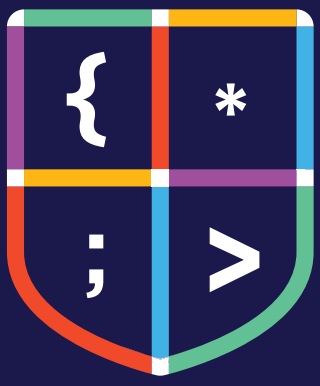


Codewards

Une Heure de Code

En 80 minutes

Manuel pour l'enseignant

Une Heure de Code

80 minutes

Objectifs de la leçon :	1. Découvrir l'approche ludique du produit. 2. Entrer rapidement dans le sujet. 3. Se familiariser avec les concepts fondamentaux de la programmation et leurs applications dans la vie réelle.
Termes employés :	1. Commande informatique 2. Objets 3. Algorithme 4. Optimisation 5. Répétition 6. Boucle
Activité informatique :	Les étudiants doivent résoudre des tâches d'ordre divers. Une nouvelle tâche apparaît dès que la précédente est résolue. Commandes : move, rotate, load, put (avancer, tourner, charger, poser) Arguments : right, left (droite, gauche) Structure: loop…end (boucle...fin) Objets : robot, crane
Matériel nécessaire :	Ordinateurs (tablettes) avec accès au système CODEWARDS.

Plan de la leçon *(Version 1)*

1. Présentation de l'histoire.
2. Résolution des problèmes + explication individuelle des concepts.

Partie 1. Préambule

Temps estimé – 5 minutes

Montrez la vidéo et présentez le sujet.

L'enseignant : *« Aujourd'hui, nous sommes sur une mission de la plus haute importance ! Nous devons rétablir le fonctionnement du système* d'exploitation d'une station *sous-marine. En relevant ce défi, nous allons apprendre à nous servir de commandes informatiques afin de contrôler différents objets. »*

Partie 2. Activité informatique

Temps estimé – 70 minutes

Résoudre les tâches successives :
Tâche n° 1
Tâche n° 2
Tâche n° 3
Tâche n° 4
Tâche n° 5
Tâche n° 6
Tâche n° 7
Tâche n° 8
Tâche n° 9
Tâche n° 10
Tâche n° 11
Tâche n° 12

Pendant que les élèves résolvent les tâches, passez parmi eux et donnez des explications individuelles, si nécessaire.

Partie 3. Conclusion

Temps estimé : 5 minutes

L'enseignant: *« Aujourd'hui, nous avons utilisé du code informatique pour réparer le dôme et le pipeline d'une station sous-marine. Nous avons fait du bon travail, mais il reste encore beaucoup de choses à réparer. J'espère que notre unité se réunira à nouveau pour remplir d'autres missions passionnantes. »*

Plan de la leçon *(Version 2)*

Partie 0. Présentation

Tout le monde se présente et fait connaissance.

Partie 1. Préambule

Temps estimé : 10 minutes

Montrez la vidéo et présentez le sujet.

L'enseignant: « Aujourd'hui, nous sommes sur une mission de la plus haute importance ! Nous devons rétablir le fonctionnement du système d'exploitation d'une station sous-marine. En relevant ce défi, nous allons apprendre à nous servir de commandes informatiques afin de contrôler différents objets. *informatiques afin de contrôler différents objets. »*

Concept Nº1 :

QUI + QUOI + COMMENT

Partie 2. Activité informatique

Estimated time : 55 minutes

L'enseignant: *« Ouvrons le panneau de configuration et regardons comment il fonctionne. »*

- **Préambule à la tâche n°1**

Avant de commencer, présentez le système de commandes.
Expliquez que le moyen le plus sûr pour contrôler des objets est d'utiliser le système de commandes : QUI + QUOI (ce qu'il faut faire) + COMMENT (combien de pas, dans quel sens, etc.)

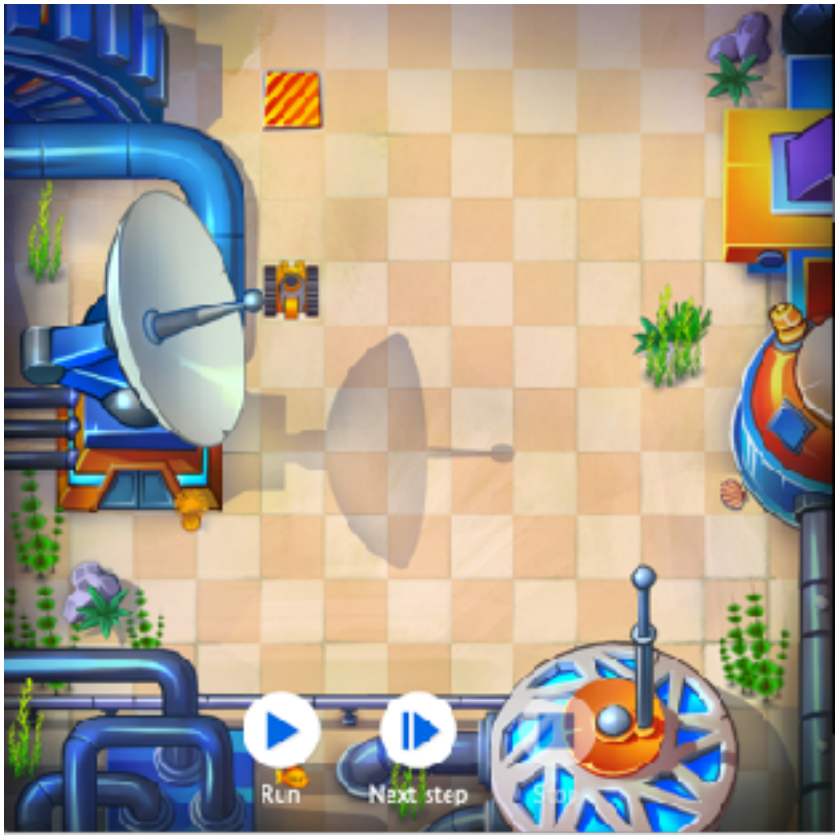
Mettez en pratique grâce au jeu d'action **« Je vous programme »**

Version 1. Les élèves doivent « programmer » l'enseignant en utilisant la méthode QUI + QUOI + COMMENT. Les enfants disent à voix haute les commandes et l'enseignant les exécute *(ex : il tourne à droite)*.

Version 2. Les élèves vont au tableau un par un, une image du robot dans la main.
Les autres enfants « programment le robot », l'enseignant écrit leurs commandes sur le tableau.

L'enseignant : « Essayons de faire la même chose sur ordinateur : il faut résoudre les tâches n°1 et n°2 ».

Tâche n°1



Votre tâche : Allez sur la ligne en surbrillance. Écrivez vous-même le code.

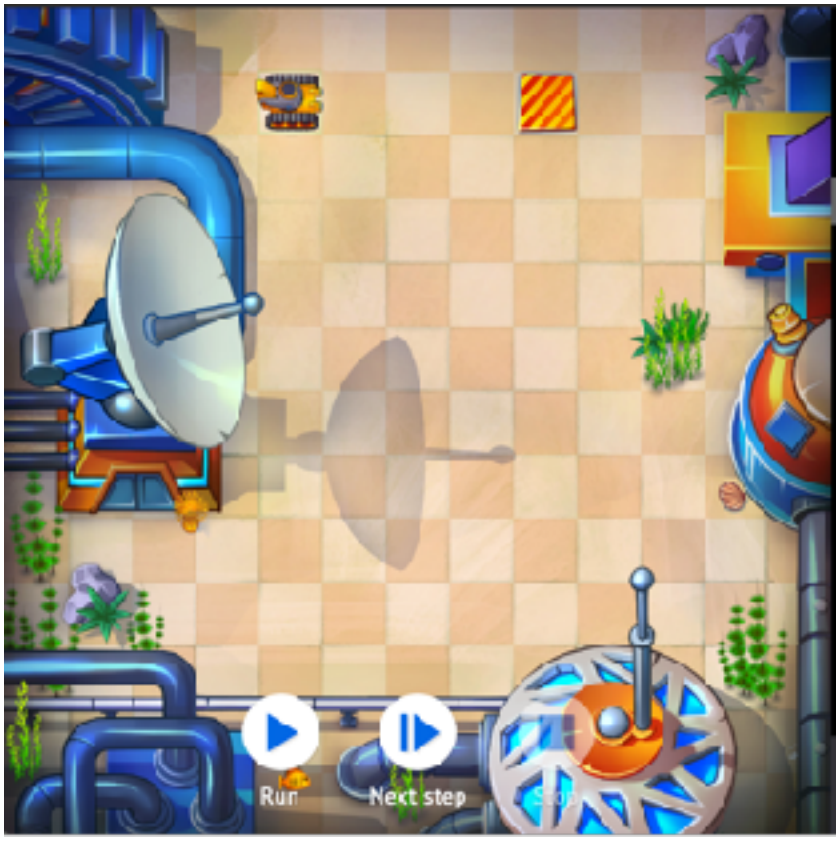
Dans ce problème, le code n'est pas écrit, mais l'élève n'a qu'à se laisser guider pour le compléter. Les apprenants doivent cliquer sur "Démarrer" pour exécuter le programme et voir ce qui se produit !

Pas de code source

Code final :

```
1.  robot.move 3
```

Tâche n°2



Votre tâche : Allez sur la case en surbrillance. Vous savez le faire.

Pour cette tâche, le code n'est pas écrit. Les élèves doivent écrire le code eux-mêmes et démarrer le programme.

Pas de code source

Code final :

```
1.  robot.move 4
```

Trucs et astuces

- Si un apprenant n'arrive pas à accéder à la tâche n°2, montrez-lui le bouton de navigation : 
- Si un apprenant n'arrive pas à calculer la distance, donnez comme exemple des pas humains ou des cases d'un jeu d'échecs.

Une fois terminé, montrez sur votre écran le résultat de la tâche n°4. N'hésitez pas à faire une erreur volontairement dans le code. La ligne comportant l'erreur va se mettre en surbrillance : c'est l'occasion d'attirer l'attention des enfants sur le fait que nous ne sommes pas des robots et faisons souvent des erreurs.

Dans le monde informatique les erreurs nous aident à comprendre ce qui doit être corrigé pour atteindre notre but. Montrez que le système Codewards affiche en surbrillance la ligne pour signaler l'erreur.

L'enseignant : *« Quand l'objet que vous contrôlez ne connaît qu'une commande, c'est simple. Mais certains mécanismes sont plus complexes. Par exemple, un lave-linge peut laver un pull en laine grâce à un programme et lavera un jean avec un autre programme. Un four à micro-ondes* réchauffe *la nourriture, mais peut aussi la décongeler, etc. »*

Si le mécanisme peut effectuer des actions multiples, quelle « partie » du système de commandes aura plus d'une option : « qui », « quoi » ou « comment » ?

La plupart des enfants répondront, à juste titre, « quoi ». Félicitez-les et faites-leur remarquer qu'à présent, il y a deux « zones variables » :

1. *« comment » (ils l'ont vu dans l'exercice précédent, quand le nombre de pas du robot pouvait être différent) ;*
2. *« quoi » (si le robot peut effectuer des actions différentes).*

Combien d'actions le robot doit-il effectuer pour atteindre son objectif ?

Le robot doit effectuer 3 actions, comprenant deux commandes différentes, exécutées successivement.

Auparavant, nous utilisions une commande pour atteindre notre objectif.

Quand on écrit des commandes dans une séquence conduisant le mécanisme à son but, cette séquence est appelée un **algorithme** : autrement dit, une séquence des commandes effectuées l'une après l'autre.

- **Préambule à la tâche n°3**

Nous allons apprendre une nouvelle commande – **rotate**

> rotate – tourner. Cette commande va de pair avec la direction : tourner vers où ? A Gauche (left) ou à droite (right).
>
> Ainsi, la commande complète sera :
>
> ```
> robot.rotate left
> robot.rotate right
> ```

Si vous voulez savoir vers où diriger le robot, il faut d'abord déterminer où se trouve la tête du robot, puis se mettre à sa place pour comprendre où tourner.

Ainsi, avant d'écrire un programme, il faut imaginer comment le robot va parvenir à sa destination. Ensuite on divise ce processus en une séquence claire de commandes, que le robot connaît. Enfin on écrit ces étapes dans notre langage codé.

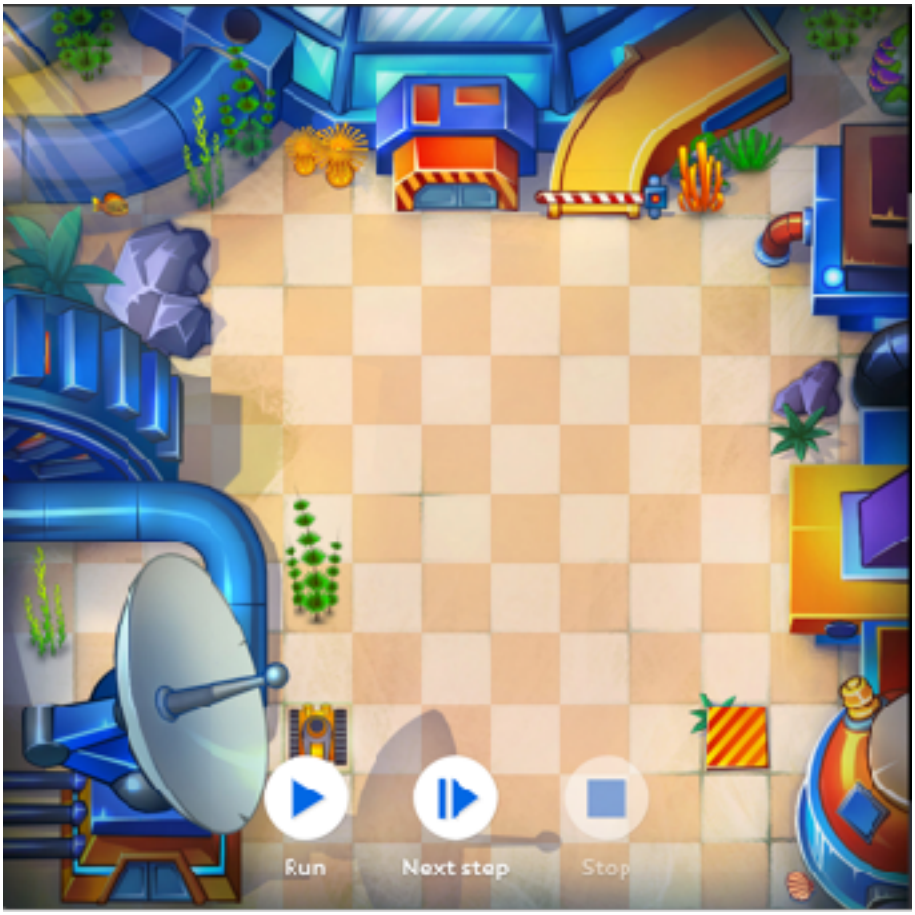
C'est fondamentalement ce que font tous les développeurs : ils écrivent des algorithmes spécifiques (des séquences des commandes) pour que les robots exécutent ce qu'ils ont planifié.

Compléter les tâches n°3, n°4 et n°5.

Afin d'aider les élèves à résoudre les tâches plus vite, répétez-leur ces questions :

•Combien d'actions doit exécuter le robot ?
•Quelle commande correspond à cette action ?
•Comment la commande doit-elle être modifiée ?

Tâche n° 3



Votre tâche : Allez sur la case en surbrillance. Écrivez le code vous-mêmes.

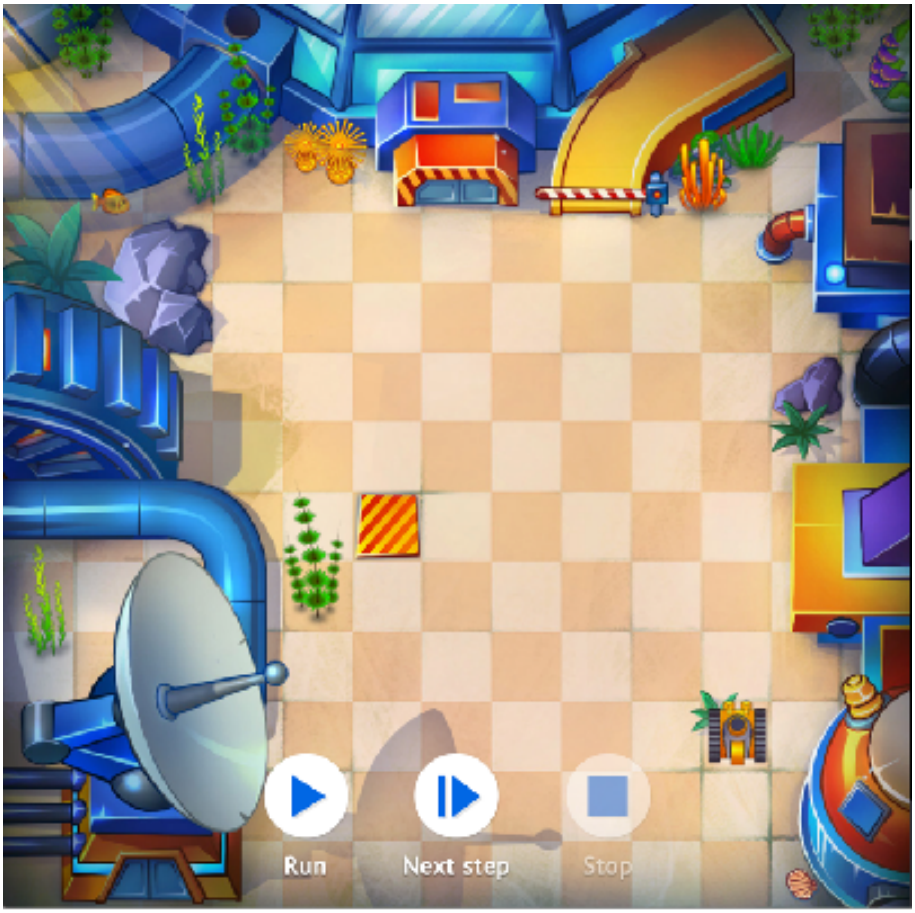
Pour cette tâche, le code n'est pas écrit. Les élèves doivent écrire le code eux-mêmes et démarrer le programme.

Pas de code source

Code final :

```
robot.rotate right
robot.move 6
```

Tâche n° 4



Votre tâche : Allez sur la case en surbrillance. Écrivez le code vous-mêmes.

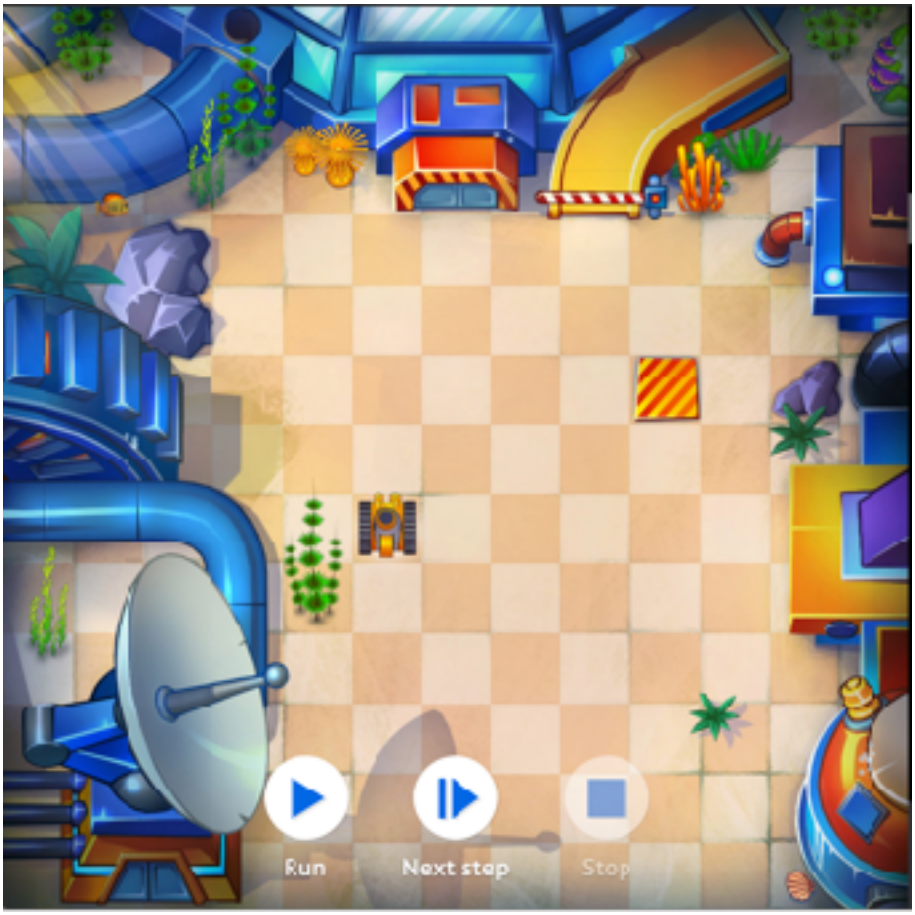
Pour cette tâche, le code n'est pas écrit. Les élèves doivent écrire le code eux-mêmes et démarrer le programme.

Pas de code source

Code final :

```
1.  robot.move 3
2.  robot.rotate left
3.  robot.move 5
```

Tâche n°5



Votre tâche : Allez sur la case en surbrillance. C'est parti !

Pour cette tâche, le code n'est pas écrit. Les élèves doivent écrire le code eux-mêmes et démarrer le programme.

Pas de code source

Code final :

```
1.  robot.move 2
2.  robot.rotate right
3.  robot.move 4
```

Trucs et astuces

- Si un élève a écrit quelque chose d'incorrect et ne sait pas comment l'effacer, montrez-lui comment utiliser la touche Retour arrière avec ou sans sélection du texte.
- Si un élève a écrit un algorithme différent qui fonctionne et a reçu moins de 3 étoiles pour la résolution de la tâche, dites-lui de ne pas s'inquiéter, vous reviendrez sur ce point un peu plus tard. Il est probable que l'algorithme contiennent plus d'étapes que nécessaire.
- Si un élève démarre le programme chaque fois qu'il a écrit une ligne de code, le système Codewards affichera « Essaie encore une fois ». Expliquez qu'avant de démarrer le programme, il faut avoir écrit la séquence de commandes entière et ensuite seulement l'exécuter. C'est la compétence clé que nous allons voir maintenant.

- **Découverte de la tâche n°6**

Posez les questions suivantes :

- Est-ce que le robot atteint son but ? Oui
- De combien de cases le robot avance-t-il ? 5
- Combien y a-t-il de commandes dans l'algorithme ? 5
- Est-il possible de réécrire le programme afin de réduire le nombre de lignes ou le nombre de cases que le robot parcourt ? Oui

Expliquez que si nous voulons que les robots travaillent efficacement, nous devons écrire des programmes optimaux, qui économisent leur temps et leurs autres ressources.

Exécutez les tâches n°6 et n°7

Tâche n°6

Votre tâche : Lancez le programme pour réparer les fissures du dôme !

Pour cette tâche, le code est déjà écrit. Les élèves doivent cliquer sur « Démarrer ».
Précisez que pour réparer la fissure, le robot doit aller dessus. La fissure qui doit être colmatée est celle qui clignote.

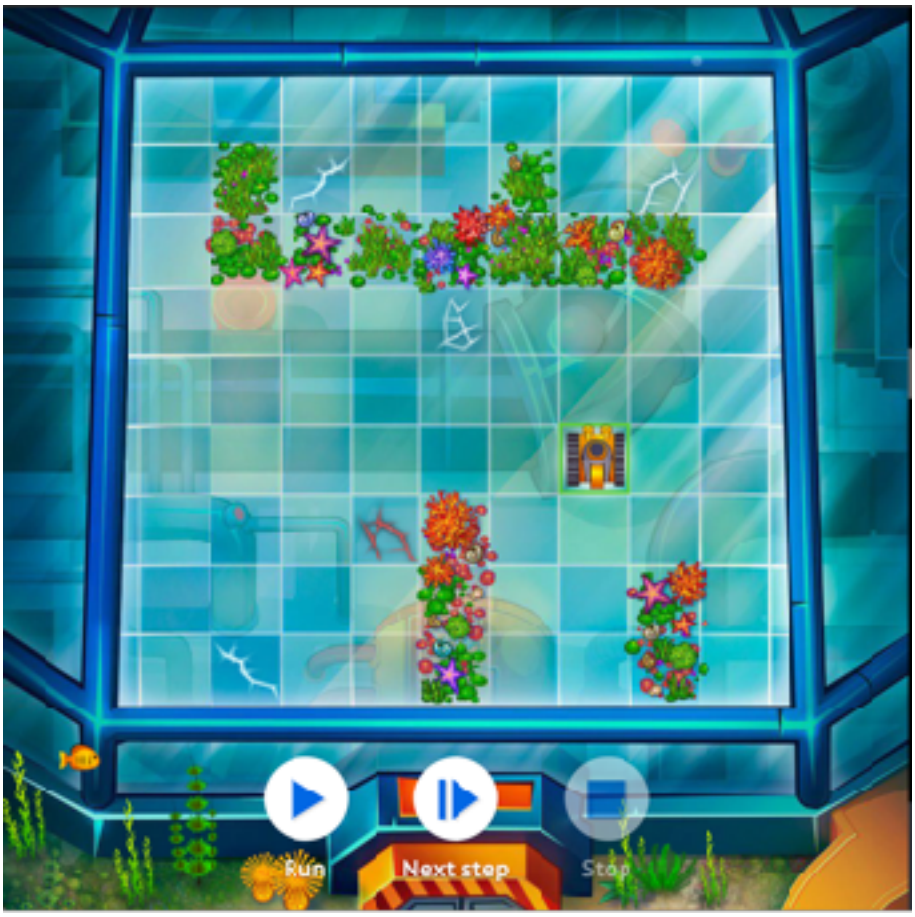
Code source :

```
1.  robot.move 3
2.  robot.rotate left
3.  robot.move 2
```

Code final :

```
1.  robot.move 3
2.  robot.rotate left
3.  robot.move 2
```

Tâche n° 7



Votre tâche : Réparez les fissures. Écrivez le code manquant.

Pour cette tâche, le code est incomplet. Les élèves doivent compléter les lignes et démarrer le programme. La fissure qui doit être colmatée est celle qui clignote.

Code source:

```
1.  ___________
2.  robot.move 3
3.  ___________
4.  robot.move 1
```

Code final :

```
1.  robot.rotate left
2.  robot.move 3
3.  robot.rotate left
4.  robot.move 1
```

L'enseignant : *« Revenons sur ce que nous avons découvert aujourd'hui ».*

-Les mécanismes et les robots sont contrôlés par des commandes selon la structure suivante : QUI + QUOI + COMMENT

-Certains mécanismes ne peuvent réaliser qu'une commande, mais la plupart d'entre eux sont multi-tâches.

-La séquence de commandes qui permet d'atteindre un certain objectif est appelé un **algorithme**.

-Un algorithme peut être **optimal** ou **non-optimal**. Un algorithme non-optimal prend plus de ressources. Il ne faut écrire que des algorithmes optimaux afin d'économiser du temps et de l'énergie.

L'enseignant : **«** *Il reste u*ne dernière tâche importante à résoudre : allons-*y ! »*

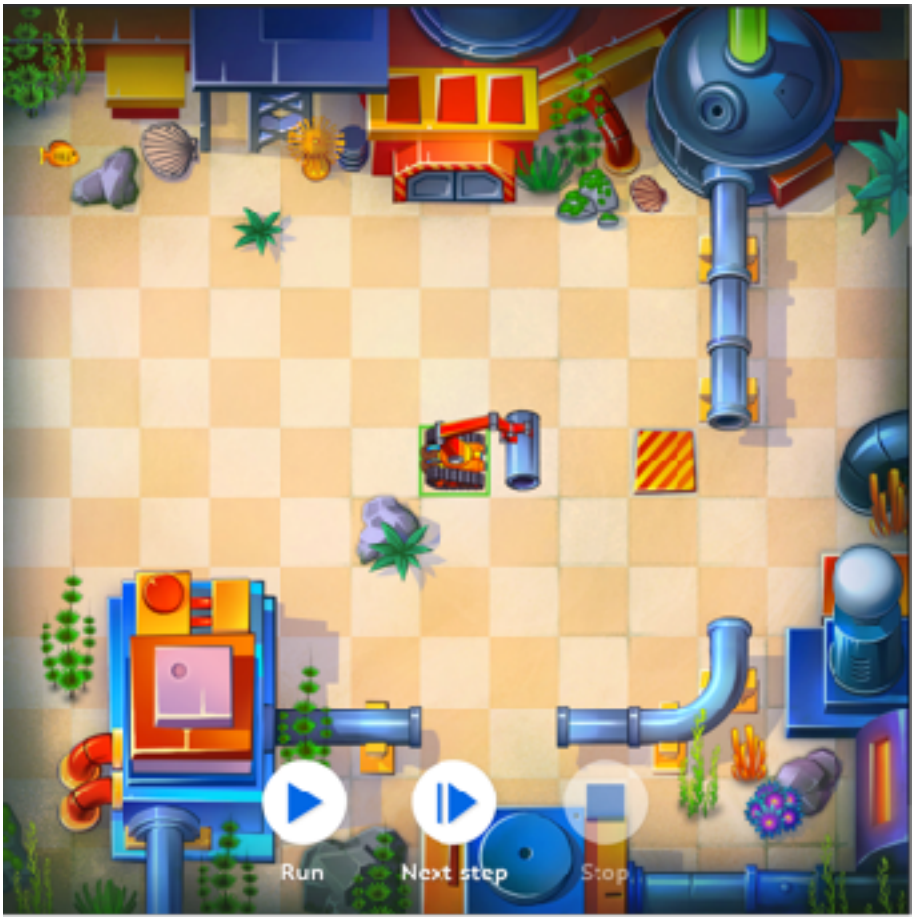
Ce qui change :

1.Ce robot peut réaliser plusieurs commandes.

2.Nouvelles commandes : load pour charger, prendre le tuyau ; put pour poser, mettre le tuyau.

3.Il peut y avoir plusieurs solutions, aux élèves de trouver la solution optimale.

Tâche n° 8



Votre tâche : *Prenez la grue (crane) pour charger le bout de tuyau et placez-le dans la partie du pipeline la plus proche. Ecrivez vous-mêmes le code.*

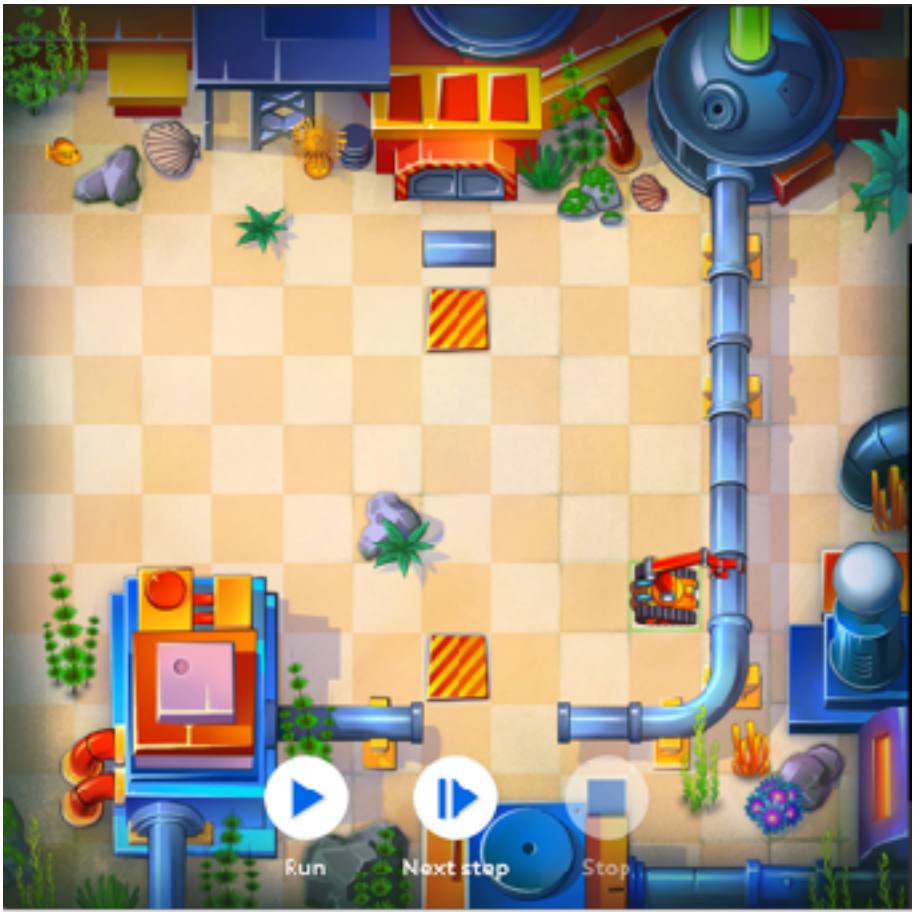
Pour cette tâche, le code n'est pas écrit. Les élèves doivent écrire le code eux-mêmes et démarrer le programme.

Pas de code source

Code final :

```
1.  crane.load
2.  crane.move 3
3.  crane.put
```

Tâche n° 9



Votre tâche : Prenez le tube et positionnez-le dans l'espace laissé par le tronçon manquant de pipeline. Puis chargez la pièce à l'aide de la grue et conduisez la grue à la case clignotante. Écrivez votre code.

Pour cette tâche, le code n'est pas écrit. Les élèves doivent écrire le code eux-mêmes et démarrer le programme.

Pas de code source

Code final :

```
1.  crane.rotate right
2.  crane.rotate right
3.  crane.move 3
4.  crane.rotate right
5.  crane.move 4
6.  crane.load
7.  crane.rotate right
8.  crane.rotate right
9.  crane.move 5
10. crane.put
```

L'enseignant : *«Avez-vous remarqué que certaines choses dans nos vies se répètent ? Pensons à des choses qui arrivent périodiquement. A quoi pensez-vous ? »*
Exemple: les saisons, les vacances, etc.

Prenons par exemple la chanson d'anniversaire :

Joyeux anniversaire,
Joyeux anniversaire,
Joyeux anniversaire,
Joyeux anniversaire.

Cochez les lignes qui se répètent.
L'enseignant : « *N'est-il pas ennuyeux de copier des lignes identiques ?* »

Elles sont parfaitement identiques et se répètent trois fois. Ce ne serait pas génial d'avoir un outil qui permette de répéter les choses identiques ?

Cet outil existe bel et bien, il s'appelle la ***boucle***. Pour comprendre comment il fonctionne, jouons !

L'enseignant devient un objet et doit faire un carré en répétant deux commandes quatre fois.

teacher.move4
teacher.turn right

Ceci doit se produire quatre fois.

L'enseignant : *« Exécutons le programme » :*

L'enseignant : *« La boucle est en marche. Combien de fois répète-t-on les commandes ? »*
Les élèves : *« Quatre »*
L'enseignant : *« Que dois-je faire ? »*
Les élèves : *« Faire quatre pas »*
L'enseignant : *« Autre chose ? »*
Les élèves : *« Tourner à droite »*

L'enseignant fait un carré.

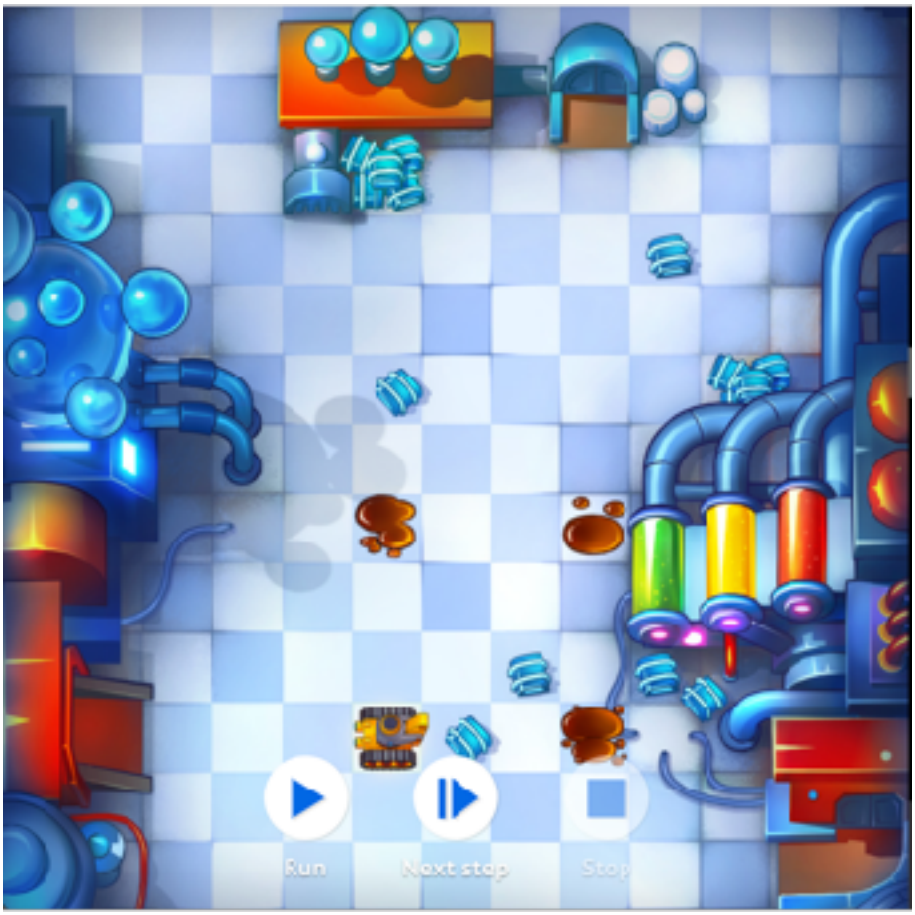
- **Préambule à la tâche n°10.**

L'enseignant : *« Voyons à quoi ressemble le code avec une boucle et comment le robot l'exécute. »*

Soyez attentif au fait que « la commande de fin » n'est pas déclenchée tant que le programme n'a pas effectué le nombre indiqué de «loop». Le programme déclenchera la dernière commande après l'achèvement de la boucle.

Il faudra expliquer que si nous voulons que notre programme distingue des instructions répétitives, il faudra placer ces commandes au sein d'une boucle.

Tâche n°10



Votre tâche : Le robot effectue parfois les mêmes actions en continu. On utilise des boucles (loops) pour éviter d'écrire des instructions répétitives. Nous allons voir comment cela fonctionne. Le code est déjà écrit, démarrez le programme.

Pour cette tâche, le code est déjà écrit. Les élèves cliquent sur « Démarrer » pour exécuter le programme.

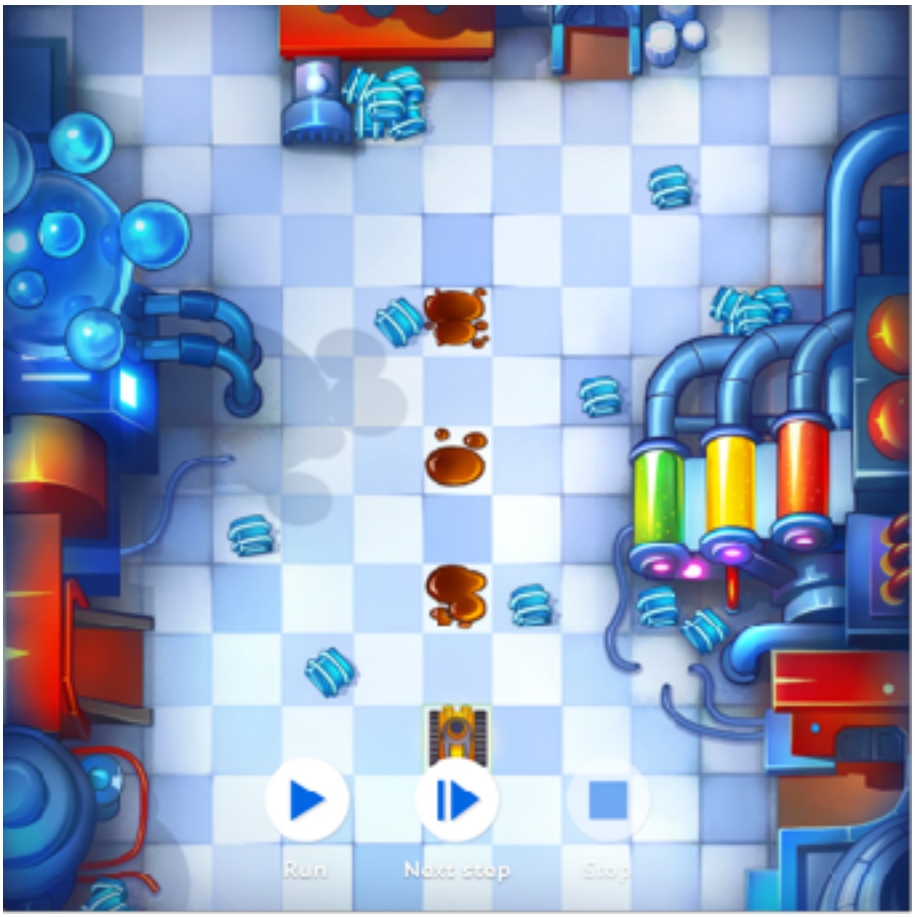
Code source :

```
1.  loop 3
2.     robot.rotate right
3.     robot.move 3
4.  end
```

Code final :

```
1.  loop 3
2.     robot.rotate right
3.     robot.move 3
4.  end
```

Tâche n°11



Votre tâche : Nettoyez toutes les taches de pétrole. Ecrivez le programme en utilisant une boucle (loop).

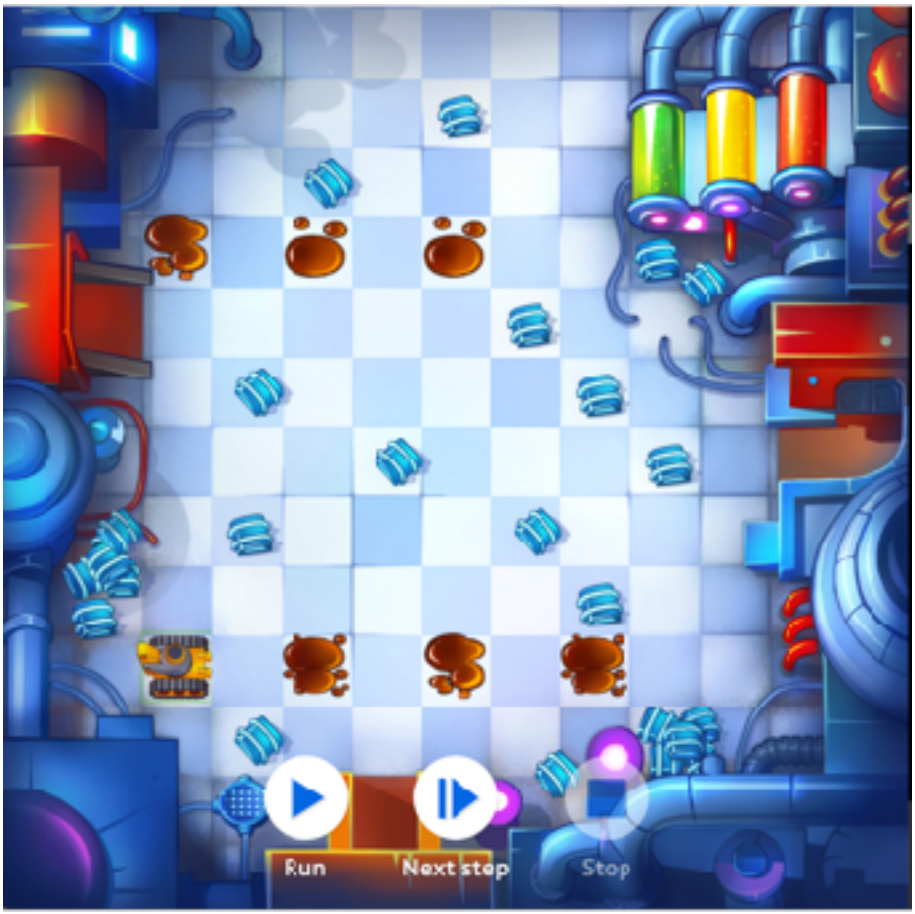
Pour cette tâche, le code n'est pas écrit. Les élèves doivent écrire le code eux-mêmes et démarrer le programme.

Pas de code source

Code final :

```
loop 3
   robot.move 2
end
```

Tâche n°12



Votre tâche : Nettoyez toutes les taches de pétrole. Écrivez vous-mêmes le code.

Pour cette tâche, le code n'est pas écrit. Les élèves doivent écrire le code eux-mêmes et démarrer le programme.

Pas de code source

Code final :

```
loop 3
   robot.rotate left
   robot.move 6
   robot.rotate right
   robot.rotate right
   robot.move 6
   robot.rotate left
   robot.move 2
end
```

Partie 3. Conclusion

Temps estimé : 5 minutes

L'enseignant : *« Aujourd'hui, grâce à notre codage informatique, nous avons réussi à réparer le dôme et le pipeline d'une station sous-marine. Nous avons fait du bon travail, mais il reste encore beaucoup de choses à réparer. J'espère que notre unité de sauvetage pourra se réunir à nouveau pour accomplir d'autres missions passionnantes. »*