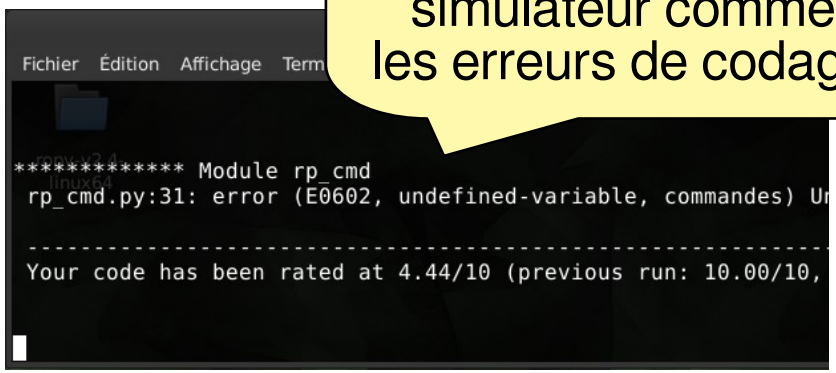
Introduction à la programmation avec Ropy

https://forge.aeif.fr/blender-edutech/ropy

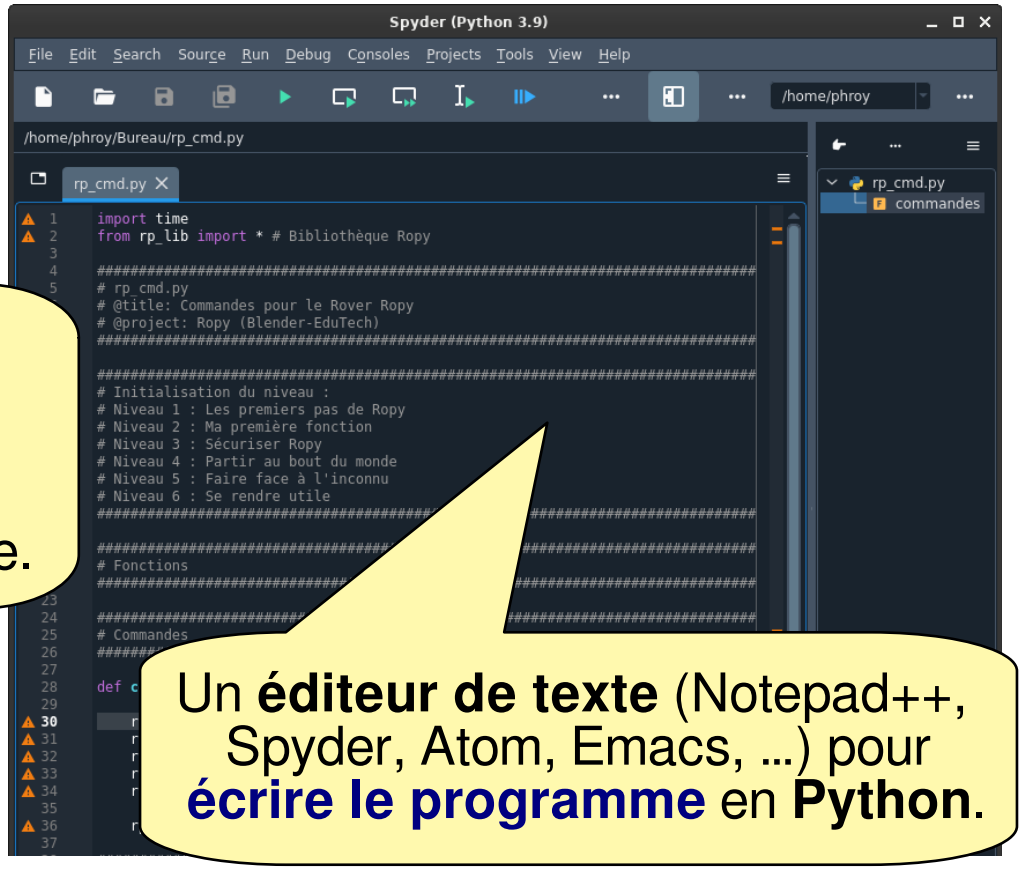
Présentation de Ropy et de son environnement de programmation

Ropy est un rover martien qui se commande grâce au langage **Python**. L'interface de programmation se décompose en **3 fenêtres** : un éditeur de texte, le simulateur et la console.

Le **simulateur** permet de **visualiser l'évolution du Rover**.



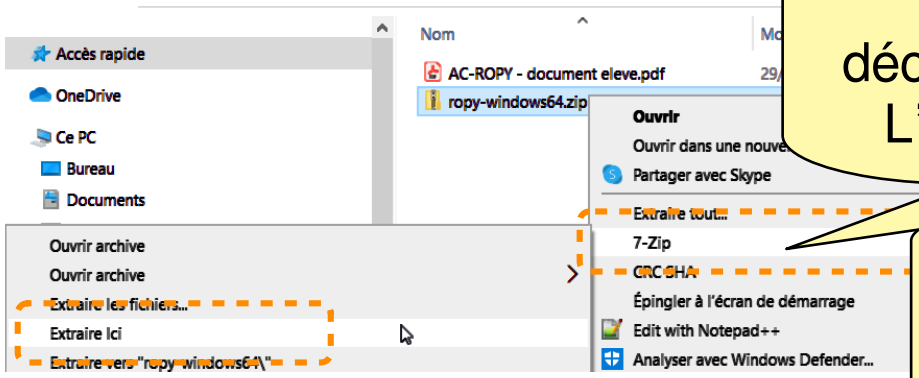
La **console** pour **visualiser les informations** du simulateur comme les erreurs de codage.



Un **éditeur de texte** (Notepad++, Spyder, Atom, Emacs, ...) pour **écrire le programme** en **Python**.

Mettre en place l'environnement de dé

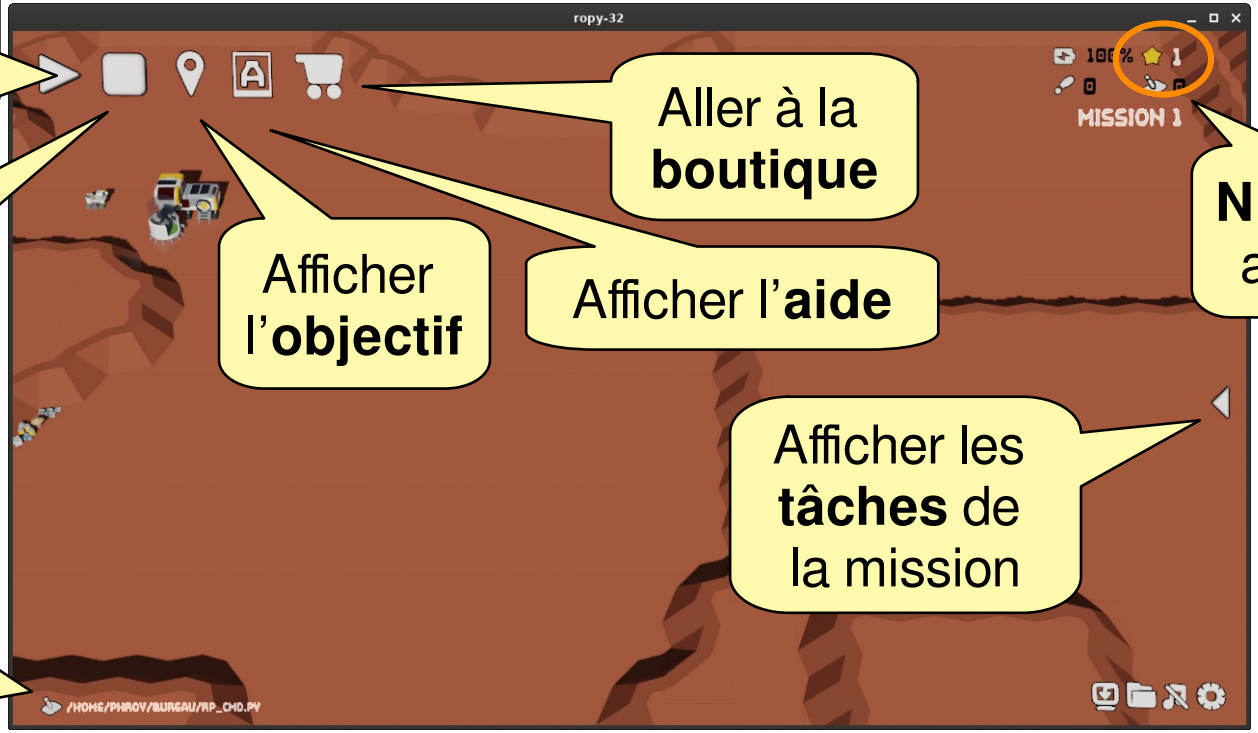
1 : Récupérer l'archive **ropy-windows64.zip** et la décompresser avec **7-Zip** sur le **bureau**. L'extraction va créer le répertoire ropy



Le **simulateur** et la **console** se lancent en même temps avec **ropy.bat** (situé dans le répertoire créé).

Exécuter le programme

Arrêter et **réinitialiser**



Aller à la **boutique**

Afficher l'**objectif**

Afficher l'**aide**

Niveau actuel

Afficher les **tâches** de la mission

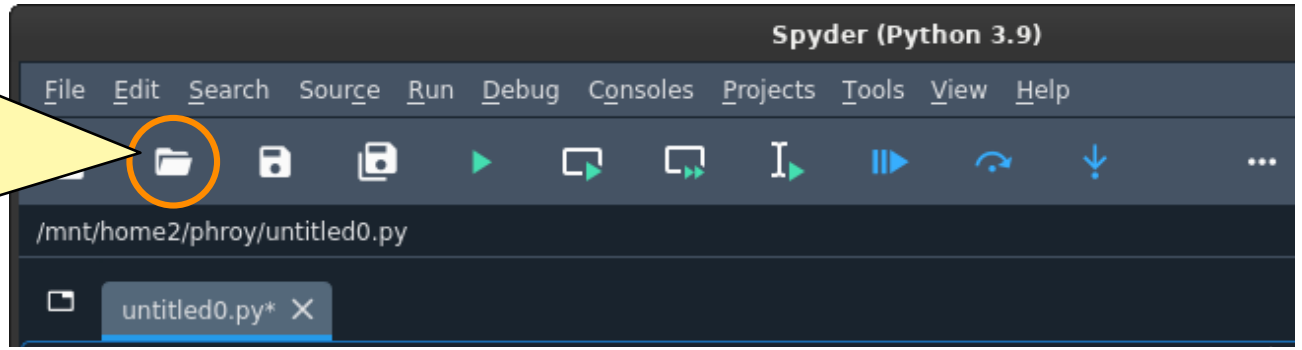
Fichier de commandes

Mettre en place l'environnement de développement

2 : Copier **dans votre répertoire** le fichier de commandes : **ropy_cmd.py** (ropy commandes).

3 : Lancer **Spyder**.

4 : Dans **Spyder** **ouvrir** le fichier de commandes qui a été précédemment copié dans votre répertoire.



5 : Dans le **simulateur**, **définir** votre fichier comme fichier de commandes.

6 : Le nom de votre fichier doit apparaître ici.

Contenu du fichier rp_cmd.py

Le fichier **rp_cmd.py** comporte 4 sections.

```
import bge # Bibliothèque Blender Game Engine (UPBGE)
import time
from rp_lib import * # Bibliothèque Ropy

##############################################################################
# rp_cmd.py
# @title: Commandes pour le Rover Ropy
# @project: Ropy (Blender-EduTech)
##############################################################################

##############################################################################
# Initialisation du niveau :
# Niveau 1 : Les premiers pas de Ropy
# Niveau 2 : Ma première fonction
# Niveau 3 : Sécuriser Ropy
# Niveau 4 : Partir au bout du monde
# Niveau 5 : Faire face à l'inconnu
# Niveau 6 : Se rendre utile
##############################################################################

##############################################################################
# Fonctions
##############################################################################



##############################################################################
# Commandes
##############################################################################

def commandes():

    rp_gauche()
    rp_avancer()
    rp_avancer()
    rp_avancer()
    rp_avancer()

    rp_fin() # A garder


##############################################################################
# En: Externals calls << DONT CHANGE THIS SECTION >>
# Fr: Appels externes << NE PAS MODIFIER CETTE SECTION >>
##############################################################################

if __name__=='start':
    thread_cmd_start(commandes)
if __name__=='stop':
    thread_cmd_stop()
```

Le code doit être indenté (décalé sur la droite) avec la touche Tab

Import des bibliothèques
Ne pas modifier cette section

Fonctions : section pour le codage de **vos fonctions**

Commandes : section pour le codage des commandes du robot

La commande `rp_fin()` est à conserver.

Appels du simulateur
(Blender Game Engine)
Ne pas modifier cette section

Mission 1 – Les premiers pas de Ropy

Instruction et structure linéaire

Objectif 1 : Il faut aider **Ropy** à sortir du son emplacement et atteindre la case à l'est de la station. Afin de visualiser le trajet, il faudra marquer les cases.

Vous avez à disposition plusieurs **commandes élémentaires** pour diriger **Ropy** :

- Avancer : `rp_avancer()`
- Tourner à gauche : `rp_gauche()`
- Tourner à droite : `rp_droite()`
- Marquer la case : `rp_marquer()`

Le « `rp_` » dans le nom des fonctions permet d'identifier les fonctions de **Ropy**.

```
##############################################
# Commandes
##############################################

_____________________________________________

_____________________________________________

_____________________________________________
```

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

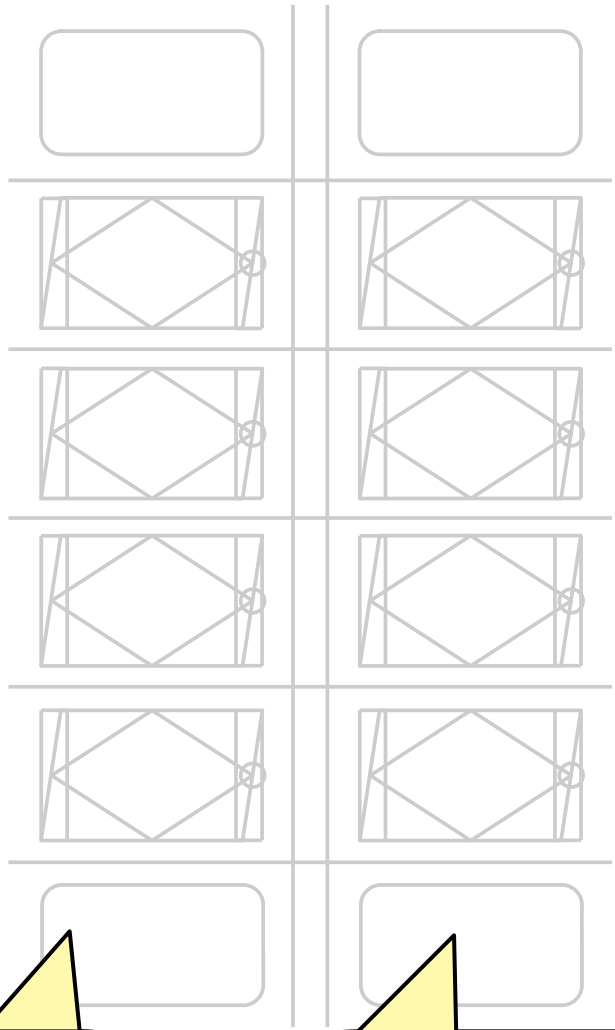
_____

Mission 2 – Ma première fonction

Création d'une fonction

Objectif 2 : Créer la fonction `mrp_avancer()` regroupant avancer et marquer.

```
##########################################
# Fonctions
##########################################

_____________________________________________

_____________________________________________

_____________________________________________

_____________________________________________

_____________________________________________

_____________________________________________

_____________________________________________


##########################################
# Commandes
##########################################

_____________________________________________

_____________________________________________
```



Représenter que le premier appel

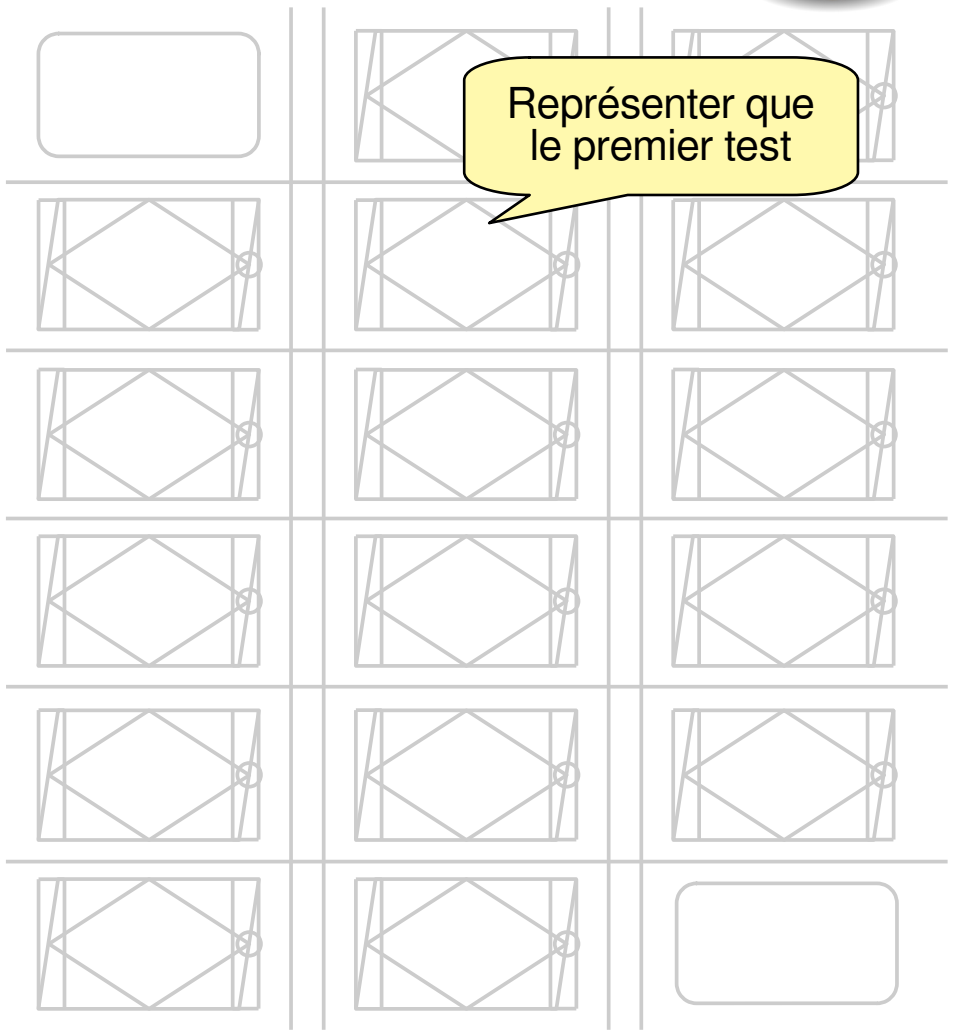
Représenter la fonction

Mission 3 – Apprendre le danger

Structure conditionnelle (si, alors, sinon)

Objectif 3.1 : Aller à la mission 3, provoquer une collision avec un obstacle en avançant et observer ce qui se passe. Il vous faut donc sécuriser l'avance du robot.

```
##############################################
# Commandes
##############################################

_____________________________________________

_____________________________________________

_____________________________________________

_____________________________________________

_____________________________________________

_____________________________________________

_____________________________________________
```



La fonction pour **détecter un mur** est : `rp_detect()`. La fonction retourne `True` si il a un mur et `False` si il n'y a pas de mur.

Mission 3 – Apprendre le danger

Structure conditionnelle (si, alors, sinon)

Objectif 3.2 : Intégrer le test de sécurisation dans votre fonction `mrp_avancer()`.

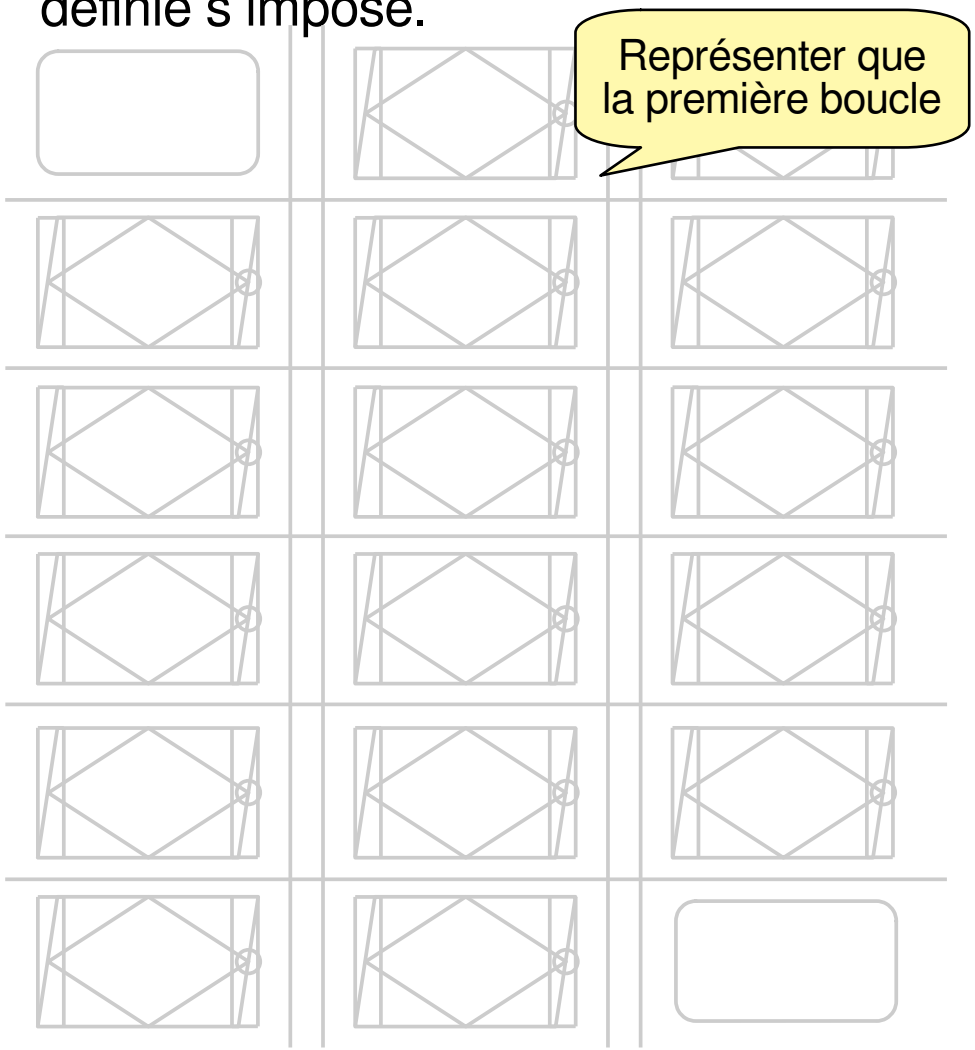
```
##############################################
# Fonctions
##############################################
```

```
##############################################
# Commandes
##############################################
```

Mission 4 – Partir au bout du monde

Structure itérative - boucle définie

Objectif 4.1 : Aller à la mission 4, **Ropy** est maintenant prêt pour l'aventure et donc atteindre une case éloignée. Pour un tel voyage, l'utilisation d'une boucle définie s'impose.




```
##############################################
# Commandes
##############################################
```

Mission 4 – Partir au bout du monde

Passage d'argument (dans une fonction)

Objectif 4.2 : Afin d'enrichir notre bibliothèque, nous allons créer une fonction pour avancer d'un nombre de pas : **`mrp_avancer_nbpas(pas)`**.

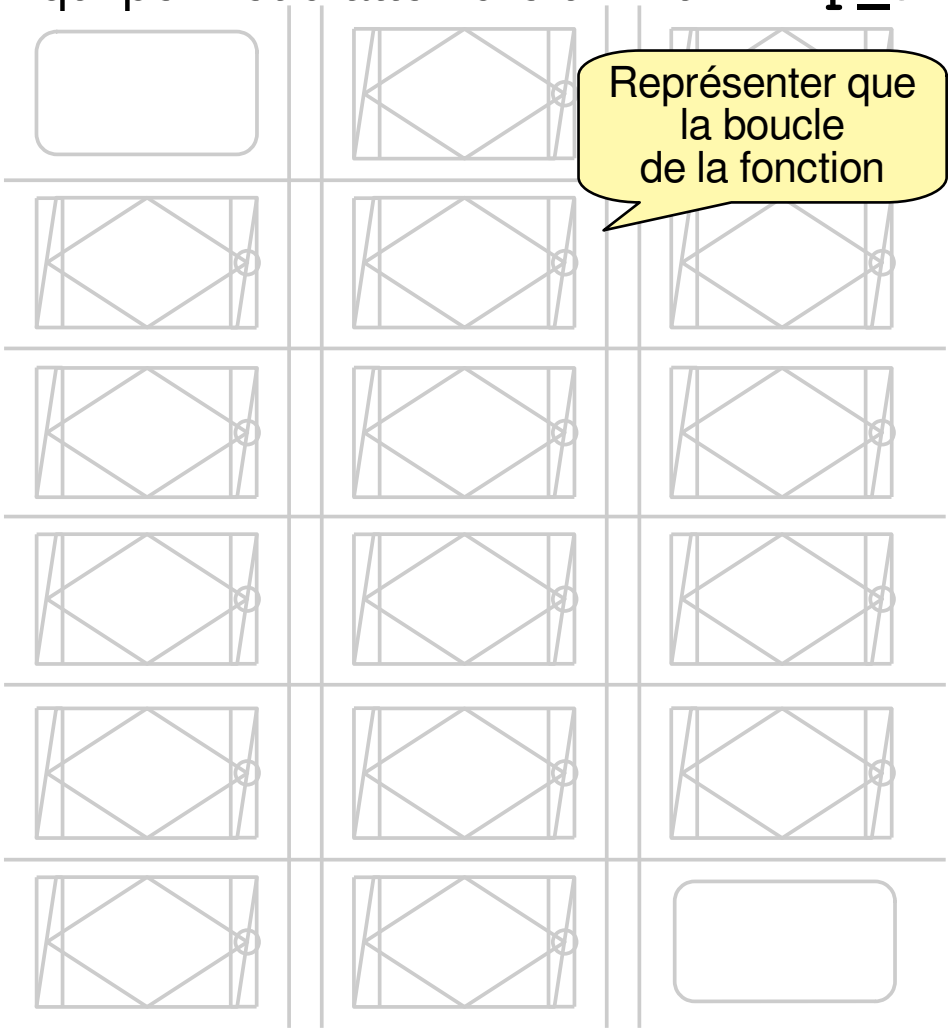
```
##############################################
# Fonctions
##############################################
```

```
##############################################
# Commandes
##############################################
```

Mission 5 – Faire face à l'inconnu

Structure itérative - boucle indéfinie (tant que)

Objectif 5 : Aller à la mission 5, **Ropy** doit toujours atteindre la même case, mais son lieu de départ change à chaque fois. Nous allons créer une fonction qui permet d'atteindre un mur : **`mrp_avancer_mur()`**.




```
##############################################
# Fonctions
##############################################
```

Mission 6 – Se rendre utile

Objectif 6.1 : Afin d'analyser la roche sur une zone, **Ropy** doit y prélever des carottes. Pour effectuer les forages, **Ropy** doit passer sur toutes les cases.

```
##############################################
# Fonctions
##############################################
```

```
##############################################
# Commandes
##############################################
```

Mission 6 – Se rendre utile

```
##############################################
# Fonctions
##############################################
```

```
##############################################
# Commandes
##############################################
```

Mission 6 – Se rendre utile … certes, mais avec classe !

Objectif 6.2 : **Ropy** est devenu esthète. C'est le même objectif, mais il faut parcourir le terrain en **colimaçon**.

```
##############################################
# Fonctions
##############################################
```

```
##############################################
# Commandes
##############################################
```

Mission 6 – Se rendre utile … certes, mais avec classe !

```
##############################################
# Fonctions
##############################################
```

```
##############################################
# Commandes
##############################################
```