Introduction à la programmation avec Ropy

LA FORGE
des communs numériques éducatifs

CC BY SA

Philippe Roy <philippe.roy@ac-grenoble.fr>

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
##############################################

_____________________________________________

_____________________________________________

_____________________________________________
```

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

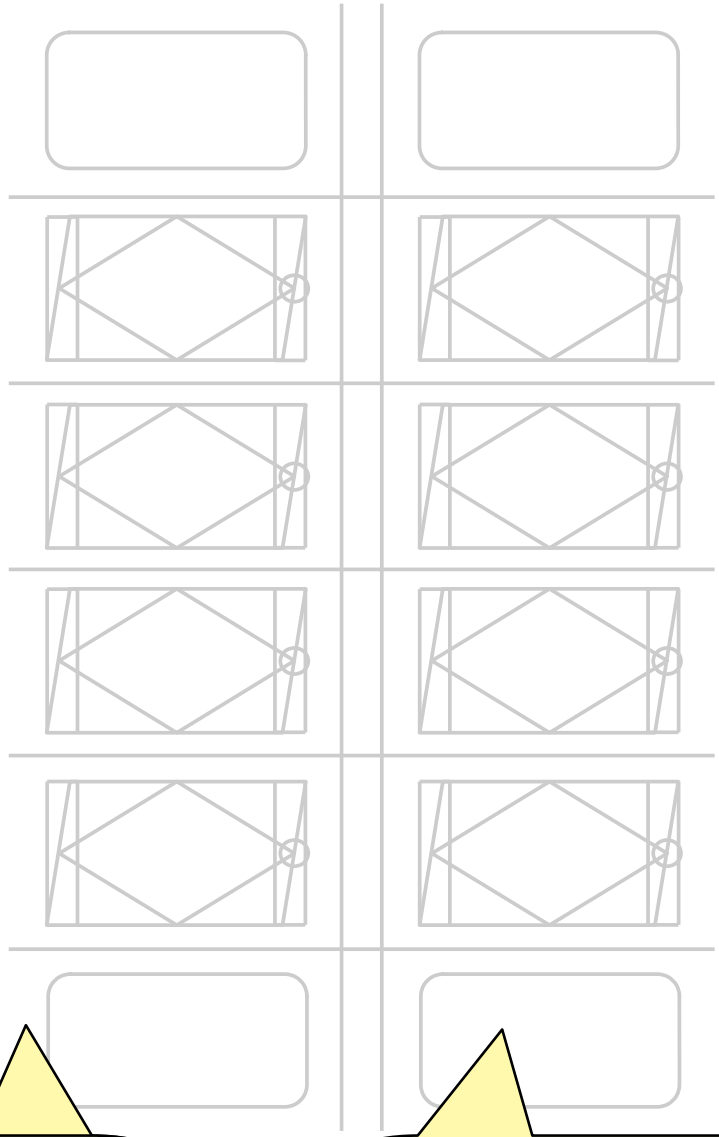
_____

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

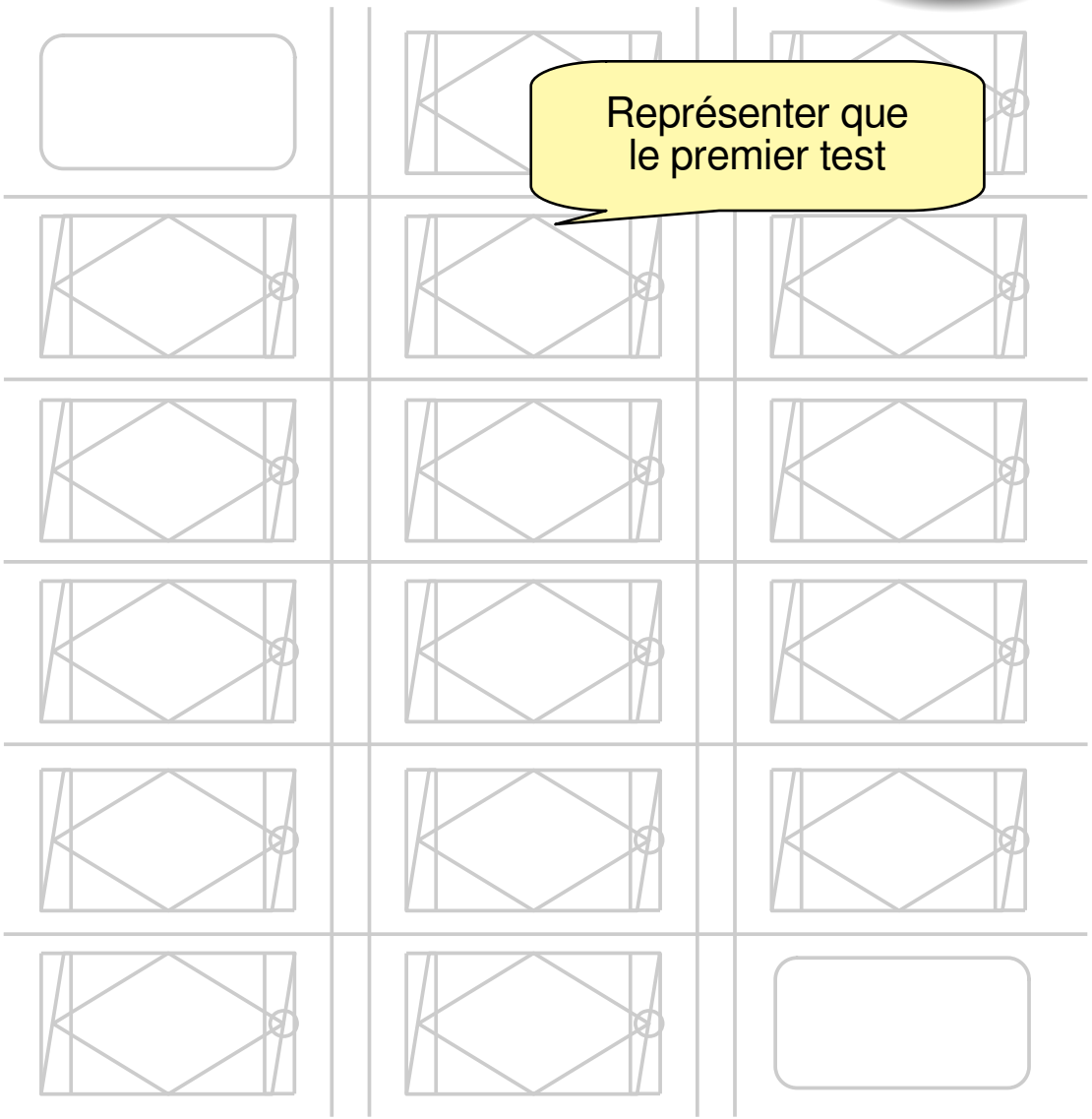
Représenter la fonction

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()`**.

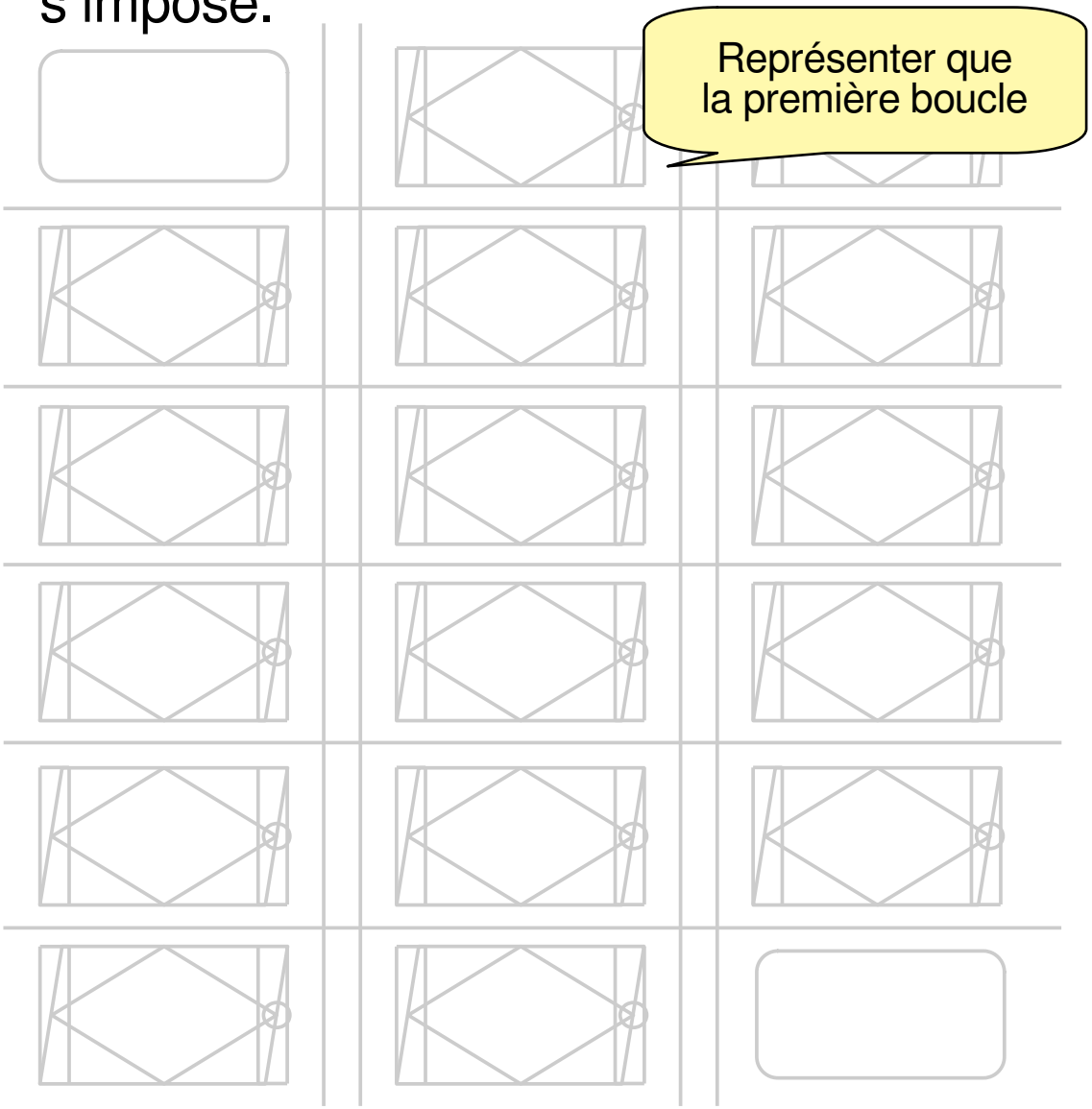
```
##############################################
# Fonctions
##############################################
```

```
##############################################
# 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)`**.

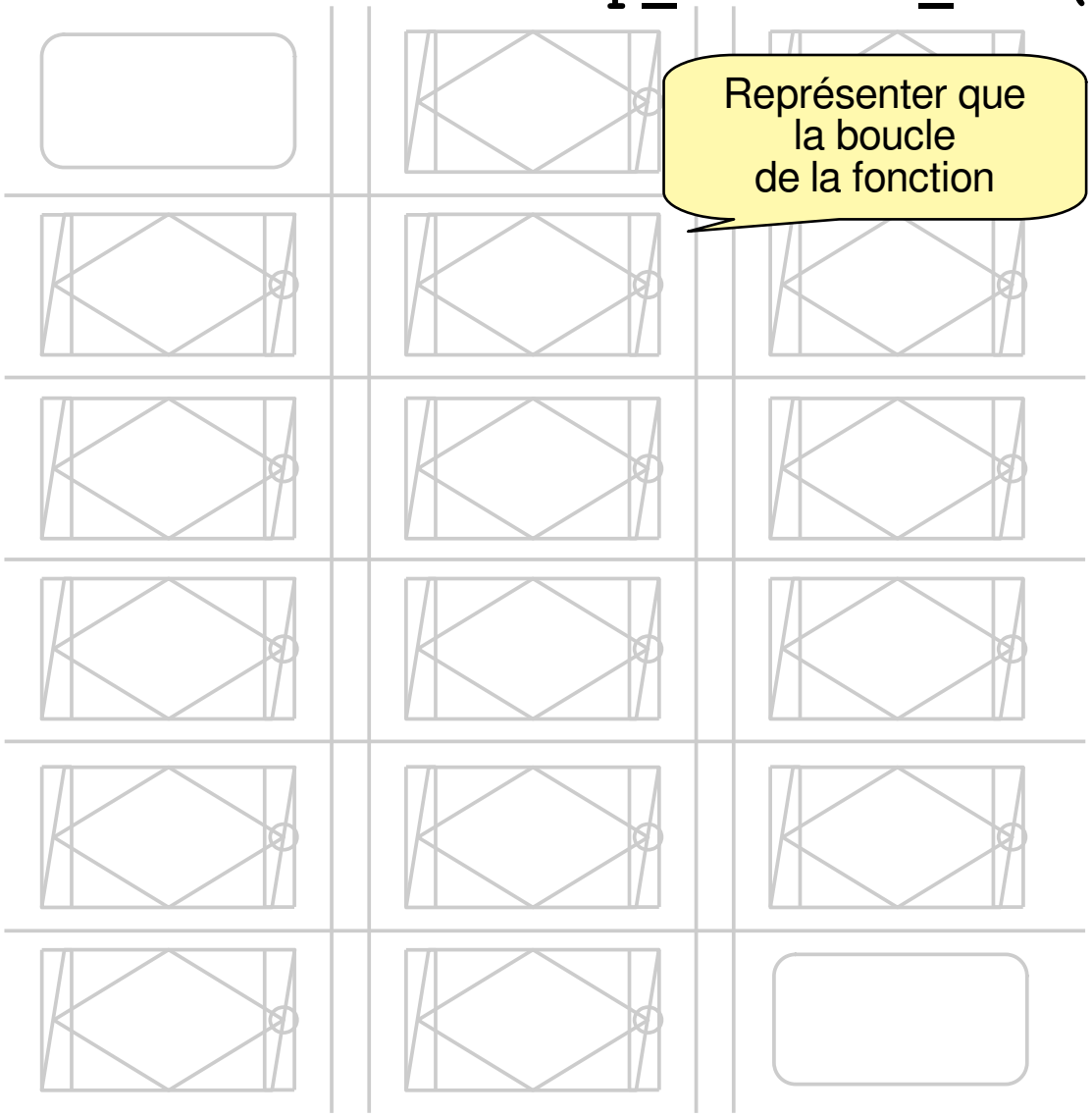
```
##############################################
# Fonctions
##############################################
```

```
##############################################
# 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
##############################################
```