publié le 20/12/2009 - mis à jour le 21/12/2009

# SCRATCH

## Un logiciel pour créer des algorithmes

*Descriptif :*

Un document qui aide à la prise en main du logiciel Scratch, créateur d'algorithmes.

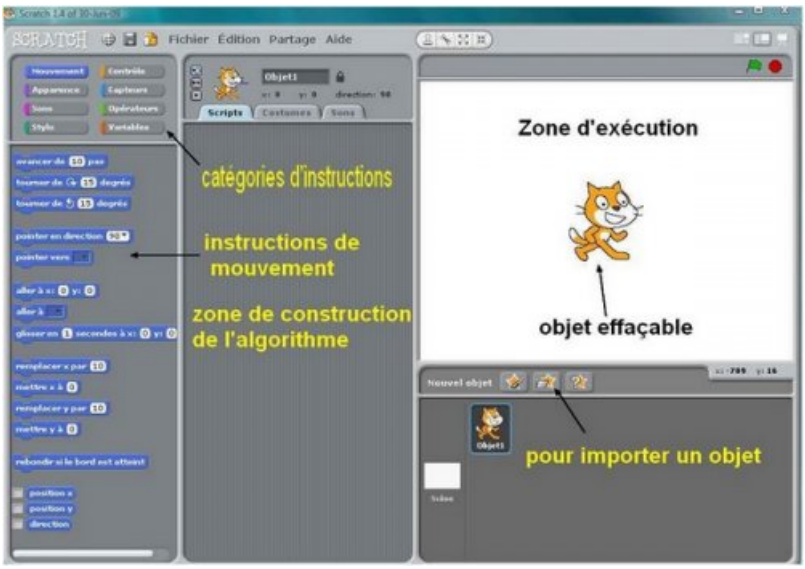
*Sommaire :*

- Présentation :
- Créer un algorithme
- Autres exemples.
- Avantages
- Inconvénients
- Conclusion

### ● Présentation :

Le logiciel « SCRATCH » permet de réaliser des programmes ou les erreurs de syntaxes sont impossibles. En effet il s'agit d'empiler ou d'emboiter des pièces sur lesquelles sont déjà marquées les instructions.

Voilà ce que l'on voit après avoir ouvert le programme puis cliqué sur la catégorie « mouvement » en haut à gauche :




Tout en haut à gauche apparaissent 8 dossiers, reconnaissables par leurs noms ou leurs couleurs :

- mouvement
- apparence
- son
- stylo
- contrôle
- capteurs
- opérateurs
- variables.

Cliquer sur l'un des dossiers, ou catégories, fait apparaitre dans la zone de gauche, toutes les instructions qu'il contient.
Chaque instruction-pièce est rangée dans l'une des catégories suivant sa fonction :
Par exemple, l'instruction :


est rangée dans la catégorie « mouvement ».
L'instruction


rangée dans la catégorie apparence, permet de cacher le chat ou tout autre objet qui vous énerve. Par contre, si vous trouvez ce chat mignon, vous pouvez le


L'instruction :


qui se trouve dans la catégorie contrôle, permet de réaliser une boucle, c'est à dire de répéter un certain nombre de fois, tout un bloc d'instructions.

## ● Créer un algorithme

On réalise un algorithme, ou un programme, en utilisant les pièces-instructions et en les associant dans un ordre précis.
Ainsi, l'algorithme suivant :

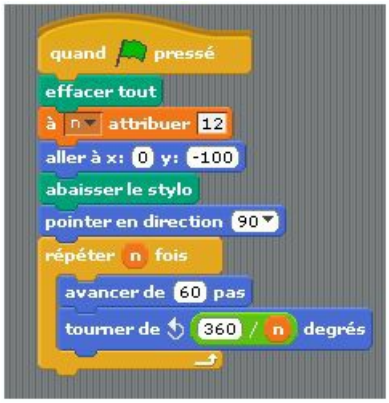

dessine un carré dans la partie réservée à l'exécution du programme.

La manipulation des différentes pièces est aisée, et le bloc ci dessus peut être construit en déposant les 4 pièces dans n'importe quel ordre. La réalisation de l'algorithme s'en trouve simplifiée car les corrections (enlever, rajouter ou déplacer une instruction) se font simplement avec la souris.
Pour démarrer l'exécution d'un algorithme, on peut cliquer sur la première instruction. En général il est préférable de chapeauter le bloc instructions par :


et on démarre alors le programme en cliquant sur le drapeau vert qui se trouve au dessus de la zone d'exécution.
L'avantage d'utiliser cette procédure est que l'on peut démarrer en même temps plusieurs algorithmes en cliquant sur le drapeau vert, si chacun d'entre eux est chapeauté par l'instruction ci-dessus.

Le petit programme, qui trace un carré, peut être le départ d'une activité avec les élèves. On peut leur demander de modifier ce programme pour qu'il trace un triangle équilatéral ou un pentagone régulier. Puis faire évoluer cet algorithme pour qu'il puisse tracer n'importe quel polygone régulier à n côtés, ce qui nécessite l'introduction d'une

variable et la création de formules.
Voilà cet algorithme à titre d'exemple :

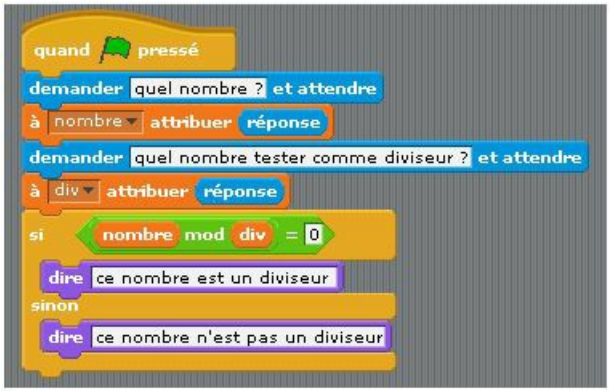



● Autres exemples.

Dans la suite, on part d'un algorithme simple et on l'enrichit progressivement. Les étapes sont :

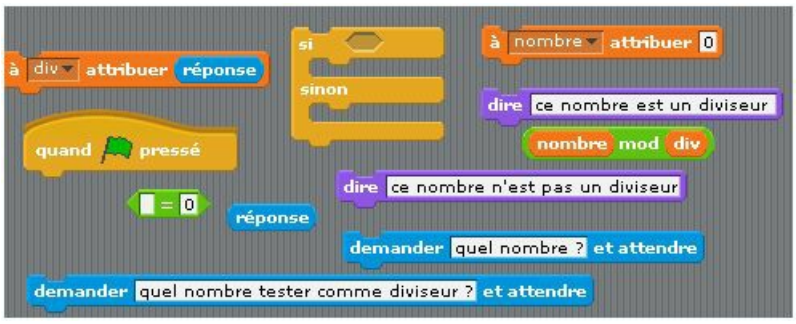
1. Tester si un nombre entier est divisible par un autre nombre entier.
2. Etablir la liste des diviseurs d'un nombre entier.
3. Faire la somme des diviseurs d'un nombre entier.
4. Tester si un nombre entier est parfait ( égal à la somme de ses diviseurs stricts)

Voilà le premier algorithme :




Dans cet algorithme, on a utilisé une instruction spécifique pour entrer les 2 données dont on a besoin.
Cette instruction n'a pas été utilisée dans les exemples précédents, mais aurait pu l'être dans l'algorithme qui construit un polygone de n côtés.
Quand l'algorithme est lancé, il faut répondre aux questions posées et valider, pour que l'algorithme continue son travail.

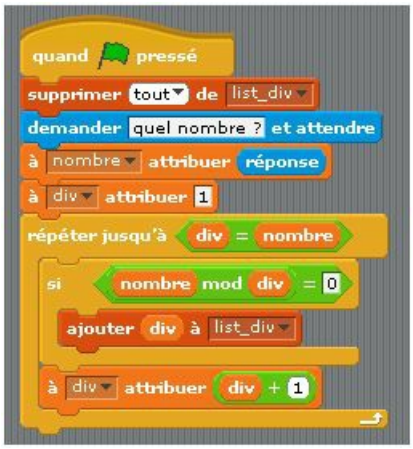
Si réaliser un tel algorithme vous semble une tâche trop difficile pour les élèves, on peut imaginer de leur donner tout ou une partie des briques , en désordre, et leur demander de les remettre dans l'ordre.
On peut par exemple leur donner l'image suivante, puis lui les laisser se débrouiller pour construire l'algorithme.
Beaucoup mêleront réflexion et progression par essais-erreurs.



Il faut reconnaître cependant que réaliser un tel travail pour tester la divisibilité d'un entier par un autre n'est pas très utile s'il n'y a pas de prolongement envisagé. En effet pour faire ce test, une simple calculatrice suffit, ou même un papier et un crayon si l'on est courageux.

Il faut donc faire évoluer cet algorithme vers la deuxième étape, la recherche de tous les diviseurs d'un entier. Cela pourra permettre par exemple d'aborder sereinement une des activités ( page 14) du document d'accompagnement des programmes sur les fonctions.

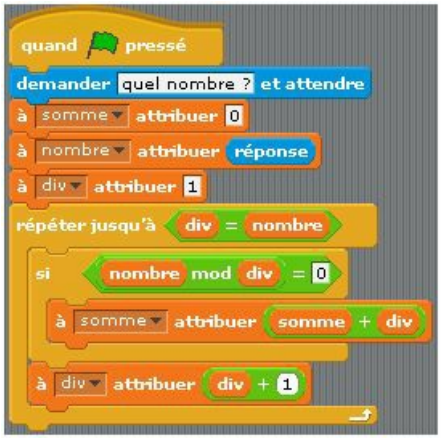
On obtient l'algorithme suivant :




Cet algorithme nécessite la création d'une liste capable de stocker tous les diviseurs du nombre choisi, au fur et à mesure qu'ils sont trouvés.
Bien sûr, il nécessite aussi une boucle pour réaliser le même calcul répétitif, mais c'est là tout l'intérêt de faire un algorithme.

On peut maintenant s'intéresser à la somme des diviseurs. Faire évoluer l'algorithme précédent vers celui-ci nécessite la création d'une nouvelle variable, »somme » . En revanche, on peut supprimer la liste si on n'a plus besoin des diviseurs mais seulement de leur somme.

On obtient ce qui suit :



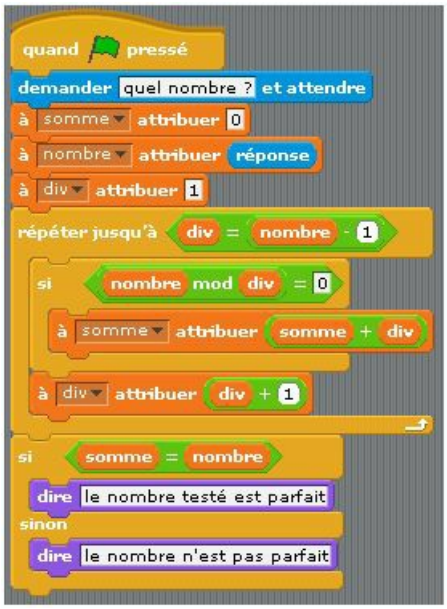

Depuis très longtemps, les mathématiciens se sont intéressés aux entiers qui étaient égaux à la somme de leurs diviseurs propres, tel 6=1+2+3 qui est le plus petit entier à posséder cette propriété.

Ces entiers sont qualifiés de parfait. Personne n'a encore prouvé si l'ensemble des entiers parfaits est fini ou infini, ni s'il existe un entier parfait impair.

Il est naturel, et c'est un travail facile, de faire évoluer l'algorithme précédent pour qu'il devienne un test pour savoir si un entier est parfait ou non.

Il suffira de rajouter un bloc d'instruction à la fin de l'algorithme précédent pour un bonne présentation des résultats, et de modifier une borne de la boucle.

On obtient :




Un dernier prolongement souhaitable serait de rechercher tous les entiers parfaits jusqu'à 1000 ou 10000. Il est alors indispensable d'améliorer l' algorithme précédent afin de diminuer le nombre d'opérations. Certains pourront s'y essayer mais j'ai constaté là les limites de SCRATCH(voir fichier joint).
Pas de partie entière, mais surtout une lenteur extrême pour les calculs.

Pour obtenir les 3 entiers parfaits inférieurs à 500, SCRATCH demande 3 minutes de calcul et plus d'une heure pour obtenir le 4ème.

En faisant le même programme avec Algobox, j'ai obtenu les 4 premiers entiers parfaits(6 ;28 ;496 ;8128) en 3s.

## ● Avantages

- Le principal intérêt de « scratch » est la simplicité de manipulation conjugué avec la possibilité de faire un programme sans se préoccuper de syntaxe.

- L'environnement ludique, les couleurs, la présentation agréable donne envie aux élèves de travailler avec cet outil.

- Il est possible d'importer plusieurs objets ou personnages sur la zone d'exécution et de contrôler chacun d'eux par un programme. A l'exécution, tous les programmes s'exécutent en même temps, gérant tous les objets à la fois. La réalisation de jeux simples ou plus élaborés est possible.

- On trouve dans « scratch » des commandes originales pour gérer les mouvements d'objets, ou contrôler le déroulement du programme, ou encore réaliser des sons. Ces commandes ne se trouvent pas dans d'autres logiciels.

- Scratch est un logiciel très développé en Amérique du nord, beaucoup de ressources sont disponibles sur Internet, en particulier dans le domaine des mathématiques .

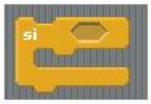
## ● Inconvénients

- Scratch n'est pas un logiciel adapté pour faire des algorithmes qui nécessitent beaucoup d'opérations, car Scratch est très lent.

- La zone réservé à l'exécution du programme est un repère, dont l'origine est le centre de la fenêtre. L'unité est le pixel, c'est dire que l'abscisse varie entre -237 et 237, et l'ordonnée varie entre -177 et 177.
Cependant un repère utilisable pour placer des points n'existe pas. Il faudrait le programmer, si on voulait tracer des courbes de fonctions.

- Certaines associations de pièces-instruction sont impossibles. Par exemple, la brique :


ne peut se mettre au dessus de :




car il n'y a pas d'encoche.

- La fonction partie entière manque.

- Lorsqu'une variable est affichée, on ne peut obtenir qu'un seul chiffre après la virgule, sauf si cette variable est placée dans une liste, et alors là, c'est le contraire, il y a trop de décimales.

## ● Conclusion

SCRATCH est un bon logiciel pour initier les élèves à l'algorithmique. Sa véritable fonction est de réaliser des animations ou des jeux. On peut bien sur faire des mathématiques avec , à condition de choisir des activités pas trop gourmandes en calcul.

Pour télécharger le logiciel, et voir des exemples :

page d'accueil de SCRATCH

## Documents joints

liste des diviseurs d'un entier (Binary Data de 49.4 ko)
algorithme en scratch qui trouve les diviseurs d'un entier

liste entiers parfaits (Binary Data de 54 ko)
Algorithme en scratch qui fait la liste des entiers parfaits jusqu'à 500.

polygone régulier de n cotés (Binary Data de 51.6 ko)
Algorithme en Scratch qui dessine un polygone régulier de n côtés.

test parfait (Binary Data de 52.4 ko)
Teste si un entier est parfait

teste un diviseur pour un entier (Binary Data de 48.5 ko)
algorithme en scratch qui teste si un entier divise un autre entier.