

ALGORITHMIQUE ET PROGRAMMATION

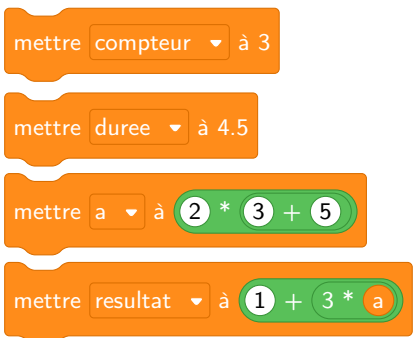
1 Instructions élémentaires

ALGORITHMIQUE


SCRATCH

PYTHON



1.1 Affecter une valeur numérique à une variable (*entier ou flottant*)

<pre>compteur ← 3 duree ← 4,5 a ← 2 × (3 + 5) resultat ← 1 + 3a</pre>		<pre>compteur = 3 duree = 4.5 a = 2 * (3 + 5) resultat = 1 + 3 * a</pre>
---	--	--

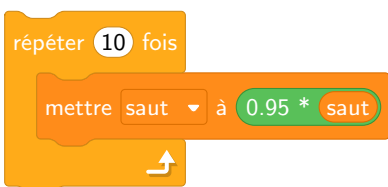
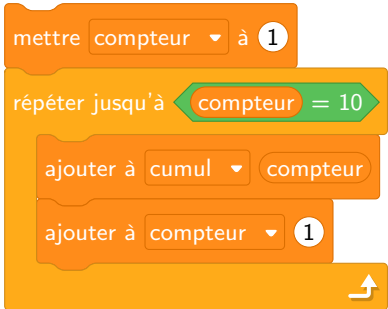
1.2 Affecter une chaîne de caractères à une variable

<pre>message ← "Bonjour"</pre>		<pre>message = "Bonjour"</pre>
--------------------------------	---	--------------------------------


1.3 Utiliser une instruction conditionnelle

<pre>Si compteur > 20 alors score ← score + 1 (finSi)</pre>		<pre>if compteur > 20 : score = score + 1</pre>
<pre>Si compteur > 20 alors score ← score + 1 sinon score ← score - 1 (finSi)</pre>		<pre>if compteur > 20 : score = score + 1 else : score = score - 1</pre>

1.4 Utiliser une boucle bornée (nombre d'itérations fixé)

<pre>Répéter 10 fois saut ← saut × 0,95 (finRépéter)</pre>		<pre>for i in range (10): saut = saut * 0.95</pre>
<pre>Pour compteur allant de 1 à 9 cumul ← cumul + compteur (finPour)</pre>		<pre>for compteur in range (1,10): cumul = cumul + compteur</pre>

1.5 Utiliser une boucle non bornée

<p>Tant que $saut \geq 1$ $saut \leftarrow saut \times 0,95$ (finTantQue)</p>		<pre>while saut >= 1 : saut = saut * 0.95</pre>
---	---	--

2 Instructions d'entrée-sortie



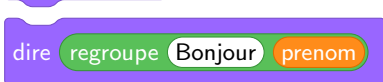
En mathématiques au lycée, on privilégiera à ces instructions d'entrées-sortie l'usage de fonction : les entrées sont les paramètres de la fonction et la sortie est l'objet renvoyé par l'instruction `return()` (voir paragraphe 4).

ALGORITHMIQUE

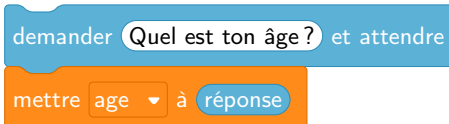
SCRATCH

PYTHON

2.1 Afficher un message et/ou la valeur d'une variable

Afficher la valeur de la variable <i>compteur</i>		<pre>print(compteur)</pre>
Afficher "Bonjour"		<pre>print("Bonjour")</pre>
Afficher "Bonjour " <i>prenom</i>		<pre>print("Bonjour", prenom)</pre>

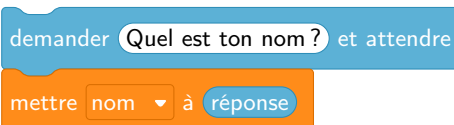
2.2 Demander la valeur d'une variable de type entier

Demander la valeur de <i>age</i>		<pre>age = int(input("Quel est ton âge ?"))</pre>
----------------------------------	--	---

2.3 Demander la valeur d'une variable de type flottant

Demander la valeur de <i>longueur</i>		<pre>longueur = float(input("Longueur ?"))</pre>
---------------------------------------	---	--

2.4 Demander la valeur d'une variable de type chaîne de caractères

Demander la valeur de <i>nom</i>		<pre>nom = input("Nom ?")</pre>
----------------------------------	--	---------------------------------

3 Compléments

ALGORITHMIQUE

SCRATCH

PYTHON

3.1 Tests et opérations booléennes

L'égalité $A = B$ est-elle vraie ?		<code>A == B</code>
L'inégalité $A \neq B$ est-elle vraie ?		<code>A != B</code>
L'inégalité $A > B$ est-elle vraie ?		<code>A > B</code>
L'inégalité $A \geq B$ est-elle vraie ?		<code>A >= B</code>

3.2 Fonctions numériques usuelles

x^2		<code>x ** 2</code>
\sqrt{x}		<code>from math import sqrt sqrt(x)</code>
Quotient dans la division euclidienne de a par b		<code>a // b</code>
Reste dans la division euclidienne de a par b		<code>a % b</code>

3.3 Nombres aléatoires

Nombre aléatoire entier entre 1 et 6		<code>from random import randint randint(1, 6)</code>
Nombre aléatoire décimal entre 0 et 1	non défini par défaut	<code>from random import uniform uniform(0, 1)</code>

3.4 Constante(s) usuelle(s)

π	non défini par défaut	<code>from math import pi pi</code>
-------	-----------------------	---

3.5 Opérations sur des chaînes de caractères

Concaténation ¹ de A et B		<code>A + B</code>
Longueur de A		<code>len(A)</code>
Première lettre de A		<code>A[0]</code>
n-ième lettre de A		<code>A[n-1]</code>

1. La concaténation de A = "bon" et B = "jour" donne la chaîne de caractères "bonjour"

4 Fonctions

Il n'est pas possible actuellement de construire de véritables fonctions dans Scratch. Néanmoins le fonctionnement des blocs peut s'en rapprocher, en affectant le résultat à une variable globale (si la fonction doit renvoyer un résultat). Ici la variable globale introduite est nommée `résultat`, quel que soit son type.

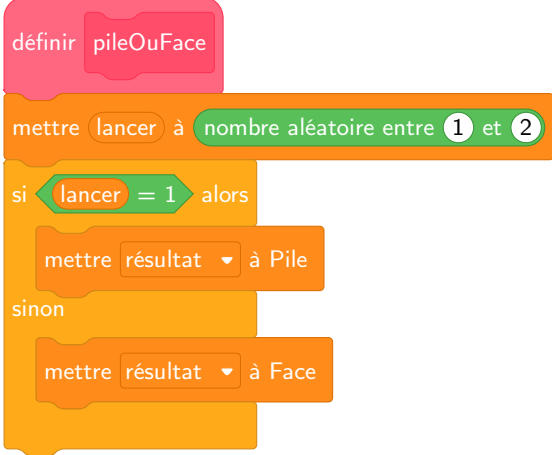
ALGORITHMIQUE

SCRATCH

PYTHON

4.1 Fonction sans paramètre

4.1.1 Définition de la fonction

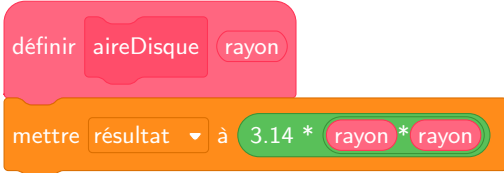
<p>Fonction <code>pileOuFace()</code> Affecter à <code>lancer</code> un nombre entier aléatoire entre 1 et 2 Si <code>lancer = 1</code> alors renvoyer "Pile" sinon renvoyer "Face" (finSi)</p>	 <pre> définir pileOuFace mettre lancer à nombre aléatoire entre 1 et 2 si lancer = 1 alors mettre résultat à Pile sinon mettre résultat à Face </pre>	<pre> from random import randint def pile_ou_face(): lancer = randint(1, 2) if lancer == 1: return "Pile" else : return "Face" </pre>
--	--	---

4.1.2 Appel de la fonction

<p><code>pileOuFace()</code></p>		<p><code>pile_ou_face()</code></p>
----------------------------------	--	------------------------------------

4.2 Fonction avec paramètre(s)

4.2.1 Définition de la fonction

<p>Aire d'un disque de rayon <code>r</code> : Fonction <code>Aire(r)</code> renvoyer πr^2</p>	 <pre> définir aireDisque rayon mettre résultat à 3.14 * rayon * rayon </pre>	<pre> from math import pi def aire_disque(rayon): return pi * rayon**2 </pre>
---	---	---

4.2.2 Appel de la fonction

<p>Aire d'un disque de rayon 5 : <code>Aire(5)</code></p>		<p><code>aire_disque(5)</code></p>
--	---	------------------------------------

5 Le Stylo et la Tortue

Le tracé lié à des déplacements peut être réalisé pour tout lutin dans Scratch à l'aide du Stylo et à l'aide du module *turtle* dans Python. Le chargement des fonctions de ce module se fait grâce à l'instruction `:from turtle import *`. Le programme principal devra comporter l'instruction `mainloop()` comme dernière instruction.

On pourra consulter en ligne la liste des fonctions disponibles dans le module *turtle*.

ALGORITHMIQUE

SCRATCH

PYTHON

5.1 Paramètres du lutin, de la tortue

Rendre le lutin visible		<code>showturtle()</code>
Rendre le lutin invisible		<code>hideturtle()</code>
Choix de la forme <i>tortue</i>	le choix d'un lutin se fait dans une fenêtre dédiée	<code>shape("turtle")</code>
Abscisse du lutin		<code>xcor()</code>
Ordonnée du lutin		<code>ycor()</code>
Direction du lutin		<code>heading()</code>

5.2 Paramètres de tracé

Placer le crayon en position d'écriture		<code>pendown()</code>
Relever le crayon		<code>penup()</code>
Mettre la taille du crayon à 4		<code>pensize(4)</code>
Prendre un stylo de couleur rouge		<code>pencolor("red")</code>
Effacer tous les tracés		<code>clear()</code>

5.3 Mouvement (relatif)

Avancer de 100		<code>forward(100)</code>
Reculer de 100		<code>backward(100)</code>
Tourner de 60° à droite		<code>right(60)</code>
Tourner de 60° à gauche		<code>left(60)</code>

5.4 Mouvement (absolu)

Aller au point de coordonnées (25; 50)		<code>goto(25, 50)</code>
S'orienter vers la droite de la fenêtre		<code>setheading(0)</code>
Donner à l'abscisse la valeur 50		<code>setx(50)</code>
Ajouter 50 à l'abscisse		<code>setx(xcor() + 50)</code>