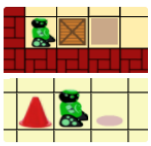
SNT 2de	Activités	*Le web*	Date : … / … / ……
A210	Concours Algoréa		NOM : Prénom : Classe :

https://concours.castor-informatique.fr/?team=prep2018

Se préparer / Commencer une préparation

Catégorie blanche →	• séquences d'instructions • appels de fonctions simples • boucles répéter simples
Catégorie jaune →	• instructions conditionnelles • boucles répéter imbriquées • imbrication de boucles et d'instructions conditionnelles
Catégorie orange →	• variables • opérateurs arithmétiques et booléens • boucles "tant que"
Catégorie verte →	• création de fonctions • tableaux, listes et chaînes de caractères **Participation individuelle uniquement (pas de binômes)**

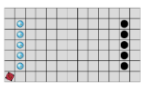
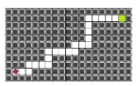
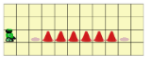
Blockly →		**Ce que nous vous conseillons pour cette activité.**
Scratch →		Si vous avez l'habitude de Scratch. Attention : ne fonctionne bien qu'avec les navigateurs Google Chrome ou Mozilla Firefox récents.
Python →		Si vous maîtrisez bien ce langage. Attention : ne fonctionne bien qu'avec les navigateurs Google Chrome ou Mozilla Firefox récents.

Préparation Algoréa Jaune Python, Billes et Parcours →	
Préparation Algoréa Jaune Python, Caisses et Plots →	
Préparation Algoréa Jaune Python 2017 →	

Retenir numéro : 

Retenir numéro : 

Compléter les étoiles au fur et à mesure :

Ranger les billes 3p  ☆☆☆☆	Suivre le parcours 4p  ☆☆☆☆	Ranger les billes 5p  ☆☆☆☆	Suivre le parcours 5p  ☆☆☆☆
Pousser les caisses 2p  ☆☆☆☆	Déposer des plots 4p  ☆☆☆☆	Pousser les caisses 3p  ☆☆☆☆	Déposer des plots 5p  ☆☆☆☆

Pour reprendre un parcours : https://concours.castor-informatique.fr/

SNT 2de	Activités	*Le web*	Date : … / … / ……
A210	Rappel Python 		NOM : Prénom : Classe :

Programmation en Python Python permet de créer des programmes à partir d'instructions.

Par exemple, l'instruction **droite()** peut faire déplacer un robot d'une case vers la droite.

Un programme formé d'instructions les unes en dessous des autres, exécute ces instructions l'une après l'autre.

```
from robot import *
droite()
haut()
droite()
```

Le programme ci-contre fait déplacer le robot vers la droite, puis vers le haut, puis de nouveau vers la droite.

Boucle de répétition

Pour exécuter plusieurs fois la même instruction, on peut utiliser l'instruction **for loop in range(...)**.

Par exemple, plutôt que de mettre 4 fois la même instruction :

```
droite()
droite()
droite()
droite()
```

On peut écrire la boucle suivante :

```
for loop in range(4):
    droite()
```

On peut aussi mettre plusieurs instructions dans une boucle :

```
for loop in range(5):
    droite()
    haut()
```

Boucles imbriquées

Il est possible d'utiliser des boucles imbriquées, c'est-à-dire que l'on peut mettre des boucles **for**, à l'intérieur d'autres boucles **for**.

```
for loop in range(5):
    droite()
    for loop in range(3):
        haut()
        droite()
```

Ce programme répétera 5 fois un déplacement d'1 case vers la droite, 3 cases vers le haut et 1 case vers la droite.

L'instruction if

Avec l'instruction **if**, on peut exécuter une instruction uniquement dans certaines conditions.

```
if caseMarquee():
    peindre()
```

Par exemple, le programme ci-contre teste le contenu de la case du robot, et ne la peint que si elle est marquée.

On peut aussi placer plusieurs instructions dans une instruction **if**, comme illustré ci-dessous :

```
if caseMarquee() :
    peindre()
    droite()
```

l'instruction if / else

On peut utiliser une instruction **if** / **else**, pour effectuer des opérations différentes selon la situation. Par exemple :

```
if caseMarquee():
    peindre()
else:
    haut()
```

Dans le programme ci-contre, si la case du robot est marquée, le robot la peint, sinon il ne la peint pas mais se déplace vers le haut.

l'instruction while

On peut utiliser une instruction **while**, comme dans l'exemple ci-dessous.

```
while not surTrou():
    droite()
```

L'instruction « while » exécute en boucle les instructions placés en dessous, indentées vers la droite, tant que la condition est vraie. Si la condition est fausse dès le départ, les instructions ne sont jamais exécutées.

Ce type de boucle est utile lorsque l'on ne connaît pas à l'avance le nombre de répétitions. Par exemple dans un cas comme ci-dessous, où l'on ne connaît pas le nombre de cases entre la bille et le trou où il faut la déposer.

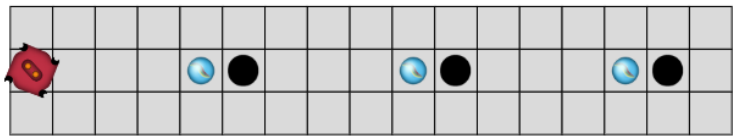
https://amazon.quick-pi.org/

v6pwysb3

Ranger les billes 3P

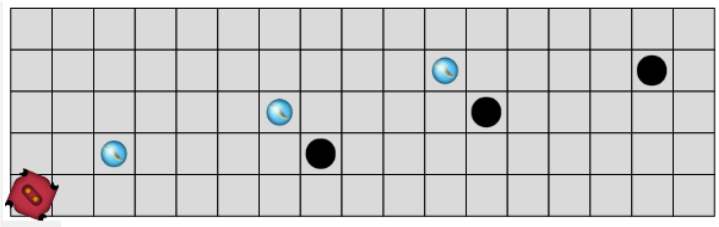
Fonctions disponibles : droite(), gauche(), haut(), ramasserBille(), deposerBille().
Mots-clés autorisés : for.

```
from robot import *
for loop in range(3):
    for loop in range(4):
     droite()
    ramasserBille()
    droite()
    deposerBille()
```

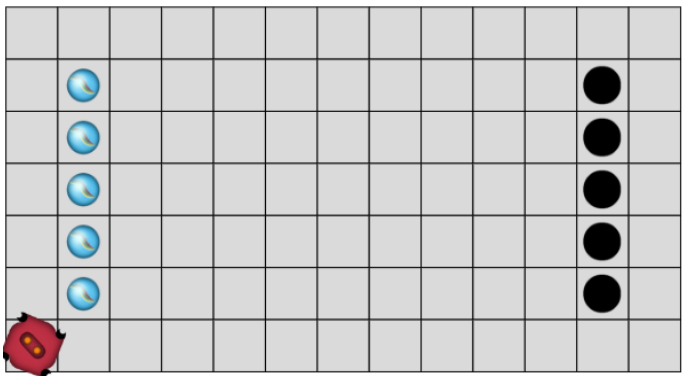


Fonctions disponibles : droite(), gauche(), haut(), ramasserBille(), deposerBille().
Mots-clés autorisés : for.

```
from robot import *
droite()
droite()
haut()
for loop in range(3):
    ramasserBille()
    for loop in range(5):
       droite()
    deposerBille()
    gauche()
    haut()
```



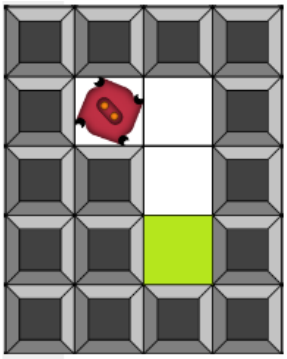
```
from robot import *
droite()
haut()
for loop in range(5):
  ramasserBille()
  for loop in range(10):
     droite()
  deposerBille()
  for loop in range(10):
     gauche()
  haut()
```

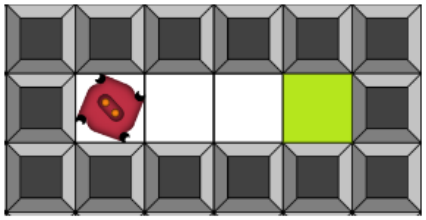


Suivre le parcours 4P

Fonctions disponibles : droite(), haut(), bas(), obstacleDroite().
Mots-clés autorisés : if, else.

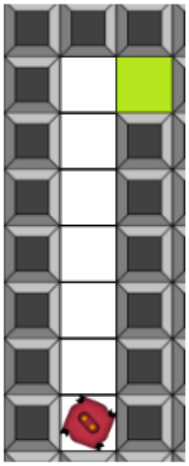
```
1  from robot import *
2  droite()
3  if obstacleDroite():
4   bas()
5   bas()
6  else:
7       droite()
8       droite()
```

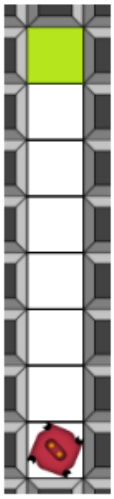




Fonctions disponibles : droite(), haut(), bas(), obstacleHaut().
Mots-clés autorisés : if, else, for.

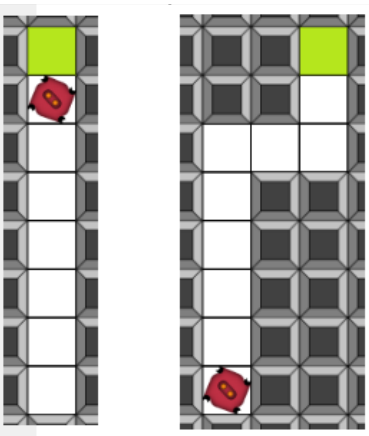
```
1  from robot import *
2  for loop in range(6):
3      haut()
4  if obstacleHaut():
5      droite()
6  else:
7      haut()
```





Fonctions disponibles : droite(), haut(), bas(), obstacleHaut(), obstacleDroite().
Mots-clés autorisés : if, else, for.

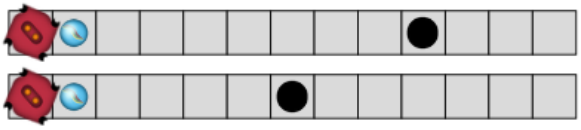
```
1  from robot import *
2  for loop in range(5):
3      haut()
4  if obstacleHaut():
5      droite()
6      droite()
7      haut()
8      haut()
9  else:
10     haut()
```



Ranger les billes 5P

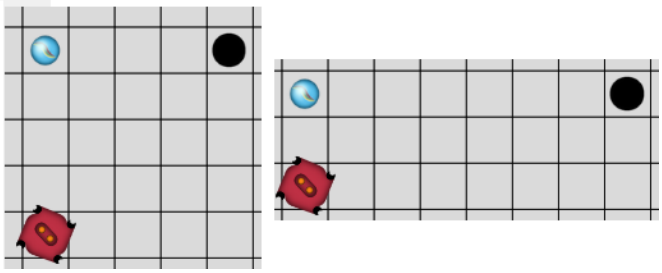
Mots-clés autorisés : while, not.

```
1  from robot import *
2  droite()
3  ramasserBille()
4  while not surTrou():
5      droite()
6  deposerBille()
```



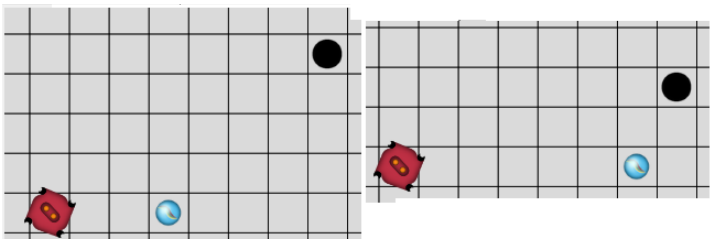
Mots-clés autorisés : while, not.

```
1  from robot import *
2  while not surBille():
3      haut()
4  ramasserBille()
5  while not surTrou():
6      droite()
7  deposerBille()
```



Mots-clés autorisés : while, not.

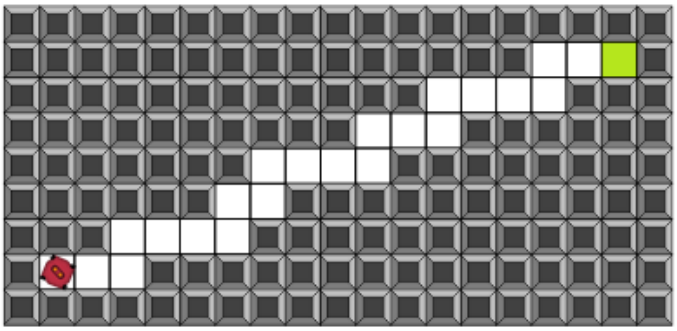
```
1  from robot import *
2  while not surBille():
3      droite()
4  ramasserBille()
5  while not bordGrilleDroite():
6      droite()
7  gauche()
8  while not surTrou():
9      haut()
10 deposerBille()
```



Suivre le parcours 5P

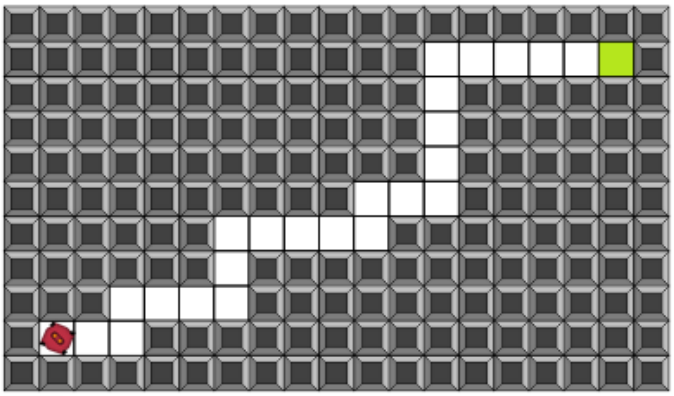
Mots-clés autorisés : for, if, else.

```
1  from robot import *
2  for loop in range(22):
3      if obstacleDroite():
4          haut()
5      else:
6          droite()
```



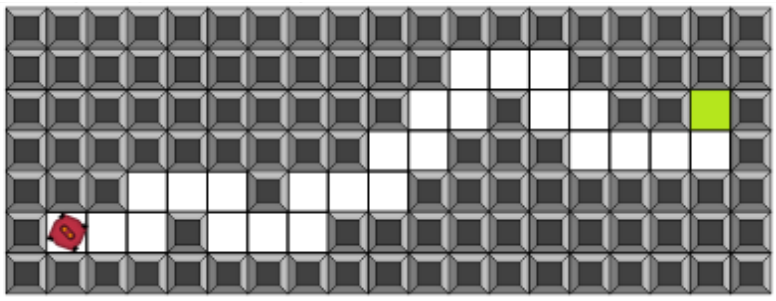
Mots-clés autorisés : for, if, else.

```
1  from robot import *
2  for loop in range(24):
3      if obstacleDroite():
4          haut()
5      else:
6          droite()
```



Mots-clés autorisés : for, if, else.

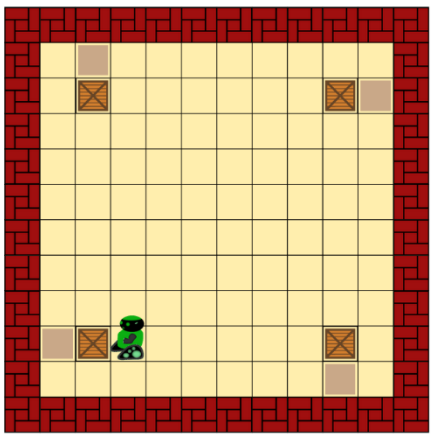
```
1  from robot import *
2  for loop in range(16):
3      droite()
4      if obstacleDroite():
5          if obstacleHaut():
6              bas()
7          else:
8              haut()
```



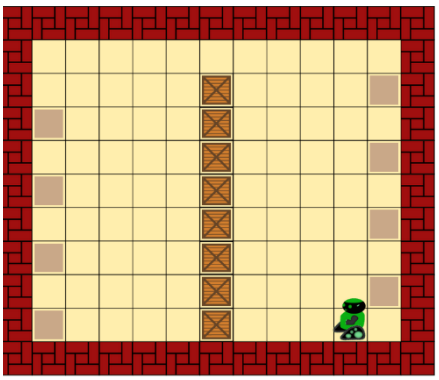
v6pwysb3

Pousser les caisses 2P

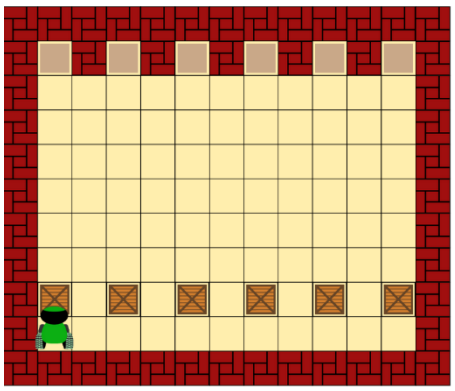
```
from robot import *
for loop in range(4):
    pousserCaisse()
    tournerDroite()
    avancer()
    avancer()
    avancer()
    avancer()
    avancer()
    avancer()
```



```
from robot import *
for loop in range(4):    for loop in range(3):
        avancer()
    for loop in range(5):
        pousserCaisse()
    tournerDroite()
    avancer()
    tournerDroite()
    for loop in range(3):
        avancer()
    for loop in range(5):
        pousserCaisse()
    tournerGauche()
    avancer()
    tournerGauche()
```

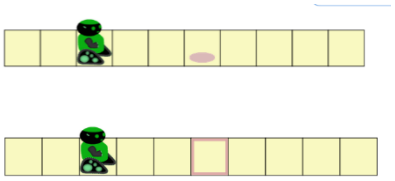


```
from robot import *
for loop in range(5):
    for loop in range(7):
        pousserCaisse()
    tournerDroite()
    avancer()
    tournerDroite()
    for loop in range(7):
        avancer()
    tournerGauche()
    avancer()
    tournerGauche()
for loop in range(7):
    pousserCaisse()
```

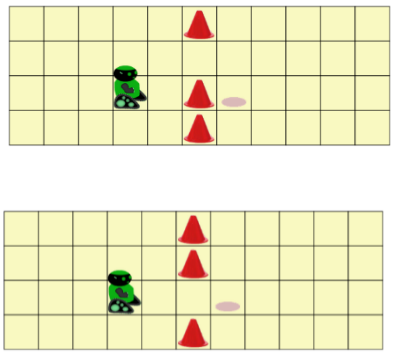


Déposer des plots 4P

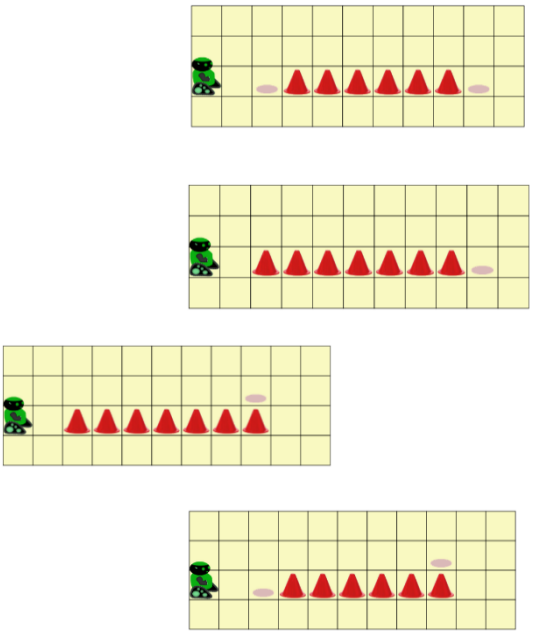
```
from robot import *
avancer()
avancer()
avancer()
if surCaseMarquee():
    deposerPlot()
```



```
from robot import *
avancer()
if plotDevant():
    tournerGauche()
    avancer()
    tournerDroite()
    avancer()
    avancer()
    tournerDroite()
    avancer()
else:
    avancer()
    avancer()
deposerPlot()
```

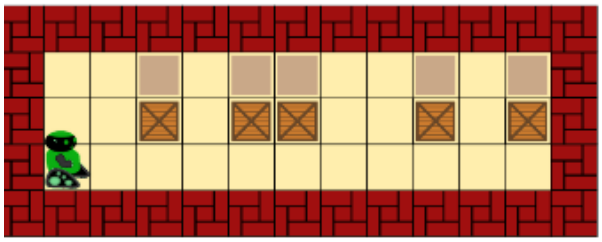


```
from robot import *
tournerGauche()
avancer()
tournerDroite()
avancer()
for loop in range(9):
    tournerDroite()
    if plotDevant():
        tournerGauche()
    else:
        avancer()
        if surCaseMarquee():
            deposerPlot()
        tournerDroite()
        tournerDroite()
        avancer()
        tournerDroite()
    avancer()
    if surCaseMarquee():
        deposerPlot()
```

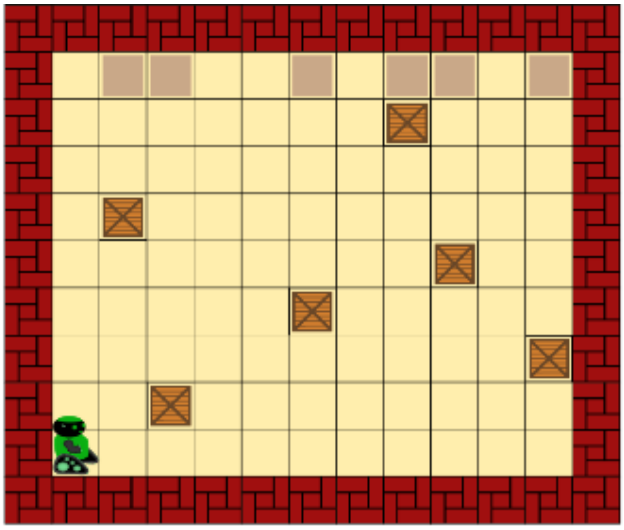


Pousser les caisses 3P

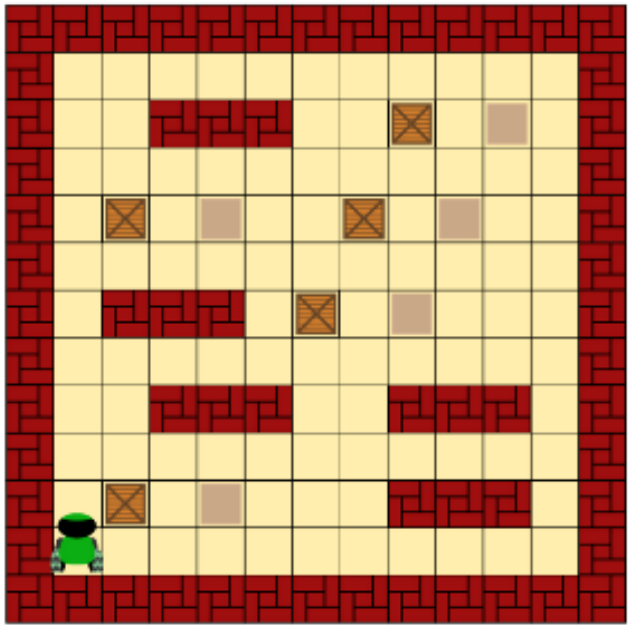
```
from robot import *
for loop in range(10):
    avancer()
    tournerGauche()
    if caisseDevant():
        pousserCaisse()
        tournerDroite()
        tournerDroite()
        avancer()
        tournerGauche()
    else:
        tournerDroite()
```



```
from robot import *
for loop in range(10):
    avancer()
    tournerGauche()
    for loop in range(7):
        if caisseDevant():
            pousserCaisse()
        else:
            avancer()
    tournerDroite()
    tournerDroite()
    for loop in range(7):
        avancer()
    tournerGauche()
```

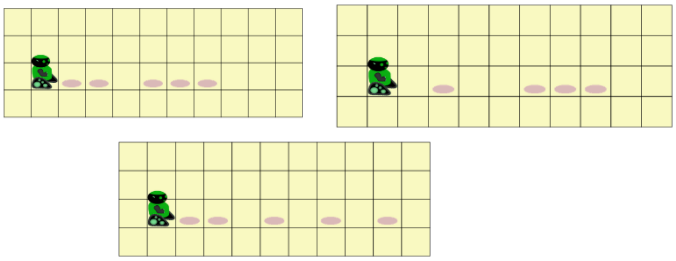


```
from robot import *
```

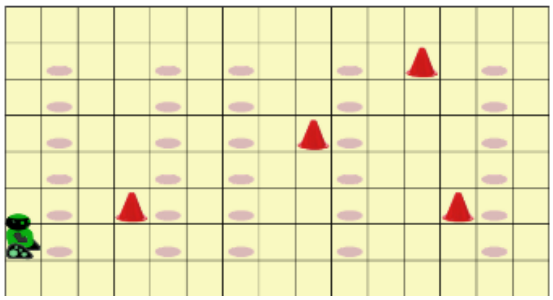


Déposer des plots 5P

```
from robot import *
for loop in range(9):
   avancer()
   if surCaseMarquee():
      deposerPlot()
```



```
from robot import *
for loop in range(9):
   avancer()
   if surCaseMarquee():
      tournerGauche()
      for loop in range(6):
         deposerPlot()
         avancer()
      tournerDroite()
      avancer()
      tournerDroite()
      for loop in range(6):
         avancer()
      tournerGauche()
```



```
from robot import *
for loop in range(4):
   for loop in range(8):
      if plotDevant():
         tournerGauche()
         avancer()
         tournerDroite()
         avancer()
         avancer()
         tournerDroite()
         avancer()
         tournerGauche()
      else:
         avancer()
         if surCaseMarquee():
            deposerPlot()
   tournerGauche()
   avancer()
   tournerGauche()
   for loop in range(9):
      avancer()
   tournerDroite()
   avancer()
   tournerDroite()
```