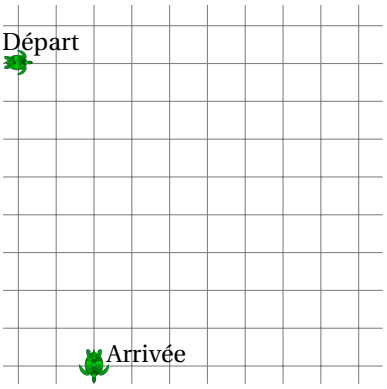
La tortue

I Trace ton chemin...

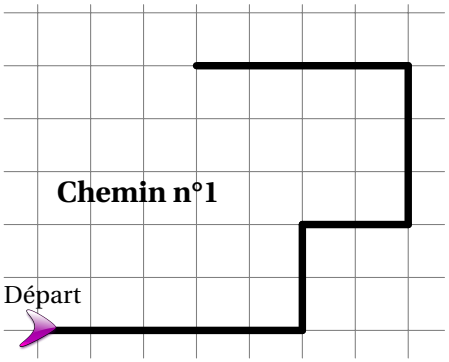
1. L'algorithme ci-dessous permet de déplacer une tortue :

```
tortue.avance(7)
tortue.droite(90)
tortue.avance(5)
tortue.droite(90)
tortue.avance(5)
tortue.gauche(90)
tortue.avance(3)
```



Dessiner le chemin parcouru par la tortue.

2. Écrire un algorithme qui permet de déplacer la tortue comme dans la figure ci-dessous :



. . . . . .
. . . . . .
. . . . . .
. . . . . .
. . . . . .
. . . . . .
. . . . . .
. . . . . .
. . . . . .

- - - - - - - - - - - - - - - ✂ -

Il est possible d'inventer de nouvelles commandes pour déplacer la tortue !

Nom de la nouvelle commande :

. . . . . .

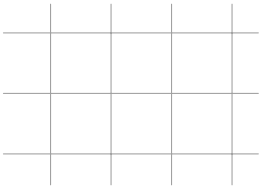
Votre programme :

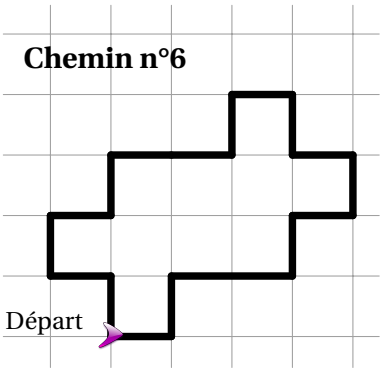
. . . . . .
. . . . . .
. . . . . .
. . . . . .
. . . . . .

Le résultat :

. . . . . .
. . . . . .
. . . . . .
. . . . . .
. . . . . .
. . . . . .
. . . . . .
. . . . . .
. . . . . .

Dessin de la nouvelle commande :



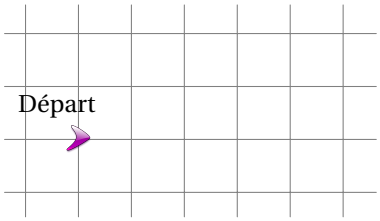


II Définir et appeler une procédure

Une idée, pour alléger le code, est de créer de nouvelles **procédures** offrant à la tortue un nouveau déplacement. Voici ci-contre une façon de définir une **procédure** : « *zigzag* ».

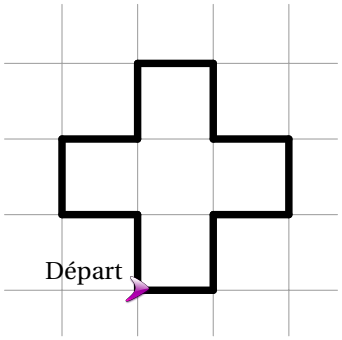
```
def zigzag() :
    tortue.avance(1)
    tortue.gauche(90)
    tortue.avance(1)
    tortue.droite(90)
    tortue.avance(1)
```

1. Que va dessiner la tortue avec l'algorithme ci-dessous ?




```
zigzag()
zigzag()
```

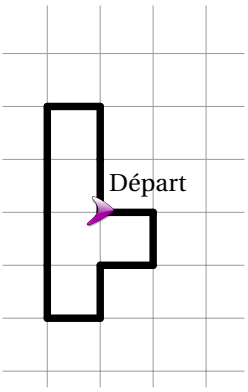
2. En utilisant la **procedure** **zigzag**(), écrire un algorithme permettant de dessiner la figure ci-dessous :



3. Définir une procédure permettant de réaliser le chemin ci-dessous en un minimum de lignes de codes :

Votre procédure :

Votre programme :



4. Demander une carte à votre professeur. Écrire un algorithme permettant de reproduire le dessin de la carte. Les **procédures** suivantes peuvent être utiles :

```
def demiTourGauche() :
    tortue.avance(1)
    tortue.gauche(90)
    tortue.avance(1)
    tortue.gauche(90)
    tortue.avance(1)
```

```
def demiTourDroite() :
    tortue.avance(1)
    tortue.droite(90)
    tortue.avance(1)
    tortue.droite(90)
    tortue.avance(1)
```

```
def doubleZigzag() :
    zigzag()
    tortue.gauche(90)
    tortue.avance(1)
    tortue.droite(90)
    tortue.avance(1)
```

```
def carre() :
    tortue.avance(2)
    tortue.droite(90)
    tortue.avance(2)
    tortue.droite(90)
    tortue.avance(2)
    tortue.droite(90)
    tortue.avance(2)
```

Vous pouvez aussi inventer vos procédures et venir les marquer au tableau !

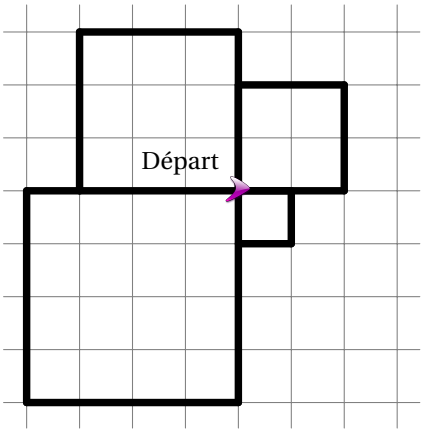
III Procédure à paramètre

1. Nadia a écrit un algorithme permettant de dessiner la figure ci-dessous en utilisant une **procédure carre**(x). Compléter l'algorithme de Nadia.

```
def carre(x) :
    tortue.avance(x)
    tortue.droite(90)
    tortue.avance(x)
    tortue.droite(90)
    tortue.avance(x)
    tortue.droite(90)
    tortue.avance(x)
```

Le programme de Nadia :

```
carre(1)
.....
.....
.....
```

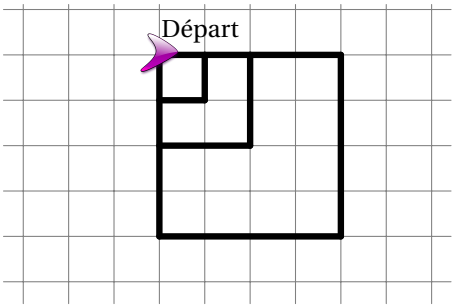


☞ La **procédure carre**(x) demande un **paramètre** x.

☞ Comme pour une fonction, on utilise cette **procédure** en remplaçant le **paramètre** x par une valeur.

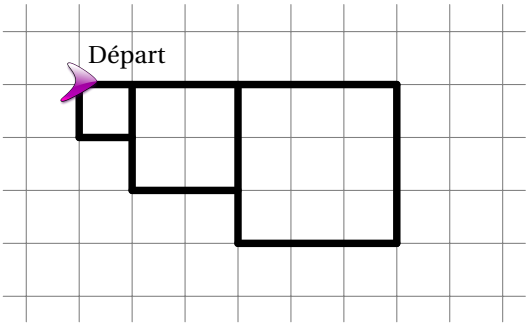
2. Écrire un algorithme qui permet de déplacer la tortue comme dans la figure ci-dessous :

```
.....
.....
.....
.....
.....
```

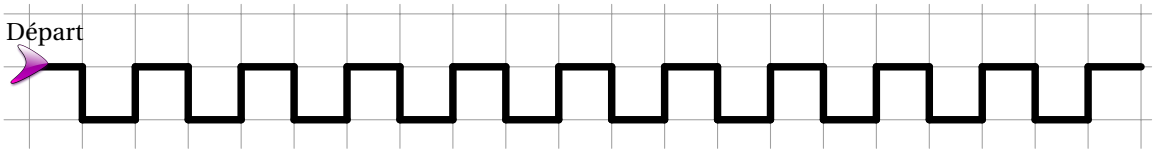


3. Même question avec la figure ci-dessous :

```
.....
.....
.....
.....
.....
.....
.....
```

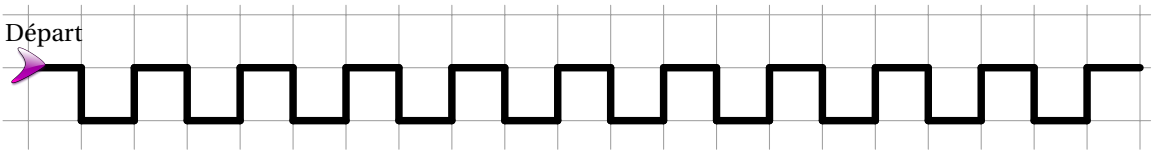


4. Écrire un algorithme qui permet de déplacer la tortue comme dans la figure ci-dessous :



IV Vers les boucles

1. Écrire un algorithme qui permet de déplacer la tortue comme dans la figure ci-dessous :




```
# (Répéter 10 fois :)
for i in range(10):
    ..............................
    ..............................
    ..............................
    ..............................
    ..............................
    ..............................
    ..............................
    ..............................
..................................
..................................
```

☞ « `Répéter 10 fois :` » n'existe pas en *Python*.

☞ On écrit : « `for i in range(10):` »

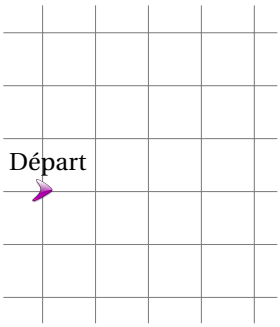
☞ On peut le traduire par : « *Pour i allant de 0 à 9 faire :* »

☞ Le **compteur** `i` parcourt les valeurs 0 ; 1 ; 2 . . . jusqu'à 9.

☞ À la sortie de la **boucle** `for`, `i` aura la valeur 9.

2. Que va dessiner la tortue avec l'algorithme ci-dessous ?

```
tortue.avance(1)
for i in range(4):
    tortue.avance(2)
    tortue.gauche(90)
tortue.droite(90)
tortue.avance(1)
```



La fin de l'**indentation** annonce la fin du contenu de la **boucle**.

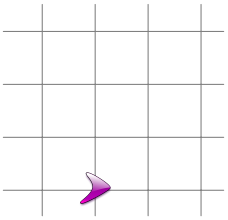
Voici deux **procédures** déjà utilisées :

```
def zigzag() :
    tortue.avance(1)
    tortue.gauche(90)
    tortue.avance(1)
    tortue.droite(90)
    tortue.avance(1)
```

```
def carre(x) :
    tortue.avance(x)
    tortue.droite(90)
    tortue.avance(x)
    tortue.droite(90)
    tortue.avance(x)
    tortue.droite(90)
    tortue.avance(x)
```

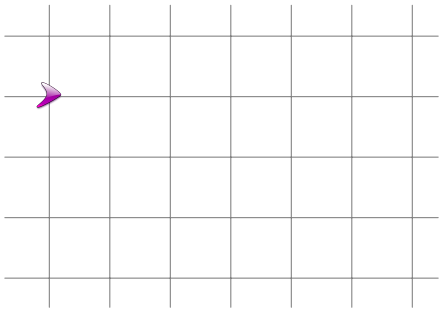
3. Que va dessiner la tortue avec l'algorithme ci-dessous ?

```
for i in range(4):
    zigzag()
    tortue.gauche(90)
```

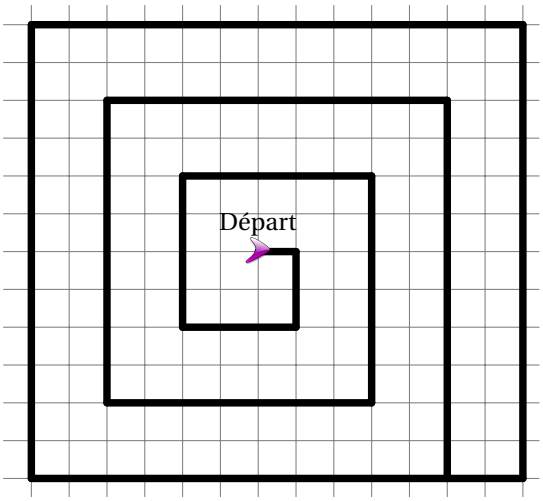


4. Que va dessiner la tortue avec l'algorithme ci-dessous ?

```
for i in range(1,4):
    carre(i)
    tortue.droite(90)
    tortue.avance(i)
```



5. Écrire un algorithme permettant de dessiner la figure ci-dessous :



.....
.....
.....
.....
.....
.....
.....

V Sur machine

1. Ouvrir *Pyzo* et tester le code suivant :

```
import turtle as tortue

tortue.reset()

#Vos procedures:
def zigzag():
    tortue.fd(10)
    tortue.left(90)
    tortue.fd(10)
    tortue.right(90)
    tortue.fd(10)

#Votre algorithme:
zigzag()
tortue.fd(10)
zigzag()

#Fenetre graphique:
tortue.mainloop()
```

a) Que signifie la commande : `tortue.fd(10)` ?

.....

b) Que signifie la commande : `tortue.left(90)` ?

.....

c) Que signifie la commande : `tortue.right(90)` ?

.....

d) Que signifie la commande : `tortue.reset()` ?

.....

2. Écrire un programme permettant de dessiner les figures ci-dessous :

Figure n°1

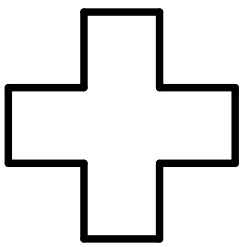


Figure n°2

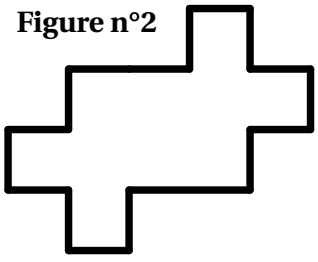


Figure n°3

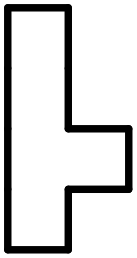


Figure n°4

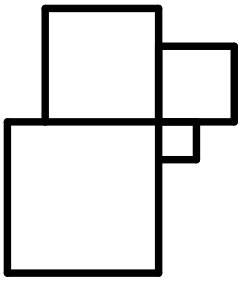


Figure n°5

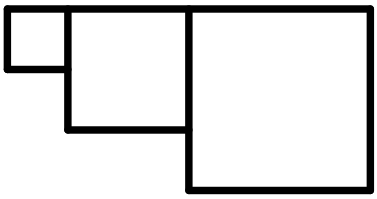


Figure n°6

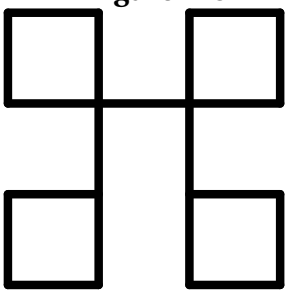


Figure n°7

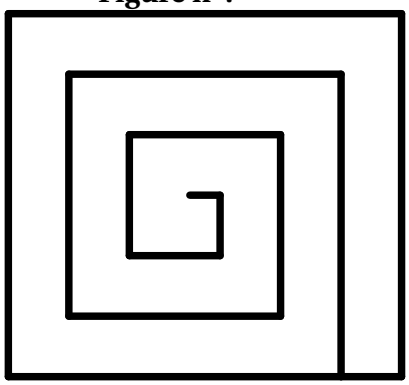


Figure n°8

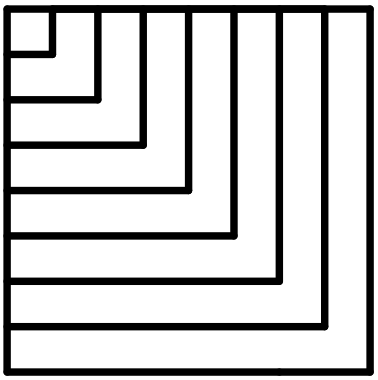


Figure n°9

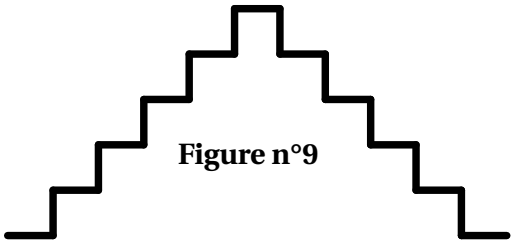


Figure n°10

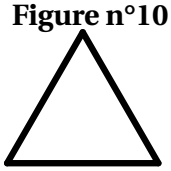
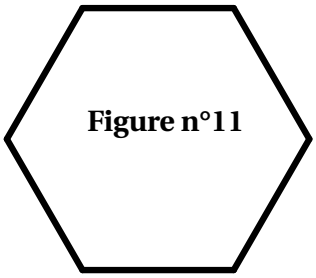


Figure n°11



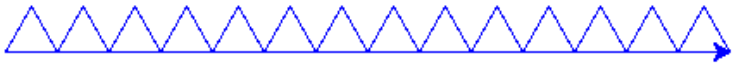


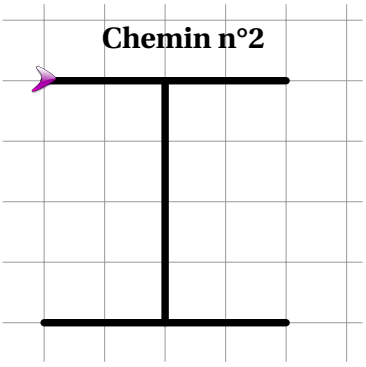
Figure n°12

Écrire un programme qui demande un nombre puis qui dessine un polygone régulier avec ce nombre de côtés.

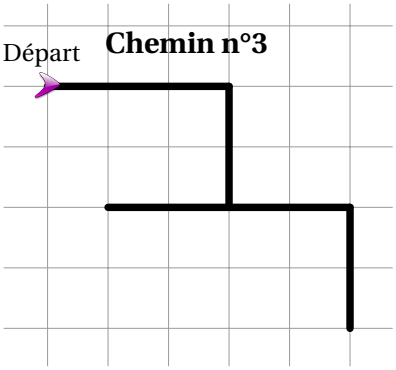
Voici une idée pour commencer :

```
1 import turtle as tortue
2 import random
3 tortue.reset()
4
5 n=int(input("Donner le nombre de cotes : "))
```

Chemin n°2

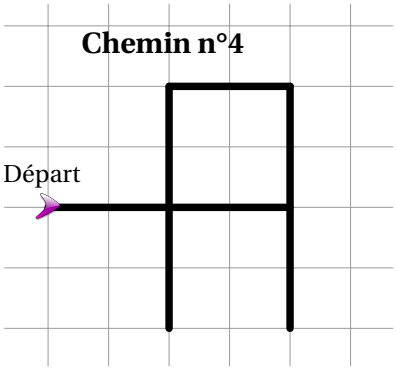


Départ **Chemin n°3**

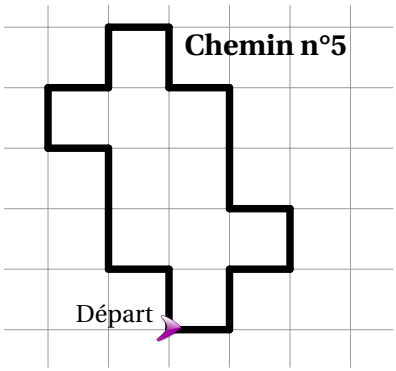


Chemin n°4

Départ

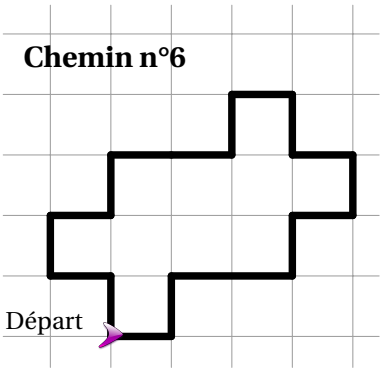


Chemin n°5



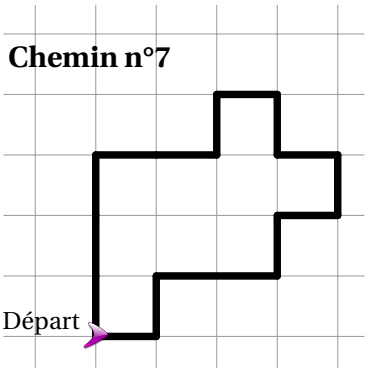
Départ

Chemin n°6

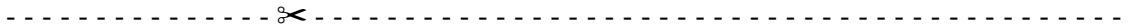


Départ

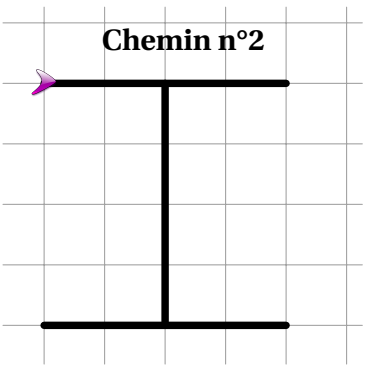
Chemin n°7



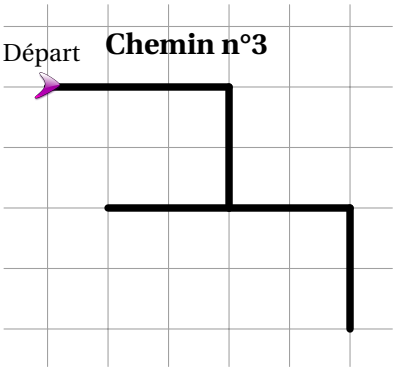
Départ



Chemin n°2

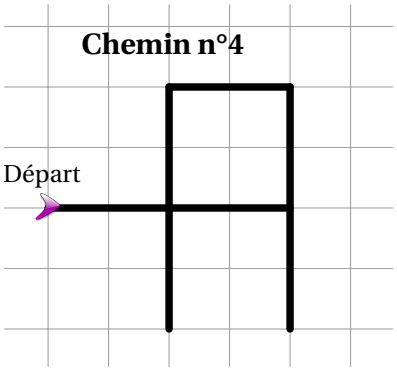


Départ **Chemin n°3**

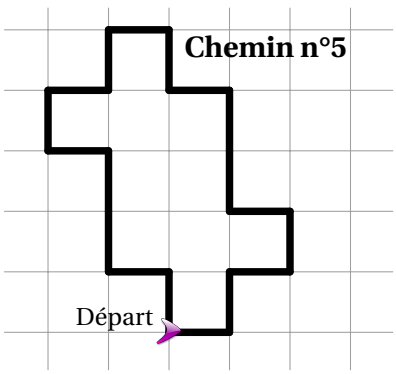


Chemin n°4

Départ

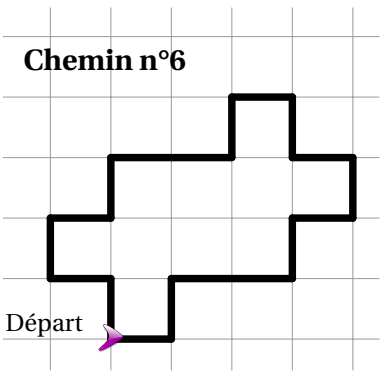


Chemin n°5



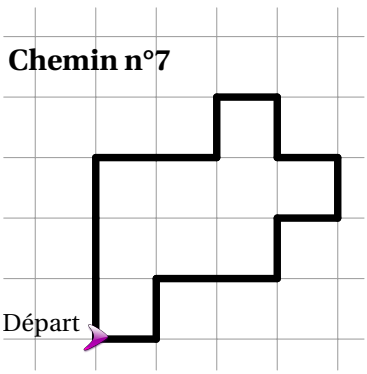
Départ

Chemin n°6



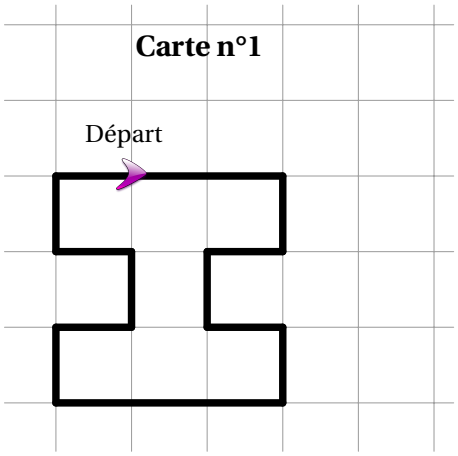
Départ

Chemin n°7

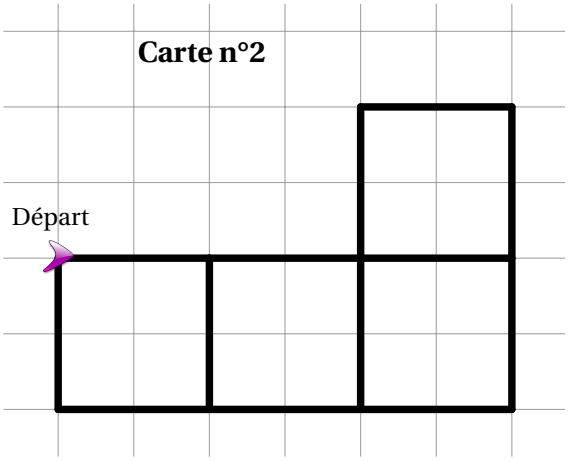


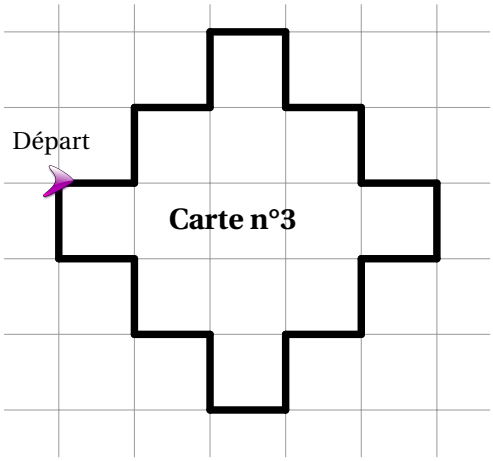
Départ

Carte n°1



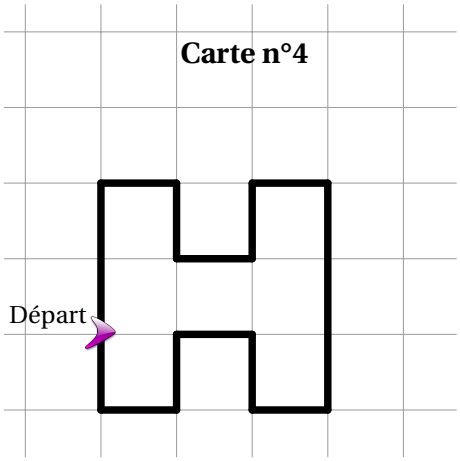
Carte n°2



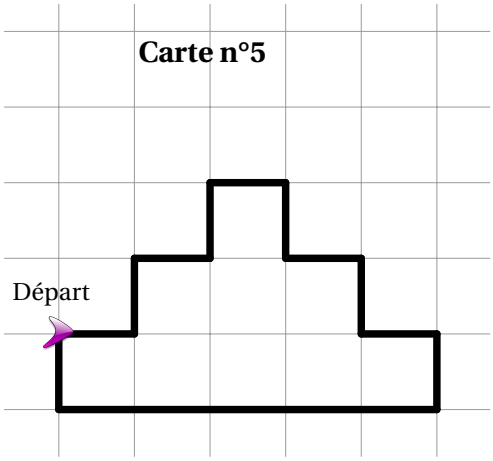


Carte n°3

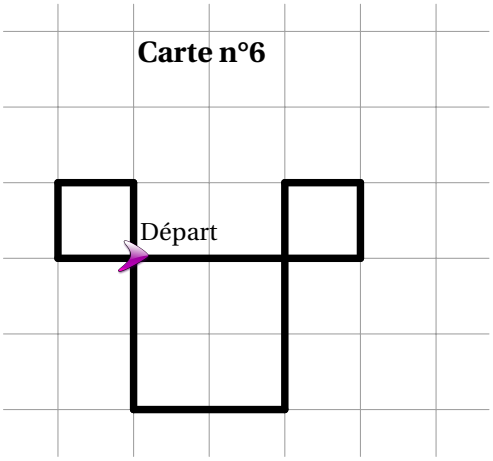
Carte n°4



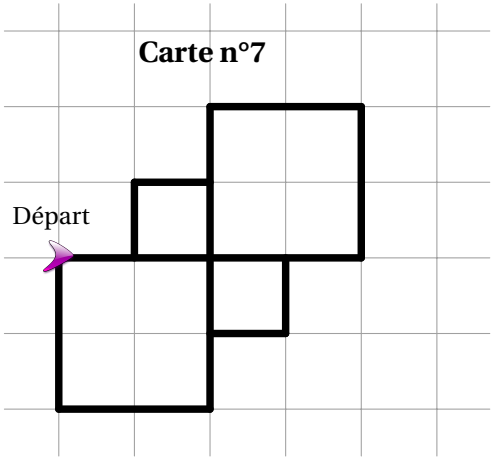
Carte n°5



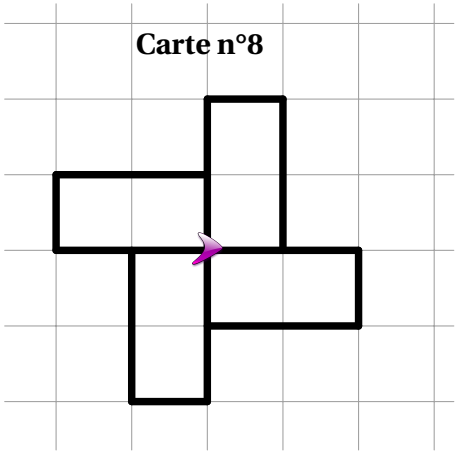
Carte n°6

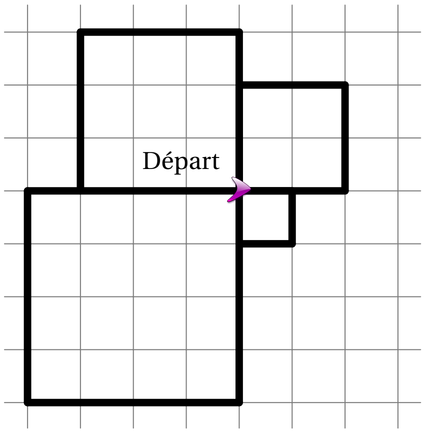
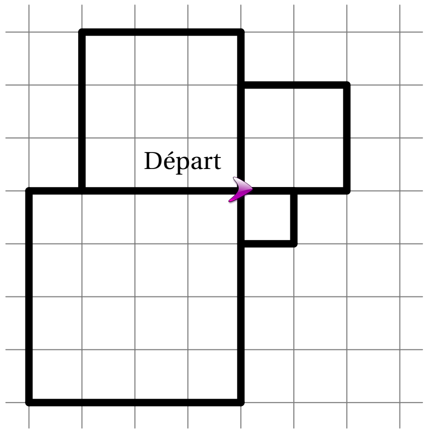
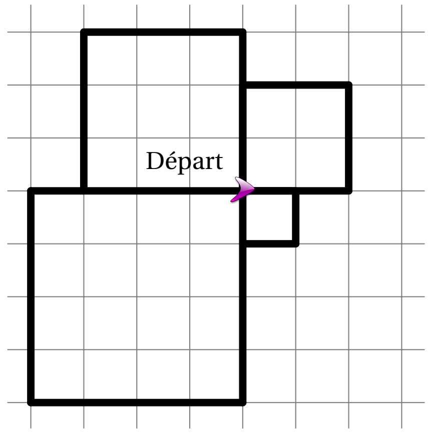
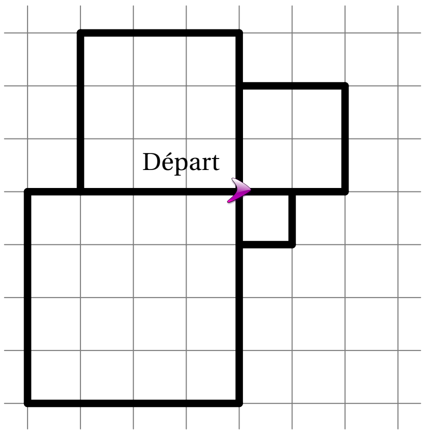
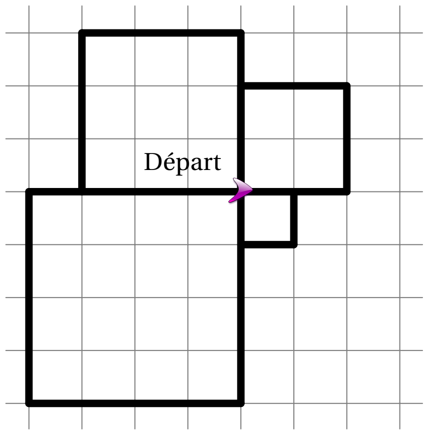
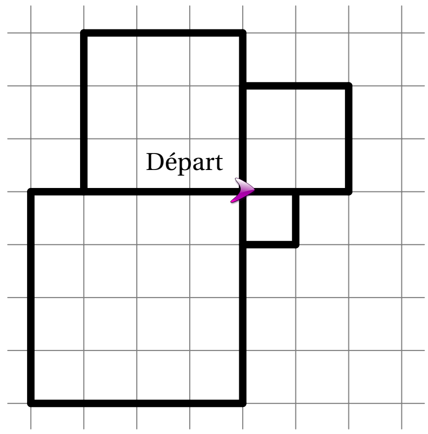


Carte n°7

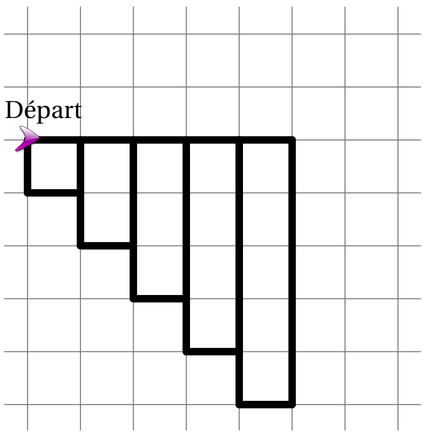
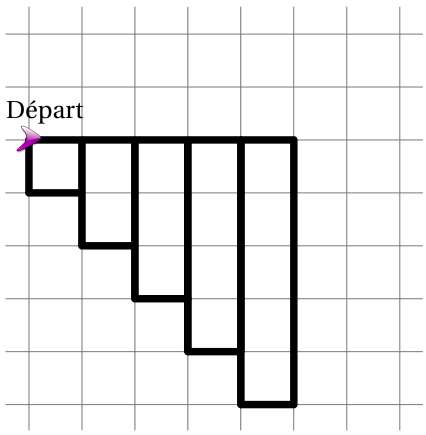
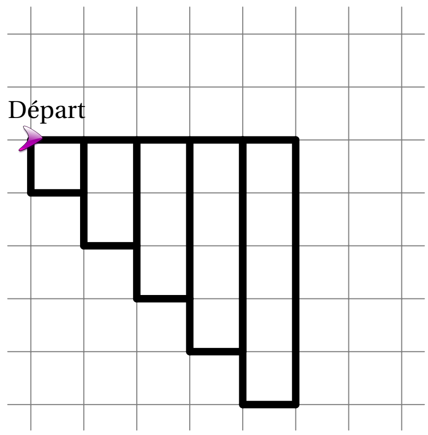
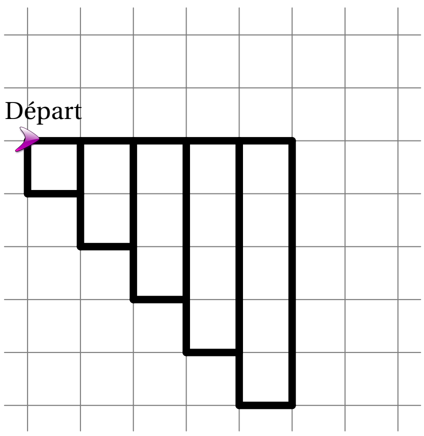
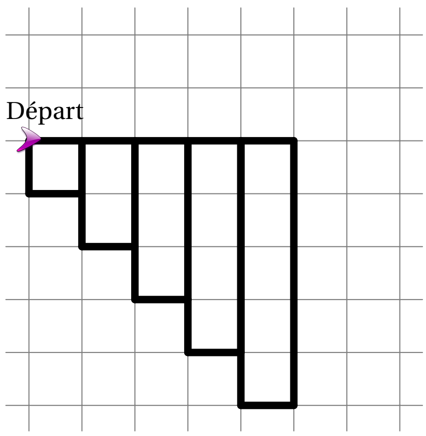
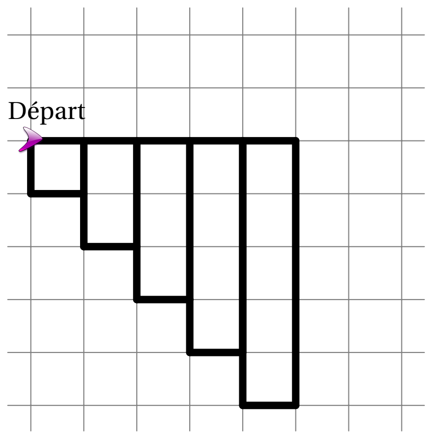


Carte n°8




Départ

Départ

Départ

Départ

Départ

Départ

✂


Départ

Départ

Départ

Départ

Départ

Départ

Chemin ou Carte n°......

. . . .

. . . .

. . . .

. . . .

. . . .

. . . .

. . . .

. . . .

. . . .

. . . .

Chemin ou Carte n°......

. . . .

. . . .

. . . .

. . . .

. . . .

. . . .

. . . .

. . . .

. . . .

. . . .

- - - - - - - - - - - - - - - ✂ -

Chemin ou Carte n°......

. . . .

. . . .

. . . .

. . . .

. . . .

. . . .

. . . .

. . . .

. . . .

. . . .

Chemin ou Carte n°......

. . . .

. . . .

. . . .

. . . .

. . . .

. . . .

. . . .

. . . .

. . . .

. . . .

- - - - - - - - - - - - - - - ✂ -

Chemin ou Carte n°......

. . . .

. . . .

. . . .

. . . .

. . . .

. . . .

. . . .

. . . .

. . . .

. . . .

Chemin ou Carte n°......

. . . .

. . . .

. . . .

. . . .

. . . .

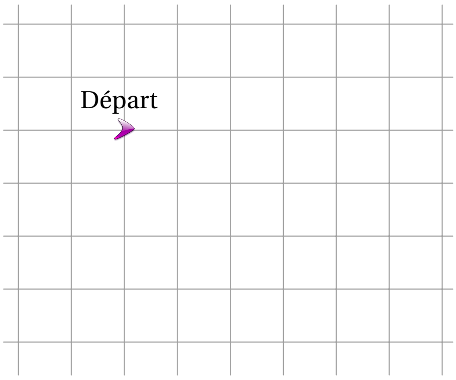
. . . .

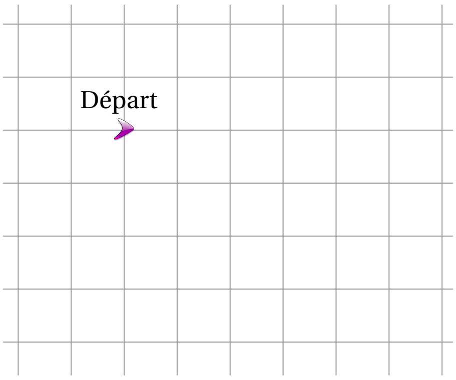
. . . .

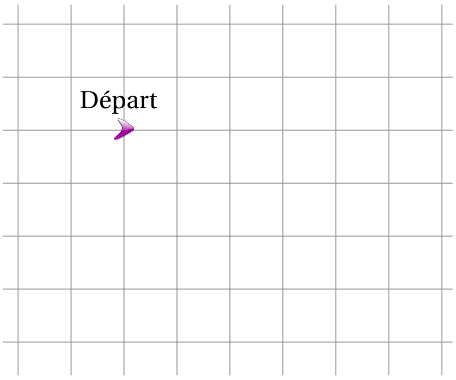
. . . .

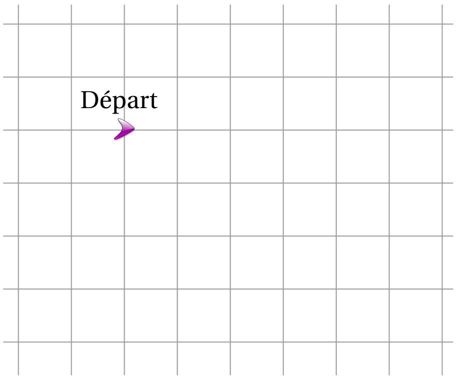
. . . .

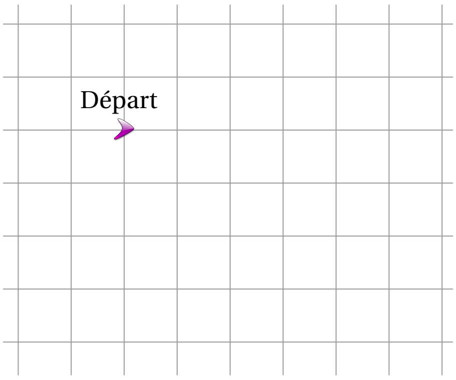
. . . .

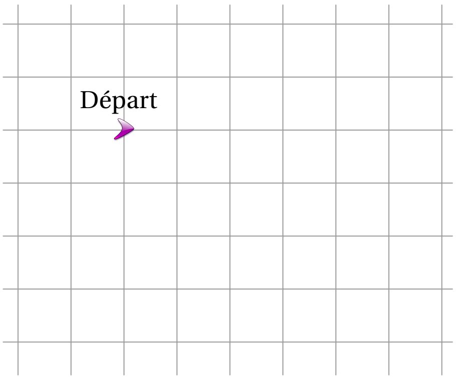

Départ

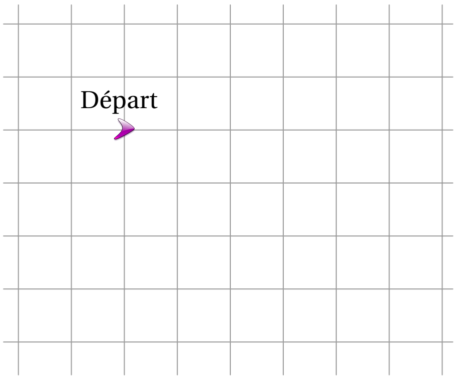

Départ

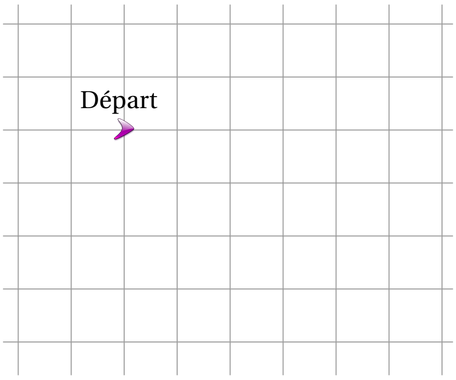

Départ

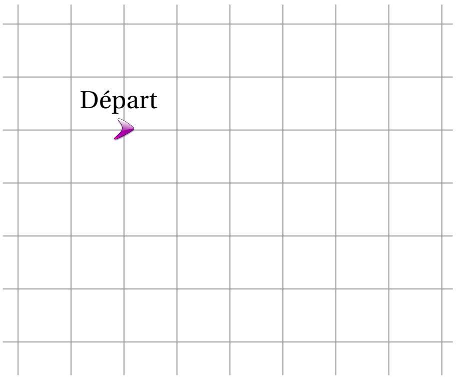

Départ

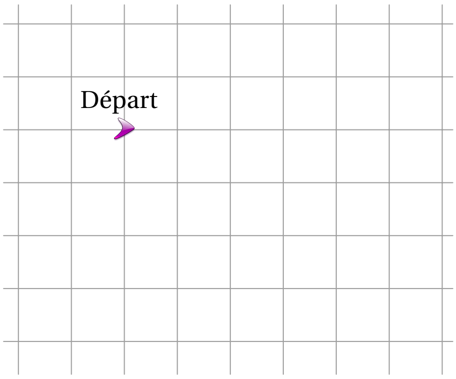

Départ

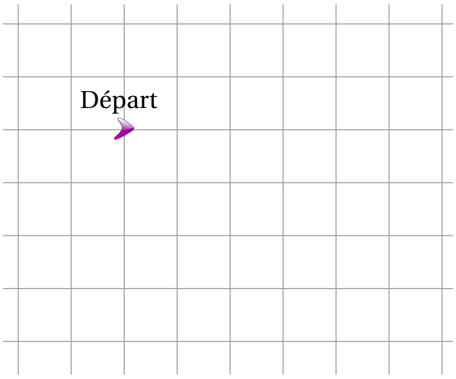

Départ

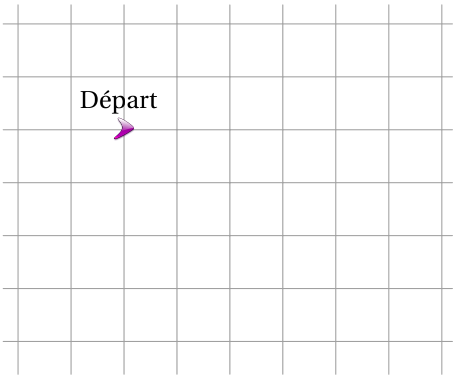

Départ

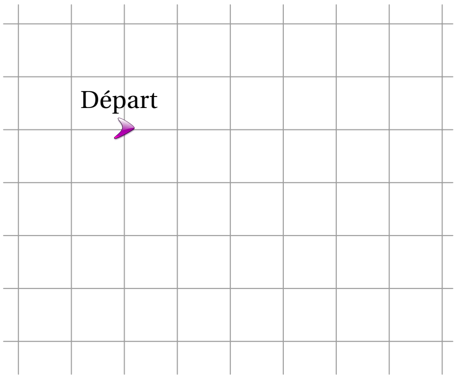

Départ

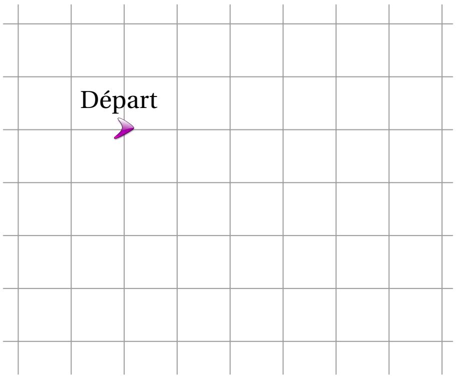

Départ

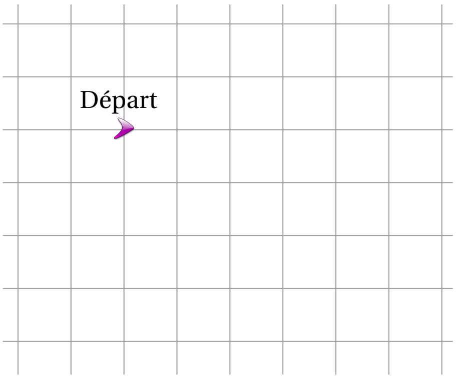

Départ


Départ


Départ


Départ


Départ


Départ