Scratch et algorithmes

DURÉE ESTIMÉE

25 minutes

Un algorithme est une suite d'instructions qui permet de résoudre un problème. Le logiciel Scratch sert à écrire ces instructions sous forme de blocs colorés que l'on assemble. On obtient alors un programme visuel, facile à lire et à modifier.

1 - Notion d'algorithme

Algorithme

Un **algorithme** est une suite finie d'instructions, exécutées dans un ordre précis, qui permet de résoudre un problème ou d'effectuer un calcul.

Exemple

Le programme de calcul suivant est un algorithme :

- Choisir un nombre.
- Ajouter 4 au nombre choisi.

- Multiplier le résultat par 2.
- Annoncer le résultat final.

Pour le nombre 3, on obtient $(3 + 4) \times 2 = 7 \times 2 = 14$.
Pour le nombre 10, on obtient $(10 + 4) \times 2 = 14 \times 2 = 28$.

Remarque

Un algorithme comporte toujours trois parties :

- les **données d'entrée** (ce que l'on connaît au départ) ;
- le **traitement** (les instructions à exécuter dans l'ordre) ;
- la **sortie** (le résultat que l'on obtient à la fin).

2 - L'environnement Scratch

Scratch

Scratch est un logiciel de programmation visuelle. On y construit un programme en assemblant des **blocs** colorés dans la zone de script. Un personnage appelé **lutin** exécute alors les instructions sur la scène.

L'interface de Scratch est organisée en plusieurs zones :

- la **scène** : la zone où le lutin se déplace et dessine ;
- la **zone de script** : l'endroit où l'on empile les blocs ;

- la **palette de blocs** : la liste des blocs disponibles, rangés par catégories de couleurs.

Les catégories de blocs

Chaque catégorie de blocs a sa couleur :

Remarque

Pour lancer un programme, on place un bloc chapeau du menu Événements. Le plus courant est :

Il suffit ensuite de cliquer sur le drapeau vert, en haut de la scène, pour que Scratch exécute les instructions l'une après l'autre.

3 - Déplacements et tracés

Le lutin peut se déplacer sur la scène et laisser une trace grâce au **stylo**. On combine les blocs de mouvement et les blocs du stylo pour dessiner des figures.

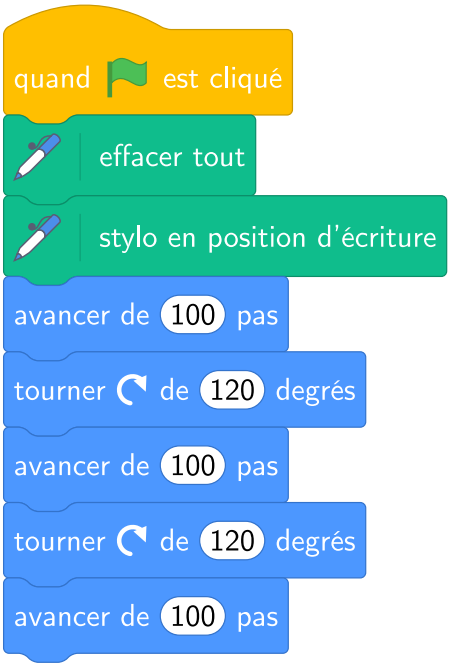
Blocs de déplacement et de tracé

Les blocs les plus utiles sont :

- **avancer de ... pas** : le lutin avance dans la direction courante ;
- **tourner de ... degrés** : le lutin pivote vers la droite ou vers la gauche ;
- **stylo en position d'écriture** : le lutin laisse une trace quand il se déplace ;
- **relever le stylo** : le lutin se déplace sans dessiner ;
- **effacer tout** : efface tous les tracés déjà présents sur la scène.

Exemple

Ce programme trace un triangle équilatéral dont les côtés mesurent 100 pas :



Télécharger le projet Scratch .sb3 ?

A chaque sommet, le lutin tourne de $\frac{360}{3} = 120$ degrés. Trois côtés et deux rotations suffisent pour fermer la figure.

Attention

Dans Scratch, l'angle qui apparaît dans le bloc « tourner de » n'est **pas** l'angle intérieur du polygone. C'est l'angle dont le lutin pivote en arrivant à un sommet. Pour un polygone régulier à n côtés, cet angle vaut $\frac{360}{n}$ degrés.

4 - Boucles

Quand les mêmes instructions se répètent plusieurs fois, on utilise une **boucle** plutôt que de recopier les blocs. Le programme devient plus court et plus facile à lire.

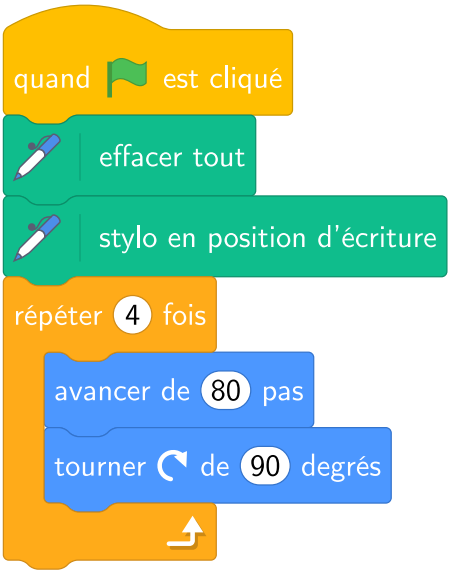
Boucle

Une **boucle** est une instruction qui fait répéter un bloc d'instructions.

- **Répéter ... fois** : le bloc est répété un nombre de fois fixé à l'avance ;
- **Répéter jusqu'à ...** : le bloc est répété tant qu'une condition n'est pas vérifiée.

Exemple

Pour tracer un carré de 80 pas de côté, on répète 4 fois la même séquence d'instructions :



Télécharger le projet Scratch .sb3 ?

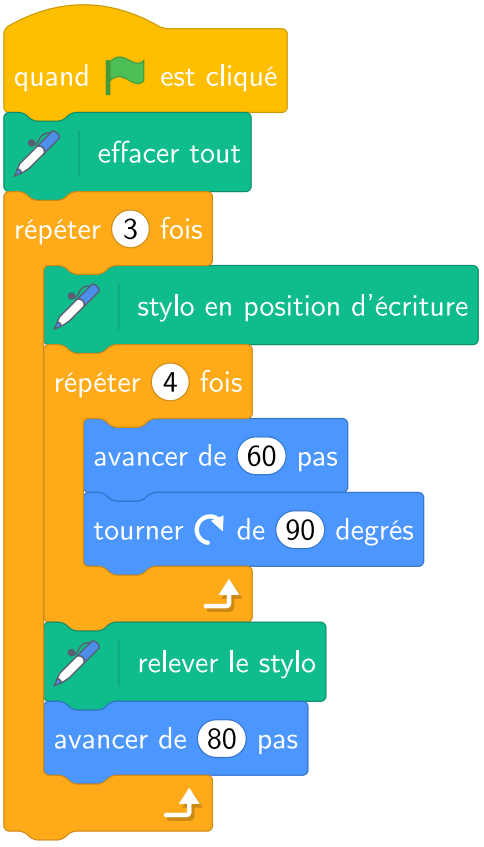
Un carré a 4 côtés et chaque rotation vaut $\frac{360}{4} = 90$ degrés. Sans la boucle, il aurait fallu empiler 8 blocs au lieu de 3.

Remarque

On peut placer une boucle à l'intérieur d'une autre : on parle alors de **boucles imbriquées**. La boucle intérieure s'exécute complètement à chaque passage de la boucle extérieure.

Exemple

Ce programme trace 3 carrés de 60 pas de côté, espacés de 80 pas :



Télécharger le projet Scratch .sb3 ?

La boucle extérieure se répète 3 fois (un tour par carré). A chaque tour, la boucle intérieure trace un carré complet, puis le lutin se déplace sans dessiner pour se placer au début du carré suivant.

5 - Variables

Variable

Une **variable** est un espace de mémoire qui porte un **nom** et qui contient une **valeur** (un nombre ou un texte). L'**affectation** consiste à donner une valeur à une variable.

Dans Scratch, le menu Variables permet de :

- **créer** une variable et choisir son nom ;
- **mettre ... à ...** : affecter une valeur à la variable ;
- **ajouter ... à ...** : augmenter la valeur de la variable.

Exemple

Ce programme crée une variable appelée **score**, lui donne la valeur 0, puis lui ajoute 7 :

Télécharger le projet Scratch .sb3

A la fin, le lutin affiche la valeur 7.

Attention

Lorsqu'on utilise « mettre … à … », l'**ancienne valeur** de la variable est **remplacée**. Si score valait 10 et que l'on écrit « mettre score à 5 », la variable contient désormais 5, et non 15. Pour ajouter à la valeur existante, il faut utiliser « ajouter … à … ».

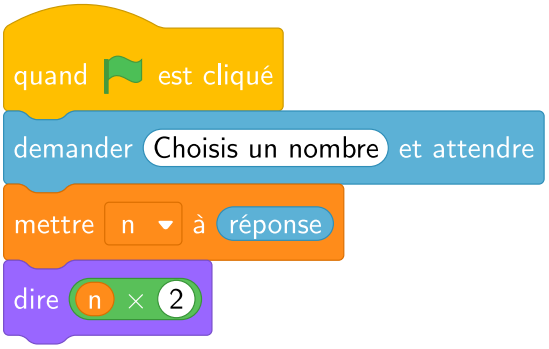
Entrée et sortie

Pour communiquer avec l'utilisateur :

- **Entrée** : le bloc « demander … et attendre » (menu Capteurs) affiche une question et range la valeur tapée dans la variable spéciale « réponse » ;
- **Sortie** : le bloc « dire … » (menu Apparence) affiche un message dans une bulle au-dessus du lutin.

Exemple

Ce programme demande un nombre et affiche son double :



Projet : double d'un nombre .sb3 ?

Si l'utilisateur entre 6, le lutin affiche $6 \times 2 = 12$.

Suivi pas à pas des variables

Pour comprendre un programme, on peut construire un **tableau de valeurs** qui note la valeur de chaque variable après chaque ligne exécutée. Par exemple :

Projet : évolution de deux variables .sb3 ?

Ligne	A	B
2	3	
3	3	5
4	8	5

A la fin, la variable A vaut $3 + 5 = 8$.

6 - Instructions conditionnelles

Instruction conditionnelle

Une **instruction conditionnelle** (ou **test**) exécute un bloc d'instructions seulement lorsqu'une condition est vraie. Il existe deux formes :

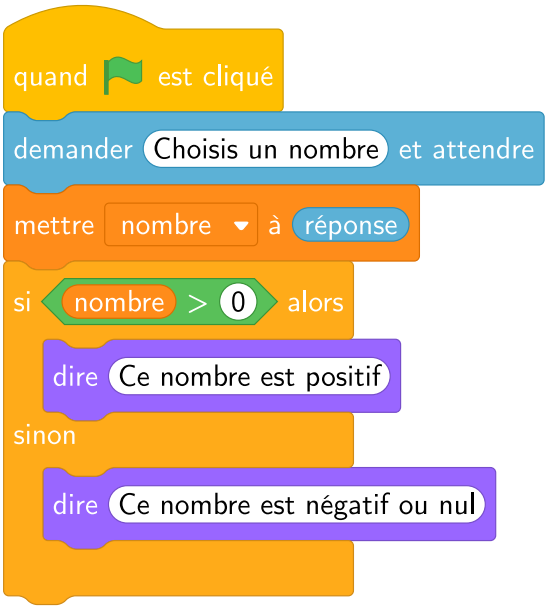
- **si ... alors** : les instructions s'exécutent uniquement si la condition est vérifiée ;
- **si ... alors ... sinon** : un premier bloc s'exécute si la condition est vraie, un autre si elle est fausse.

Remarque

Une **condition** est un test qui peut seulement répondre « vrai » ou « faux ». Les comparaisons les plus courantes sont $=$, $<$ et $>$. On peut aussi combiner deux conditions avec les opérateurs « et », « ou » ou « non ».

Exemple

Ce programme demande un nombre, puis indique s'il est positif ou non :

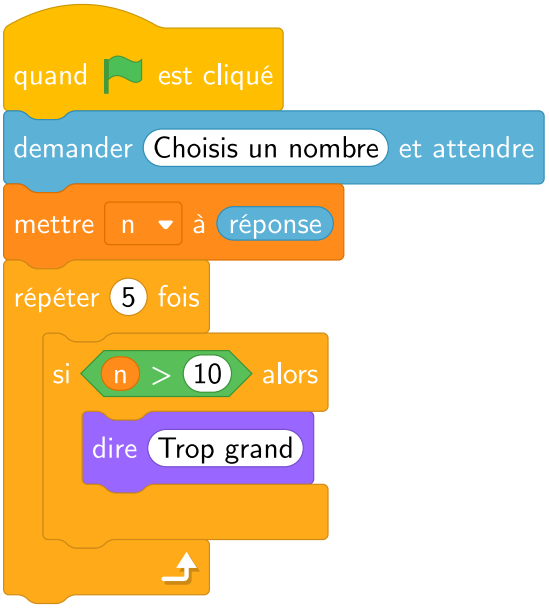


Projet : positif ou négatif .sb3 ?

Si l'utilisateur saisit 4, la condition $4 > 0$ est vraie et le lutin affiche « Ce nombre est positif ». S'il saisit -3, la condition est fausse et l'autre message s'affiche.

Boucle et condition combinées

Ce programme demande un nombre et affiche « Trop grand » tant qu'il est strictement supérieur à 10 :



Projet : afficher Trop grand .sb3 ?

La boucle passe cinq fois dans le test. A chaque passage, le message s'affiche seulement si la condition $n > 10$ est vraie.

7 - Programmes de calcul

Programme de calcul

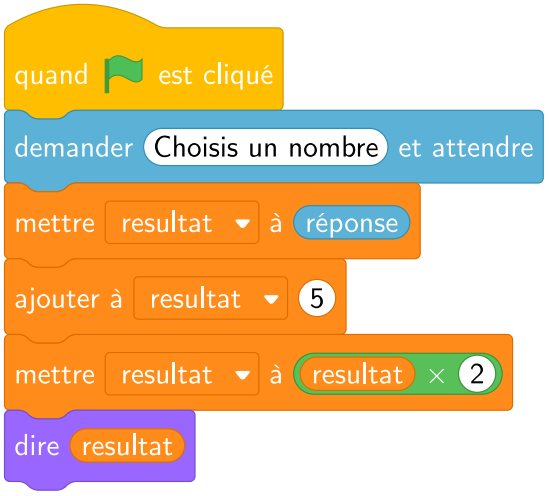
Un **programme de calcul** est une suite d'opérations appliquées, dans l'ordre, à un nombre choisi au départ. Il peut se traduire en programme Scratch.

Exemple

Soit le programme de calcul suivant :

- Choisir un nombre.
- Ajouter 5 au nombre choisi.
- Multiplier le résultat par 2.
- Annoncer le résultat final.

Sa traduction en Scratch, en utilisant une variable **resultat** :



Projet : programme (x+5) × 2 .sb3

Vérification pour le nombre 3 : $(3+5) \times 2 = 8 \times 2 = 16$. Le lutin affiche bien 16.

Remarque

Pour traduire un programme de calcul en Scratch :

- demander le nombre de départ et le ranger dans une variable ;
- appliquer chaque opération l'une après l'autre, dans l'ordre du programme ;

- afficher la valeur finale de la variable avec « dire ».

⚠ Attention

Avant toute boucle qui utilise une variable, il faut **initialiser cette variable** (lui donner une valeur de départ). Sinon, la variable conserve sa valeur précédente et le résultat peut être faux. Par exemple, une variable **somme** doit être mise à 0 avant de commencer à accumuler des nombres.

Les questions essentielles

1. Comment construire une figure avec Scratch ?

Il faut utiliser les blocs du menu Stylo (effacer tout, stylo en position d'écriture) et les blocs de mouvement (avancer, tourner). Pour un polygone régulier à n côtés, on place une boucle « répéter n fois » avec un angle de rotation de $\frac{360}{n}$ degrés.

Voir la fiche méthode : Construire une figure géométrique avec Scratch

2. Comment choisir entre recopier les blocs et utiliser une boucle ?

Dès qu'une même séquence d'instructions doit être répétée plusieurs fois, on utilise une boucle « répéter ... fois ». Le programme est plus court, plus lisible, et plus facile à modifier.

Voir la fiche méthode : Utiliser une boucle dans Scratch

3. Comment utiliser une variable dans Scratch ?

On crée la variable dans le menu Variables, on l'initialise avec « mettre ... à ... », puis on la modifie avec « ajouter ... à ... » ou en lui

affectant une nouvelle valeur. Pour suivre son évolution, on peut construire un tableau de valeurs pas à pas.

Voir la fiche méthode : Utiliser une variable dans Scratch

4. Comment choisir entre « si alors » et « si alors sinon » ?

On utilise « si ... alors » lorsqu'il n'y a rien à faire de spécial quand la condition est fausse. On utilise « si ... alors ... sinon » lorsqu'il faut traiter deux cas différents (un cas si la condition est vraie, un autre sinon).

Voir la fiche méthode : Utiliser une instruction conditionnelle dans Scratch

5. Comment traduire un programme de calcul en Scratch ?

On demande le nombre de départ avec « demander ... et attendre », on le range dans une variable, puis on applique chaque opération du programme l'une après l'autre. On affiche enfin la valeur finale avec le bloc « dire ».

Voir la fiche méthode : Traduire un programme de calcul en Scratch

Télécharger en PDF