ACADÉMIE DE ROUEN

Algorithmique et programmation

Éléments d'intégration dans nos cours de mathématiques

Pôle élargi de Compétences Disciplinaires de Mathématiques
Novembre 2016

On peut retrouver la version enrichie de ce livret sur le site académique :
http://maths.spip.ac-rouen.fr/spip.php?rubrique142.

Ce livret a été élaboré par des professeurs de mathématiques de l'académie de Rouen. Diffusé lors des journées de formation pour accompagner les nouveaux programmes du collège à l'automne 2016, il a pour objectif de proposer aux enseignants de mathématiques du cycle 4 du collège quelques pistes de travail en classe autour de la programmation avec le logiciel *Scratch*.

Comme l'indique la ressource thématique sur l'algorithmique et la programmation sur le site Éduscol (http://eduscol.education.fr/cid99696/ressources-maths-cycle.html), une séance d'apprentissage de l'algorithmique et de la programmation ne saurait se dérouler sous une forme descendante. Vous trouverez donc dans ce recueil des pistes d'activités favorisant le travail en autonomie des élèves, classées selon les intitulés : *tâches intermédiaires*, *activités avec prise d'initiative* et *projets*.

Figurent également dans ce recueil d'autres travaux conçus pour des lycéens qu'il nous a semblé pertinent de présenter dans une rubrique intitulée *liaison collège-lycée*.

À la suite de ces diverses activités, au cœur de ce livret, un chapitre particulier intitulé *repère de progressivité* est proposé. À partir d'un même thème décliné sur les quatre niveaux du collège se dessine un exemple de progressivité qui permet d'estimer quelles pourraient être les compétences exigibles, pour chaque période du collège, pour mener à son terme un projet.

Les deux derniers chapitres sont consacrés à l'évaluation : quelques idées de questions « flash » liées à *Scratch* sont suivies d'une réflexion à propos d'une évaluation sur une durée plus longue.

Comme dans le précédent livret d'avril-mai 2016, aucune des activités proposées dans ce recueil n'a l'ambition d'être un quelconque modèle des attendus en classe. L'état d'esprit prévaut sur le contenu.

Toujours à l'image des travaux présentés dans le premier livret, sont privilégiées les entrées ludiques (« *jeu* ») ou liées à un aspect plus traditionnellement mathématique.

Un tableau introductif précède chacune des activités afin d'y retrouver notamment les objectifs, les compétences travaillées ou les modalités d'organisation.

Certains des documents présentés dans ce recueil ont déjà été mis en œuvre dans des classes de lycée et de collège, y compris en éducation prioritaire. Des retours d'expériences ou des témoignages accompagnent alors les descriptifs.

L'ensemble des activités proposées contribuent à la composante « *Comprendre, s'exprimer en utilisant les langages mathématiques, scientifiques et informatiques* » du domaine 1 du socle commun de connaissances, de compétences et de cultures.

Ce livret, ainsi que ses ressources numériques (*Scratch*, …), sont disponibles sur la page « mathématiques » du site académique : http://maths.spip.ac-rouen.fr.

1. TÂCHES INTERMÉDIAIRES 6
A) Dessine-moi un mouton 6
B) Décrire un chemin 7
C) Où vais-je ? 9
D) Où suis-je ? 10
E) Tracé d'une courbe 11
F) Somme d'entiers consécutifs 12
G) Puissances entières 13
H) Construction géométrique 14
2. ACTIVITÉS AVEC PRISE D'INITIATIVE 16
A) Le crabe 16
B) Le système Terre-Lune 17
C) Le maître-nageur 19
D) Deux modifications de programmes 22
3. PROJETS 24
A) Un chronomètre à aiguilles 24
B) Un jeu d'échecs 27
C) Chiffrement et déchiffrement 29
4. LIAISON COLLÈGE - LYCÉE 34
A) Qui veut gagner des millions ? 34
B) Attrape les chiffres & Attrape-moi 38
C) Memory 43
5. REPÈRES DE PROGRESSIVITÉ 48
A) 1re étape - Un repérage dans Arles ! (activité débranchée en fin de cycle 3) 49
B) 2e étape - Une balade dans Arles ! (fin de cycle 3) 50
C) 3e étape - Une promenade dans Arles ! (début de cycle 4) 51
D) 4e étape - Une poursuite dans Arles ! (milieu de cycle 4) 52
E) 5e étape - Une traque dans Arles ! (fin de cycle 4) 53
F) Un exemple de progressivité 54
6. QUESTIONS FLASH 55
A) Lien avec la géométrie 55
B) Lien avec le repérage dans le plan 57
C) Lien avec des programmes de calcul 58
7. UNE ÉVALUATION POSSIBLE 59
Dans le cas d'un problème de construction géométrique 59

Le tableau ci-dessous reprend l'ensemble des connaissances algorithmiques rencontrées, activité par activité.

Par souci de cohérence entre ce livret et celui de la formation précédente, les notions recensées sont les mêmes dans l'un et dans l'autre.

	Tâches intermédiaires								Activités avec prise d'initiative				Démarche de projet			Liaison Collège-Lycée			Repères de progressivité				
Activité	**A**	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**A**	**B**	**C**	**D**	**A**	**B**	**C**	**A**	**B**	**C**	**A**	**B**	**C**	**D**	**E**
Mouvements	X	X	X		X			X	X	X	X	X	X	X			X			X	X	X	X
Messages													X	X	X	X	X	X					X
Événements au clavier	X		X		X				X			X	X	X	X	X						X	X
Événements souris													X	X				X					
Gestion des costumes									X	X				X		X	X	X					
Variables				X		X		X		X	X	X	X	X	X	X	X	X				X	X
Boucles							X	X	X	X	X	X	X	X	X	X	X	X			X	X	X
Instructions conditionnelles			X	X		X			X	X	X	X	X	X	X	X	X	X			X	X	X
Programmation événementielle	X		X	X	X				X		X	X	X	X	X	X	X	X			X	X	X
Programmation d'actions parallèles									X				X	X			X					X	X
Clones																							
Blocs supplémentaires								X						X			X						X

1. Tâches intermédiaires

A) Dessine-moi un mouton

Objectif	*Créer un programme par enchaînement de briques*
Connaissances travaillées et/ou mobilisées	✔ *Notion de programmation séquentielle* ✔ *Notion de programmation événementielle*
Prérequis	*Aucun.*
Modalités d'organisation	*En salle informatique*
Positionnement dans le cycle	*Début de cycle 4*
Durée estimée	*1 séance*

➢ *Objectif* : Créer un jeu qui dessine des figures.

1. Choisis un lutin, diminue sa taille pour ne pas que sa taille soit trop grande, recopie le script ci-contre et exécute ce script (clic sur drapeau vert). La brique « *attendre 1 seconde* » permet de ralentir le script. Tu peux essayer de les enlever pour voir ce qui se passe.

2. Complète ce script pour que le lutin dessine un carré de côté de taille 100.

3. Sur le même modèle, crée un nouveau script qui, quand on clique sur la touche « *2* », fait dessiner au lutin, dans le sens des aiguilles d'une montre, un carré de côté de taille 50.

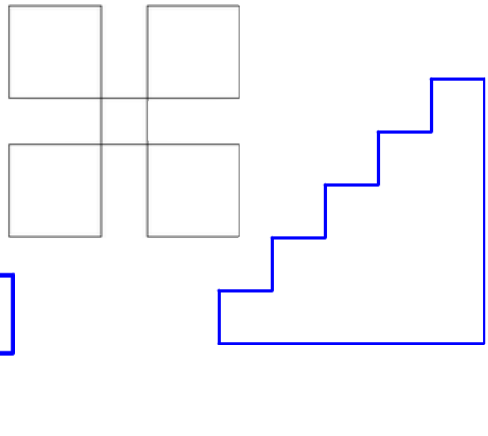
4. Sur le même modèle, crée un nouveau script qui, quand on clique sur la touche « *3* », fait dessiner au lutin, un rectangle de la taille que tu veux.

5. Sur le même modèle, crée un nouveau script qui, quand on clique sur la touche « *4* », fait dessiner au lutin, un losange de côté de taille 70.

6. Sur le même modèle, crée un nouveau script qui, quand on clique sur la touche « *5* », fait dessiner au lutin, un parallélogramme.

   Remarque de conception de l'activité : Si ces scripts (pas forcément tous) sont construits et que l'élève joue à son jeu, l'objectif initial est atteint. On peut espérer que l'image mentale sur les quadrilatères se soit encore accentuée.

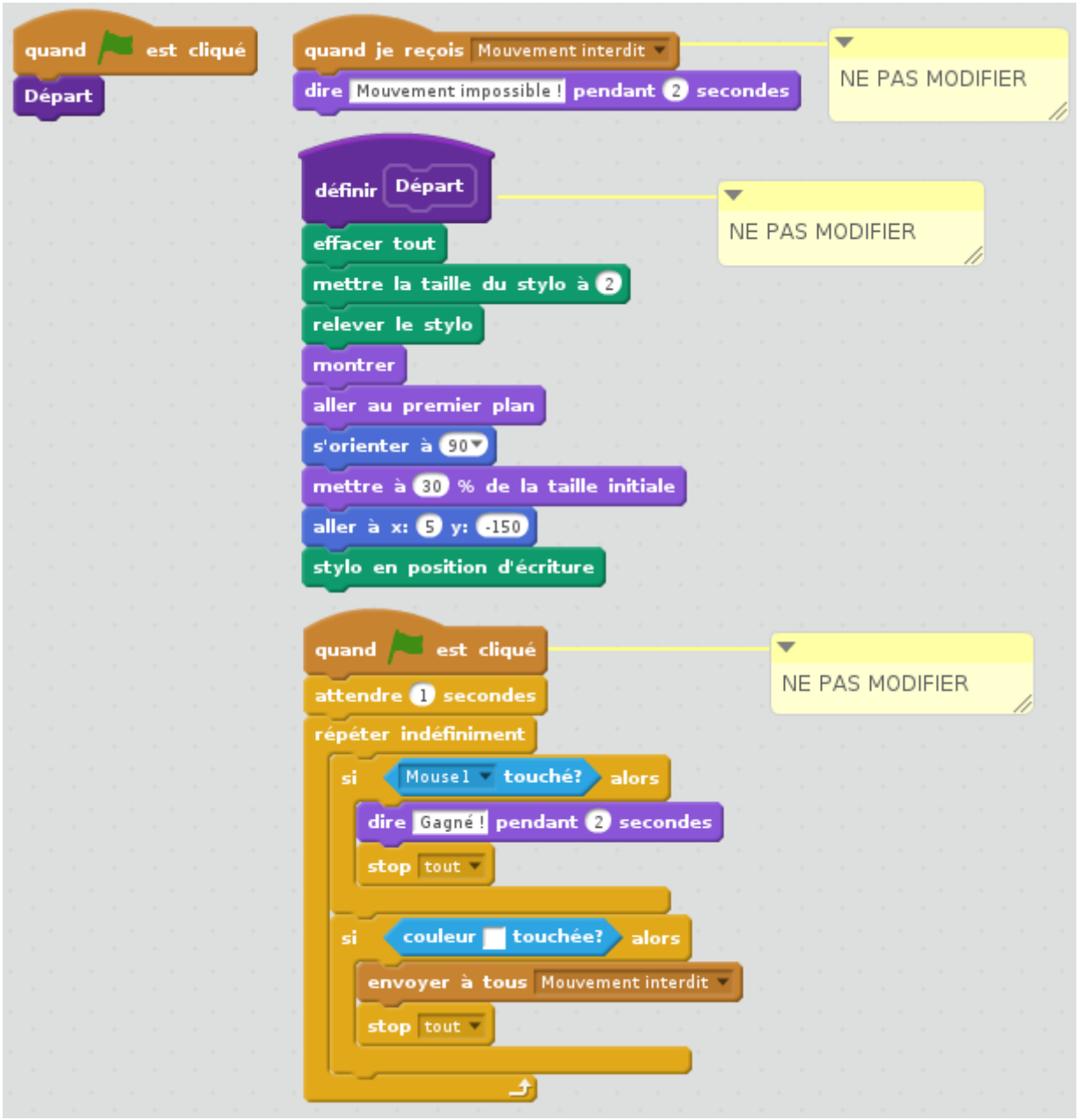
7. Grâce à la réussite de ton jeu, tu peux essayer de créer un nouveau script qui, grâce à la touche « *6* », réalise une des ces figures :



B) Décrire un chemin

Objectifs	*Se repérer, se déplacer*
Connaissance travaillée et/ou mobilisée	*Programmation séquentielle simple*
Prérequis	*Ce serait mieux d'avoir travaillé en débranché au cycle 3 sur un plan, une carte*
Modalités d'organisation	*Travail individuel en salle informatique*
Positionnement dans le cycle	*Fin de cycle 3*
Durée estimée	*2 séances*

L'objectif de cette activité est de demander aux élèves d'écrire, au sein d'un programme déjà existant, une suite d'instructions permettant d'aller d'un point à un autre. Le programme ci-dessous (avec l'arrière-plan adéquat) est fourni aux élèves.

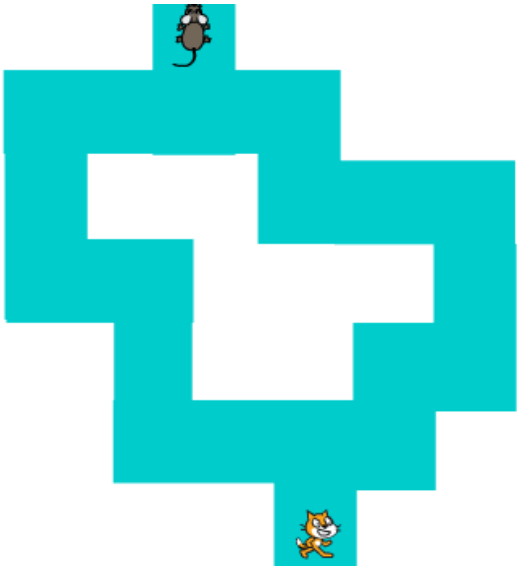


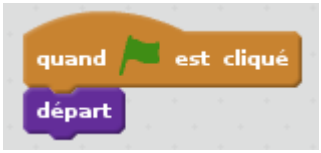
Dans le programme donné aux élèves, il y a 3 scripts à ne pas modifier :

- un bloc « *Départ* » qui correspond à l'initialisation ;
- un script qui détecte, à la fois, l'arrivée et une éventuelle détection de sortie de route ;
- Un script qui signale une éventuelle sortie de route.

Remarque : Si on utilise *Scratch* en ligne, on peut placer le programme dans « *le sac à dos* ». Ainsi, les élèves ne seront pas tentés de le modifier.

L'arrière-plan est indispensable pour faire l'activité mais le programme ne l'est pas et cette activité est facilement adaptable.





Scénario

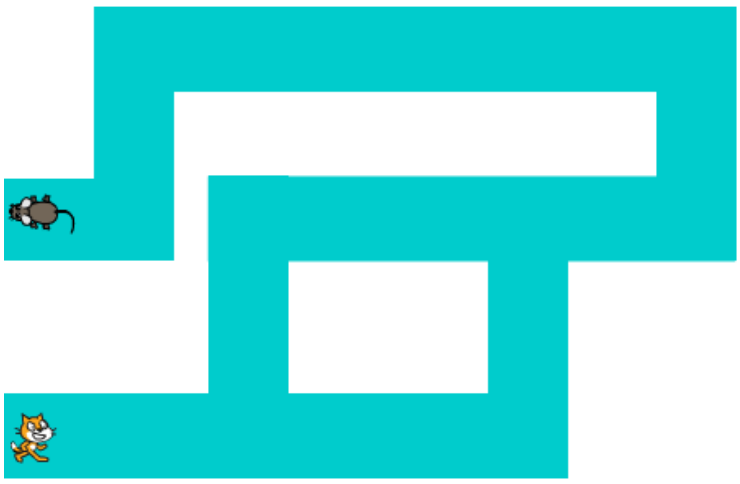
À partir du fichier fourni aux élèves, ceux-ci ont à commencer le script ci-dessus par une suite d'instructions permettant au chat de rejoindre la souris.

Chaque mouvement doit être séparé par le bloc ci-contre pour que le déplacement du chat soit bien visible.

Les mouvements autorisés du chat sont ceux des instructions ci-contre.

Les élèves peuvent maintenant tenter de répondre aux questions suivantes :

1. Quel est le trajet le plus court ?
2. Quelle est la longueur du trajet ?



Témoignages

C'est une activité qui fonctionne bien. Les élèves tâtonnent beaucoup au début. Plusieurs confondent gauche et droite.

Il faut prévoir plusieurs arrière-plans supplémentaires car certains élèves sont très rapides.

C) Où vais-je ?

Objectif	*Introduire la somme de relatifs*
Connaissances travaillées et/ou mobilisées	✔ *Mouvement de lutins* ✔ *Programmation événementielle*
Prérequis	*Avoir compris l'imbrication des instructions dans Scratch*
Modalités d'organisation	*En salle informatique*
Positionnement dans le cycle	*Début de cycle 4*
Durée estimée	*1 séance*

➢ *Objectif* : Créer un jeu qui consiste à faire attraper un lutin par un autre lutin.

1. Choisis deux lutins, l'un sera le chasseur, l'autre sera le chassé.

2. Crée un script pour le lutin chasseur qui, quand tu appuies sur la flèche « *haut* », permet à ton lutin de se déplacer vers le haut.

3. Si tu n'as pas d'idées pour créer ce script, alors avec les briques ci-dessous, à mettre dans le bon ordre, tu peux fabriquer ce script. Deux façons de faire sont proposées : privilégie la 2nde possibilité.

1re possibilité

2nde possibilité

4. Sur le même modèle, crée trois nouveaux scripts qui commandent le mouvement du lutin chasseur par l'appui des flèches « *bas* », « *gauche* » et « *droite* ».

Remarque de conception de l'activité : Si ces scripts sont construits et que l'élève joue à son jeu, l'objectif initial est atteint (gestion du mouvement d'un lutin). Si l'élève a utilisé la 2nde possibilité, on peut espérer qu'il ait assimilé qu'ajouter -10 revient bien à soustraire 10.

5. Sur chacun des scripts, rajoute un test pour savoir si le lutin chassé a été touché et si tel est le cas, fais dire « *Touché* » au lutin chasseur.

6. Tu peux améliorer ton jeu pour que ton lutin chasseur puisse passer directement du côté gauche au côté droit.

   - Pour cela, sur le script associé à la flèche « *droite* », rajoute un test pour savoir si l'abscisse est supérieure à celle du bord droit (240) et dans ce cas, mets -240 comme nouvelle abscisse au lutin.
   - Tente de faire cette amélioration pour les autres flèches.

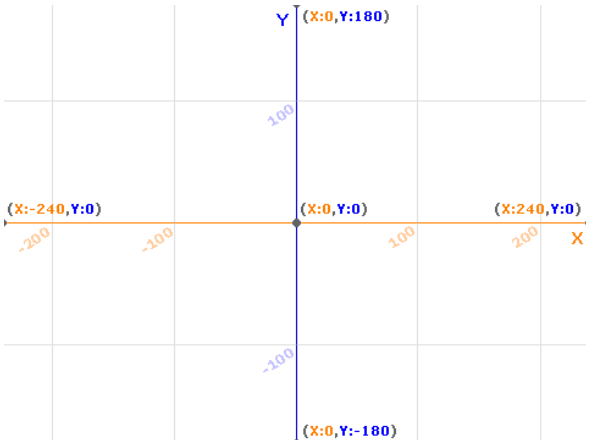
D) Où suis-je ?

Objectif	*Améliorer le repérage dans le plan*
Connaissances travaillées et/ou mobilisées	✔ *Variables prédéfinies (réponse, abscisse x)* ✔ *Conditionnelle*
Prérequis	*Avoir compris l'imbrication des instructions dans Scratch*
Modalités d'organisation	*En salle informatique*
Positionnement dans le cycle	*Début de cycle 4*
Durée estimée	*1 séance*

➢ *Objectif* : Créer un jeu qui utilise l'abscisse (et l'ordonnée) d'un lutin.

1. Tout d'abord, fais apparaître une nouvelle scène qui s'appelle « *xy-grid* ».

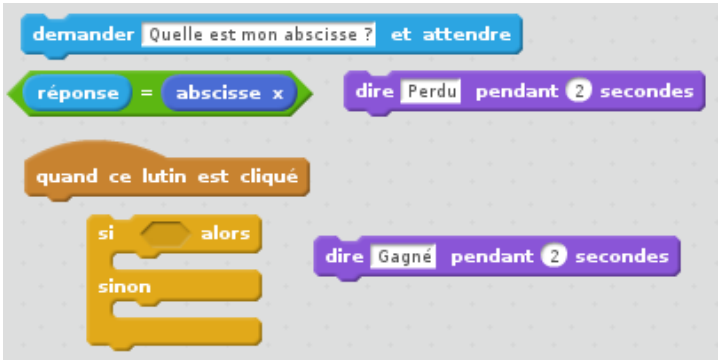
Une abscisse est désignée par la lettre « x ».
Une ordonnée est désignée par la lettre « y ».



Grâce au repère, tu peux constater que :
Les abscisses sont comprises entre ... et ... .
Les ordonnées sont comprises entre ... et ... .

2. Choisis un nouveau lutin et place-le où tu veux.

3. Crée un script réagissant ainsi, lorsque le lutin est cliqué : on demande l'abscisse du lutin et on répond « *Gagné* » ou « *Perdu* » si la bonne réponse a été trouvée ou pas.



Remarque : L'abscisse du lutin correspond à l'abscisse du centre du lutin.
Si tu n'as pas d'idées pour créer ce script, alors avec les briques ci-contre, à mettre dans le bon ordre, tu peux fabriquer ce script.

4. Sur le même modèle, crée un nouveau script qui demande l'ordonnée, cette fois quand on appuie sur la touche « *espace* ».

Remarque de conception de l'activité : Si ces scripts sont construits et que l'élève joue à son jeu, l'objectif initial est atteint : on peut espérer avoir créé une image mentale des abscisses et ordonnées (positives ou négatives) d'un point tout en ayant fait utiliser Scratch aux élèves.

Ce jeu est trop compliqué car il n'est pas facile de gagner. Tu vas essayer de l'améliorer.

5. Quand on donne une réponse, si elle n'est pas exacte, on va tester si elle est plus petite ou plus grande que l'abscisse du lutin et répondre « *trop petit* » ou « *trop grand* » selon la réponse.

6. Tu peux appliquer aussi cette amélioration aux ordonnées.

E) Tracé d'une courbe

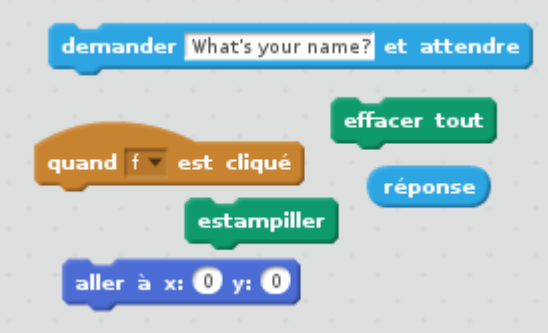
Objectif	*Tracer la courbe d'une fonction*
Connaissances travaillées et/ou mobilisées	✔ *Programmation événementielle* ✔ *Manipulation, imbrication des opérateurs (priorités opératoires)* ✔ *Antécédent, image*
Prérequis	*Aucun, si ce n'est avoir compris l'imbrication des instructions dans Scratch*
Modalités d'organisation	*En salle informatique*
Positionnement dans le cycle	*Fin de cycle 4*
Durée estimée	*1 séance*

$f(x)=3x-100$ $\quad g(x)=-\frac{1}{2}x+50$ $\quad h(x)=0,03x^2+3,75x+21$ $\quad m(x)=\frac{x^3-50x^2+2x}{100}+88$

➢ *Objectif*

En tapant sur une lettre choisie par tes soins et à partir de nombres demandés par le lutin « *Croix* », calcule leur image respective par une fonction donnée (parmi les quatre dont les expressions algébriques sont ci-dessus) et place les points associés dans un repère.

- Tu auras besoin de l'arrière-plan « *xy-grid* » et du lutin prédéfini « *Button 5* » dont tu diminueras la taille.
- Tu peux avoir besoin au moins de ces briques ci-contre.
- Il faudra penser à créer un script qui, quand on tape sur une autre lettre, efface tout.



➢ *Amélioration*

Une fois qu'il fonctionne, tu vas améliorer ce script. À chaque fois qu'est créé un point, tu vas tirer un trait entre ce point et le précédent point placé.

- Tu devrais normalement avoir un aperçu de la représentation graphique de la fonction telle que tu pourrais la faire dans ton cahier.
- À un moment, tu peux avoir besoin des briques ci-contre.

➢ *Prolongements possibles*

- En tapant sur un chiffre choisi par tes soins, à partir d'une fonction donnée (parmi les quatre dont les expressions algébriques sont ci-dessus), trace sa représentation graphique.
- Tu peux constater que comme l'espace est limité, certains points ne peuvent pas être placés car ils ont une ordonnée (et/ou une abscisse) trop grande. Ces points sont alors bloqués contre la fenêtre d'affichage. De ce fait, une ligne horizontale est tracée à tort. Ne relie que les points correctement placés.

F) Somme d'entiers consécutifs

Objectifs	✔ *Créer un programme utilisant une variable* ✔ *Limites du logiciel* ✔ *Démonstration à l'aide du calcul littéral*
Connaissances travaillées et/ou mobilisées	✔ *Notion d'algorithme et de programme* ✔ *Notion de variable informatique* ✔ *Divisibilité*
Prérequis	*Opérateurs*
Modalités d'organisation	*En salle informatique, puis mise en commun de la démonstration utilisant le calcul littéral en classe*
Positionnement dans le cycle	*Cycle 4*
Durée estimée	*1 séance*

➢ *Étape 1* : Écrire un programme qui calcule la somme de trois entiers consécutifs, en utilisant par exemple les blocs ci-contre, autant de fois que besoin.

- Pour simplifier le programme, on peut stocker les calculs intermédiaires dans une variable et procéder étape par étape.

➢ *Étape 2* : Créer une variable « *somme* » puis réécrire le premier programme à l'aide des blocs ci-contre.

➢ *Étape 3* : Modifier le programme pour qu'il teste si la somme obtenue est divisible par 3. On pourra utiliser la commande modulo qui donne le reste de la division euclidienne d'un nombre par un autre (par exemple, le reste de la division euclidienne de 42 par 5).

- Tester le programme précédent avec une dizaine de nombres différents. Quelle conjecture peut-on faire ?
- Tester le programme précédent avec un très grand nombre (un million de milliards par exemple).
- Prouver la conjecture émise.

➢ *Étape 4* : Modifier le programme pour qu'il teste si la somme de n entiers consécutifs est divisible par n.

➢ *Prolongement possible*

En algorithmique débranchée, en classe entière, faire tester la compréhension de programmes du même acabit par vidéo-projection de scripts.

G) Puissances entières

Objectifs	✔ *Créer un programme utilisant une boucle (introduction de cette notion)* ✔ *Observer les limites du logiciel* ✔ *Mobiliser la définition d'une puissance entière*
Connaissances travaillées et/ou mobilisées	✔ *Notion d'algorithme et de programme* ✔ *Boucles* ✔ *Puissances*
Prérequis	*Opérateurs, affectation de variables*
Modalités d'organisation	*En salle informatique*
Positionnement dans le cycle	*Milieu de cycle 4*
Durée estimée	*1 séance*

➢ *Étape 1* : Écrire un programme qui calcule le carré d'un nombre demandé, et son cube.

- *Scratch* ne possédant pas d'opérateur permettant de calculer directement le carré d'un nombre, ni aucune autre puissance, la création de ce programme est pertinent.

➢ *Étape 2* : Écrire un programme qui demande un nombre et calcule ce nombre à la puissance 12.

- Pour éviter d'avoir à recopier plusieurs fois un même bloc, on peut demander à *Scratch* de répéter un certain nombre de fois une même procédure, à l'aide du bloc ci-contre. On réalise ainsi ce qu'on appelle **une boucle**.

➢ *Étape 3* : Modifier le programme précédent pour qu'il demande un nombre et l'exposant auquel on veut élever ce nombre puis compléter à l'aide de Scratch :

2^7 = 5^9 = 7^8 = 15^7 =

➢ *Étape 4* : Calculer les premières puissances de 5. Quelle conjecture peut-on faire ?
Calculer 5^{30}. Que penser du résultat affiché ?
En procédant à différents essais, déterminer le nombre de chiffres exacts que le logiciel est capable de donner.

➢ *Étape 5* : Modifier le programme pour qu'il permette également le calcul de la puissance d'un nombre d'exposant négatif.

➢ *Prolongement possible en algorithmique débranchée*

Sur l'exemple ci-contre, compléter pour que les deux programmes donnent le même résultat

H) Construction géométrique

Objectif	*Introduire la notion de « nouveau bloc» ou sous-programme de commandes à partir d'une situation géométrique*
Connaissances travaillées et/ou mobilisées	✔ *Notion simple d'angle, de carré, de nombres aléatoires* ✔ *Notion d'algorithme et de programme* ✔ *Notion de sous-programme, de variable informatique* ✔ *Déclenchement d'une action par un événement, séquences d'instructions, boucles, tests*
Prérequis	*Déplacer un lutin, boucles itératives, programmation événementielle*
Modalités d'organisation	*En binôme en salle informatique (avec mise en commun lors des présentations et conclusions en classe entière)*
Positionnement dans le cycle	*Milieu de cycle 4*
Durée estimée	*1 séance*

Scénario

Les scripts des étapes 1 et 2 sont donnés ; les élèves recherchent en binôme ; ils testent avec le logiciel. L'étape 3 est alors proposée. Puis l'étape 4 . Les étapes 5 et 6 peuvent être données pour les plus rapides ou en devoir à la maison…

➢ *Étape 1* : On veut construire un carré… Chercher la (ou les) erreur(s) dans le script ci-contre.

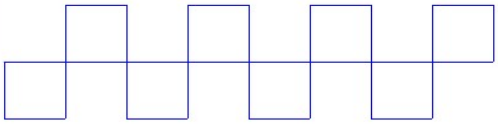
➢ *Étape 2* : Écrire le script de construction d'un carré dont la longueur du coté est 100.

- Indice : Utiliser les blocs d'instruction ci-contre.
- Vérifier le tracé.

➢ *Étape 3* : Créer un nouveau bloc.

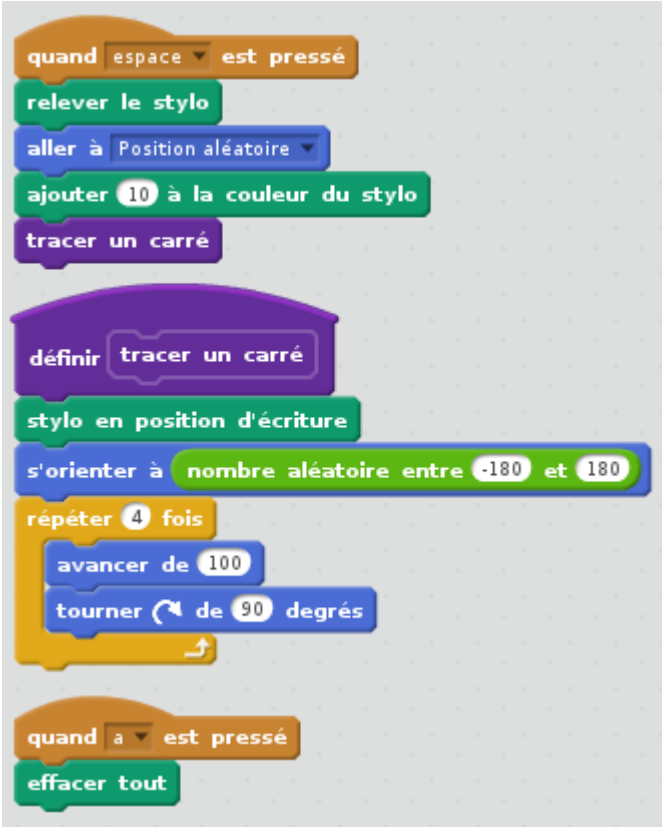
- Indice : Définir un sous-programme qui correspond à la construction d'un carré dont la longueur du côté est demandée.

➢ *Étape 4* : Créer une des deux frises.



Aide

- Expliquer le rôle de chaque bloc d'instructions du programme donné ci-contre.
- Modifier l'algorithme pour que le lutin demande la longueur du carré à construire.

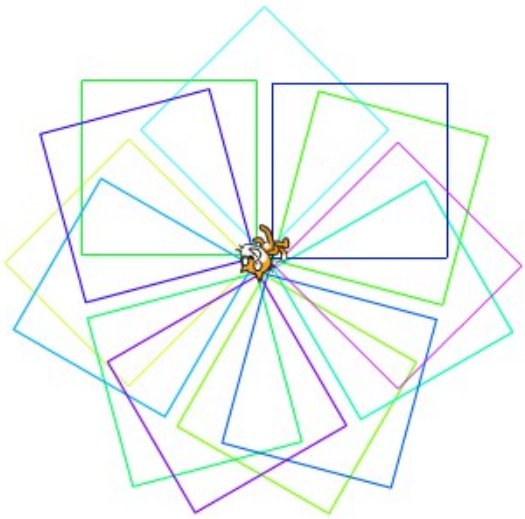


➢ *Étape 5* : Prolongements possibles

- Que doit-on modifier pour tracer des triangles équilatéraux ? Et des rectangles de longueur 100 et de largeur 50 ?
- Tracer un carré dont le côté mesure un nombre de pas qui est demandé, et orienté de façon aléatoire.

➢ *Étape 6* : Défi

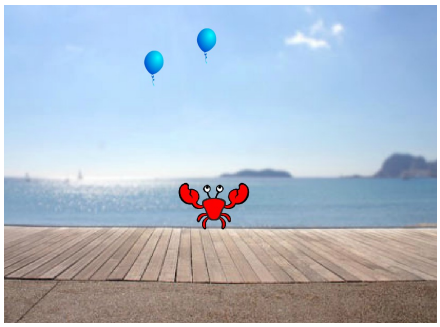
- Écrire le script correspondant à cette figure (on pourra placer le lutin dans une position aléatoire et l'orienter à une valeur aléatoire).



2. Activités avec prise d'initiative

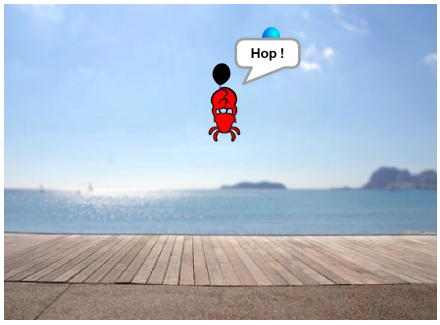
A) Le crabe

Objectif	*Programmer un jeu ayant un cahier des charges précis*
Connaissance travaillée et/ou mobilisée	*Notion d'algorithme et de programme*
Prérequis	*Connaître les différents mouvements d'un lutin, les conditionnelles et les boucles*
Modalités d'organisation	✔ *Aucune ressource particulière n'est donnée au départ* ✔ *Une fiche méthode rappelant les différents mouvements possibles est donnée aux élèves en difficulté*
Positionnement dans le cycle	*Milieu de cycle 4*
Durée estimée	*2 séances*



Scénario

La société *Infogame* vous demande d'écrire le programme d'un jeu.



Il y a trois contraintes dans ce jeu :

- un lutin, par exemple un crabe, se déplace horizontalement avec les flèches et peut sauter ;
- un autre lutin, par exemple un ballon, se déplace de manière autonome ;
- le premier lutin (le crabe) doit intercepter le deuxième lutin (le ballon).

Différents arrière-plans et différents lutins sont à votre disposition dans la bibliothèque de *Scratch*.

Prolongements possibles

1. Ajouter plusieurs ballons, un compteur de ballons, un test de fin de jeu.
2. Ajouter un chronomètre.
3. Ajouter de l'aléatoire pour le deuxième lutin.

Témoignages

Très peu d'élèves ont choisi le modèle du crabe. La très grande majorité a préféré créer son propre jeu.

Les plus rapides ont voulu ajouter des contraintes supplémentaires (chronomètre, compteur, …).

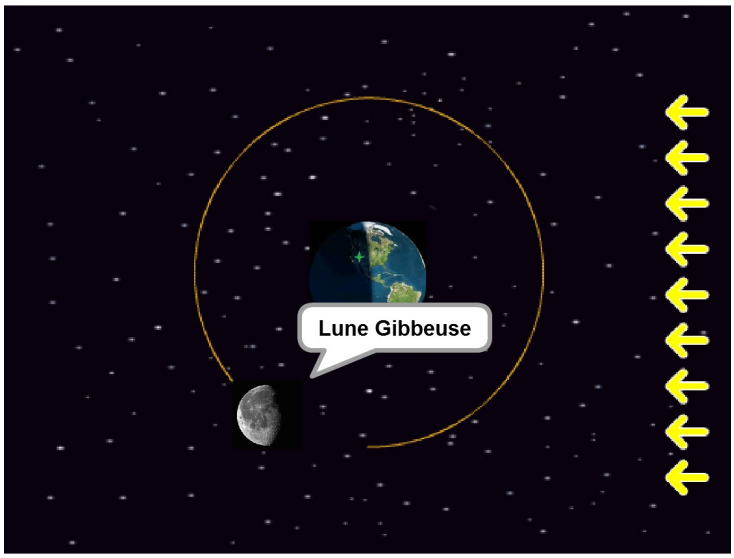
Plusieurs élèves n'ont pas su gérer le saut du premier lutin. Un script gérant la montée a donc été donné. Charge à eux de l'adapter pour gérer la descente.

B) Le système Terre-Lune

Objectif	*Créer un programme pour modéliser une situation vue en Sciences Physiques*
Connaissances travaillées et/ou mobilisées	✔ *Notion d'algorithme et de programme* ✔ *Déclenchement d'un changement de costume par un test*
Prérequis	✔ *Savoir faire un cercle avec Scratch en utilisant un polygone régulier à 360 côtés* ✔ *Connaître les conditionnelles et les boucles*
Modalités d'organisation	*En binôme en salle informatique*
Positionnement dans le cycle	*Fin de cycle 4*
Durée estimée	*2 séances*

Un lutin « *Terre* » est donné sous forme d'image. Un autre lutin, « *Lune* », est un « *gif* ». Il est composé de vingt-quatre images qui deviennent les différents costumes du lutin « *Lune* ». Le but est de modéliser la révolution de la Lune autour de la Terre.

Cette activité est un prolongement du travail qui a été fait sur les polygones réguliers. Il a été vu que construire un polygone régulier à 360 côtés était une méthode efficace pour tracer un cercle avec *Scratch*. Cette activité a été l'occasion de comparer la construction d'un cercle avec *Geogebra* et *Scratch*. *Geogebra* est plus efficace car le centre et le rayon sont donnés mais *Scratch* permet un changement de costume plus pertinent pour présenter (modéliser) les différentes phases de la Lune.



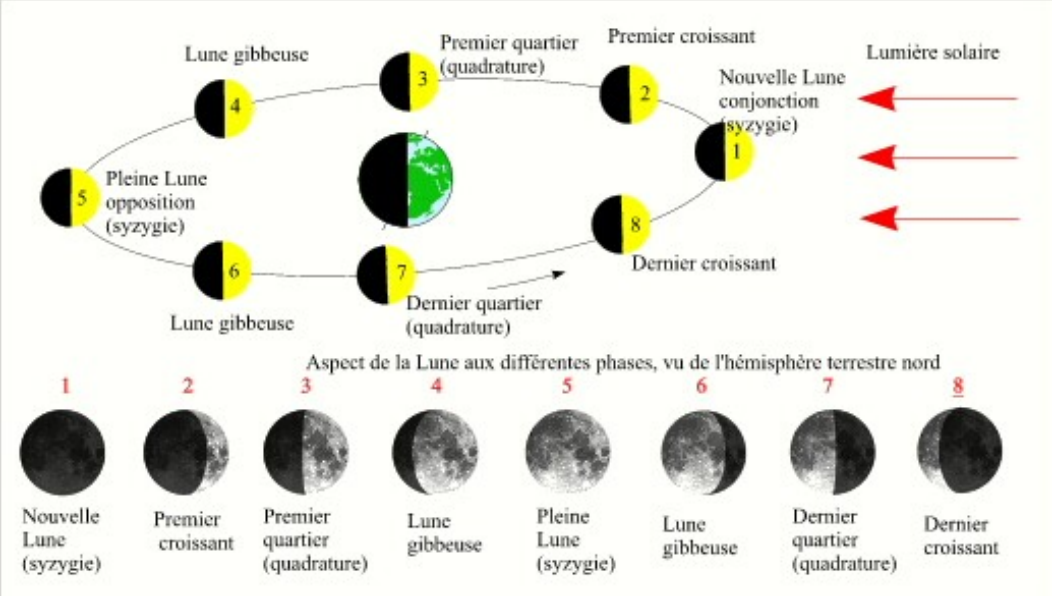
Copie d'écran du travail de deux élèves

Scénario

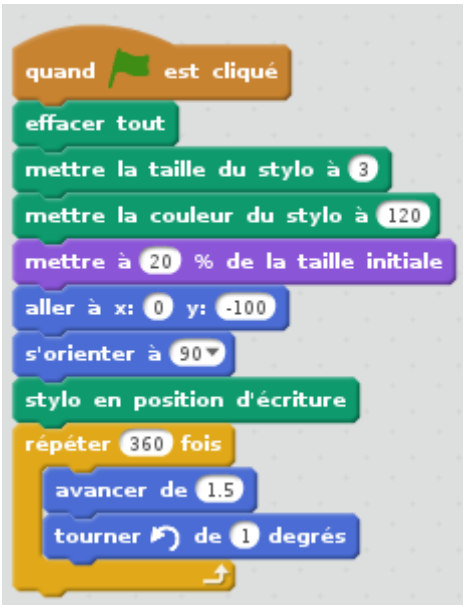
➢ *Dans un nouveau projet Scratch*
- Importer l'arrière-plan « *stars.png* » et insérer l'arrière-plan « *xy-grid* » de la bibliothèque.
- Importer les lutins suivants « *Terre.jpg* » et « *phases de la Lune.gif* ».
- Créer un algorithme simulant la rotation de la Lune autour de la Terre.

➢ *Les contraintes sont les suivantes* :
- La Terre sera centrée sur l'origine.
- Un lutin « *rayons du Soleil* » sera placé au bon endroit.
- La Lune devra tourner autour de la Terre sur un cercle de rayon 100.
- Il faudra utiliser les 24 costumes de la Lune et faire apparaître le nom des phases au bon moment.

Document 1



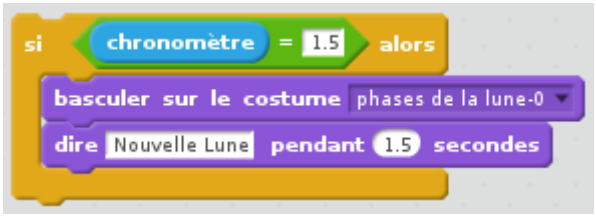
Document 2 : Le script ci-contre permet de tracer un « *cercle* » avec *Scratch*.



Témoignages

➢ *Deux difficultés sont sous-jacentes à cette activité.*

- Obtenir un cercle de rayon 100.
- Faire en sorte qu'il y ait un basculement de costume tous les 15° (360°/24 costumes).
  - Une utilisation de l'opérateur « *modulo* » avait été anticipée pour le changement de costume.
  - De nombreux élèves ont trouvé d'autres méthodes pour déclencher ce changement de costume.
  - Par exemple, certains ont utilisé le chronomètre pour mesurer le temps mis par la Lune pour effectuer une révolution (36 secondes).
    36 secondes / 24 = 1,5 seconde. Le changement de costume est déclenché par une série de conditionnelles comme ci-contre.

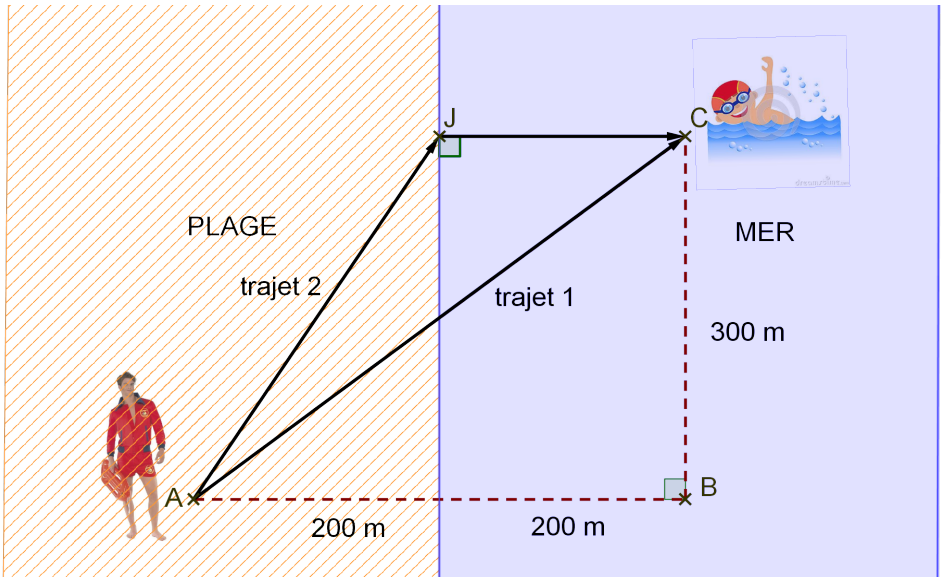


C) Le maître-nageur

Objectif	*Créer un programme pour modéliser une situation étudiée en classe lors d'une tâche avec prise d'initiative*
Connaissances travaillées et/ou mobilisées	*Théorème de Pythagore, théorème de Thalès, la notion de vitesse moyenne pour la question 1*
Prérequis	*Connaître les différents mouvements d'un lutin, les conditionnelles et les boucles pour la question 2*
Modalités d'organisation	*Travail individuel ou en binômes en classe pour la question 1 puis en salle informatique pour la question 2*
Positionnement dans le cycle	*Milieu de cycle 4*
Durée estimée	*2 séances*

Partie 1 - Tâche à prise d'initiative : le maître-nageur

En classe



Mitch, maître-nageur, sauveteur en mer, vient de repérer un nageur en difficulté.

Il hésite entre deux trajets possibles pour rejoindre le nageur.

Mitch court sur la plage à la vitesse de 5 m/s et il nage à 1,25 m/s.

Lequel des deux trajets est le plus rapide ?

➢ *Présentation de cette activité en classe entière*

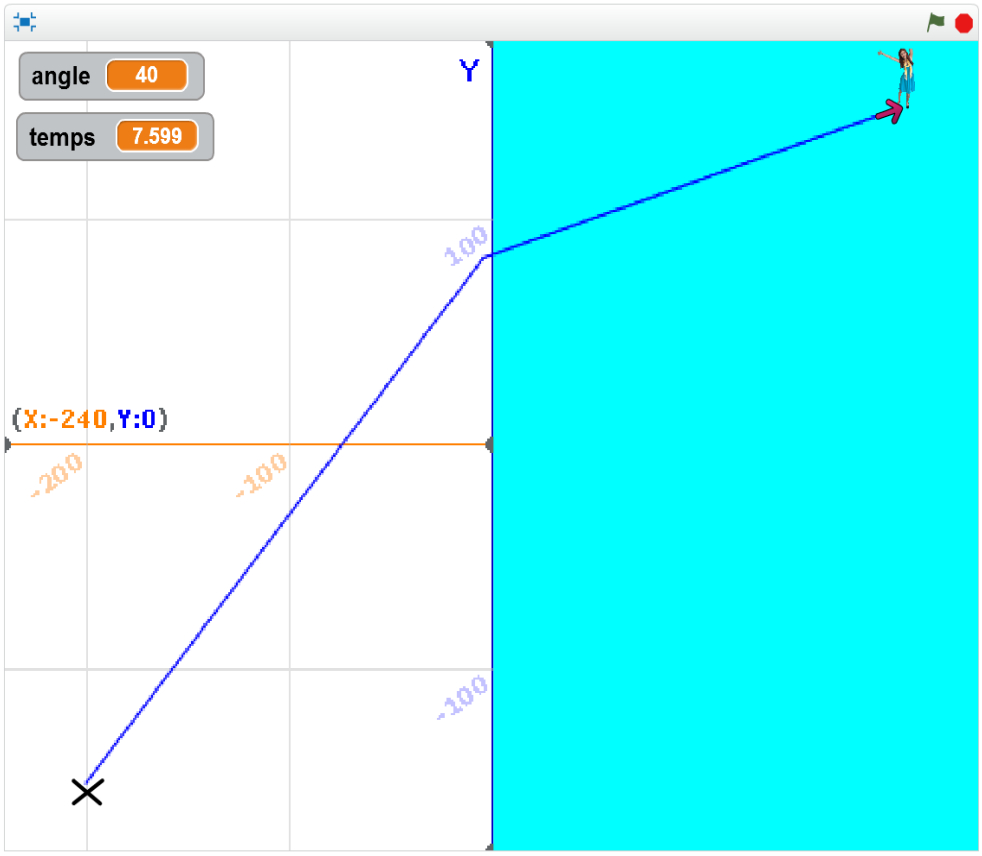
C'est une activité avec prise d'initiative utilisant la notion de vitesse moyenne, le théorème de Pythagore et le théorème de Thalès (ou théorème des milieux).

Le chemin le plus rapide n'est pas le plus court. La « ligne droite » fait perdre du temps à Mitch, ce qui déstabilise certains élèves.

Partie 2 – Modélisation avec *Scratch*

L'objectif est de modéliser la situation dans laquelle se trouve Mitch avec *Scratch* et d'utiliser le chronomètre pour trouver au moins un trajet encore plus rapide.

En salle informatique



La modélisation des lieux est relativement simple avec les coordonnées. En revanche une discussion est nécessaire pour trouver un moyen de modéliser la notion de vitesse.

Certains élèves proposent d'utiliser les blocs ci-contre pour simuler Mitch courant sur la plage et Mitch nageant dans la mer. Après un temps d'échanges, les élèves sont convaincus que cela ne donnera pas le temps réel mais que le temps donné par *Scratch* donnera un temps proportionnel à la réalité.

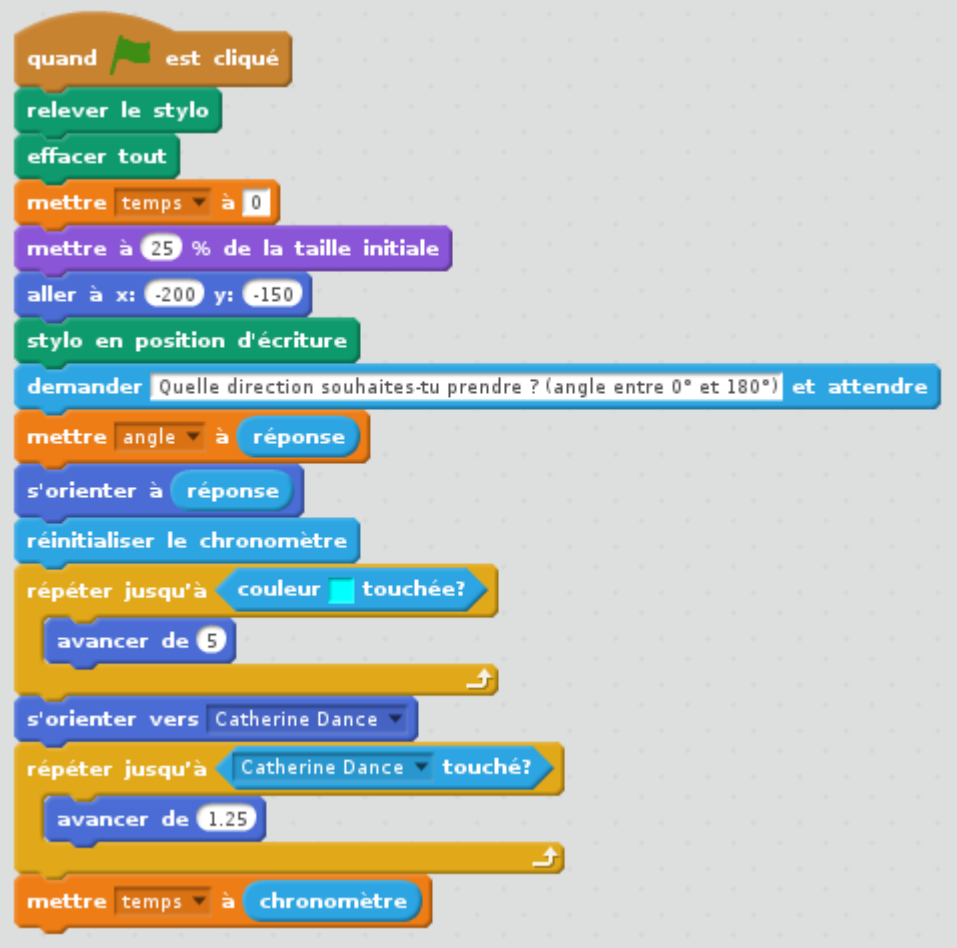
Cette situation fait intervenir des vitesses différentes dans deux milieux différents. Résoudre un tel problème de minimisation avec des moyens classiques mathématiques est hors de portée des élèves de cycle 4.

Scratch permet d'apporter des éléments de réponses.

Prolongements possibles

1. Créer une liste pour garder en mémoire les différents temps.

2. Travailler sur la proportionnalité : Le temps indiqué par le script n'est pas le temps réel (c'est une modélisation) mais il est proportionnel au temps réel mis par Mitch pour sauver le nageur. Le temps réel est 30 fois plus grand si on utilise le script ci-dessous.



D) Deux modifications de programmes

Objectifs	✔ *Le principe de cette activité est de demander aux élèves d'améliorer, d'une manière ou d'une autre, un programme qui donne « quelque chose » .* ✔ *Essayer de valoriser toute tentative de modification même non aboutie pour des élèves en mesure de formuler ce qu'ils veulent modifier et qui mettent en œuvre une stratégie pour le faire.*
Connaissances travaillées et/ou mobilisées	*Elles dépendent des propositions des élèves : les plus à l'aise sont complètement autonomes et trouvent seuls des idées ; il faut solliciter les autres en demandant et/ou proposant quelques pistes d'améliorations.*
Prérequis	✔ *Avoir déjà manipulé au moins une fois ou deux le logiciel : déplacer un lutin, boucles, structures conditionnelles, programmation événementielle* ✔ *Quelques démonstrations rapides en classe entière permettent aussi de comprendre l'esprit : « Le programme fonctionne mais il peut être modifié et rendu plus efficace, plus joli ou plus complexe... »*
Modalités d'organisation	*En binôme en salle informatique (avec mise en commun lors des présentations et conclusions en classe entière)*
Positionnement dans le cycle	*Début de cycle 4*
Durée estimée	*1 ou 2 séances*

Exemple 1 – Autour du rectangle

Témoignages

Des élèves d'une classe de 5e ont travaillé sur le script ci-contre et, suite à sa compréhension, ont proposé les évolutions suivantes :

- Changer et/ou choisir le lutin pour tracer le rectangle.
- Tracer d'une couleur différente les longueurs et les largeurs.
- Refuser de tracer le rectangle si la longueur proposée est inférieure à la largeur.
- Cliquer sur l'écran pour choisir le point de départ de la construction.
- Calculer le périmètre et/ou l'aire du rectangle.
- Demander de calculer le périmètre et/ou l'aire du rectangle et valider ou corriger la réponse donnée.

Exemple 2 – Autour du calcul mental

Témoignages

Des élèves d'une classe de 5e ont travaillé sur ce script et, suite à sa compréhension, ont proposé les évolutions suivantes.

- Faire avancer le lutin quand la réponse est bonne et reculer quand elle est fausse.
- Ajouter un chronomètre et afficher le temps mis pour donner dix bonnes réponses.
- Faire en sorte qu'au moins un des deux nombres soit supérieur ou égal à 5 pour éviter les calculs « *trop faciles* ».
- Proposer d'autres opérations que la multiplication.
- Certains élèves ont envisagé un choix aléatoire de l'opération mais sans succès.

3. Projets

A) Un chronomètre à aiguilles

Objectif	*Simuler le fonctionnement d'un chronomètre à aiguilles avec Scratch*
Contribution spécifique aux domaines du socle	*D4 - Concevoir le prototype d'une solution numérique*
Connaissances travaillées et/ou mobilisées	✔ *Notion simple de proportionnalité* ✔ *Notion simple d'angle* ✔ *Notion d'algorithme et de programme* ✔ *Notion de variable informatique* ✔ *Déclenchement d'une action par un événement, séquences d'instructions, boucles, tests*
Prérequis	*Déplacer un lutin, boucles itératives, structures conditionnelles, programmation événementielle*
Modalités d'organisation	*En binôme en salle informatique (avec mise en commun lors des présentations et conclusions en classe entière)*
Positionnement dans le cycle	*Milieu de cycle 4*
Durée estimée	*2 ou 3 séances*

Partie 1 - Un chronomètre électronique

L'objectif est de simuler, à l'aide du logiciel *Scratch*, le fonctionnement d'un chronomètre.

➢ *Présentation de cette partie en classe entière*

- Exposé de la situation-problème : concevoir tout d'abord un chronomètre électronique à l'aide de *Scratch* pour envisager ensuite un chronomètre à aiguilles.

- Si besoin, on peut fournir aux élèves des chronomètres et leur demander de lister les principales fonctionnalités d'un chronomètre (remise à zéro, start, stop, affichage des heures, minutes, secondes).

- Dans l'idée d'une introduction aux variables informatiques, le professeur peut vidéo-projeter au tableau comment créer une variable informatique et comment l'incrémenter.

- On pourra aussi, par exemple, questionner les élèves sur les structures algorithmiques qui vont être nécessaires pour simuler ce chronomètre, ainsi que le nombre de variables *a priori*. On pourra aussi les inviter à réfléchir à l'ordre dans lequel elles peuvent être créées.

- Les tâches des étapes 2 et 3 qui suivent ne sont pas nécessairement données aux élèves qui déterminent eux-mêmes les étapes de la construction de leur programme.

➢ *Construction des secondes (en binôme)*

Les binômes ont comme tâche de créer une variable qui simule le décompte des secondes.

- Les blocs ci-contre peuvent par exemple être proposés aux élèves pour construire un premier script intermédiaire qui, lorsque le drapeau est cliqué, initialise les secondes à zéro et enclenche le décompte des secondes.

- Plusieurs solutions sont envisageables selon les directions et initiatives prises par les élèves : avec un bloc « *répéter jusqu'à …* », avec un test, avec un bloc « *répéter … fois* », …

- Le professeur peut outiller les binômes pour les conduire à la réalisation de la tâche en les invitant, selon les solutions envisagées par les élèves, par exemple à :
  - s'aider d'un organigramme incomplet fourni ou à construire,
  - utiliser le bloc « *répéter jusqu'à ...* » à partir d'une documentation fournie,
  - utiliser la division euclidienne et le bloc « *modulo* ».

➢ *Construction des heures et des minutes (en binôme)*

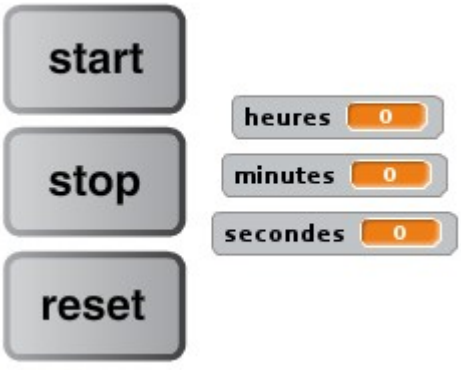
Les binômes ont comme tâche de créer les variables « *minutes* » et « *heures* » en repérant les similarités et différences avec l'incrémentation des secondes.

- Plusieurs solutions sont envisageables : un seul script ou plusieurs avec utilisation des messages (solution retenue ci-dessous).

- S'ils ne prennent pas l'initiative de le faire, on peut demander aux élèves qu'ils proposent un moyen de tester si leur décompte des heures fonctionne sans devoir attendre effectivement une heure.

- Pour gagner du temps, on pourra fournir les boutons « *start* », « *stop* » et « *reset* ». Les deux premiers sont facultatifs et peuvent être avantageusement remplacés par le Drapeau et le Stop.

➢ *Exemple de rendu final*



➢ *Bilan (classe entière) et prolongement*

- Questionnements : Qu'ont permis de réaliser les structures itératives ? À quoi ont servi les variables ?
- Récapitulation des instructions permettant d'utiliser les variables, éventuellement sous forme d'une fiche. L'activité a également permis de mettre en évidence une notion de temporalité (états successifs des cases mémoires secondes, minutes, heures).
- On peut interroger sur la précision du chronomètre créé.
- On peut envisager de demander aux différents binômes de comparer leurs solutions algorithmiques et leurs programmes. Cela peut conduire à une première approche, informelle, de la documentation d'un programme.

Partie 2 – Un chronomètre à aiguilles

Les élèves réutilisent ici le programme qu'ils ont créé lors de la précédente séance pour concevoir un chronomètre à aiguilles. Le professeur peut fournir l'arrière-plan et les lutins « *aiguilles* » déjà préparés dans un fichier *Scratch*.

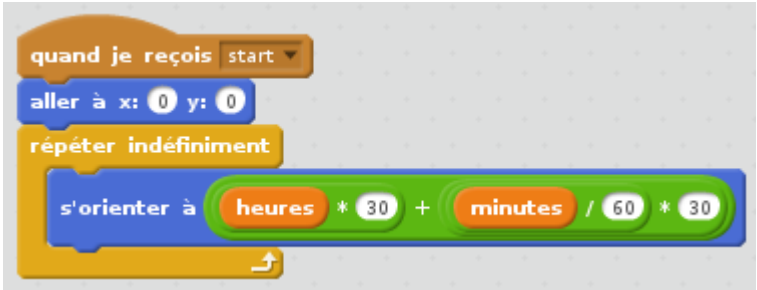
➢ *Présentation de cette partie en classe entière*

- Exposé de la situation-problème : concevoir un chronomètre à aiguilles.
- Questionnements : combien de lutins seront a priori nécessaires ? En quoi le précédent programme conçu peut être réutilisé ici ?

➢ *Rotation des aiguilles (en binôme)*

Les élèves ont comme tâche de programmer le mouvement des aiguilles.

- Par exemple, pour les secondes, voir ci-contre.
- On peut demander aux élèves d'envisager un mouvement de l'aiguille des heures un peu plus réaliste à l'aide d'un script comme ci-dessous.

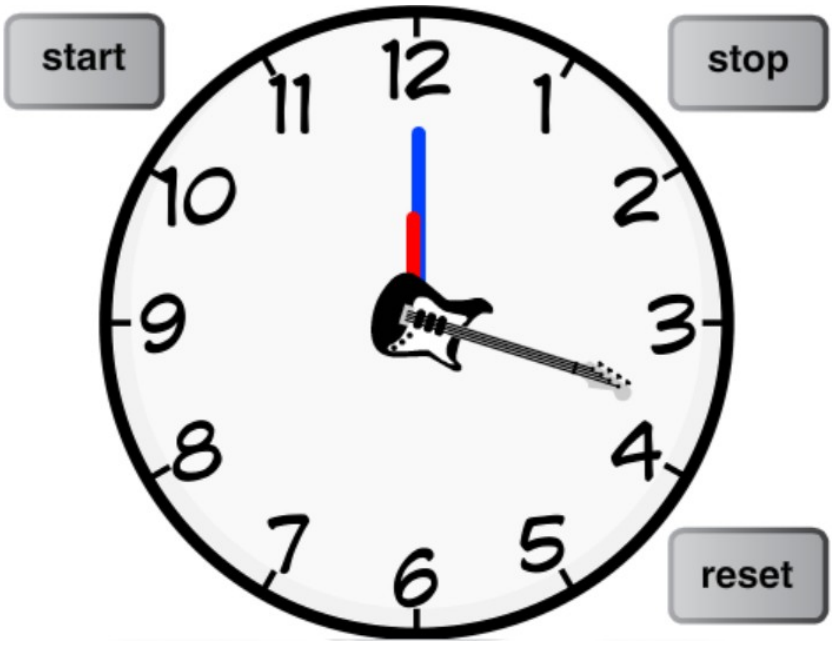


- Pour aller plus loin, on pourra faire modifier le programme pour en faire une horloge (en utilisant l'heure machine disponible dans la rubrique « *Capteurs* », ou encore un réveil).

➢ *Au-delà de la séance*

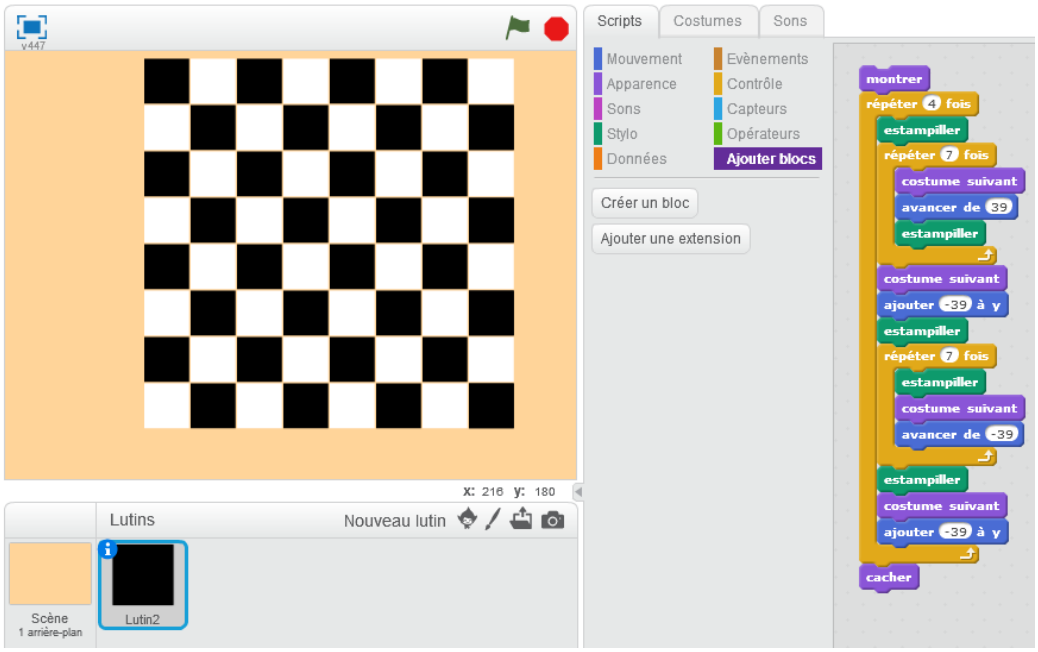
- Une recherche documentaire sur l'origine des chronomètres de marine et des horloges ainsi que leur utilité peut être envisagée.

➢ *Exemple de rendu final*



B) Un jeu d'échecs

Objectif	*Simuler un échiquier sur lequel deux élèves pourront jouer une partie en face à face*
Contribution spécifique aux domaines du socle	*D4 - Concevoir le prototype d'une solution numérique*
Connaissances travaillées et/ou mobilisées	✔ *Notion de repérage dans le plan* ✔ *Parallélisme et perpendicularité* ✔ *Symétrie axiale* ✔ *Notion d'algorithme et de programme* ✔ *Notion de variable informatique* ✔ *Déclenchement d'une action par un événement, séquences d'instructions, boucles, tests*
Prérequis	*Aucun. Pour mener à bien le projet, les élèves vont apprendre petit à petit à travailler avec Scratch*
Modalités d'organisation	*Seul ou en binôme en salle informatique selon les effectifs. Mise en commun possible*
Positionnement dans le cycle	*Fin de cycle 3*
Durée estimée	*Toute l'année (à raison d'une séance au moins tous les 15 jours) et aussi séances débranchées*



➢ *Questionnements possibles*

- Comment gérer le déplacement de chaque sorte de pièces ?
  Cela peut faire l'objet d'un travail différencié (le déplacement du roi est plus simple à envisager que celui du cavalier…).
- Comment ajouter un chronomètre comme dans les parties professionnelles ?
- Doit-on refuser des déplacements illicites ? (Prolongement pour les plus motivés)

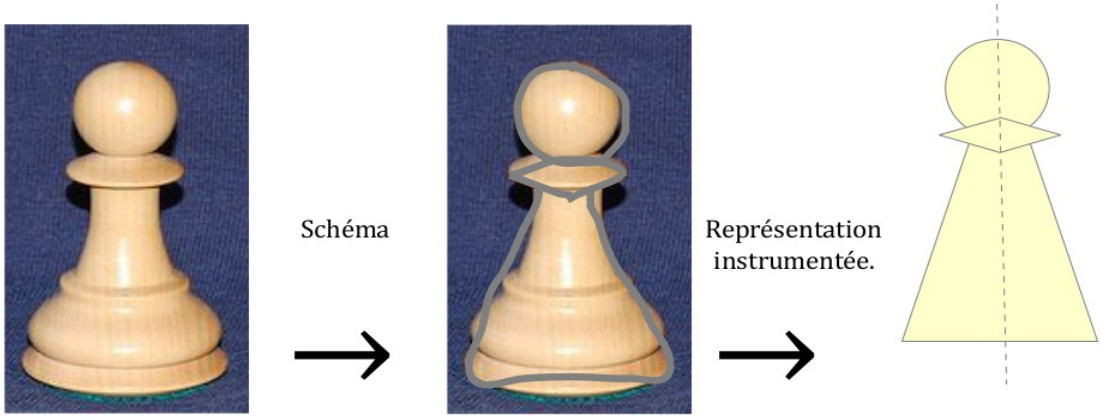
Pour chaque piste, il y a des solutions *Scratch* plus ou moins complexes à mettre en œuvre ; il convient de rester modeste dans la production finale car l'objectif n'est pas de concevoir un programme qui serait capable d'évaluer une situation d'échec et mat, par exemple.

Deux joueurs qui connaissent les règles du jeu (et qui ne trichent pas !) doivent simplement pouvoir jouer une partie avec plaisir.

Lien avec le travail « *non algorithmique* »

Les compétences mathématiques pourraient, par exemple, se travailler dans la continuité de ce projet autour de séquences consacrées à l'étude des polygones particuliers :

REPRÉSENTER : Passer d'une photo à une représentation utilisant des figures usuelles.
Construire une figure avec les instruments.
MODÉLISER : Tracer les figures avec un logiciel de géométrie dynamique (ou *Scratch*).
COMMUNIQUER : Rédiger le programme de construction correspondant à une représentation.



C) Chiffrement et déchiffrement

Objectif	*Créer deux programmes :* ✔ *L'un permettant de déchiffrer un message chiffré reçu* ✔ *L'autre permettant de chiffrer un message à envoyer*
Contributions spécifiques aux domaines du socle	*D4 – Concevoir le prototype d'une solution numérique d'un procédé utilisé depuis des centaines d'années* *D5 – Développer une conscience du temps historique*
Connaissances travaillées et/ou mobilisées	✔ *Notion d'algorithme et de programme* ✔ *Notion de variable informatique* ✔ *Notion de boucles, de tests*
Prérequis	✔ *Notion de variables* ✔ *Division euclidienne* ✔ *Addition et soustraction avec des nombres relatifs* ✔ *Expression littérale*
Modalités d'organisation	✔ *En classe entière : Présentation de l'activité, de la cryptographie* ✔ *En groupe : Tests non instrumentés en salle de classe* ✔ *En binôme : Pour programmer en salle informatique* ✔ *Tous ensemble : Pour s'échanger des messages chiffrés ou à déchiffrer*
Positionnement dans un cycle	*Début de cycle 4*
Durée estimée	*3 à 4 séances*

En classe entière, dans la salle de classe : une activité magique…

- Expliquer ce qu'est la cryptographie, ses origines, des exemples d'applications actuelles…
- Proposer les deux programmes de calculs ci-dessous.

Programme de calcul n°1

- Prendre un nombre.
- Le multiplier par 5.
- Soustraire 12.
- Si le résultat est négatif, alors :
  - Ajouter 26 au résultat.

  Sinon
  - Prendre le reste de la division euclidienne du résultat par 26.

Programme de calcul n°2

- Prendre un nombre.
- Lui ajouter 12.
- Multiplier le résultat par 21.
- Prendre le reste de la division euclidienne du résultat par 26.

- Faire appliquer ces deux programmes de calcul dans cet ordre :
  - Choisir un nombre entier positif inférieur à 26.
  - Lui appliquer le programme de calcul n°1 et conserver ce résultat.
  - Prendre ce résultat et lui appliquer le programme de calcul n°2.
  - On constate qu'on retrouve le nombre choisi au départ. On a donc deux programmes différents et pourtant le programme de calcul n°2 semble faire « *l'inverse* » du programme de calcul n°1.

Conclusion : Les mathématiques sont magiques !

- Introduire que, dans notre cas, le programme de calcul n° 1 est notre algorithme de chiffrement et le programme de calcul n° 2, notre algorithme de déchiffrement.

- On a alors pour clé publique (5 ; 12) et pour clé privée (21). Inutile dans la clé privée de préciser le 12.

- Faire alors un test avec un message.
  - Associer à chaque lettre un nombre en commençant par A=0, B=1, …

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

  - Prendre par exemple le message : « *BIENVENUE* ».

Message clair	B	I	E	N	V	E	N	U	E
Association avec les nombres	1	8	4	13	21	4	13	20	4
Après le programme 1	19	2	8	1	15	8	1	10	8
Association avec les lettres	T	C	I	B	P	I	B	K	I
Association avec les nombres	19	2	8	1	15	8	1	10	8
Après le programme 2	1	8	4	13	21	4	13	20	4
Association avec les lettres	B	I	E	N	V	E	N	U	E

Par binôme, dans la salle de classe

- Les élèves se lancent dans les calculs... On peut commencer les calculs à la main pour les 3 premières lettres et constater que les calculs sont fastidieux. Ainsi, on finit les calculs à l'aide de la calculatrice. Les élèves constateront que certaines opérations peuvent être plus rapides, mais qu'obtenir le reste de la division euclidienne reste compliqué. La calculatrice ne simplifie donc pas forcément le travail.

- Pourquoi, alors, ne pas écrire un programme sur l'ordinateur qui nous donnerait directement le résultat pour un nombre donné ?

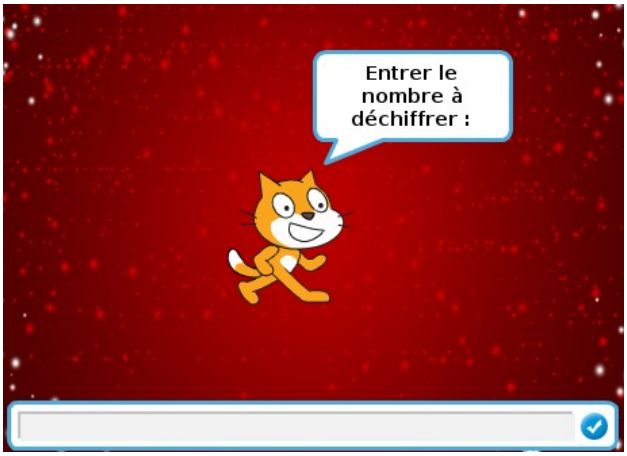
  Par exemple :
  - Pour le programme n°1 : on entre dans le programme le nombre 13 (associé à la lettre N) et le programme nous donne directement la réponse 1 (associée à la lettre B).
  - Pour le programme n°2 : on entre dans le programme le nombre 1 (associé à la lettre B) et le programme nous donne directement le réponse 13 (associée à la lettre N).

- Il est possible qu'un élève pense à écrire une unique expression littérale permettant de déterminer le résultat obtenu pour un nombre x donné après le programme de calcul 2. Dans ce cas, le professeur pourra encourager l'élève à l'utiliser en salle informatique. Il n'est pas nécessaire de l'exiger des autres élèves. Ces derniers y penseront peut-être une fois en salle informatique, *Scratch* les aura ainsi guidés vers le calcul littéral…

Remarques sur cette partie en classe

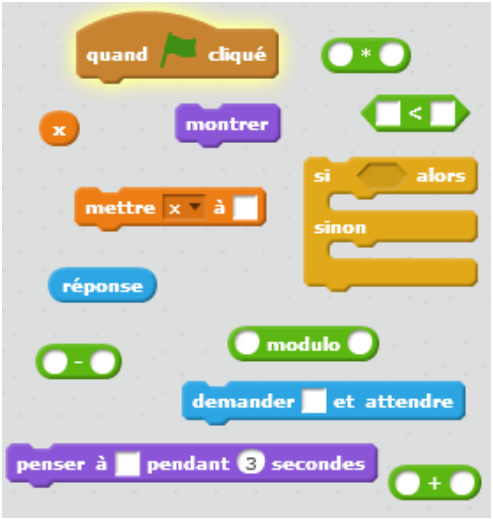
- Il est important d'avoir la condition « *Si le résultat est négatif* », dans le programme de calcul n°1, afin de ne pas avoir à faire une division euclidienne d'un nombre négatif.
- Il est possible de proposer cette activité à des élèves en fin de cycle 4 ou en 2nde avec les fonctions, ou encore à des Terminales S en utilisant les congruences…
- On peut également prendre des couples de nombres (de clés) différents de 5 et de 21. Par exemple : 7 avec 15 ou bien 11 avec 19 ou encore 17 avec 23, etc.
- De même, le nombre 12 peut être remplacé par n'importe quel nombre, à condition de le modifier à la fois dans le programme n°1 et le programme n°2.

Par binôme, en salle informatique



Algorithme de chiffrement

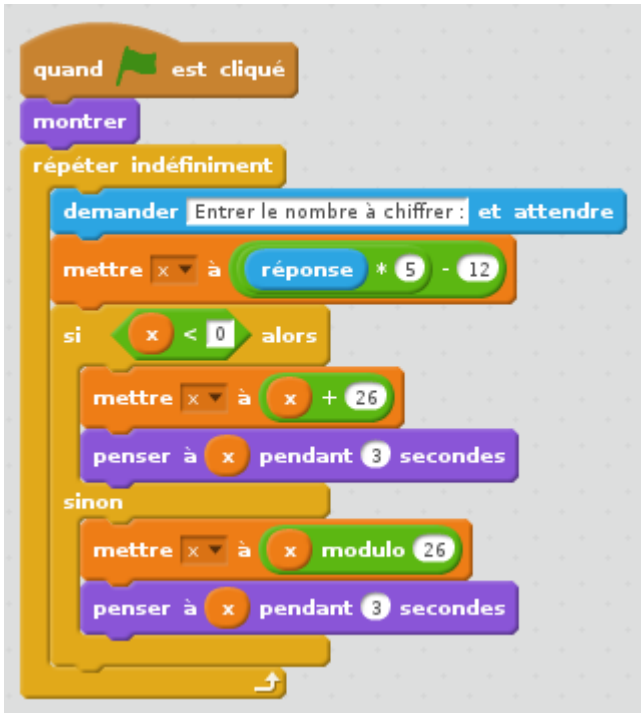
o ***Pour le calcul permettant de passer du nombre clair au nombre chiffré***

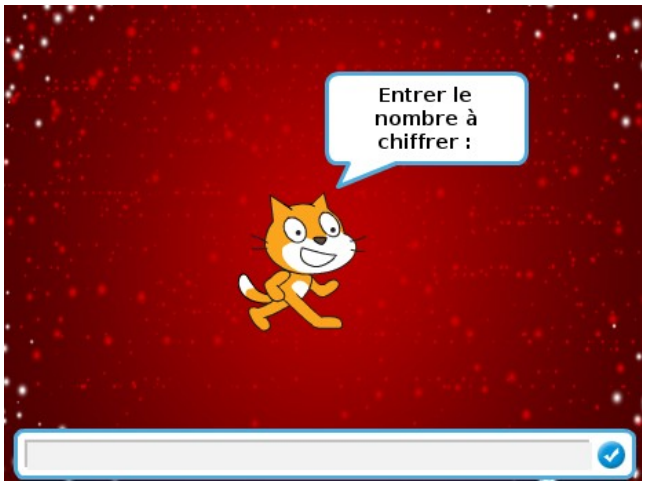


- Il faut demander le nombre à chiffrer, puis appliquer à ce nombre le programme de calcul n°1 (l'algorithme de chiffrement).
- Il sera nécessaire de créer une variable (ici x), qui aura pour valeur la « *réponse* » entrée au clavier suite à la demande.
- Pour cela, les blocs ci-contre, complétés avec les bonnes valeurs, pourront être utiles pour la réalisation du script.
- Le professeur devra indiquer l'équivalence entre le modulo et le reste de la division euclidienne.

o ***Pour ne pas être obligé de relancer le programme pour chaque nombre à chiffrer***

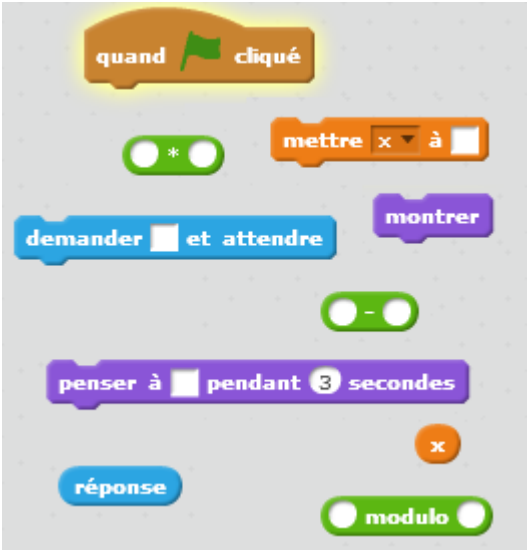
- Certains élèves pourraient avoir envie de trouver une solution afin de ne pas devoir relancer leur programme pour chaque nouveau nombre à chiffrer.
- Le professeur pourra alors les laisser chercher une solution dans le logiciel. Ils découvriront ainsi par eux-mêmes les boucles (par exemple la boucle « *répéter* »). Il faudra ensuite qu'ils la placent correctement.
- Le programme ci-contre est un exemple de solution à laquelle les élèves pourraient aboutir.





Algorithme de déchiffrement

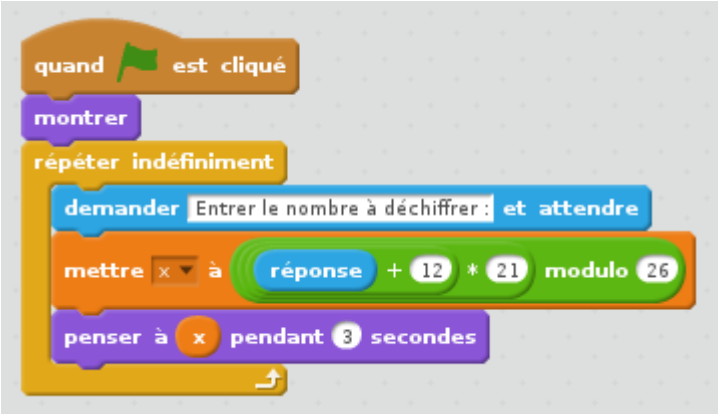
o ***Pour le calcul permettant de passer du nombre chiffré au nombre clair***



- Il faut demander le nombre à déchiffrer, puis appliquer à ce nombre le programme de calcul n°2 (l'algorithme de déchiffrement).
- Il sera nécessaire de créer une variable (ici x), qui aura pour valeur la « *réponse* » entrée au clavier suite à la demande.
- Pour cela, les blocs ci-contre, complétés avec les bonnes valeurs, pourront être utiles pour la réalisation du script.
- Le professeur devra, si besoin, de nouveau indiquer l'équivalence entre le modulo et le reste de la division euclidienne.

o ***Pour ne pas être obligé de relancer le programme pour chaque nombre à déchiffrer :***

- Certains élèves pourraient avoir envie de trouver une solution afin de ne pas devoir relancer leur programme pour chaque nouveau nombre à déchiffrer.
- Le professeur pourra alors les laisser utiliser la même solution pour ceux qui l'avaient trouvée dans le programme précédent. Et pour les autres élèves, c'est une autre nouvelle occasion de découvrir la boucle « *répéter* ».
- Le programme ci-dessous est un exemple de programme auquel les élèves pourraient aboutir.



Une fois chaque algorithme créé

Les élèves pourront échanger avec leurs camarades leurs propres messages, chiffrés au préalable avec leur algorithme de chiffrement. Les messages reçus pourront ensuite être déchiffrés à l'aide de leur algorithme de déchiffrement.

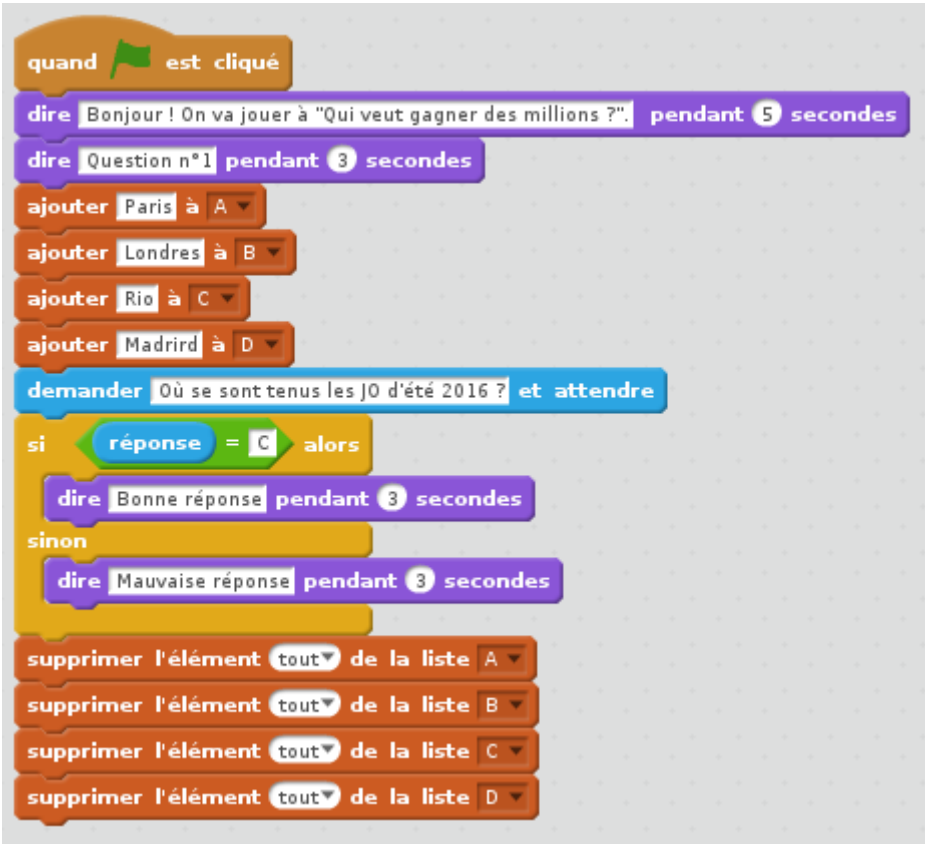
4. Liaison Collège - Lycée

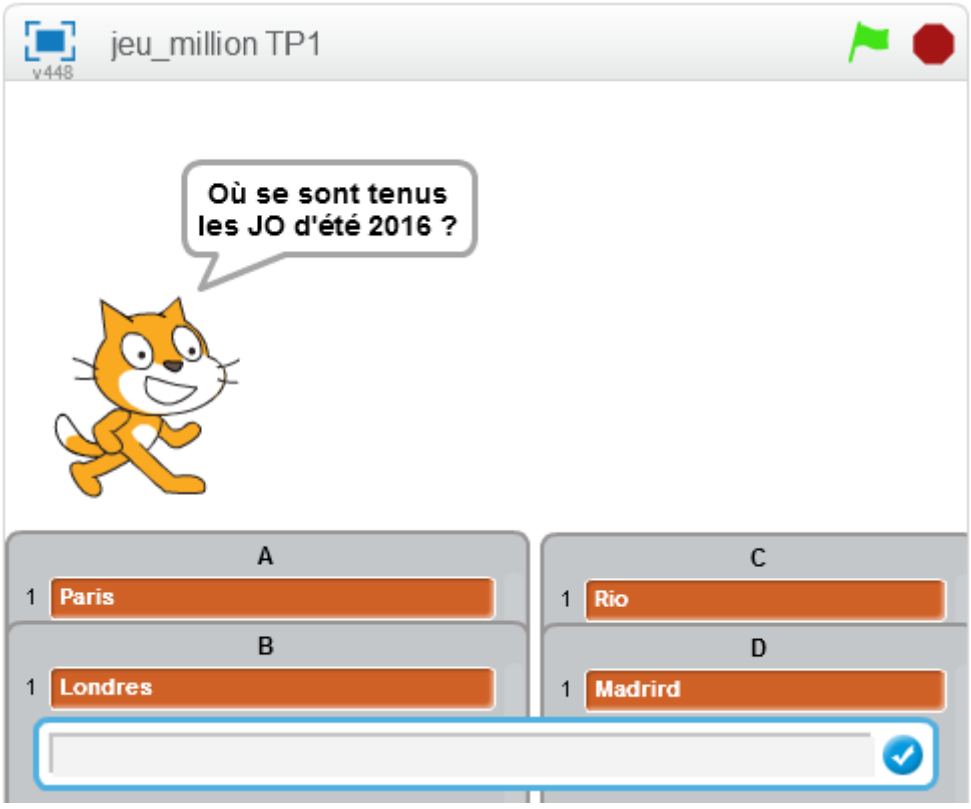
A) Qui veut gagner des millions ?

Objectif	*Créer à travers plusieurs TP un programme qui pose des questions à un joueur. Le programme final sélectionne 5 questions aléatoires parmi 1024 séries possibles de 5 questions. Le programme est capable de dire si la réponse proposée est correcte. Si tel est le cas, on passe à la question suivante (dont la difficulté s'accentue). Sinon le jeu s'arrête.*
Contribution spécifique aux domaines du socle	*D1 – Les langages pour penser et communiquer* *D4 – Les systèmes naturels et les systèmes techniques*
Connaissances travaillées et/ou mobilisées	✔ *Notion de liste (juste pour l'affichage d'une question dans un premier temps puis comme lieu de stockage par la suite)* ✔ *Notion de variable informatique* ✔ *Déclenchement d'une action par un événement, séquences d'instructions, boucles, tests*
Prérequis	*Aucun*
Modalités d'organisation	✔ *Demi-classe. Un élève par poste en salle informatique à raison de 1h30 par semaine (enseignement d'exploration ICN).* ✔ *Le projet mené par l'élève est personnel. La progression de chaque élève dans les TP qui cadrent le travail se différencie assez rapidement. Certains TP peuvent prendre jusqu'à 3 séances (voire plus au TP6 si on n'avertit pas les élèves qu'il faut préparer 20 questions). La plupart du temps, la séance commence par une copie du fichier de la séance précédente que l'on renomme et que l'on ouvre pour apporter de nouvelles modifications.*
Positionnement dans un cycle	*Enseignement d'exploration ICN (Informatique et Création Numérique) en 2nde*
Durée estimée	*Un trimestre, voire un semestre, sous la forme de 7 TP (tout dépend du détail et de la liberté apportés à la fiche élève de chaque TP et de l'investissement de l'élève)*

Le résumé de l'activité qui suit, présente une suite chronologique possible de TP pour aboutir à la réalisation du jeu.

- *TP1* : Poser une question et pouvoir répondre avec un affichage bonne réponse ou mauvaise réponse.

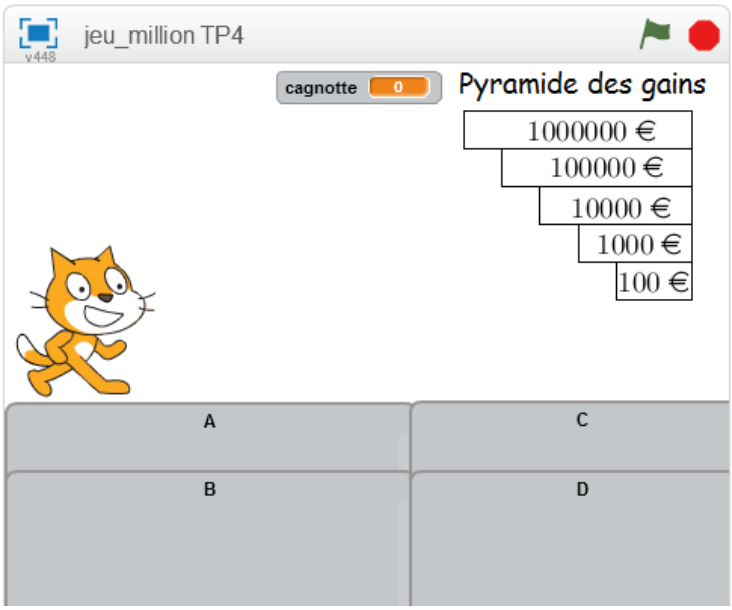




- *TP2* : Poser deux questions et pouvoir répondre (version sans boucle) avec un affichage bonne réponse ou mauvaise réponse.

- *TP3* : Poser deux questions et pouvoir répondre (version avec boucle + introduction des listes) avec un affichage bonne réponse ou mauvaise réponse.

- *TP4* : À partir du TP3 + création d'une pyramide des gains et incrémentation d'une variable cagnotte en cas de bonne réponse.



  - L'initialisation des listes (« *Question* », « *base A* », « *base B* », « *base C* », « *base D* », « *Réponse* ») reste inchangée.

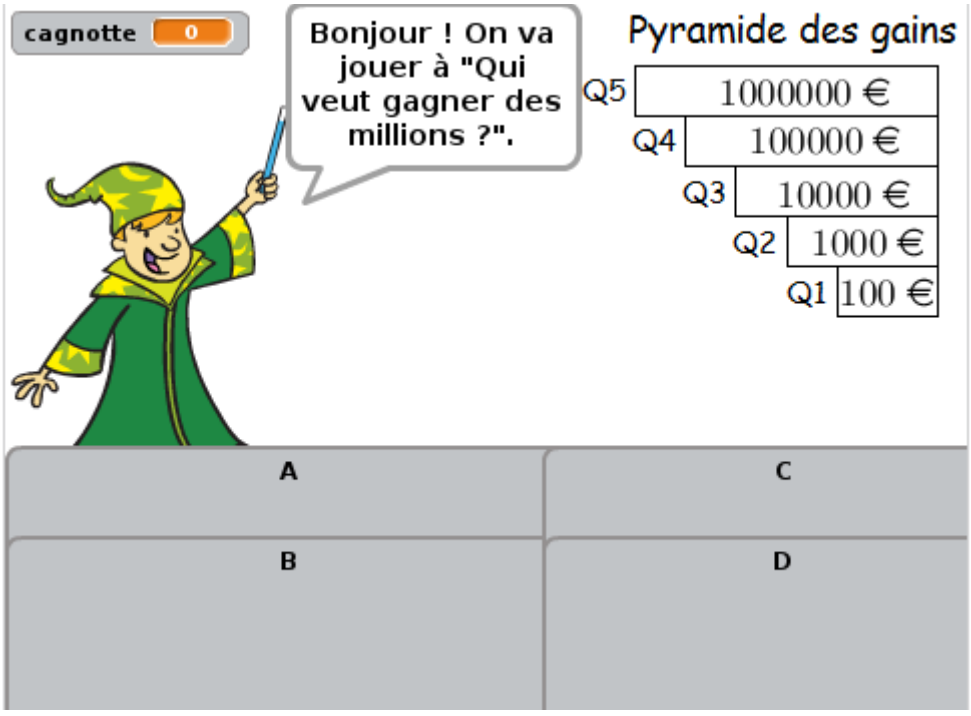
➢ *TP5* : À partir du TP4 + arrêt en cas d'erreur et message de félicitations en cas de victoire.

- L'extrait ci-contre montre le besoin de l'apparition d'un nouveau script annexe.

➢ *TP6* : À partir du TP5 mais avec 5 questions sélectionnées aléatoirement parmi 20 dans des groupes de difficulté croissante de 4 questions.

- Il suffit d'enrichir la liste « *Questions* » et de choisir aléatoirement une question dans la liste adéquate.
- On peut même insérer une musique en cas de victoire finale.
- Dans le script principal, la boucle tourne maintenant 5 fois (contre 2 auparavant).

➢ *TP7* : (facultatif) Mettre des arrière-plans (éventuellement les animer), personnaliser le lutin selon la thématique des questions, mettre des sons pour annoncer la question.



« Le jeu est la forme la plus élevée de la recherche. »
Albert EINSTEIN

B) Attrape les chiffres & Attrape-moi

Objectif	*Mettre en place la programmation événementielle par blocs.* *Il s'agit ici de faire prendre en main le logiciel Scratch pour, a priori, la première fois, au travers de deux jeux.* *Les notions rencontrées sont à réinvestir au fur et à mesure par les élèves.*
Contributions spécifiques aux domaines du socle	*D1 – Les langages pour penser et communiquer* *D4 – Les systèmes naturels et les systèmes techniques*
Connaissances travaillées et/ou mobilisées	✔ *Notion de variable informatique* ✔ *Interaction entre lutins* ✔ *Déclenchement d'une action par un événement, séquences d'instructions, boucles, tests*
Prérequis	*Aucun : prise en main du logiciel et du type de programmation*
Modalités d'organisation	*Séances en salle informatique.* *Les élèves travaillent par binôme sur un ordinateur, afin de faciliter la réalisation par leurs échanges.* *L'enseignant projette au tableau sa version, qu'il construit au fur et à mesure au travers de débats avec les élèves.*
Positionnement dans un cycle	*Enseignement d'exploration ICN (Informatique et Création Numérique) en 2nde*
Durée estimée	*Cinq séances de 1h30*

Premier jeu – Attrape les chiffres

Un chien, guidé au clavier, doit attraper le chiffre 1 apparu au hasard à l'écran.

Voici la progression prévue pour ce jeu.

- *Étape 1* : Permettre le mouvement du chien au clavier.
- *Étape 2* : Faire apparaître le chiffre 1 à une position au hasard.
- *Étape 3* : Détecter le contact entre le chien et le chiffre 1.
- *Amélioration 4* : Faire afficher successivement les chiffres de 1 à 9 pour que le chien les attrape.
- *Amélioration 5* : Gérer le chronomètre pour afficher en fin de jeu le temps réalisé.
- *Améliorations suivantes* : Suivant l'inspiration !

➢ *Étape 1* : Permettre le mouvement du chien au clavier.

- Introduction du nouveau lutin « *Chien* »  , réduit à 50 %, à la place du lutin « *Chat* » basique.

- Dans un premier temps, élaboration d'un algorithme pour déplacer le lutin « *Chien* » vers la droite et la gauche :

- Puis observation de la nécessité d'introduire le bloc d'orientation pour les mouvements verticaux :

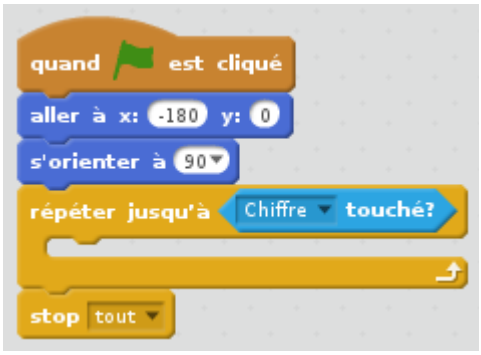
- À noter la possibilité de définir pour le lutin « *Chien* » le style de rotation à un mouvement latéral, afin d'éviter qu'il ait la tête en bas :

➢ *Étape 2* : Faire apparaître le chiffre 1 à une position au hasard.

- Introduction du nouveau lutin « *Chiffre* »  réduit à 50 %.
- Positionnement initial du lutin « *Chiffre* » à une position au hasard, à l'appui sur le drapeau vert :

- De fait, positionnement initial du lutin « *Chien* » à gauche, orienté vers la droite :

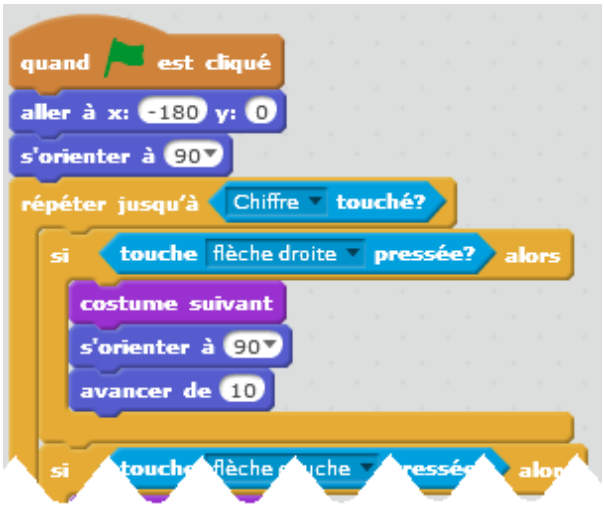
➢ *Étape 3* : Détecter le contact entre le chien et le chiffre.



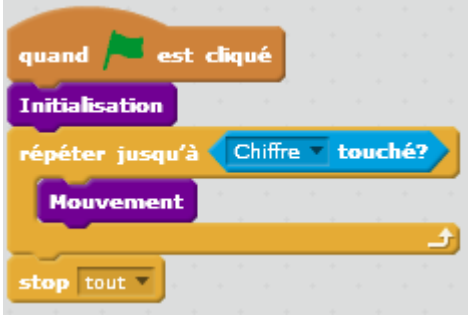
- Rajout pour le lutin « *Chien* » d'une boucle d'attente de contact avec le lutin « *Chiffre* » comme ci-contre.

- À remarquer que le jeu ne s'arrête cependant pas lors du contact bien que le drapeau vert s'éteigne ; cela est dû à la programmation en parallèle des algorithmes du lutin « *Chien* ».

- Il convient donc d'inclure dans la boucle d'attente de contact les différents mouvements comme dans l'extrait ci-contre.

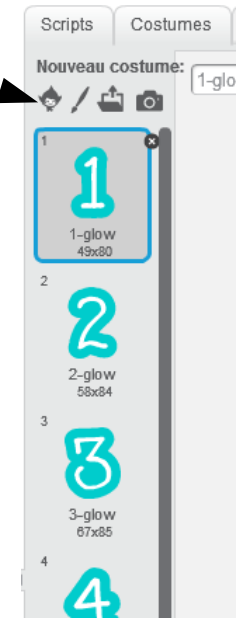


- Afin de faciliter et compartimenter la lecture de l'algorithme, on peut définir des sous-parties « *Initialisation* » et « *Mouvement* » à l'aide de la bibliothèque ***Ajouter Blocs*** comme ci-dessous.

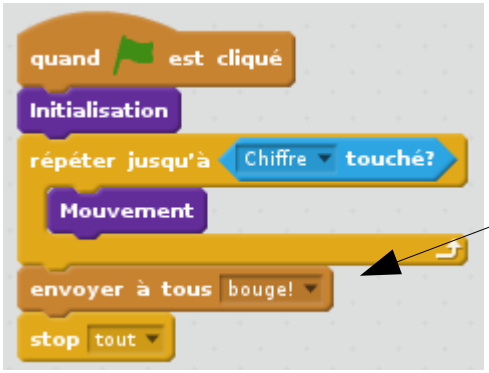


➢ *Amélioration 4* : Faire apparaître successivement les chiffres 1 à 9.

- Dans un premier temps, rajout des costumes de 2 à 9 dans le lutin « *Chiffre* », en sélectionnant ***Choisir un nouveau costume dans la bibliothèque***.

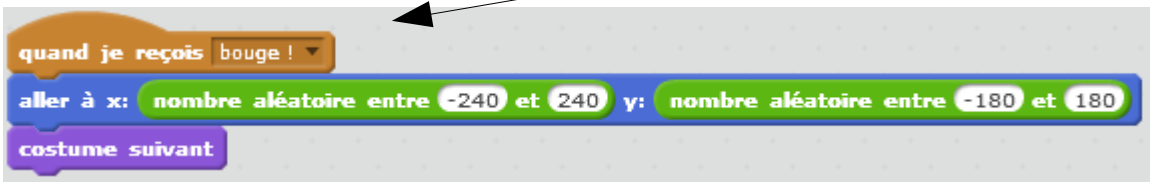


- Modification en conséquence de l'initialisation du lutin « *Chiffre* ».

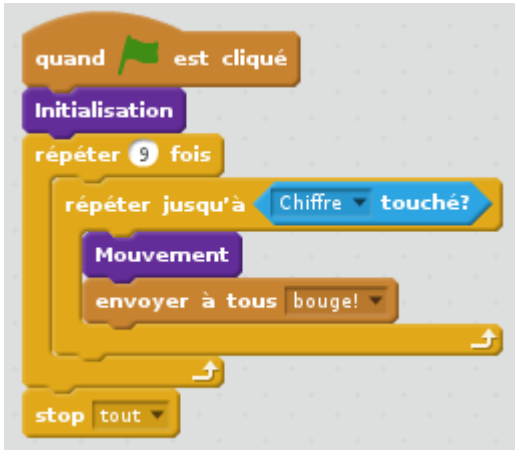


- Le contact entre les deux lutins étant détecté par le lutin « *Chien* », ajout pour celui-ci de l'émission d'un message à l'intention du lutin « *Chiffre* ».

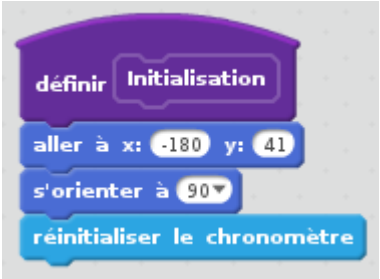
- En conséquence, gestion pour le lutin « *Chiffre* » à réception de ce message.



- Puis retour au lutin « *Chien* » pour créer une boucle permettant le défilement des neuf chiffres.



➢ *Amélioration 5* : Gérer le chronomètre pour afficher en fin de jeu le temps réalisé.



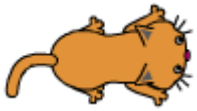
- Il s'agit, en premier lieu, de réinitialiser en début de jeu le chronomètre à zéro, au niveau du lutin « *Chien* ».

- Puis d'afficher, en fin de jeu, le temps réalisé.

Second jeu – Attrape-moi

Puisque le jeu précédent fonctionne, étendons-en le principe :

Un chat, guidé par un premier joueur, doit attraper une souris, guidée par un second joueur.



Voici la progression prévue pour ce jeu.

- *Étape 1* : Déterminer la position initiale du chat et de la souris.
- *Étape 2* : Permettre les mouvements du chat et de la souris au clavier.
- *Étape 3* : Détecter le contact entre le chat et la souris.
- *Amélioration 4* : Faire défiler successivement différentes proies pour que le chat les attrape.
- *Amélioration 5* : Gérer le chronomètre pour afficher en fin de jeu le temps réalisé.
- *Amélioration 6* : Faire afficher des scènes d'arrière-plan différentes suivant la proie.
- *Améliorations suivantes* : Suivant l'inspiration !

C) Memory

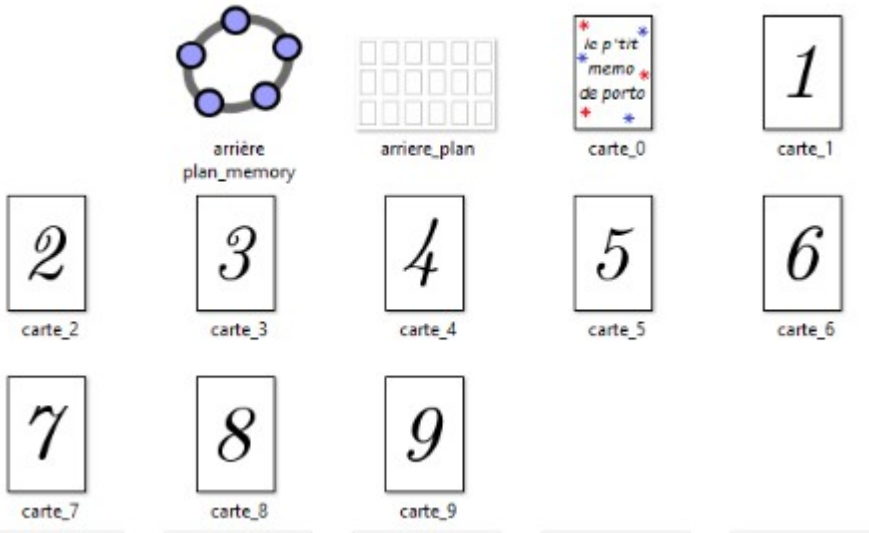
Objectif	*Créer à travers plusieurs TP un jeu Memory. C'est à la base un jeu solitaire où le but est de retrouver (retourner) toutes les paires de cartes en un minimum de coups. Le retournement des cartes se fait par 2. Si les deux cartes sont identiques, on les laisse découvertes. Dans le cas contraire, on les retourne faces cachées après une courte durée imposée (de quelques secondes). Il faut donc mémoriser ces faces cachées de façon à les retourner en un minimum de coups. Le jeu s'arrête quand il n'y a plus de carte à retourner.*
Contributions spécifiques aux domaines du socle	*D1 – Les langages pour penser et communiquer* *D4 – Les systèmes naturels et les systèmes techniques*
Connaissances travaillées et/ou mobilisées	✔ *Notion de variable informatique* ✔ *Notion de liste* ✔ *Duplication de lutin* ✔ *Interaction entre lutins* ✔ *Déclenchement d'une action par un événement, séquences d'instructions, boucles, tests*
Prérequis	*Aucun*
Modalités d'organisation	✔ *Demi-classe. Un élève par poste en salle informatique à raison de 1h30 par semaine (enseignement d'exploration ICN).* ✔ *Le projet mené par l'élève est personnel. La progression de chaque élève dans les TP qui cadrent le travail se différencie assez rapidement. Certains TP peuvent prendre plusieurs séances. Souvent, la séance commence par une copie du fichier de la séance précédente que l'on renomme et que l'on ouvre pour apporter de nouvelles modifications.*
Positionnement dans un cycle	*Enseignement d'exploration ICN (Informatique et Création Numérique) en 2nde*
Durée estimée	*Un trimestre, voire un semestre, sous la forme de 7 TP (tout dépend du détail et de la liberté apportés à la fiche élève de chaque TP et de l'investissement de l'élève)*

Le résumé de l'activité qui suit, présente une suite chronologique possible de TP pour aboutir à la réalisation du jeu.

➢ *TP1* : Mettre en place la partie graphique (création des costumes).

- Créer (sous un autre logiciel) un arrière-plan pour placer les cartes du *Memory* et créer les 10 faces de cartes (9 + le dos de carte neutre).

Les 10 cartes doivent être de même taille et seront l'ensemble des costumes pour chaque lutin. Ainsi, chaque lutin sera une duplication d'un autre. La manipulation de l'image peut se faire sur un logiciel de retouche image.



TP1 : carte_0, carte_1, carte_2, ... : ensemble des costumes de chaque lutin

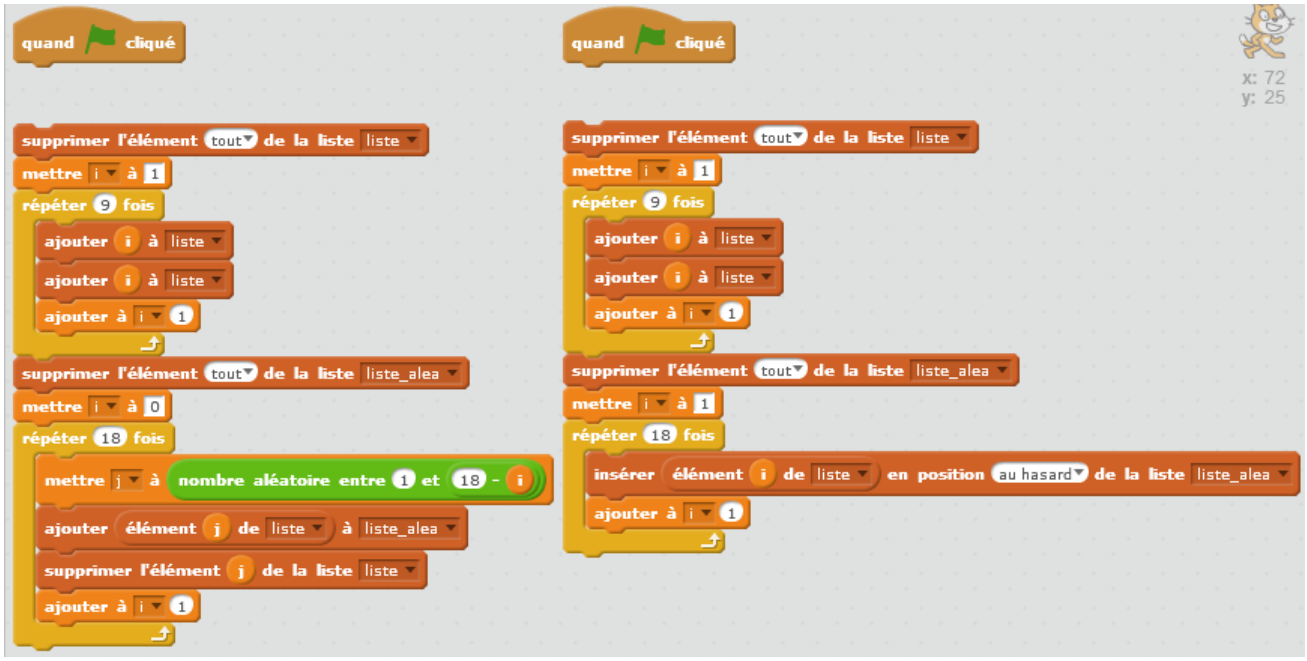
➢ *TP2* : Créer sur Scratch deux listes nommées « *liste* » et « *liste_alea* ».

- « *liste* » contient dans cet ordre les 18 éléments suivants :
  - 1 , 1 , 2 , 2 , 3 , 3 , 4 , 4 , 5 , 5 , 6 , 6 , 7 , 7 , 8 , 8 , 9 , 9.
- « *liste_alea* » contient exactement les éléments de liste mais dans un ordre aléatoire qui change à chaque exécution du programme.

Ce TP illustre encore bien le fait que sur *Scratch*, on peut arriver à ses fins de plusieurs manières :

- Création des listes manuellement, élément par élément.
- Différentes méthodes d'utilisation de boucles pour créer les listes par l'ajout automatique d'éléments.

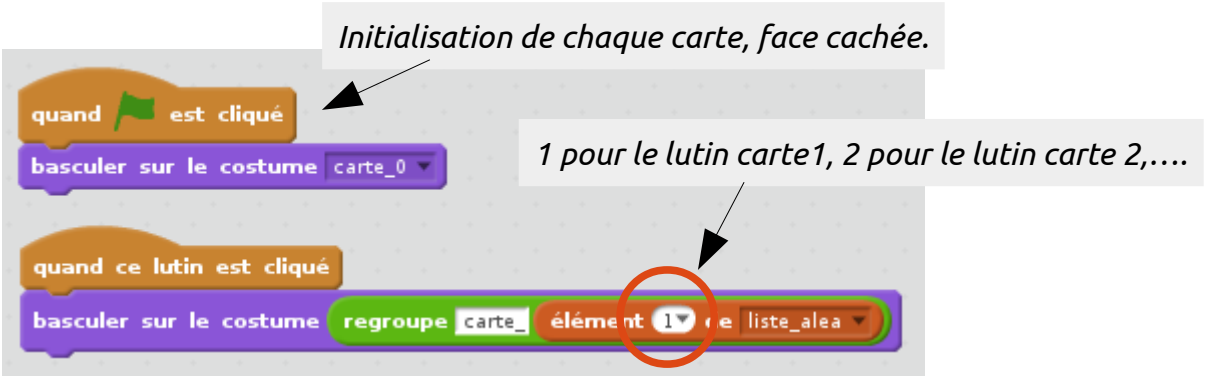
Cette création doit se faire à l'initialisation du jeu, elle est indépendante des cartes de jeu et peut donc se faire comme script dans l'arrière-plan créé au TP1. Les élèves sont guidés et accompagnés afin d'obtenir un des deux scripts suivants.



TP2 : Deux scripts différents pour réaliser la même chose : initialisation des listes

➢ *TP3* : Mettre en place des costumes de chaque carte.

- Importer une carte, la placer sur le plateau et lui associer les 10 costumes créés au TP1.
- Initialiser le programme avec les cartes face cachée, et faire se retourner une carte quand on clique dessus.
  - L'affichage de la carte doit correspondre au numéro du costume contenu en 1re position dans « *liste_alea* ».
  - Dupliquer deux fois la carte et apporter les modifications de façon à ce que la deuxième carte, une fois retournée, affiche le costume dont le numéro est en 2e position dans « *liste_alea* ».
  - Idem avec la troisième carte, pour le numéro de costume en 3e position dans « *liste_alea* »…



TP3 : Premiers scripts du lutin carte1

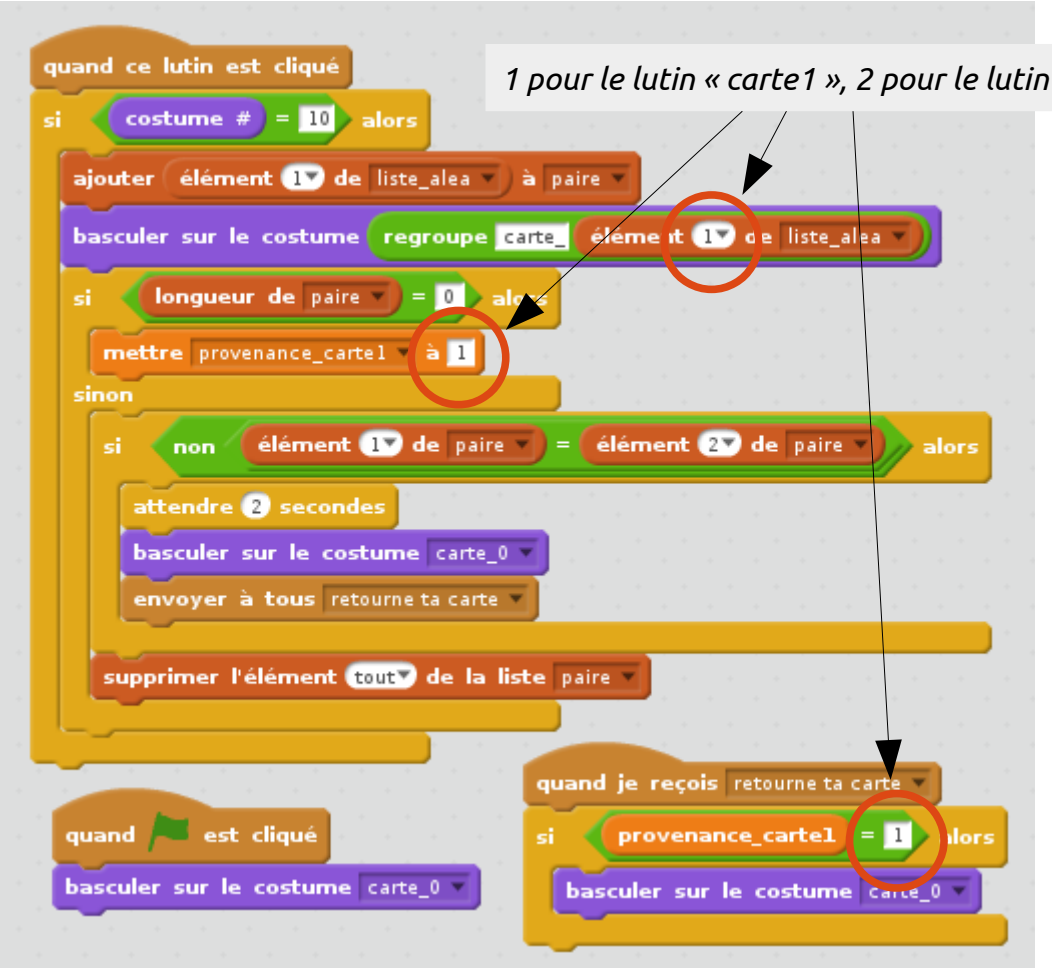
➢ *TP4* : Créer la liste « *paire* ».

- Création de la liste « *paire* » qui possédera, au maximum, deux éléments. Cette liste permet de savoir, pendant la partie, si le joueur dévoile la 1re carte d'une paire ou la 2e.

- Inclure la condition « *costume n° = 10* » dans le test permet de ne pas tenir compte des clics parasites sur une carte déjà retournée mais de se contenter uniquement du clic sur une carte face cachée. Le costume n°10 correspond au dos de la face (« *carte_0* »).

- La création de la variable « *provenance_carte1* » permet de connaître quelle carte a été dévoilée en premier. Ainsi, en cas de mauvaise paire, les cartes pourront être retournées et en cas de bonne paire, les cartes seront maintenues en l'état grâce au message « *retourne ta carte* ».

- Penser à rajouter lors de l'initialisation du jeu, la remise à zéro de la liste « *paire* ».

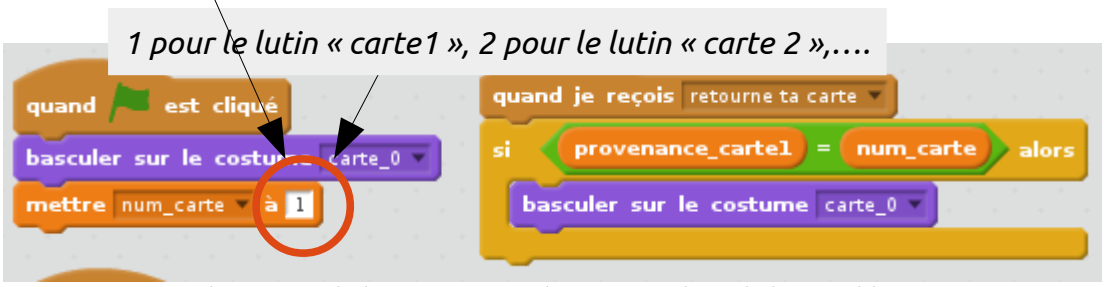


TP4 : Inclusion de la liste « paire » dans le script du lutin « carte1 »

➢ *TP5* : Créer la variable « *num_carte* ».

- On peut remarquer qu'un script quasi identique à un nombre près (reproduit plusieurs fois), existe sur chaque lutin « *carte* ».

- Pour éviter d'avoir, après duplication des scripts, à modifier le script dans chacun des lutins, on va créer une variable « *num_carte* » qui nous permettra d'avoir des scripts parfaitement identiques.
  - Cette variable « *num_carte* » doit être créée pour chaque lutin (on appelle cela une variable *locale*) et portera à chaque fois, le même nom.
  - On initialise cette variable quand le drapeau vert est cliqué.
  - On remplace alors tous les nombres cerclés dans l'image précédente, par la variable « *num_carte* ».



TP5 : Extrait des scripts du lutin carte1 après mise en place de la variable « num_carte »

➢ *TP6* : Compter le nombre de coups et modifier le délai après le choix de 2 cartes.

- Création de la variable « *nb_de_coups* » qui compte le nombre total de coups joués pendant la partie.
- Création de la variable « *temps_d'attente* » qui permet de paramétrer différemment la durée d'affichage d'un mauvais couple de carte.
- L'initialisation de ces variables se fait avec au même moment que l'initialisation des listes précédentes.

➢ *TP7* : Créer un lutin « *partie_terminée* ».

- Création d'un nouveau lutin « *partie_terminée* » qui s'affichera à la réception d'un message.
- Création d'une variable « *nb_paires_à_retourner* » qui permet à un dernier lutin « *partie_terminée* » de s'afficher quand le compteur « *nb_paires_à_retourner* » atteint 0.
- La variable « *nb_paires_à_retourner* » est initialisée à 9 au début du script de l'arrière-plan.

TP7 : Fin d'un des scripts d'un lutin carte

Prolongements possibles :

➢ *TP8* : Proposer de créer un arrière-plan avec une disposition plus originale, mettre une musique de fond.

➢ *TP9* : Programmer une version 2 joueurs.

➢ *TP10* : Programmer une version de 1 à 4 joueurs.

« Le jeu est le premier poème de l'existence. »
Jean-Paul SARTRE

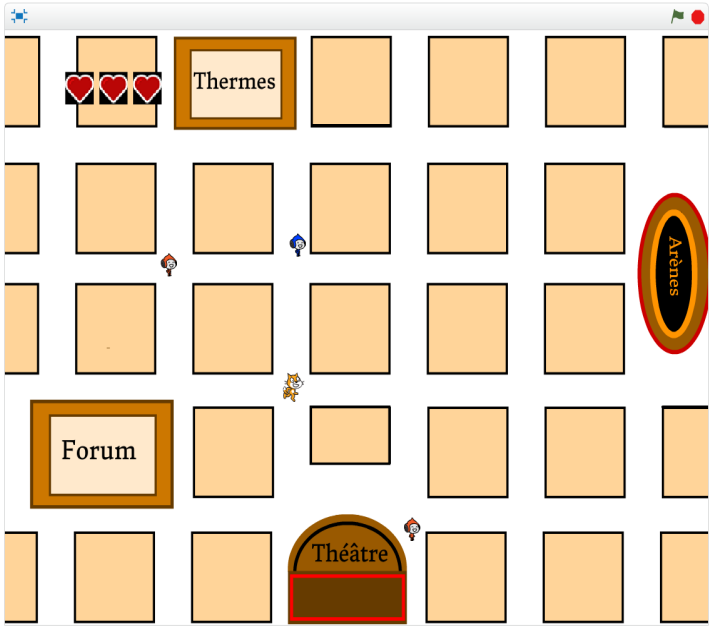
5. Repères de progressivité

Ce chapitre est issu de la réflexion suivante : « *Après quatre ans de collège, on aimerait que les élèves soient capables en 3e de programmer un jeu de type poursuite, dans une ville dont le plan est en damier. Quelles compétences doivent alors être développées et travaillées au cours du cycle 4 pour parvenir à gérer les mouvements dans un tel jeu ?* »

Dans ce qui suit, il est proposé de traiter cette problématique en se focalisant sur le thème des mouvements dans *Scratch*.

Les activités suivantes ne sont pas prévues pour être proposées telles quelles aux élèves. Elles sont juste présentées pour visualiser un même concept (aller d'un point à un autre) sur une durée de quatre ans. En utilisant un même support (Arles, son forum, son théâtre, ses thermes), on obtient une déclinaison de la notion « *mouvement* » de la 6e à la 3e.

Objectif visé après 4 années au collège : Une traque dans Arles !



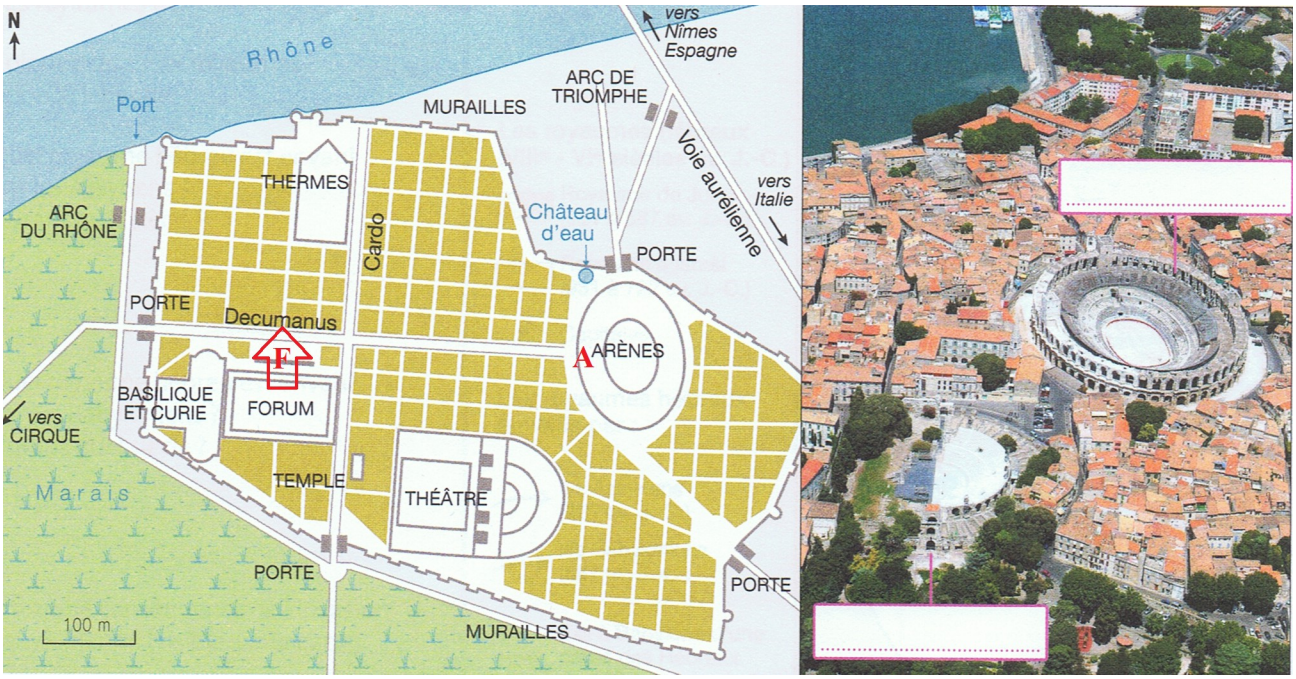
Cattus se promène dans la ville d'Arles. Il part du forum et veut aller jusqu'aux arènes. Il doit absolument passer par les thermes et le théâtre. Le problème est que trois Picos veulent l'en empêcher… Les contraintes du jeu sont les suivantes :

- Cattus est dirigé par les quatre flèches.
- Deux Picos choisissent une allée horizontale ou verticale au hasard et Cattus doit les éviter.
- Un troisième Pico tourne autour d'un bloc de maison.
- Les personnages ne peuvent se diriger que sur les rues blanches.
- Cattus dispose de trois tentatives (trois « *vies* ») pour parvenir à ses fins.
- Possibilité d'ajouter des Picos à volonté.

Pour parvenir à cet objectif, on pourrait imaginer cinq étapes pour jalonner ce cheminement. Les cinq activités suivantes commencent volontairement de manière identique et ne prétendent pas recouvrir exhaustivement le programme d'algorithmique.

A) 1re étape - Un repérage dans Arles ! (activité débranchée en fin de cycle 3)

La romanisation de l'Empire romain a été étudiée cette année en Histoire avec la ville d'Arles (Arelate, Gaule narbonaise). Nous allons nous déplacer dans cette ville au cours de cette activité.



Cattus se promène dans Arles. Il part du forum (F) et veut aller jusqu'aux arènes (A). Il doit absolument passer devant les thermes et le théâtre. A toi de donner des instructions décrivant un trajet possible.

➢ *Scénario*

1re partie :

Le travail se fait en binôme. On trace le trajet au fur et à mesure sur la carte et on le traduit par une suite d'instructions en français. On appelle cette suite d'instructions le script pour aller de F à A.

2e partie :

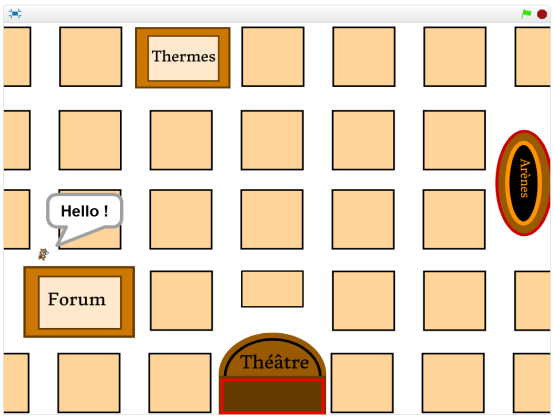
Quand tous les binômes ont écrit leurs consignes, le script est donné à un autre binôme qui doit l'appliquer sur son plan d'une autre couleur. On vérifie que la traduction sur le papier est celle attendue.

➢ *Commentaires*

On commence par de l'algorithmique débranchée : on demande aux élèves de décrire, coder ou décoder des mouvements à partir de plans ou de cartes.

Objectif : Faire émerger la nécessité d'un **langage commun** pour être compris par un logiciel comme *Scratch*. Il implique la maîtrise de codes, de règles, d'un système de signes.

B) 2e étape - Une balade dans Arles ! (fin de cycle 3)

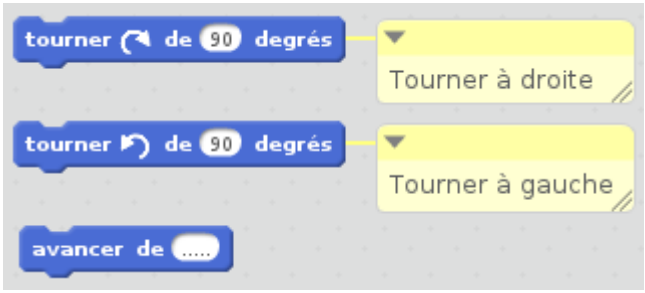


Cattus se promène dans la Arles antique. Il part du forum et veut aller jusqu'aux arènes. Il doit absolument passer devant les thermes et le théâtre. À toi de programmer son trajet.

1. Ouvre le fichier « *Arles_6eme_ELEVE* ».
Un arrière-plan avec la ville d'Arles est déjà dessiné comme ci-contre.

Commence une suite d'instructions permettant à Cattus d'effectuer son trajet.
Chaque mouvement doit être séparé par une attente de quelques secondes pour que le mouvement du chat soit bien visible.

Mouvements autorisés :



Défis pour les plus rapides :
Quelle est la longueur du trajet ?
Peux-tu trouver un trajet plus court ?

2. Enregistre ton fichier sous le nom « *Arles_6eme_ELEVE_fin*».

➢ *Commentaires*

On passe en salle informatique et on utilise *Scratch* pour programmer une suite d'instructions décrivant un trajet possible dans une ville à plan hippodamien (en damier). Seuls les changements de direction à droite ou à gauche sont possibles en 6e.

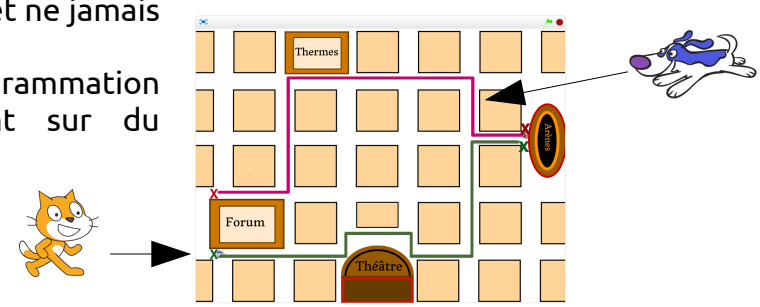
Objectifs

- Faire découvrir **la temporalité** du déroulement d'un programme. On reste sur de la **programmation séquentielle** (exécution d'une suite d'instructions dans un ordre défini).
- Comprendre l'importance de l'initialisation (orientation du lutin).

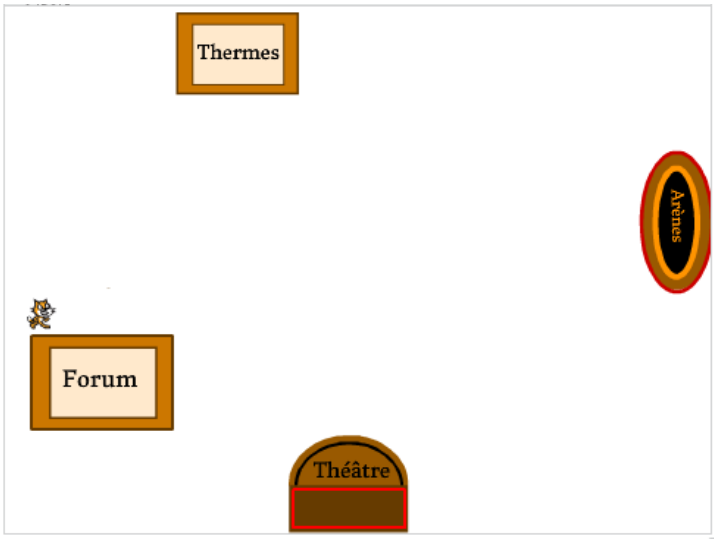
L'utilisation d'un site comme https://studio.code.org/hoc/1 complète judicieusement la démarche précédente.

➢ *Prolongement possible* : On peut reprendre le même principe avec **deux lutins**, l'un partant du forum, l'autre des arènes. Le premier doit passer par le théâtre et le deuxième par les thermes. Ils ont la contrainte suivante : ils doivent partir en même temps, arriver en même temps et ne jamais se croiser.

Objectif : Faire découvrir la programmation **en parallèle**. On reste cependant sur du **séquentiel** sur chacun des lutins.



C) 3e étape - Une promenade dans Arles ! (début de cycle 4)



Cattus se promène dans la Arles antique. Il part du forum et veut aller jusqu'aux arènes. Il doit absolument passer par les thermes et par le théâtre. Le mouvement de Cattus est géré par **les quatre flèches**.

1. Ouvre le fichier « *Arles_5ème_ELEVE* ».

Le lutin « *Cattus* » doit détecter le lutin « *thermes* » ainsi que le lutin « *théâtre* » avant de parvenir à destination.

Tu peux, par exemple, faire afficher le message « *Je suis passé par le théâtre* » quand Cattus touche le théâtre.

2. Enregistre ton fichier final sous le nom « *Arles_5ème_ELEVE_fin* ».

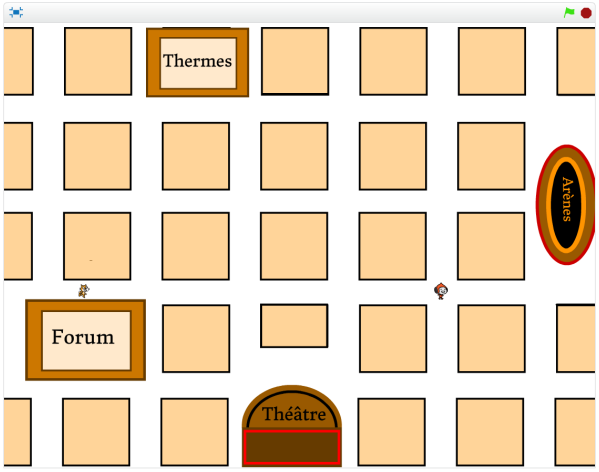
➢ Commentaires

1. Les élèves s'initient à la programmation **événementielle**.
   Objectifs
   - Faire découvrir la programmation événementielle simple. Par exemple, le programme réagit à des événements provoqués par l'utilisateur (utilisation des flèches pour déplacer un seul lutin). Ici, l'arrière-plan devient vierge (blanc) pour que les mouvements du lutin soient faciles à gérer avec les flèches.
   - Utiliser les instructions conditionnelles simples.

2. La différentiation pédagogique peut être réalisée à l'aide de défis de plus en plus compliqués.
   - Ajouter un chronomètre.
   - Comment s'assurer que Cattus est bien passé par les thermes et le théâtre ? Il peut tricher en allant directement aux arènes !

     Une possibilité : Création de deux variables « *ThermesOK* » et « *ThéâtreOK* » valant 0 ou 1.

D) 4e étape - Une poursuite dans Arles ! (milieu de cycle 4)



Cattus se promène dans la Arles antique. Il part du forum et veut aller jusqu'aux arènes. Il doit absolument passer par les thermes et le théâtre.

Problème : Pico veut l'en empêcher...

1. Ouvre le fichier « *Arles_4ème_élève* ». Un arrière-plan avec la ville d'Arles est déjà dessiné.

Cattus est dirigé par les quatre flèches.
Pico est dirigé par les touches Q, S, Z, X.

Contraintes

- Les personnages ne peuvent se diriger que sur les rues blanches.
- Cattus dispose de trois tentatives (trois « *vies* ») pour parvenir à ses fins.

2. Enregistre ton projet sous le nom « *Arles_4ème_élève_fin* ».

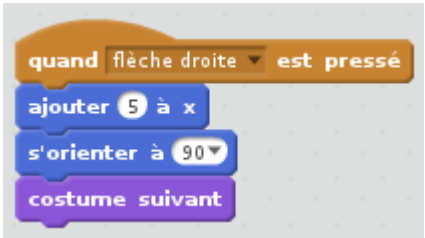
➢ *Commentaires*

1. En 4e, les élèves font de la programmation événementielle avec plusieurs lutins.
Objectif : Savoir gérer des actions en parallèle ainsi que des interactions avec des événements **intérieurs** (entre lutins) et **extérieurs** (action des flèches).

2. Le jeu ressemble à celui proposé en 5e, mais on ajoute une contrainte de type **labyrinthe** puisque l'on retrouve un plan en damier. Il faut détecter les blocs-maisons. Il faut aussi gérer un compteur de vies.

3. Pour faire bouger un personnage, on peut introduire des structures du type :

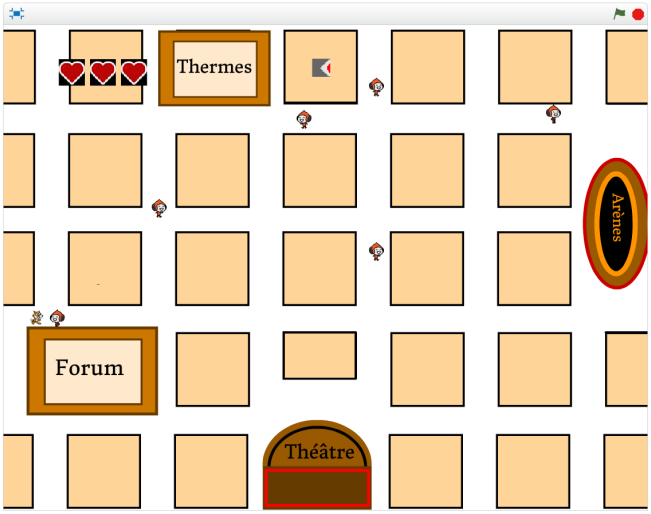
plutôt que



Le rendu de la première méthode est plus fluide quand l'algorithme est plus complexe.

4. On commence à envoyer des « *messages à tous les lutins* » du type : « *une vie en moins* ».

E) 5e étape - Une traque dans Arles ! (fin de cycle 4)



Cattus se promène dans la Arles antique. Il part du forum et veut aller jusqu'aux arènes. Il doit absolument passer par les thermes et le théâtre.

Problème : Trois picos veulent l'en empêcher...

1. Ouvre le fichier « *Arles_3ème_élève* ».
Un arrière-plan avec la ville d'Arles est déjà dessiné.

Programme un jeu avec les contraintes suivantes.

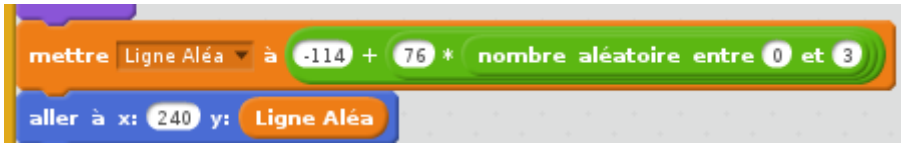
Contraintes

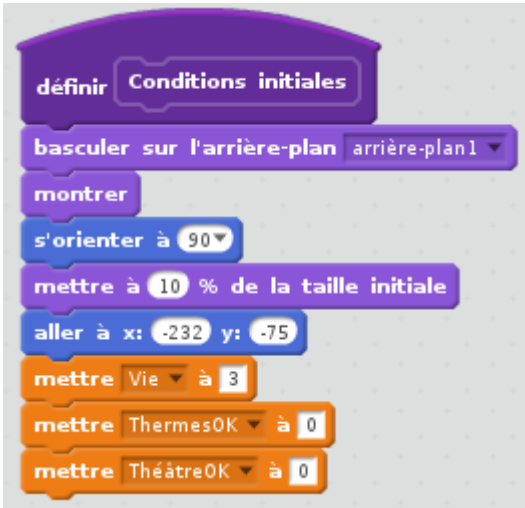
- Cattus est dirigé par les quatre flèches.
- Deux Picos choisissent une allée horizontale ou verticale au hasard et Cattus doit les éviter.
- Un troisième Pico tourne autour d'un bloc de maison.
- Les personnages ne peuvent se diriger que sur les rues blanches.
- Cattus dispose de trois tentatives (trois « *vies* ») pour parvenir à ses fins.
- Possibilité d'ajouter des Picos à volonté.

2. Enregistre ton fichier sous le nom « *Arles_3ème_élève_fin* ».

➢ *Commentaires*

- En 3e, les élèves font de la programmation événementielle avec plusieurs lutins.
- On ajoute des blocs.
- On ajoute des variables avec des **instructions d'affectation** plus complexes.



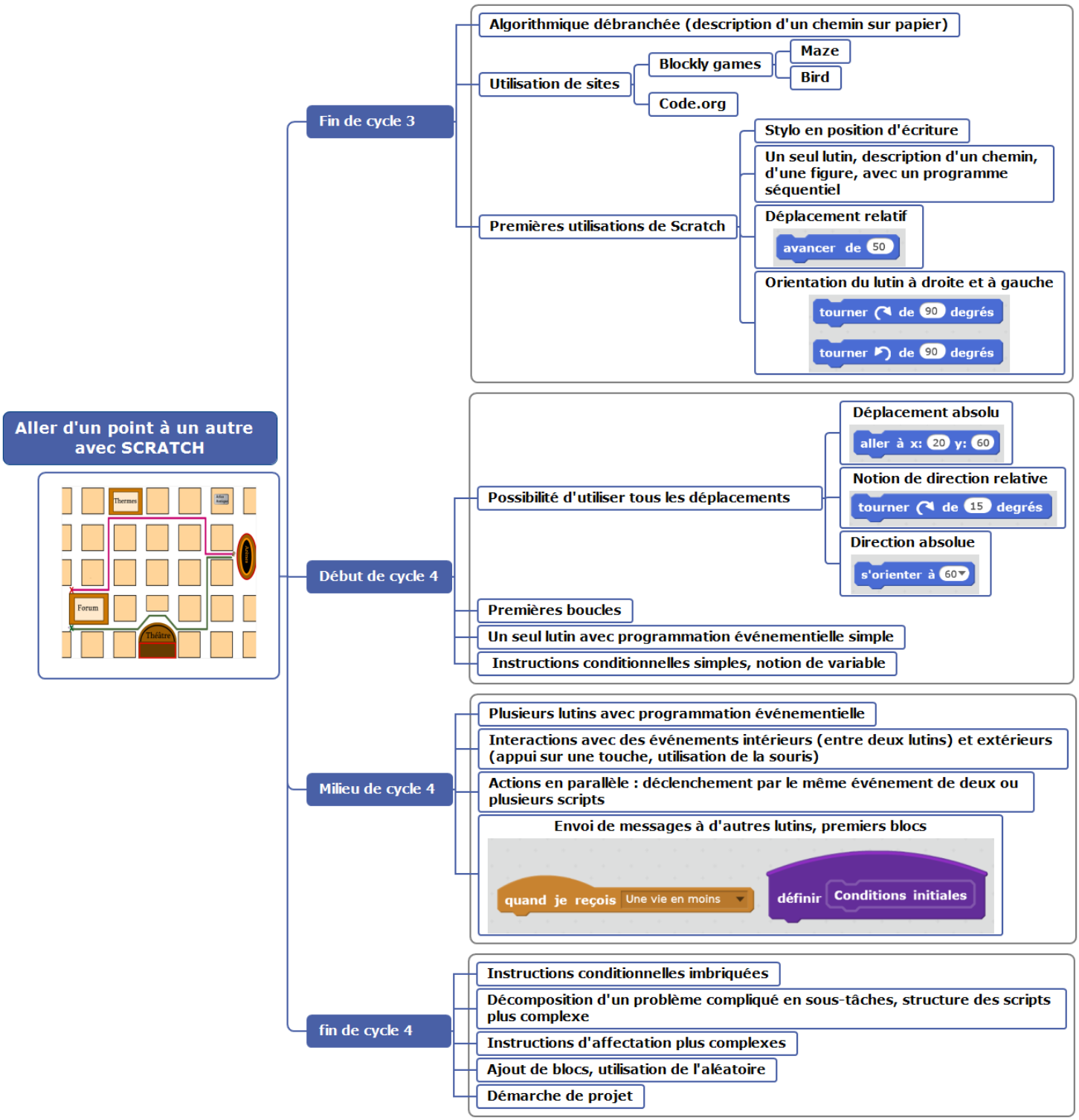


- Au fur et à mesure, on ajoute des personnages, des contraintes, (une alerte, une barrière aléatoire au niveau d'une rue par exemple) et des interactions, ce qui permet une pédagogie différenciée.

- Il y a une réelle démarche de projet, ce qui implique une répartition des tâches. Les lutins peuvent être programmés séparément et enregistrés en local. On communique, on échange.

F) Un exemple de progressivité

Suite à ces activités, voici **un exemple** de progressivité durant tout le collège, construite autour de la notion de mouvement avec *Scratch*.



6. Questions Flash

Le tableau introductif ci-dessous est fonctionnel pour toutes les questions flash, quelles que soient les connaissances abordées.

Objectif	*Faire examiner des programmes de construction géométrique réalisés sous Scratch*
Contribution spécifique aux domaines du socle	*D2 - Les méthodes et outils pour apprendre*
Connaissances travaillées et/ou mobilisées	✔ *Toutes les connaissances « non algorithmiques » sont mobilisables* ✔ *Notion d'algorithme et de programme*
Prérequis	*Déplacer un lutin, divers mouvements*
Modalités d'organisation	*Évaluation de début d'heure, notée ou non, sujet vidéo-projeté. Une ou plusieurs questions, au choix.* *La correction peut être projetée sous forme d'un programme Scratch, validant ainsi les choix effectués par les élèves.*
Positionnement dans le cycle	*Cycles 3 et 4 : adaptable à tout moment, selon les notions abordées*
Durée estimée	*Variable selon la question posée ; quelques minutes en général. Prévoir plus de temps pour des questions dont la réponse nécessite quelques calculs ou bien l'application d'un théorème.*

A) Lien avec la géométrie

Question 1 - **Objectif :** Tracer un segment parallèle au segment horizontal bleu.

Le début de la figure

Le script associé

Question : Quel est le bloc suivant ? Avec quel paramètre ?

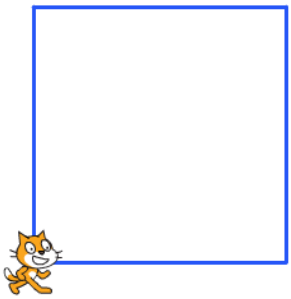
Question 2 - **Objectif :** Tracer un segment parallèle au segment horizontal bleu.

Le début de la figure

Le script associé

Question : Voici, ci-contre, le bloc suivant du script.
Quel paramètre faut-il y écrire ?

Question 3 - **Objectif :** Tracer un carré de côté 200, en 4 briques.

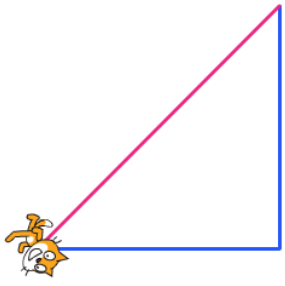


La figure à obtenir

Les briques à utiliser

Question : Dans quel ordre faut-il placer les briques pour atteindre l'objectif ?
Avec quels paramètres ?

Question 4 - **Objectif :** Tracer un triangle isocèle de côté 200, en 4 briques



La figure à obtenir

Le script commencé

Question : Voici, ci-contre, le bloc permettant de tracer le côté rouge.
Quel paramètre faut-il lui fournir ?

B) Lien avec le repérage dans le plan

Question 1

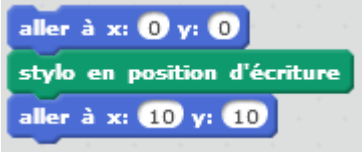
À l'exécution du script ci-contre, que va tracer le lutin ?

Question 2

Quelle est la longueur du segment tracé par le lutin ?

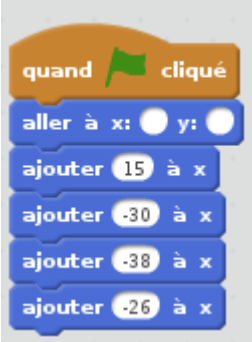
Question 3

Quelle est la longueur du segment tracé par le lutin ?



Question 4

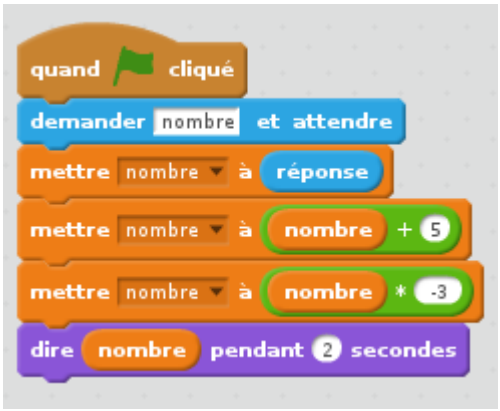
Que faut-il mettre dans le premier bloc pour que le lutin parte, à chaque exécution du script, de l'origine du repère ?



C) Lien avec des programmes de calcul

Question 1

Que donne ce programme si on choisit 7 ? -6 ?

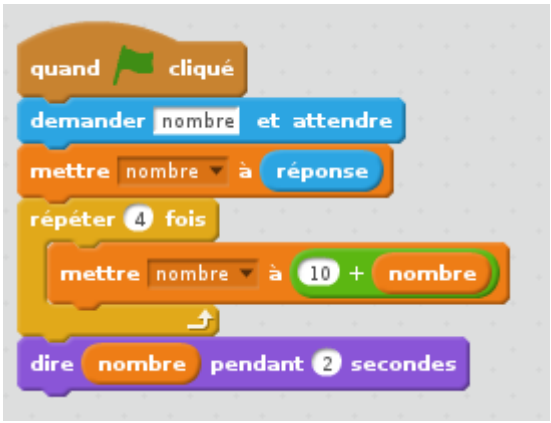


Question 2

Quel nombre faut-il choisir pour obtenir 27 ?

Question 3

Que donne ce programme si on choisit 12 comme nombre de départ ?
Que faut-il choisir pour obtenir 100 ?

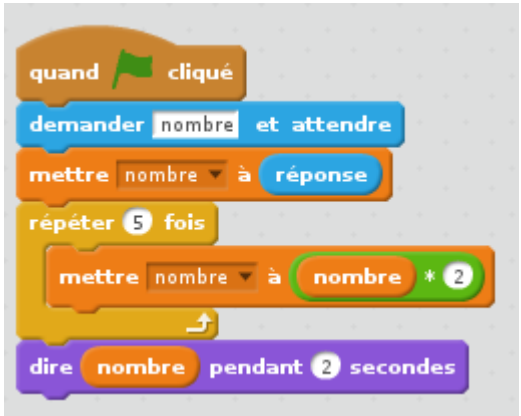


Question 4

Ce programme peut-il donner -24 ?

Question 5

Que donne ce programme si on choisit 2 comme nombre de départ ?
Que faut-il choisir comme nombre de départ pour obtenir 2^9 ?



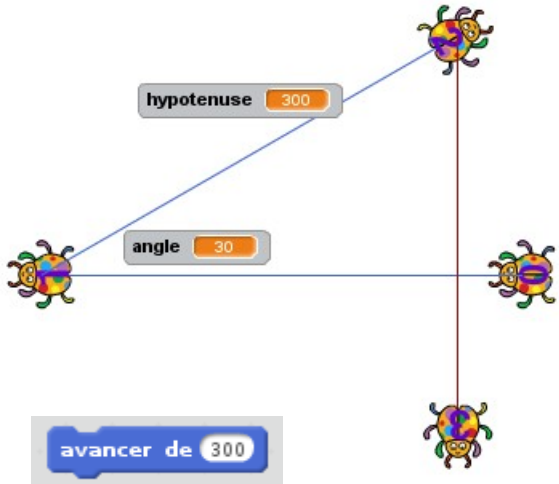
7. Une évaluation possible

Dans le cas d'un problème de construction géométrique

Objectif	*Dans le cadre d'évaluations, soit rapides en début d'heure, soit plus approfondies, il est possible de faire examiner des programmes de construction géométrique réalisés sous Scratch.*
Contribution spécifique aux domaines du socle	*D2 - Les méthodes et outils pour apprendre*
Connaissances travaillées et/ou mobilisées	✔ *Angles, géométrie du triangle* ✔ *Notion d'algorithme et de programme* ✔ *Trigonométrie*
Prérequis	*Déplacer un lutin, divers mouvements, opérateurs*
Modalités d'organisation	*Évaluation, notée ou non, sujet vidéo-projeté.* *Une fois l'objectif global compris par les élèves, chacune des trois étapes est projetée et laissée au tableau le temps nécessaire à la rédaction de la réponse.* *La correction peut être réalisée en direct sur Scratch, validant ainsi les choix des élèves.*
Positionnement dans le cycle	*Milieu et fin de cycle 4*
Durée estimée	*30 minutes, à adapter selon la classe*

Présentation du problème - **Objectif :** Améliorer la construction ci-dessous.

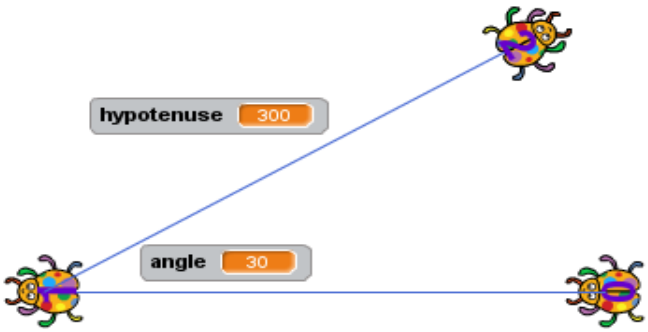
La coccinelle dessine trois segments formant un triangle rectangle ; mais en traçant le dernier segment (en rouge), elle doit s'arrêter pile sur le premier segment bleu.

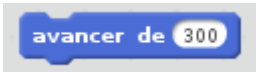


La construction du triangle complet est décrite en trois étapes.

Chaque étape est l'objet d'une question.

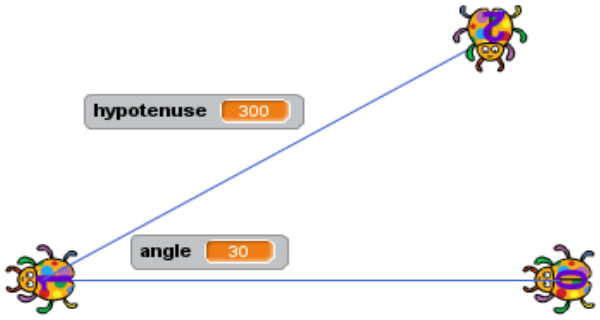
➢ *Étape 1* : Deux segments et un angle





- L'angle formé par les 2 segments doit être de 30 degrés.
- Construire une opération qui donne l'angle de rotation : un opérateur et deux valeurs parmi les trois proposées : 90, 30, 180.

➢ *Étape 2* : Deuxième angle



La coccinelle s'oriente de manière à recouper le premier segment par un angle droit.

- Quelles sont les mesures des 3 angles du triangle ? (On peut répondre par un schéma.)
- Construire une opération qui donne l'angle de rotation : un opérateur et deux valeurs parmi 90, 30 et 180.

➢ *Étape 3* : Calcul de longueur

La coccinelle est allée trop loin (segment gras et rouge). Elle doit s'arrêter sur le segment bleu exactement.

Construire la dernière instruction avec :

- la brique « *Avancer de* » ;
- deux briques vertes parmi celles ci-dessous ;
- deux nombres parmi 300, 30 et 60.