché Alors .....

Situation 3	Le compteur de temps est activé
	Répéter jusqu'à ce que Temps > Temps limite du niveau en cours Ajouter à la variable Temps + 1 Attendre 1 seconde

Situation 4	Le robot atteint la ligne d'arrivée
	Si le robot franchit la ligne d'arrivée Alors .... Repositionner le robot à l'entrée du labyrinthe

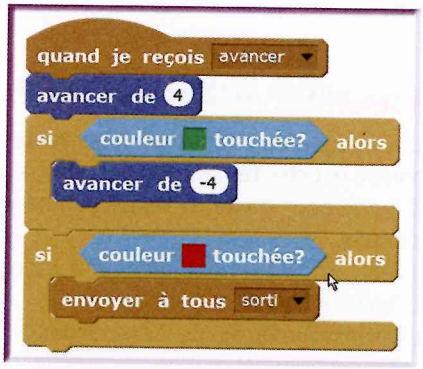
Étape 3 Écrire un programme

40'

Phase 1 : Le robot atteint la ligne d'arrivée

1. **Ouvrir le fichier Labyrinthe 1 et tester le programme à l'aide des 4 flèches de direction du clavier.** ✔ lienmini.fr/a152-labyrinthe1

2. **Sélectionner le lutin « Robot » et cliquer sur l'onglet Scripts.**



Aide

Programmation par blocs

Permet d'envoyer un message aux autres programmes (scripts) qui écoutent le message.

3. **Préciser le comportement du robot lorsqu'il touche un obstacle de couleur verte (Étape 2, situation 2).**

.....

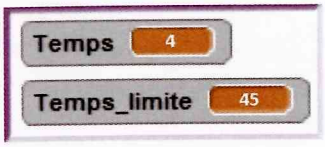
.....

4. **Noter le comportement du robot lorsqu'il touche la ligne d'arrivée rouge (vous pourrez aller sur l'arrière-plan et cliquer sur l'onglet Scripts) (Étape 2, situation 4).**

.....

.....

Phase 2 : Le jeu compte et affiche le temps écoulé



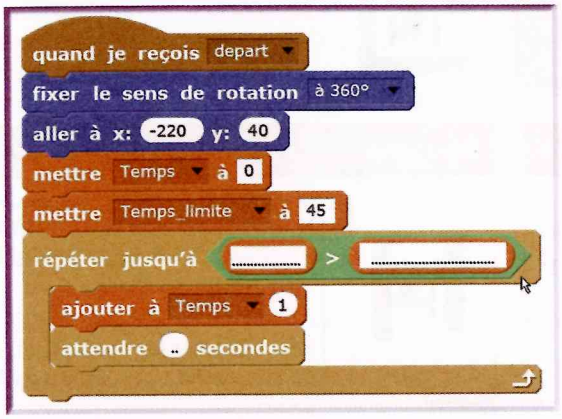
5 **Ouvrir le fichier Labyrinthe 2.** lienmini.fr/a152-labyrinthe2

6 **Sélectionner le lutin « Robot » et cliquer sur l'onglet Scripts.**

• INFORMATIONS •

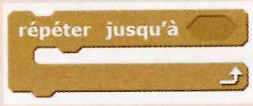
Le temps limite pour traverser un labyrinthe de niveau 1 à 3 est de 45 secondes.

7 **Compléter ci-dessous la structure répétitive « répéter jusqu'à » qui permet de compter 45 secondes.**



Aide

Programmation par blocs



Vous disposez de deux variables : « Temps » et « Temps limite » et d'un exemple de compteur fourni dans le bilan 3 page 59.

8 **Compléter le script « Quand je reçois départ » et tester le programme Labyrinthe 2 pour que le déroulement du temps s'affiche (Étape 2, Situation 3).**

9 **Préciser ce qu'il se passe si le robot franchit la ligne d'arrivée en plus de 45 secondes.**

.....

.....

Étape 4 Mettre au point et exécuter un programme

30'

Phase 3 : Le temps limite change en fonction du niveau

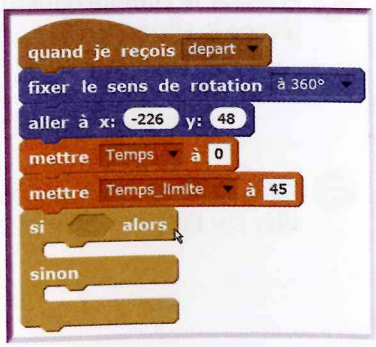
1 **Ouvrir le fichier Labyrinthe 3.**
lienmini.fr/a152-labyrinthe3

2 **Sélectionner le lutin « Robot » et cliquer sur l'onglet Scripts.**

• INFORMATIONS •

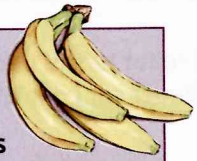
La difficulté du jeu augmente progressivement.
– Si le niveau est strictement inférieur à 4, alors le temps limite pour traverser un labyrinthe est de 45 secondes.
– Si le niveau est strictement inférieur à 7, alors le temps limite est de 30 secondes.
– Sinon, le temps limite est de 20 secondes.

3 **À l'aide d'une structure alternative « Si Alors Sinon » imbriquée, modifier et tester le programme pour que le temps limite change en fonction du niveau de jeu.**



Aller plus loin

Ajouter un objet à attraper pour gagner un bonus de temps sur le niveau en cours. De nombreux objets (lutins) sont disponibles dans la bibliothèque de Scratch.



PROJET 11 Jouer avec une raquette et une balle (jeu de Pong)

Pong est un jeu vidéo inspiré du tennis de table (ping pong) qui a été développé et programmé en 1967. Il est proposé comme console de salon à partir de 1975 et c'est le premier jeu vidéo à connaître un succès populaire mondial.

Ce jeu vidéo peut être joué seul (la raquette opposée est commandée par la machine) ou à deux joueurs, chacun commandant une **raquette**.

Une **balle** se déplace à travers l'écran, rebondissant sur les rebords du haut et du bas. Les joueurs font glisser leur raquette verticalement entre les extrémités de l'écran. Si la balle frappe la raquette, elle rebondit vers l'autre joueur. Si elle ne touche pas la raquette, l'autre joueur augmente son **score** d'un point et une nouvelle balle est engagée.

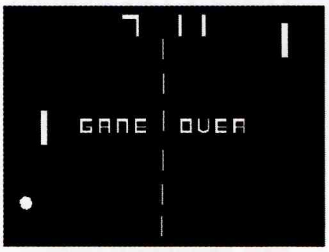
Une manche est gagnée par le joueur qui atteint le premier **11 points** avec 2 points d'écart au minimum sur son adversaire. En fin de manche, l'écran affiche « **Game over** ».

● **Comment programmer le jeu de Pong ?**

© Simon Tang/Rex-Featur/Rex/Sipa

20'

Étape 1 Analyser le fonctionnement et les règles d'un jeu vidéo



1 **Repérer les objets affichés sur l'écran ci-contre du jeu Pong, en dehors de la table et du filet (arrière-plan).**

.....

.....

2 **À l'aide de ces objets, décrire le fonctionnement de ce jeu vidéo.**

.....

.....

.....

.....

.....

3 **Préciser la condition qui permet à un joueur d'être déclaré gagnant d'une manche.**

.....

.....

.....

4 **Déterminer les bords de l'écran sur lesquels la balle ne rebondit pas.**

.....

.....

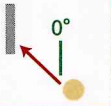
Étape 2 Modifier, compléter, écrire un algorithme

20'

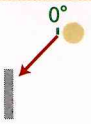
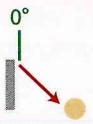
En vous aidant des schémas ci-dessous, déterminer pour les situations 2, 3 et 4 l'orientation (angle) de la balle après son rebond et compléter l'algorithme correspondant.

• INFORMATIONS •
La raquette 1 se situe à gauche de l'écran.
La raquette 2 à droite.

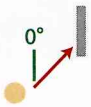
Situation 1
La raquette 1 est touchée par la balle qui vient du bas

Avant – 45°	Après + 45°	Algorithme
		Si Direction = – 45° Alors I Orienter la balle à + 45°

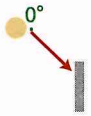
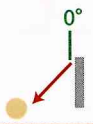
Situation 2
La raquette 1 est touchée par la balle qui vient du haut

Avant – 135°	Après	Algorithme
		SI Direction = – 135° Alors I.....

Situation 3
La raquette 2 est touchée par la balle qui vient du bas

Avant + 45°	Après	Algorithme
		Si Direction = + 45° Alors I.....

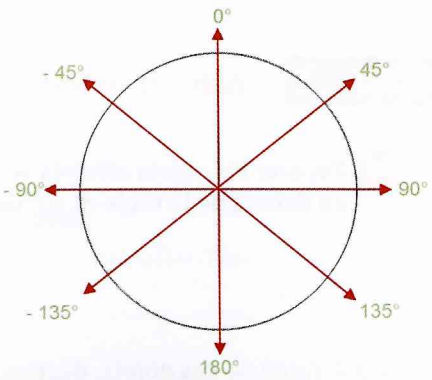
Situation 4
La raquette 2 est touchée par la balle qui vient du haut

Avant	Après	Algorithme
		Si Direction = 135° Alors I.....

Étape 3 Écrire un programme

40'

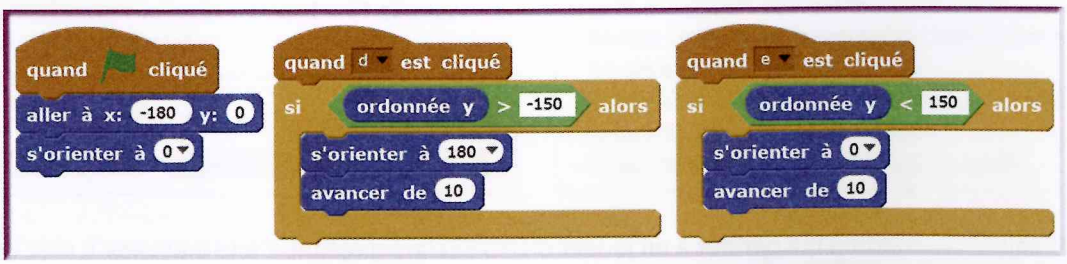
Phase 1 : Le déplacement de la balle et des raquettes



1. **Lancer le logiciel Scratch et ouvrir le fichier Pong 1.**
lienmini.fr/a152-pong1

2. **Lancer le programme et appuyer sur les touches « d » et « e » du clavier pour déplacer la raquette 1.**

3. **Cliquer sur le lutin « Raquette 1 » et sur l'onglet Scripts. Préciser le rôle des trois scripts suivants.**



Le premier script permet de .....

.....

Le deuxième script permet de .....

Le troisième script permet de .....

4. **Repérer les deux touches du clavier affectées à la montée et à la descente de la raquette 2.**

.....

Phase 2 : La balle rebondit sur la raquette

5 **Ouvrir le fichier Pong 2.** lienmini.fr/a152-pong2

6 **Sélectionner le lutin « Balle » et cliquer sur l'onglet Scripts.**

7 **Compléter le script « Balle » pour que celle-ci rebondisse lorsque la raquette est touchée.**

• INFORMATIONS •

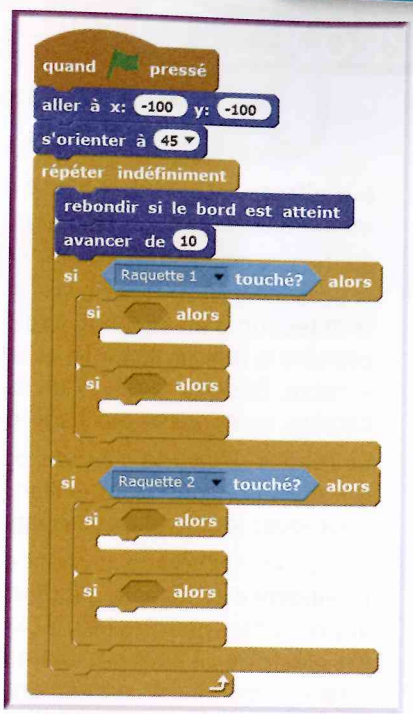
Les angles de rebond ont été précisés dans les situations 1, 2, 3 et 4 de l'Étape 2.

8 **Tester votre programme. Préciser ce qui se passe quand la balle touche le bord droit ou gauche de l'écran. Proposer une nouvelle amélioration.**

.....

.....

.....



40'

Étape 4 Mettre au point et exécuter un programme

Phase 3 : La raquette rate la balle

1 **Ouvrir le fichier Pong 3.** lienmini.fr/a152-pong3

2 **Sélectionner le lutin « Balle » et cliquer sur l'onglet Scripts.**

• INFORMATIONS •

Lorsque la balle passe au-delà de la position 200 en abscisses, le joueur 2 ne peut plus la stopper et le point doit être arrêté. À l'inverse, si la balle passe en deçà de la position – 200 en abscisses, c'est le joueur 1 qui ne peut plus la stopper. Le point doit être arrêté.
On va donc laisser la balle se déplacer jusqu'à ce que l'une des deux positions limites soit dépassée. La variable à tester est abscisse x. La condition logique OU réunit les deux tests nécessaires à l'arrêt (x < – 200) et (x > 200).

3 **Compléter la condition de la structure répéter jusqu'à afin d'arrêter le point en cours lorsque la balle ne rebondit pas sur une des deux raquettes.**

Phase 4 : La gestion du score

4 **Ouvrir le fichier Pong 4.** lienmini.fr/a152-pong4

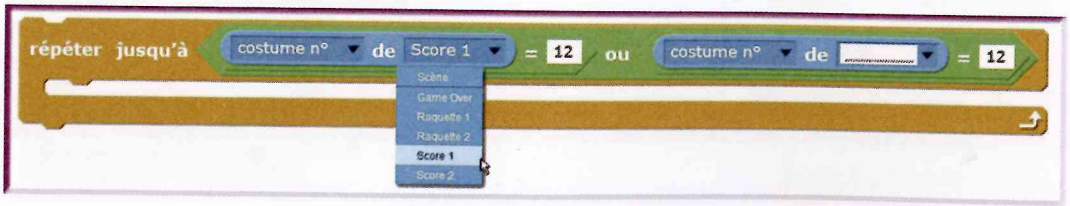
5 **Sélectionner le lutin « Score 1 » et repérer le numéro du costume correspondant au score de 11 points. Justifier le numéro choisi.**

.....

.....

6 **Sélectionner le lutin « Balle » et cliquer sur l'onglet Scripts.**

7 **Compléter la structure répéter jusqu'à pour que le programme affiche chaque point gagné, jusqu'à ce qu'un joueur atteigne les 11 points. Tester le programme.**



Aller plus loin

La balle avance avec la même vitesse (10 pas). Les parties seraient plus intéressantes si la balle prenait de la vitesse. Comment améliorer le programme afin que la balle prenne de la vitesse régulièrement ?

PROJET 12 Faire surveiller un espace par un robot

© EOS innovation

e-vigilante est un robot qui se déplace de manière autonome à l'intérieur d'un entrepôt.

Il effectue des rondes et prévient immédiatement la personne en charge de la surveillance lors de la détection d'un incident. Celle-ci peut alors prendre la main à distance et en temps réel sur le robot. Elle peut évaluer la situation grâce à la caméra, au micro et aux haut-parleurs intégrés.

Le robot e-vigilante n'a donc pas été conçu pour remplacer l'agent de surveillance mais plutôt pour jouer le rôle d'accompagnateur.

Sur le plan technique, le robot est doté d'une **intelligence artificielle** qui rend ses rondes imprévisibles pour les intrus. Ainsi, son parcours est **aléatoire** : il peut par exemple, à une intersection, choisir subitement de tourner à droite ou à gauche ou encore de faire demi-tour face à un obstacle. De plus, il gère seul son autonomie, décidant par lui-même d'aller se recharger.

● **Comment programmer un robot pour qu'il surveille un espace clos ?**

20'

Étape 1 Analyser le comportement d'un robot

1 **Préciser la fonction du robot e-vigilante et son comportement lorsqu'il fait une ronde et qu'il détecte un intrus.**

.....

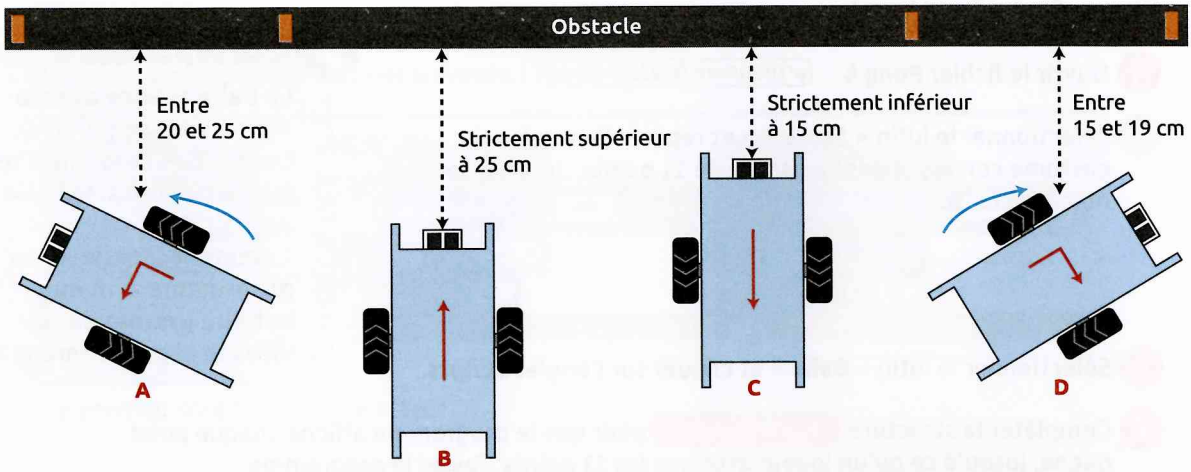
.....

.....

• INFORMATIONS •

Le robot **mBot** dispose d'un **capteur ultrason** qui renvoie la distance qui le sépare d'un obstacle (en centimètres). Caractéristiques : Distance de détection comprise entre 3 cm et 400 cm. Angle maxi de détection : 30 degrés.

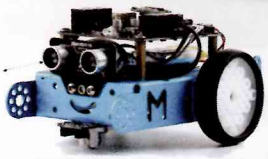
2 **En vous aidant du schéma ci-dessous, décrire dans le tableau le comportement du robot lorsque le capteur ultrason détecte un obstacle.**



Situation A L'obstacle se situe entre 20 et 25 cm	Situation B L'obstacle est à plus de 25 cm	Situation C L'obstacle est à moins de 15 cm	Situation D L'obstacle se situe entre 15 et 19 cm
.....	.....	.....	.....

Étape 2 Modifier, compléter, écrire un algorithme

20'



À l'aide du tableau précédent (Étape 1), compléter l'algorithme qui permet au robot d'éviter les obstacles. Il doit pouvoir se déplacer de manière autonome.

Algorithme

Si la distance de l'obstacle est strictement inférieure à 26 cm Alors
- Si la distance de l'obstacle est .....
  .....
- Sinon
  - Si la distance à l'obstacle est .....
    .....
  - Sinon
    .....
- Sinon
  - Faire avancer le robot

Étape 3 Écrire un programme

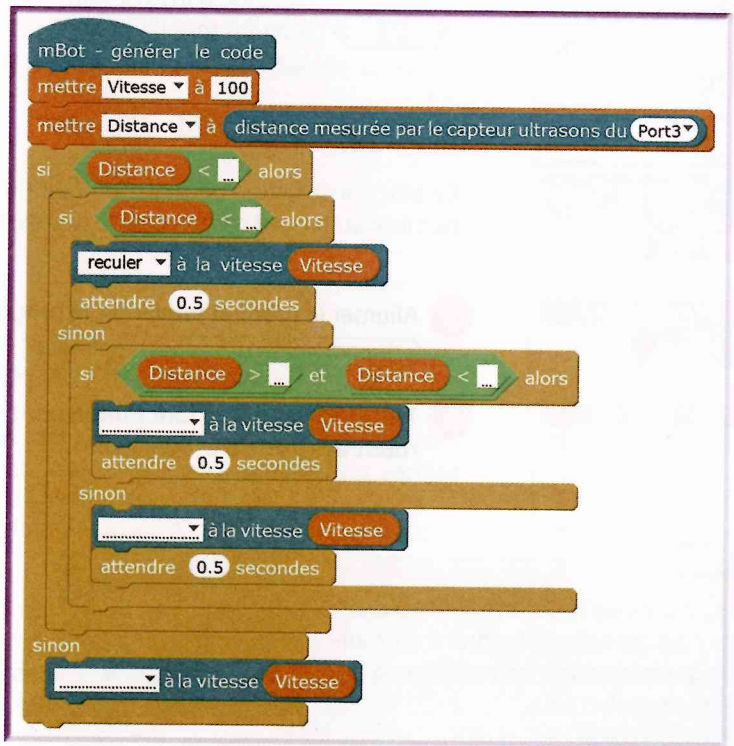
40'

Phase 1 : Détecter un obstacle et l'éviter

• INFORMATIONS •

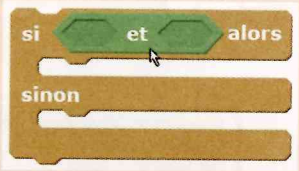
Le robot se déplace à la vitesse « 100 » (vitesse par défaut donnée par le constructeur) pendant une demi-seconde.
Le capteur ultrason est relié à un des quatre ports du robot (1, 2, 3 ou 4).

1 En vous aidant de l'algorithme (Étape 2), compléter ci-dessous le programme pour que le robot mBot puisse détecter et éviter les obstacles.



Aide

Programmation par blocs



Structure alternative qui permet de tester une condition. La condition permet de comparer deux valeurs ou de faire des opérations logiques (et, ou, non).

Ce bloc d'instruction permet de faire avancer le robot mBot à une vitesse déterminée (valeur par défaut 100).

2 Lancer le logiciel **mBlock**. Ouvrir le fichier **Surveillance 1**. lienmini.fr/a152-surveillance1

3 Sélectionner le lutin « **Robot mBot** » et cliquer sur l'onglet **Scripts**.

4 À l'aide des blocs de la rubrique **Pilotage**, compléter le programme.

5 Vérifier le numéro du port utilisé pour le capteur ultrason.

6 Allumer le robot et implanter le programme **Surveillance 1** (voir procédure dans le document ressource).
lienmini.fr/a152-robot-implanter

7 Tester votre programme en utilisant un obstacle. Que constatez-vous ?

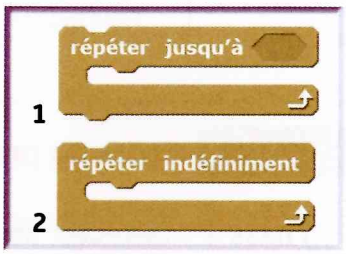
.....

.....

30'

Étape 4 Mettre au point et exécuter un programme

Phase 2 : Se déplacer de manière autonome



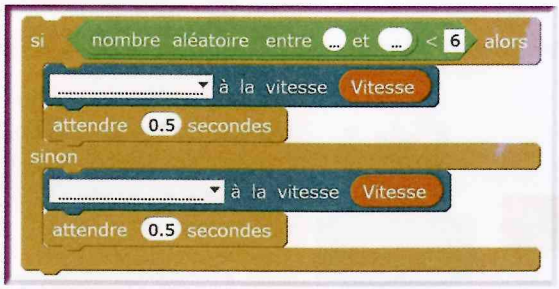
1 Parmi les deux structures répétitives ci-contre, entourer celle qui permettrait au programme de traiter toutes les informations envoyées par le capteur ultrason et ainsi de détecter en permanence tous les obstacles rencontrés.

2 Lancer le logiciel **mBlock**. Ouvrir le fichier **Surveillance 2**. lienmini.fr/a152-surveillance2

3 Introduire dans le programme la structure répétitive « répéter indéfiniment » et une fonction aléatoire pour que le robot puisse se déplacer de manière autonome et aléatoire.

• INFORMATIONS •

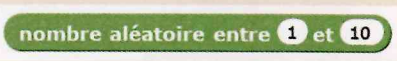
Lorsque le robot a fini de reculer, on souhaite qu'il tourne au hasard à droite ou à gauche. Pour cela on réalise un tirage aléatoire d'un nombre compris entre 1 et 10. Si ce nombre aléatoire est inférieur à 6, alors le robot tourne à droite, sinon il tourne à gauche.



Aide

Programmation par blocs

Ce bloc d'instruction conditionnelle permet de comparer deux valeurs.



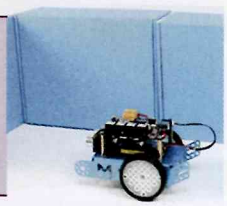
Ce bloc d'instruction permet d'obtenir un nombre au hasard compris entre deux valeurs.

4 Allumer le robot et implanter le programme.
lienmini.fr/a152-robot-implanter

5 Tester et valider le fonctionnement du robot avec votre professeur.

Aller plus loin

Lorsque le robot mBot détecte un obstacle, il doit pouvoir se placer perpendiculairement à celui-ci. Comment programmer son orientation à 90° lorsqu'il rencontre un obstacle ?



• BILAN •

L'essentiel

« Répéter jusqu'à »

→ La **structure répétitive** « **Répéter jusqu'à** » permet de répéter plusieurs fois des instructions jusqu'à ce qu'une **condition** soit vérifiée. On parle aussi de « **boucle** ». Lorsque la condition **est vérifiée**, le programme sort de la boucle.

« Si Alors (Si alors Sinon (Si...)] Sinon »

→ La **structure alternative imbriquée** « **Si Alors Sinon (Si Alors Sinon (Si...)]** » permet de tester une variable en fonction de plusieurs valeurs. Les différents tests sont introduits dans un **ordre précis**.

Aide mémoire

« Répéter jusqu'à »

Algorithmique

Créer un compteur

Initialiser la variable Seconde à 1

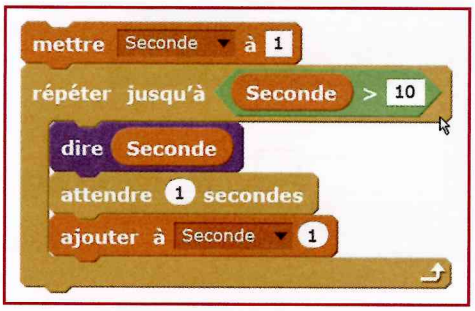
Répéter jusqu'à ce que la variable Seconde soit strictement supérieure à 10

| Afficher la variable Seconde

| Attendre 1 seconde

| Ajouter + 1 à la variable Seconde

Programmation



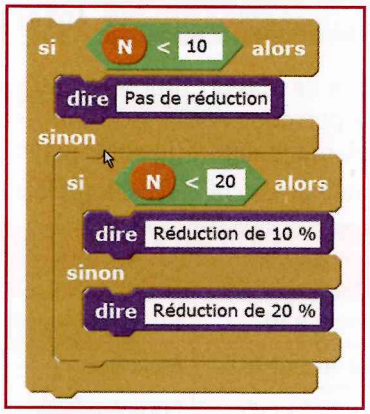
« Si Alors (Si Alors Sinon (Si...)] Sinon »

Algorithmique

Déterminer le taux de réduction (%)

Si le nombre N d'articles est < 10 Alors
| Afficher Pas de réduction
Sinon
Si le nombre N d'articles est < 20 Alors
| Afficher Réduction de 10 %
Sinon
| Afficher Réduction de 20 %

Programmation



QCM Entourer la bonne réponse.

1. La structure répétitive « Répéter jusqu'à » permet de :
a. répéter un traitement tant que la condition est vérifiée.
b. programmer un évènement.
c. répéter un traitement jusqu'à ce que la condition soit vérifiée.

2. Le programme ci-dessus permet :
a. de répéter le mot « Seconde » 10 fois.
b. d'afficher pendant 10 secondes les nombres 1 à 10.
c. de répéter l'affichage des secondes supérieures à 10.

3. La structure alternative imbriquée « Si Alors Sinon [Si alors Sinon [Si...]] » permet :
a. de répéter *n* fois une séquence d'instructions.
b. d'affecter une valeur à une variable.
c. de tester une variable en fonction de plusieurs valeurs.

4. Dans le programme ci-dessus, le bloc d'instruction « Si Alors Sinon » imbriqué permet de déduire que :
a. la réduction est de 20 % si N est supérieur ou égal à 20.
b. la réduction est de 20 % si N est strictement inférieur à 20.
c. la réduction est de 10 % si N est strictement inférieur à 10.

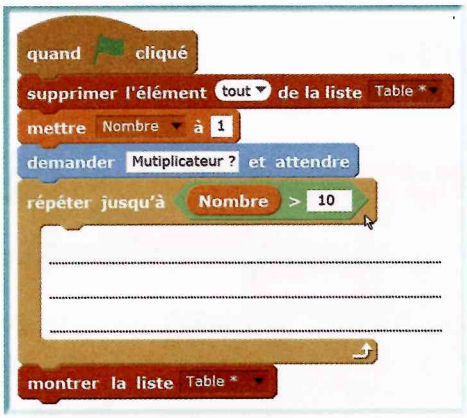
ENTRAÎNEMENT AU BREVET

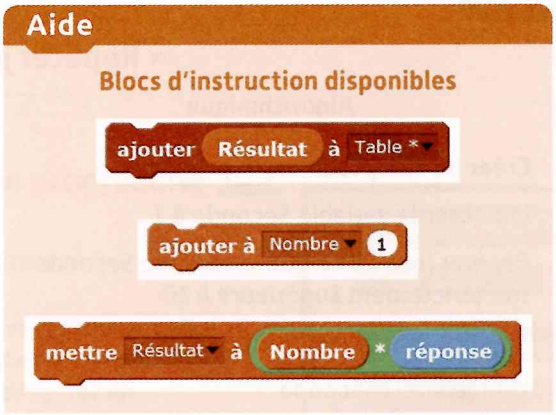
Mathématiques Sujet 1 Afficher les résultats d'une table de multiplication

Dans le cadre du développement d'un jeu éducatif pour les écoles, on vous demande de compléter le programme suivant pour afficher les 10 premiers résultats d'une table de multiplication sous la forme d'une liste.

1. **Repérer les deux variables utilisées dans ce programme.**

2. **Compléter à l'aide des blocs d'instruction qui suivent, la structure répétitive « Répéter jusqu'à » (boucle) pour que le programme affiche, pour un multiplicateur saisi, les résultats de la table de multiplication (jusqu'au nombre 10).**





3. **Lancer le logiciel Scratch. Ouvrir le fichier Table multiplication. Compléter et tester le programme.** lienmini.fr/a152-table-multiplication

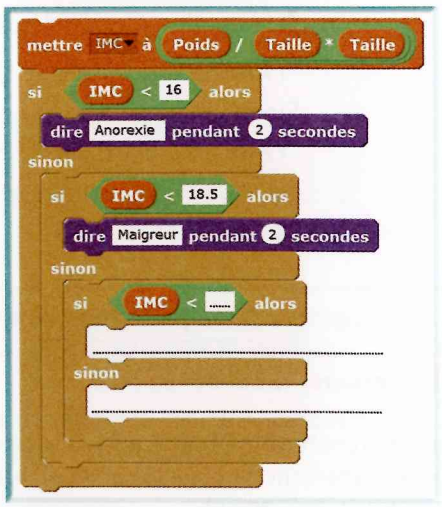
Mathématiques Sujet 2 Interpréter l'Indice de Masse Corporelle

L'Indice de Masse Corporelle (IMC) permet d'avoir une indication sur la morphologie d'une personne adulte (maigreur, surpoids....). L'IMC d'une personne est obtenu en calculant le rapport du poids (kg) sur la taille (m) au carré.

Formule de calcul : Poids (kg)/[Taille (m)]^2.

Valeurs de l'IMC (extrait)	Interprétation selon l'Organisation mondiale de la santé (OMS)
Inférieur à 16	Anorexie ou dénutrition
Entre 16,5 et 18,5	Maigreur
Entre 18,5 et 25	Corpulence classique
Entre 25 et 30	Surpoids

Par exemple, une personne d'un poids de 65 kg et d'une taille de 1,70 m a un IMC de 65/[1,7]^2 ~ 22,5. Elle est considérée de corpulence classique.



1. **Repérer les trois variables utilisées dans ce programme.**

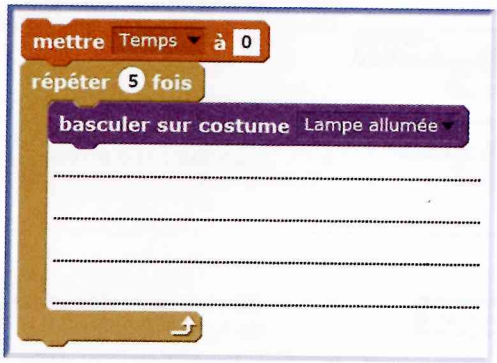
2. **Compléter la structure alternative « Si Alors Sinon » imbriquée pour que le programme affiche l'interprétation de toutes les valeurs de l'indice de masse corporelle calculé.**

3. **Lancer le logiciel Scratch. Ouvrir le fichier IMC. Compléter et tester le programme.** lienmini.fr/a152-imc

Technologie Sujet 1 — Temporiser et signaler la fermeture d'un portail automatique

Avant la fermeture d'un portail coulissant, une lampe clignote pendant dix secondes pour alerter les piétons.

1 **Compléter à l'aide des blocs d'instruction disponibles la structure répétitive « Répéter n fois » pour que le programme simule le clignotement d'une lampe pendant 10 secondes et l'affichage du temps.**



Aide

Blocs d'instruction disponibles

2 **Lancer le logiciel Scratch. Ouvrir le fichier Portail fermeture. Compléter et tester le programme.**
lienmini.fr/a152-portail-fermeture

Technologie Sujet 2 — Détecter un obstacle et l'éviter

Le robot mBot se déplace à une vitesse « V » grâce à un capteur ultrasons :
– si le capteur ultrasons détecte un obstacle à moins de 10 cm, le robot recule pendant 0,5 seconde ;
– si le capteur ultrasons détecte un obstacle entre 10 et 20 cm, le robot tourne à gauche ;
– si le capteur ultrasons détecte un obstacle à plus de 20 cm, le robot continue à avancer.
Parmi les deux programmes qui suivent, un seul permet au robot de se déplacer de manière autonome.

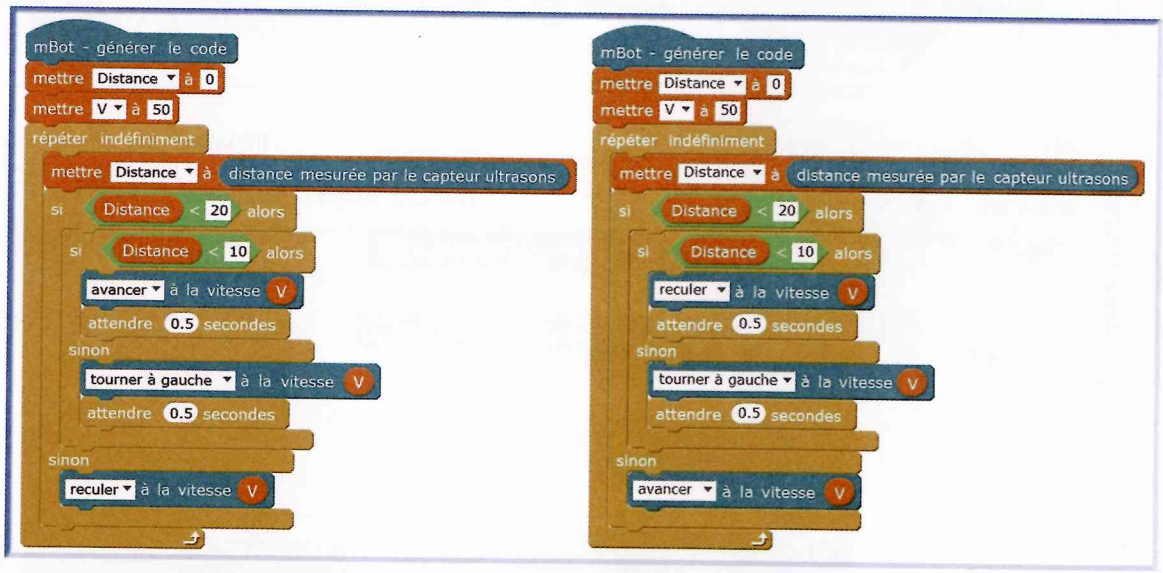
1 **Préciser à quelle valeur est initialisée la variable « V » qui paramètre la vitesse.** .....

2 **Déterminer lequel des deux programmes est faux et justifier votre réponse.**

.....

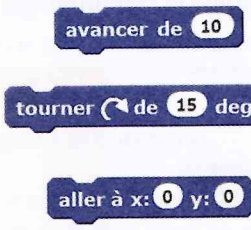
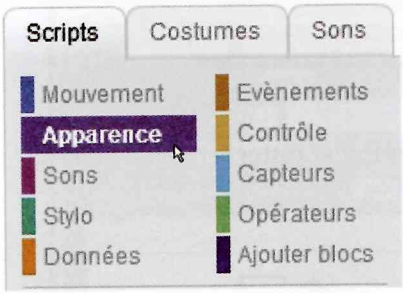
.....

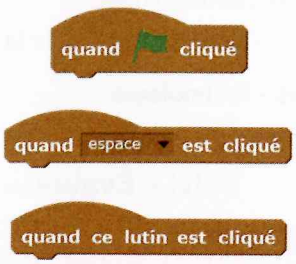
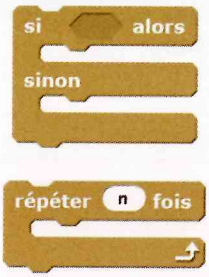
.....



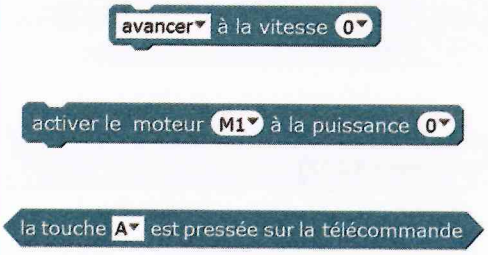
3 **Lancer le logiciel Scratch. Ouvrir, corriger et tester le programme Obstacle.** lienmini.fr/a152-obstacle

GUIDE DE PROGRAMMATION SCRATCH

	Rubrique	Blocs d'instruction	
Mouvement			Avancer d'un nombre de pas (pixel) déterminé à l'écran. Tourner vers la droite d'un angle déterminé. Positionner le lutin à un endroit précis de l'écran (Largeur : 480 pixels – Hauteur : 360 pixels).
Apparence			Afficher un message par le biais du lutin sélectionné. Basculer sur un autre costume du lutin sélectionné. Basculer sur un autre arrière-plan.
Sons			Jouer un son téléchargé. Jouer une note un laps de temps défini. Arrêter tous les sons en cours.
Stylo			Dessiner une image du lutin à l'écran. Laisser une trace à l'écran. Déplacer le stylo sans laisser de trace à l'écran.
Données			Initialiser une variable ou affecter une valeur à une variable. Ajouter une valeur à une variable. Ajouter la valeur d'une variable à une liste.

Rubrique	Blocs d'instruction	
Evènements 		Activer le script lorsque l'utilisateur clique sur le drapeau vert. Activer le script lorsque l'utilisateur appuie sur une touche du clavier. Activer le script lorsque l'utilisateur clique sur le lutin.
Contrôle 		Structure alternative qui permet de tester une condition. La condition permet de comparer deux valeurs ou de faire des opérations logiques. Structure répétitive (boucle) qui permet d'exécuter plusieurs fois une séquence d'instructions (traitement).
Capteurs 		Poser une question. La réponse à cette question étant stockée dans le bloc d'instruction « réponse ». Permet, lorsqu'un lutin touche la couleur sélectionnée, d'ordonner une nouvelle action.
Opérateurs 		Rassembler du texte, des variables. Connaître en fonction de la position choisie, le caractère d'un mot. Obtenir un nombre au hasard compris entre deux valeurs.

mBlock - Extension robotique Scratch

Rubrique	Blocs d'instruction	
Pilotage 		Faire avancer le robot mBot à une vitesse déterminée. Activer un des deux moteurs (M1 et M2) du robot mBot à une puissance déterminée. Permet lorsqu'une touche de la télécommande est activée, d'ordonner une nouvelle action.

GRILLE DE SUIVI ET D'ÉVALUATION

Objectifs de formation

Socle commun de connaissances, de compétences et de culture

Domaine 1 : les langages pour penser et communiquer

« L'élève connaît les principes de base de l'algorithmique et de la conception des programmes informatiques. »

Cycle 4 - Programmes Mathématiques - Technologie

Attendu(s) de fin de cycle : « Écrire, mettre au point et exécuter un programme. »

Suivi – Évaluation

Connaissances et compétences associées	Je sais	Je sais en parler	Je sais faire	Je sais réutiliser
Niveau découverte				
Notions d'algorithme et de programme **(Activité 1)**				
Séquences d'instructions **(Activité 2)**				
Déclenchement d'une action par un évènement **(Projet 1)**				
Niveau 1				
Notion de variable informatique **(Activité 3)**				
Structure alternative Si Alors - Instructions conditionnelles **(Activité 4)**				
Capteur, actionneur, interface **(Projet 6)**				
Niveau 2				
Boucles - Structure « Répéter » **(Activité 5)**				
Structure alternative Si Alors Sinon - Instructions conditionnelles **(Activité 6)**				
Capteur, actionneur, interface **(Projet 9)**				
Niveau 3				
Boucles - Structure « Répéter Jusqu'à » **(Activité 7)**				
Structure alternative Si Alors Sinon imbriquée - Instructions conditionnelles **(Activité 8)**				
Capteur, actionneur, interface **(Projet 12)**				
Décomposer un problème en sous-problèmes afin de structurer un programme. **(Projets 2 – 5 – 8)**				
Écrire, mettre au point (tester, corriger) et exécuter un programme commandant un système réel et vérifier le comportement attendu. **(Projet 3)**				
Programmer des scripts se déroulant en parallèle. **(Projet 4-10)**				
Analyser le comportement attendu d'un système réel et décomposer le problème posé en sous-problèmes afin de structurer un programme de commande. **(Projets 6 – 9 – 12)**				
Écrire un programme dans lequel des actions sont déclenchées par des évènements extérieurs. **(Projets 7 – 11)**				
Attendu de fin de cycle	**Acquis**		**Non acquis**	
Écrire, mettre au point (tester, corriger) et exécuter un programme en réponse à un problème donné. **(Projets 1 à 12)**				

IMPRIM'VERT Votre imprimeur agit pour l'environnement

Achevé d'imprimer en avril 2016 en Italie
par «La Tipografica Varese Srl» Varese
Éditions Delagrave – N° éditeur : 2016-0664 – Dépôt légal : avril 2016

Certifié PEFC
Ce produit est issu de forêts gérées durablement et de sources contrôlées
PEFC
PEFC/18-31-264
www.pefc-france.org

Cahier d'algorithmique et de programmation

Ce cahier couvre le nouvel **enseignement d'informatique** évalué au **brevet des collèges** et dispensé conjointement en **Mathématiques et en Technologie**. Il comprend **20 fiches** réparties sur **4 niveaux** de difficulté croissante.

Il vous permet, à travers **8 activités**, de vous familiariser progressivement aux principes de l'algorithmique. Il vous initie également à la programmation grâce à **12 projets motivants** dans les domaines du jeu, du chiffrement, du calcul et de la robotique (robot **mBot**, système **Arduino**).

En fin d'ouvrage des **exercices d'entraînement** et une **grille d'évaluation** vous aident à suivre votre progression pour une **préparation active au brevet des collèges**.

Les ressources numériques (algorithmes, programmes, etc.) liées au cahier sont fournies au format **Scratch**. Elles sont téléchargeables et accessibles **gratuitement**.

ISBN : 978-2-206-10152-1

9 782206 101521