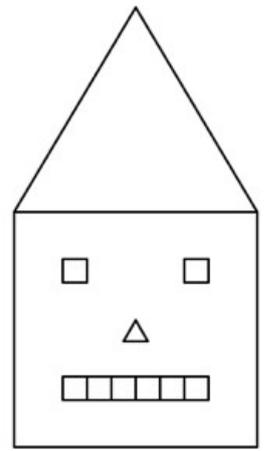
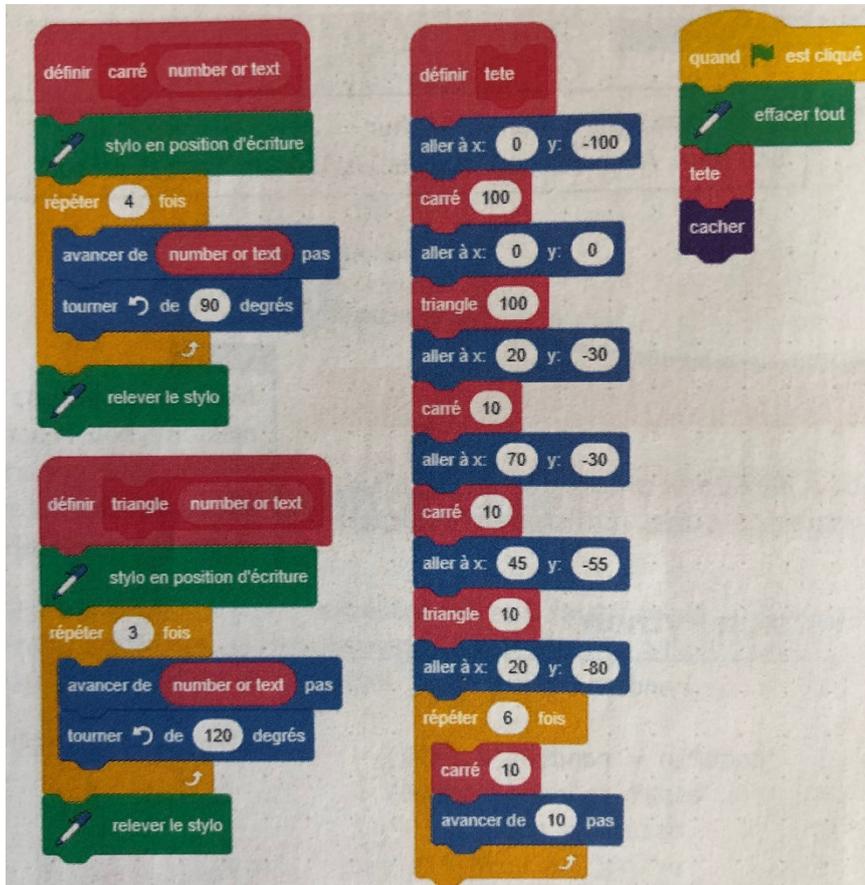


T01-04

Dans le cadre d'un projet de robotique, il s'agit de programmer un robot afin qu'il dessine la figure géométrique ci-dessous, composée de triangles et de carrés.

Version Scratch :



Version Python :

ça y est, on apprend à produire des fonction :

`def nom_de_la_fonction(variable1, variable2...)` :
on écrit la fonctions
en précisant à la fin
ce qu'on obtient

puis pour appeler la fonction dans un programme :

`nom_de_la_fonction(valeur variable1, valeur variable2...)`

Exercice :

- Combien de fonctions comporte le programme et quels sont les noms de ces fonctions ?
- Quelle différence voyez-vous entre la fonction `carre` et la fonction `tête` ?

```
maison-tete-turtle.py
1 from turtle import *
2
3 def carre(n) :
4     pendown()
5     for i in range (4):
6         forward(n)
7         left(90)
8     penup()
9
10 def triangle(n):
11     pendown()
12     for i in range (3):
13         forward(n)
14         left(120)
15     penup()
16
17 def tete():
18     goto(0,-100)
19     carre(100)
20     goto(0,0)
21     triangle(100)
22     goto(20,-30)
23     carre(10)
24     goto(45,-55)
25     triangle(10)
26     goto(20,-80)
27     for i in range (6):
28         carre(10)
29         forward(10)
30     goto(70,-30)
31     carre(10)
32
33 tete()
34 hideturtle()
35
```

c. Il est à présent temps de passer à l'action. Plusieurs niveaux de difficultés vous sont proposés : Objectif Mondrian...

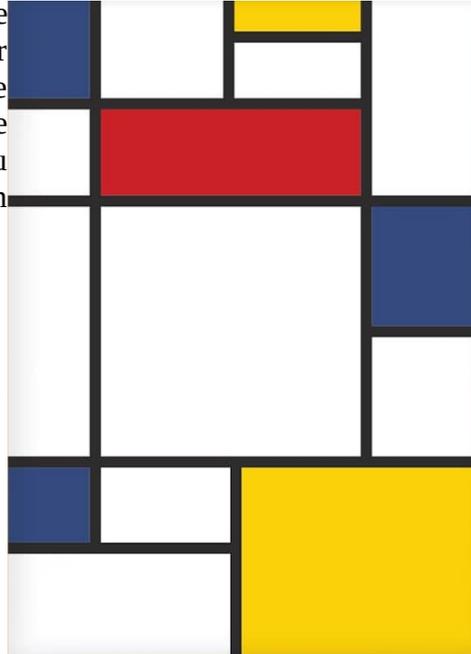
1- produire une fonction rectangle(L,l) qui permet de dessiner un rectangle de Longueur et de largeur voulues avec un trait de pourtour de 5 pixels de large..

2- 1+ produire une fonction qui demande les L et l à l'utilisateur avant de la tracer.

3- 2 + on demande la couleur du remplissage du rectangle et le rectangle tracé est de la bonne couleur.

4- 3 + on demande les coordonnées du point en haut à gauche du rectangle et il est tracé à partir de là.

Aide : Les programmes sont ici découpés en **fonctions**. Une fonction est un sous-programme qui doit pouvoir s'exécuter de manière autonome. Cette organisation permet de simplifier l'**algorithme** du programme principal (on appelle **programme principal** les instructions qui sont données au moment de l'exécution. Cela simplifie aussi la tâche si on doit modifier le programme.



Créer un « presque Mondrian » :

Vous êtes en équipe de 4 : partagez vous le travail !

Aide pour Python :

Appeler le module turtle pour dessiner en le renommant **tu** pour écrire les instructions plus rapidement.

Import turtle as tu

Chaque instruction doit donc être écrite tu.instruction. Exemple : tu.fd(150) pour avancer de 150 pixels.

les commandes de base du module turtle : Types de donnée : pixel ou angle ou couleur

<code>reset()</code>	-> Efface l'écran, recentre la tortue et remet les variables à zéro	<code>forward(distance)</code> ou <code>fd</code>	-> Avance d'une distance donnée
<code>backward(distance)</code> ou <code>bk</code>	Reculer d'une distance donnée	<code>left(angle)</code> ou <code>lt</code> <code>right(angle)</code> ou <code>rt</code>	Pivote vers la gauche/la droite
<code>up()</code> <code>down()</code>	Relève le crayon (pour pouvoir avancer sans dessiner) ou le descend pour dessiner	<code>goto(x, y)</code> /!\ Ne pas oublier d'utiliser la fonction <code>up()</code> avant d'utiliser <code>goto()</code> car sinon il tracera le parcours effectué.	Va à l'endroit de coordonnées (x, y)
<code>color(couleur)</code> "blue" => Bleu// "red" => Rouge// "green"	Détermine la couleur du tracé (noir par défaut)	<code>ht()</code> <code>st()</code>	Masque la tortue Afficher la tortue.

=> Vert// "yellow"=>Jaune "brown", purple, black, pink, white...	défaut)		
fill(color)	Remplit un contour fermé à l'aide de la couleur sélectionnée	Exemple pour remplir un carré : fillcolor('red') begin_fill() forward(100) left(90) forward(100) left(90) forward(100) left(90) forward(100) end_fill()	
circle(rayon, angle)	Trace un cercle de rayon donné. L'argument facultatif angle indique l'angle de l'arc de cercle (par défaut 360, soit un cercle complet)	width(épaisseur	Choisit l'épaisseur du tracé
bgcolor("couleur")	Préciser la couleur de l'arrière plan de la scène.	speed(vitesse) 'slowest'=1 'slow' =4 "normal"=6 "fast" =9 "fastest" =0	Choisit la vitesse à laquelle se déplace le curseur
title(titre)	Donne un titre à la fenêtre (par défaut le titre est Turtle Graphics)	write(texte)	Écrit du texte
done() bye() clear()	Maintien(done) ou fermeture(bye) de la fenêtre ouverte à la fin du dessin clear efface l'écran	setup(w, h)	Ouvre une fenêtre de taille wxh

Bug possible : si un fichier turtle.py est créé, python va tenter de retrouver les fonction appelées par les instructions dedans... Et elles n'y seront pas. Il faut donc l'effacer ou le renommer!!!(J'ai passé des heures à le comprendre, je vous fais cadeau de ce temps ! Et ça, ça n'a pas de prix!)

Tiroir pour les gueudins : générer un mondrian de façon aléatoire !!!!