Informations sur l'algorithmique en seconde

Voici ce qui pourrait servir de document ressource à télécharger sur le site académique pour les collègues.
Seule une partie sera distribuée sur papier (le paragraphe « Les Ateliers »).
Nous vidéo-projetterons les points essentiels de ce document lors de nos exposés ; en particulier, nous présenterons le matin :

- le paragraphe I (ce que dit le programme) avec nos remarques
  (5 minutes)
- une introduction possible en classe à la notion d'algorithme sur l'exemple « deviner un nombre »
  (10 minutes)

et l'après midi avant les ateliers :

- les choix techniques (paragraphe II) car l'attente est forte sur ce point à notre avis
  (5 minutes)
- encadrer les ateliers le reste du temps (paragraphe V) dont l'objectif serait la réalisation d'algorithmes répondant à des problèmes mathématiques issus de tous les chapitres du programme.

Table des matières

I. Ce que dit le programme

La démarche algorithmique est, depuis les origines, une composante essentielle de l'activité mathématique. Au collège, les élèves ont rencontré des algorithmes (algorithmes opératoires, algorithme des différences, algorithme d'Euclide, algorithmes de construction en géométrie). Ce qui est proposé dans le programme est une formalisation en langage naturel propre à donner lieu à traduction sur une calculatrice ou à l'aide d'un logiciel[1]. Il s'agit de familiariser les élèves avec les grands principes d'organisation d'un algorithme : gestion des entrées-sorties, affectation d'une valeur et mise en forme d'un calcul.

Dans le cadre de cette activité algorithmique, les élèves sont entraînés :

- à décrire certains algorithmes en langage naturel ou dans un langage symbolique ;
- à en réaliser quelques uns[2] à l'aide d'un tableur ou d'un petit programme réalisé sur une calculatrice ou avec un logiciel adapté ;
- à interpréter des algorithmes plus complexes.

Aucun langage, aucun logiciel n'est imposé.

L'algorithmique a une place naturelle dans tous les champs des mathématiques et les problèmes posés doivent être en relation avec les autres parties du programme (fonctions, géométrie, statistiques et probabilité, logique) mais aussi avec les autres disciplines ou la vie courante.

À l'occasion de l'écriture d'algorithmes et de petits programmes, il convient de donner aux élèves de bonnes habitudes de rigueur et de les entraîner aux pratiques systématiques de vérification et de contrôle.

Lu dans le document ressource disponible sur Eduscol :

l'introduction d'une familiarisation avec l'algorithmique prend sa place dans une pratique des Mathématiques dont un axe principal est la formation des élèves à la démarche scientifique sous toutes ses formes.

Dans la classe de seconde, la découverte de l'algorithmique permettra d'étudier certaines notions sous un angle différent :
comment organiser la recherche du maximum d'une fonction ? Comment représenter une droite sur un écran n'affichant que des pixels ? Comment réaliser, en statistiques, le tri des données requis pour accéder à la médiane ?

La sensibilisation de l'élève à la question de la « démarche algorithmique » pourra se faire en évitant toute technicité ou exposé systématique. On pourra sur ce thème consulter des publications réalisées dans le cadre des IREM.

Les compétences suivantes pourront être identifiées et travaillées :

– comprendre et analyser un algorithme préexistant ;

---

1 Il nous paraît judicieux d'initier les élèves en langage naturel, ce qui élimine deux difficultés parasites : le langage à apprendre et les erreurs de syntaxe. On se concentre ainsi sur le contenu et sur la démarche algorithmique.

2 le « quelques uns » confirme que l'écriture de programmes informatiques n'est pas le but de la formation. Une fois l'algorithme au point et testé, on peut passer éventuellement à sa traduction sur machine.
Remarque : Algobox, décrit plus bas, permet de regrouper les deux phases : écriture en langage naturel + exécution du programme. Cependant, pour certains problèmes, l'usage d'un tableur ou d'un logiciel spécialisé sera plus pertinent (cf le paragraphe des Ateliers).

– modifier un algorithme pour obtenir un résultat particulier ;

– analyser la situation : identifier les données d'entrée, de sortie, le traitement...;

– mettre au point une solution algorithmique : comment écrire un algorithme en « langage courant » en respectant un code, identifier les boucles, les tests, des opérations d'écriture, d'affichage... ;

– valider la solution algorithmique par des traces d'exécution et des jeux d'essais simples ;

– adapter l'algorithme aux contraintes du langage de programmation : identifier si nécessaire la nature des variables... ;

– valider un programme simple.

Les capacités attendues dans le domaine de l'algorithmique d'une part et du raisonnement d'autre part, sont transversales et doivent être développées à l'intérieur de chacune des trois parties.[3]

II. Les différents choix techniques

A notre avis, un logiciel se distingue car il colle bien aux objectifs du programme officiel : AlgoBox.

Il nous paraît très adapté à la découverte de l'algorithmique en seconde car l'élève n'aura pas à se soucier dans un premier temps de langage ni de syntaxe à apprendre. De plus, le logiciel construit le code de l'algorithme pas à pas **de façon hiérarchique et structurée** grâce à des instructions de base que l'on insère en cliquant sur les boutons de l'interface :

3 Donc pas de chapitre « Algorithmique ». L'initiation se fera au fur et à mesure des besoins pendant l'année.

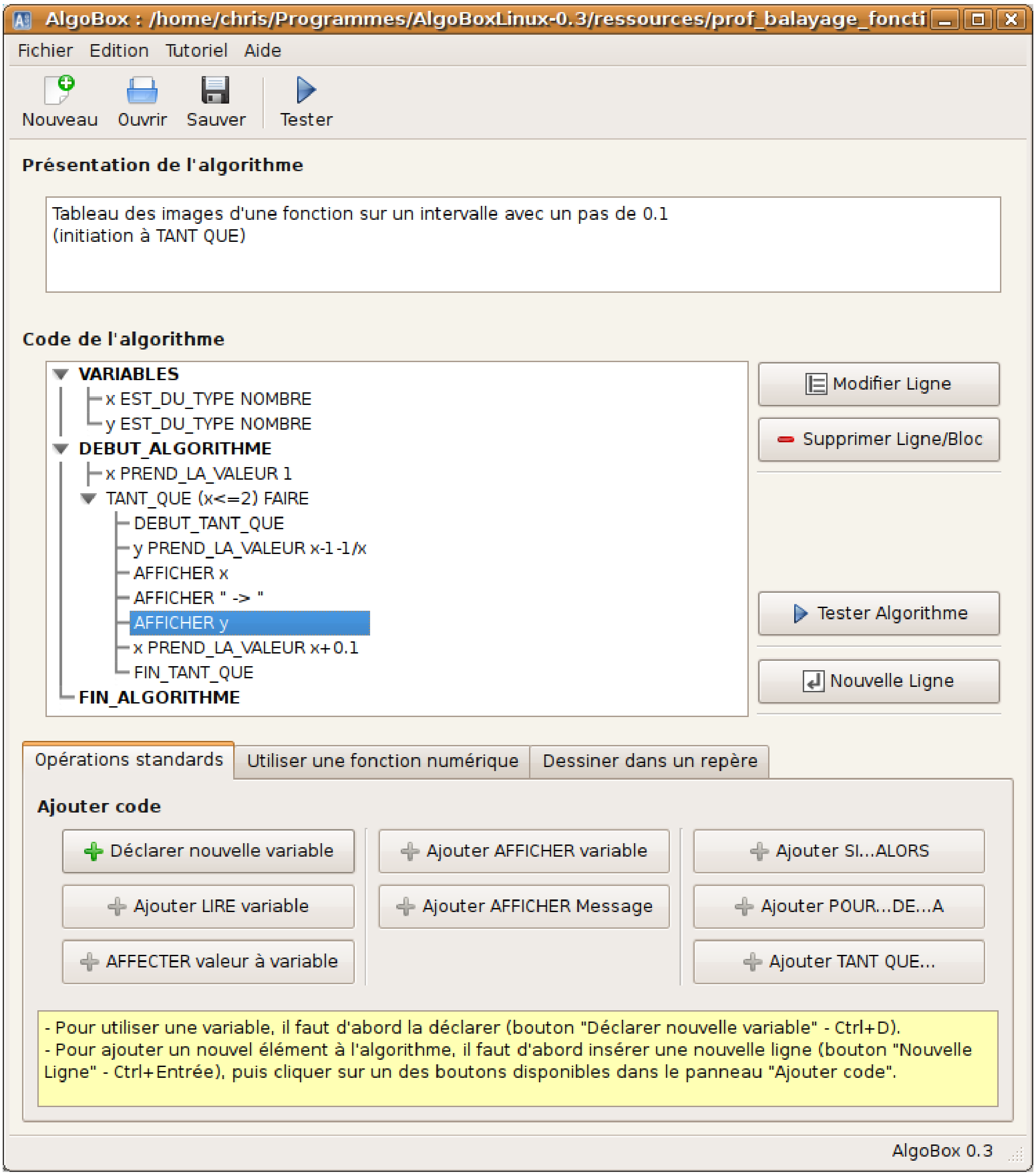


Illustration 1: Présentation de l'interface de AlgoBox

L'élève peut se concentrer ainsi sur l'algorithme lui-même et il est fortement incité, par le fonctionnement même du programme, à faire preuve d'un minimum de rigueur (il faut réfléchir à la déclaration des variables avant de pouvoir les utiliser, se demander ou insérer telle ou telle ligne (à l'intérieur de la boucle ou non, dans quelle partie du test « SI », etc...)).

Les atouts pédagogiques de Algobox nous semble indiscutables.

Cependant, pour des raisons pratiques ou techniques, d'autres solutions sont possibles. En voici quelques unes :

Matériel	Avantages	Inconvénients
AlgoBox	pédagogique, fonctionne en langage naturel, limite les	Nécessite un ordinateur (si utilisé par le prof en cours) ou

	erreurs de syntaxe, répond parfaitement aux objectifs du programme. Libre et gratuit, les élèves peuvent l'installer chez eux.	une salle info (pour les élèves). Pas de mode « pas à pas » (mais c'est en cours d'intégration pour une prochaine version)
Calculatrice	Disponible à tous les cours dans n'importe quelle salle. Les élèves peuvent travailler avec n'importe quand.	Un langage et une syntaxe à apprendre. Il y a souvent plusieurs marques de calculatrice à gérer dans une classe.
Le tableur	Pour OpenOffice Calc : libre et gratuit, les élèves peuvent l'installer chez eux.	Un langage et une syntaxe à apprendre. Nécessite un ordinateur (si utilisé par le prof en cours) ou une salle info (pour les élèves).
Scilab	Logiciel de calcul numérique libre et gratuit. Les élèves peuvent l'installer chez eux.	Un langage et une syntaxe à apprendre. Nécessite un ordinateur (si utilisé par le prof en cours) ou une salle info (pour les élèves).
Xcas	Logiciel « tout en un » pour les mathématiques. Permet l'écriture de programmes en langage TI. Libre et gratuit, les élèves peuvent l'installer chez eux.	Un langage et une syntaxe à apprendre. Nécessite un ordinateur (si utilisé par le prof en cours) ou une salle info (pour les élèves).
Scratch	Ludique, esthétique, pas de langage ni de syntaxe à apprendre.	Pas de boucle « pour », écriture laborieuse des algorithmes, beaucoup de fonctionnalités inutiles pour nous qui risquent de disperser l'attention des élèves.

III. Aborder l'algorithmique avec les élèves

Nous faisons le choix de consacrer un peu de temps à une petite initiation à l'algorithmique, par exemple en classe entière, pour que les élèves soient opérationnels ensuite à tout moment dans l'année. Voici deux exemples d'approche de l'algorithmique. D'autres idées sont développées dans le document ressource disponible sur le serveur académique.

Sans outil informatique

Afin d'introduire le vocabulaire élémentaire (les variables, leur type, les affectations, les entrées, les sorties), on donne aux élèves deux problèmes mathématiques très simples. On leur demande alors de construire un algorithme dont le déroulement résoudra le problème posé.

Question 1 : Connaissant deux nombres a et b, calculer puis afficher a^2+b^2.

Question 2 : Connaissant deux points du plan A et B, construire un point C tel que ABC soit un triangle équilatéral.

Tout est traité ici avec le crayon-papier. On pourra éventuellement demander une traduction dans un logiciel une fois les algorithmes complètement terminés. Pourquoi pas en travail à la maison ou au CDI ?

Une fois ce travail achevé, on pourra passer à des algorithmes plus élaborés (utilisant notamment des boucles, des tests, etc...)

Avec un outil informatique, en partant d'un algorithme existant

Dans cette partie, les quatre exemples-activités suivants illustrés avec AlgoBox permettent d'introduire l'algorithmique à l'aide d'un ordinateur vidéo-projeté à la classe.

a) Exemple 1 : Introduire le vocabulaire à partir d'un algorithme

Voici un algorithme :

Code de l'algorithme

```
VARIABLES
  n EST_DU_TYPE NOMBRE
  m EST_DU_TYPE NOMBRE
DEBUT_ALGORITHME
  n PREND_LA_VALEUR floor(random()*1000)+1
  LIRE m
  TANT_QUE (m!=n) FAIRE
    DEBUT_TANT_QUE
    SI (m>n) ALORS
      DEBUT_SI
      AFFICHER m
      AFFICHER " est trop grand"
      FIN_SI
    SI (m<n) ALORS
      DEBUT_SI
      AFFICHER m
      AFFICHER " est trop petit"
      FIN_SI
    LIRE m
    FIN_TANT_QUE
  AFFICHER "Bravo !"
FIN_ALGORITHME
```

1) Présenter le vocabulaire sur cet algorithme :
   les variables, leur type, les affectations
   les entrées (lire), les sorties (afficher)
   les instructions élémentaires, les blocs d'instructions
   les boucles conditionnelles (SI), les boucles répétitives (ici TANT QUE)
2) Lire l'algorithme avec la classe et faire deviner ce qu'il réalise
3) Faire jouer quelques élèves
4) Comment modifier ce programme pour qu'il affiche le nombre d'essais nécessaires à la

victoire ?

5) Faire émerger une stratégie permettant de gagner avec le moins de tentatives possibles.

Remarque : cet exemple peut servir de modèle pour, plus tard, construire un algorithme permettant d'encadrer une racine d'une équation par dichotomie (voir les Ateliers plus bas).

b) Exemple 2 : Que fait l'algorithme ci-dessous ? (extrait des exemples d'AlgoBox)

Code de l'algorithme

```
VARIABLES
  a EST_DU_TYPE NOMBRE
  b EST_DU_TYPE NOMBRE
  m EST_DU_TYPE NOMBRE
DEBUT_ALGORITHME
  LIRE a
  LIRE b
  SI (a<b) ALORS
    DEBUT_SI
    m PREND_LA_VALEUR b
    FIN_SI
    SINON
      DEBUT_SINON
      m PREND_LA_VALEUR a
      FIN_SINON
  AFFICHER m
FIN_ALGORITHME
```

Des pistes de réflexions après que la classe a trouvé l'utilité de cet algorithme :

1) A-t-on défini une fonction ? à combien de variables ?

2) Quel nom pourrait-on donner à cette fonction ?

3) Comment transformer cet algorithme pour définir la fonction « min » ?

c) Exemple 3 : quelle est cette fonction mathématique ?

Voici un algorithme :

Code de l'algorithme

```
VARIABLES
  a EST_DU_TYPE NOMBRE
  b EST_DU_TYPE NOMBRE
  c EST_DU_TYPE NOMBRE
  d EST_DU_TYPE NOMBRE
DEBUT_ALGORITHME
  LIRE a
  b PREND_LA_VALEUR a+2
  c PREND_LA_VALEUR b^2
  d PREND_LA_VALEUR c+1
  AFFICHER d
FIN_ALGORITHME
```

1) En lisant l'algorithme prévoir les valeurs de sortie pour $a=1$ puis pour $a=-4$
2) Vérifier en « testant l'algorithme »
3) Exprimer d en fonction de a uniquement et confirmer par le calcul les réponses précédentes.

d) Exemple 4 : un algorithme erroné à corriger

L'algorithme suivant a pour objectif de demander à l'utilisateur deux valeurs numériques stockées dans les variables a et b puis d'échanger le contenu des deux variables :

Code de l'algorithme

```
VARIABLES
  a EST_DU_TYPE NOMBRE
  b EST_DU_TYPE NOMBRE
DEBUT_ALGORITHME
  LIRE a
  LIRE b
  a PREND_LA_VALEUR b
  b PREND_LA_VALEUR a
  AFFICHER "a vaut "
  AFFICHER a
  AFFICHER "b vaut "
  AFFICHER b
FIN_ALGORITHME
```

1) Cet algorithme ne donne pas satisfaction. Pourquoi ? Que fait-il réellement ?

2) Que faire pour corriger cet algorithme ?

Une autre progression possible

Objectif : en 2 ou 3 séances de module, introduire la structure d'un algorithme, la notion de variable, celles d'entrée et de sortie; puis progressivement ajouter les tests et les boucles.

a) Première construction d'algorithme : dialoguer avec la machine

- Logiciel lancé sur vidéo-projecteur à l'entrée en salle (module de préférence);
- Question posée : « entrez votre nom »;
- Réponse : « Bonjour Untel ».

Avec quelques « briques » écrites au tableau ou sur une fiche (à découper ? Sur tableau blanc interactif ?), faire écrire aux élèves une suite d'instructions pour arriver à ce résultat. Comparer avec l'algorithme de départ, sur Algobox.

b) Elle sait son cours, elle !

Évolution en cours de module : faire calculer la machine / l'ordinateur à sa place.
Argument : puisque on a le droit à la calculatrice en devoir (parfois), autant lui faire vérifier nos calculs...
Quelques exemples :

- entrer m et p de mx+p; faire donner la racine, le sens de variation avec phrase associée.
- Coordonnées de A et B en repère orthonormé → distance AB, coordonnées de $\vec{AB}$.

A répéter régulièrement, en adaptant l'algorithme à sortir à chaque chapitre.
Poursuivre en analysant des algorithmes divers : décompositions de fonctions, affectations successives de variables (ci-contre)...

```
VARIABLES
  a EST_DU_TYPE NOMBRE
  p EST_DU_TYPE NOMBRE
DEBUT_ALGORITHME
  LIRE a
  p PREND_LA_VALEUR a*a
  p PREND_LA_VALEUR p*p
  p PREND_LA_VALEUR p*a
  p PREND_LA_VALEUR p*p
  AFFICHER p
FIN_ALGORITHME
```

c) Introduction des tests

Une fonction définie par morceaux pose toujours problème : beaucoup d'élèves y voient deux fonctions distinctes.
Un algorithme comprenant un test permet de calculer des valeurs en une seule suite d'opérations.
Piste :
soit la fonction f définie par :

$$f(x) = -(x-3)^2 + 1 \text{ si } x < 3 \qquad f(x) = 2(x-3)^2 + 1 \text{ si } x \geq 3$$

1. Faire construire l'algorithme de calcul des chacune des 2 fonctions (un par demi-groupe ? Sur calculatrice ou sur papier.)
2. Introduire le si... alors... sinon... pour concaténer les deux algorithmes. Faire tourner le nouvel algorithme sur Algobox.

d) Introduction du calcul répétitif

1. Passer au balayage d'un intervalle à partir de l'exemple précédent, pour dresser un tableau de valeurs d'une fonction, voire la tracer.
2. Jeu « deviner un nombre », décrit précédemment.

IV. Traductions d'algorithmes sur machine

Dans des cas particuliers, on pourra avoir besoin des fonctionnalités particulières d'un logiciel autre qu'AlgoBox pour exécuter un algorithme (tableur, logiciel de calcul formel, logiciel de géométrie dynamique, etc...). pour cela, des ressources sont disponibles sur le web. Par exemple pour l'écriture d'un programme avec Scratch, Scilab et Xcas voir : http://www.decliclycee-prof.hachette-education.com/pdf/Declic2nde_Logiciel.pdf

Nous mettons votre disposition sur le serveur académique (http://maths.ac-rouen.fr/) les traductions des algorithmes présentés ci-dessus pour les calculatrices (TI et Casio), Xcas, Scilab et tableur.

V. Ateliers

Les exemples ci-dessous sont des problèmes mathématiques qu'il est demandé de résoudre par l'écriture d'un algorithme. Ils sont classés par ordre de difficulté croissante.

Exercice 1 (d'après Bordas indice 2009) :

Le barème de l'impôt 2009 pour un foyer ayant deux parts dépend de son revenu imposable R en euros. Pour R<51 852, on a le barème suivant :

- Pour $R \leqslant 11704$, on ne paie pas d'impôt ;
- Pour $11\,705 \leqslant R \leqslant 23\,346$ l'impôt est : $0{,}055 \times R - 643{,}72$;
- Pour $R \geqslant 23347$, l'impôt est : $0{,}14 \times R - 2628{,}14$.

Écrire un algorithme qui, pour un revenu donné, renvoie la valeur de l'impôt à payer lorsque le revenu est inférieur à 51 852 €.

Le tester avec AlgoBox.

Exercice 2 (d'après Compléments en ligne de Déclic 2nde) :

On se place dans un repère orthonormé.

1) Écrire un algorithme qui permet de saisir les coordonnées des sommets d'un triangle, et qui précise si ce triangle est rectangle ou non, et si oui, en quel point.
2) Dans chacun des cas suivants, dire, à l'aide d'un programme, si le triangle ABC est rectangle ou non, et si oui, en quel point :
a) A (2 ; 1), B (1 ; 5) et C (2 ; 3) .
b) A (2 ; 1), B (2 ; 3) et C (6 ; 3) .
c) A (0 ; 2), B (3 ; 2) et C (1 ; 4) .

Exercice 3 :

On se donne une fonction f et X_{min}, X_{max}, pas trois réels. Écrire un algorithme de tracé de la courbe représentative de f sur $[X_{min} ; X_{max}]$ avec un pas valant pas. Le tester avec AlgoBox.

Exercice 4 (d'après Nathan hyperbole 2009) :

1) Écrire un algorithme qui compare les nombres $A=(x+y)^2$ et $B=x^2+y^2$.
2) traduire cet algorithme sur un tableur et comparer A et B sur une même feuille de calcul lorsque x et y prennent des valeurs aléatoires entre -5 et 5.

3) Conjecturer la comparaison de A et B suivant les valeurs de x et y
4) Démontrer la conjecture par le calcul.

Exercice 5

Les prototypes Geoplan se comportent comme des fonctions informatiques :

- elles admettent des objets en entrée
- effectuent un traitement sur ces objets
- renvoient un objet géométrique en sortie

exercice : connaissant 3 points du plan A, B et C, on veut construire le parallélogramme ABCD.

1) Écrire un algorithme effectuant cette tache
2) définir un prototype avec Géoplan effectuant cette construction.
3) Placer 3 autres points M, N et P et tester le prototype. Vérifier que la construction résiste au déplacement de M, N et P
4) Existe-t-il plusieurs parallélogrammes contenant les points M, N et P ? Si oui, les afficher tous à l'aide du prototype ainsi créé.

Exercice 6

Au jeu des petits chevaux, un joueur ne peut engager un cheval dans la partie que s'il fait un 6 avec le dé. On se demande combien de coups, en moyenne un joueur doit-il jouer pour engager un cheval.

1) Écrire un algorithme qui simule un début de partie et affiche le nombre de lancers de dé qu'il aura fallu pour faire un 6.
2) Transformer ce programme pour simuler n débuts de partie (n est entré par l'utilisateur) et afficher la répartition du nombre de lancers de dé nécessaire pour engager un cheval.
3) Tester plusieurs fois cet algorithme pour n=100. La répartition est-elle toujours la même ? Comment appelle-t-on ce phénomène ?
4) Compléter l'algorithme pour afficher en plus le nombre moyen de coups nécessaire pour engager un cheval lorsqu'on joue n parties.
5) Le résultat était-il prévisible ?

Exercice 6 bis

Au jeu des *petits chevaux*, Chaque joueur possède quatre pions (chevaux) de la même couleur et les place dans son enclos. Le premier joueur qui sortira un six sortira son premier cheval.

Le but de ce problème est de répondre à la question suivante en utilisant l'algorithmique :

Combien de tentatives sont en moyenne nécessaires pour amener un 6 sur un dé à 6 faces pour pouvoir sortir un cheval ?

1) a) Quel est la probabilité de sortir le chiffre 6 si on suppose le dé bien équilibré ?

b) Peut-on être certain de sortir le 6 en 6 lancers ?

2) a) Présenter un algorithme en langage naturel puis appliqué à un logiciel (de type

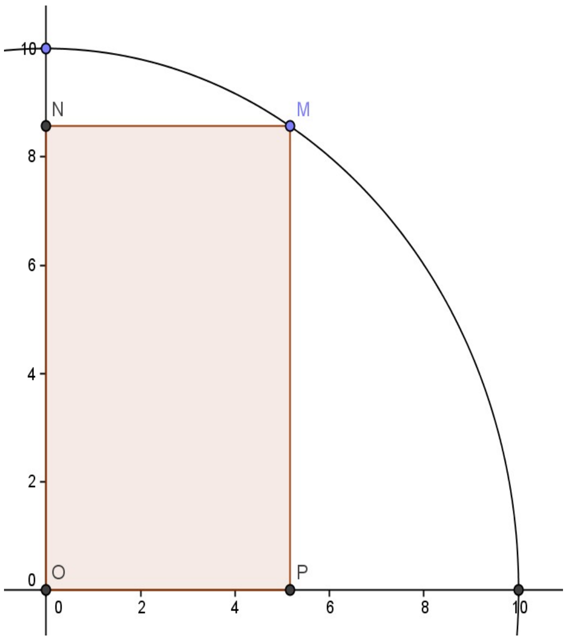
Algobox ou ...) permettant de faire un début de partie, c'est-à-dire de lancer le dé jusqu'à ce qu'il fournisse un 6 et d'afficher le nombre de coups qu'il a été nécessaire pour que ce 6 apparaisse pour la première fois.

b) Modifier le programme suivant pour calculer la moyenne de ces apparitions sur 10 parties, sur 100 parties.

Pouvez-vous répondre au problème initial ? Pourquoi ?

c) Modifier le programme précédent afin de pouvoir donner une réponse.

Exercice 7 : Balayer pour mieux cerner



Un point $M(x;y)$ parcourt le quart de cercle de centre O et de rayon 10 comme ci-contre. L'abscisse et l'ordonnée de M sont positives.

Ce point M permet de construire le rectangle $MNOP$.

Le but de ce problème est de répondre à la question suivante en utilisant l'algorithmique :

Quelle est la position du point M pour que l'aire du rectangle MNOP soit maximale ?

1) Calculer la longueur OM en fonction de x et y.
En déduire une relation entre x et y.
Montrer alors que l'aire A du rectangle peut s'écrire :

$$A(x) = x\sqrt{100 - x^2}.$$

Pour quelles valeurs de x cette fonction est-elle définie ?

2) a) Présenter un algorithme en langage naturel puis appliqué à un logiciel (de type Algobox ou ...) permettant d'afficher les images successives de la fonction pour x variant de 0 à 10 avec un pas égal à 1. Vous repèrerez la première valeur de x pour laquelle l'image décroit.

En déduire un encadrement de la valeur de x pour laquelle l'aire du rectangle est maximale.

b) Modifier le programme précédent de façon à trouver un encadrement de 0,2 puis de 0,02.

c) Donner une valeur approchée à 10^{-5} de la valeur de x pour laquelle l'aire est maximale puis une valeur approchée de l'aire maximale du rectangle recherchée.

Exercice 8 : Le triangle de Sierpinski

On utilise comme figure de départ un triangle équilatéral que l'on supposera ici de côté 10. En utilisant les milieux de ses côtés, on définit ainsi un nouveau triangle central que l'on enlève au triangle initial permettant d'obtenir la figure après la première étape. Il suffit d'appliquer ce procédé aux trois triangles restants pour obtenir la figure suivante. En itérant ce procédé une infinité de fois, on obtient ainsi le triangle (ou fanion) de Sierpinski comme ci-dessous :

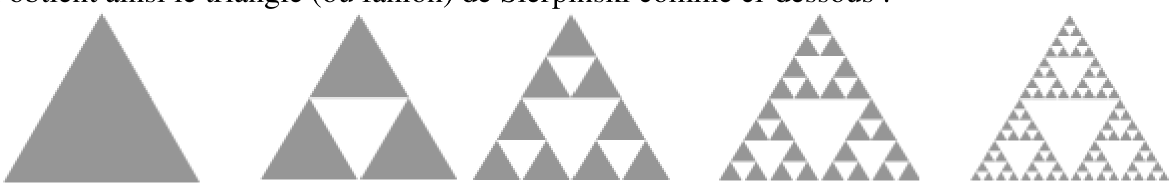


figure initiale étape 1 étape 2 étape 3 étape 4

L'étape 1 présente la création d'un triangle blanc. Celui-ci donne naissance à trois nouveaux triangles qui eux mêmes donneront naissance à de nouveaux triangles.

Le but de ce problème est de répondre à la question suivante en utilisant l'algorithmique :

Combien d'étapes seront nécessaires pour que la surface blanchie par la création des triangles représente au moins 90% de la surface totale ?

1) a) Quel est le nombre de triangles à l'étape 1, à l'étape 2, à l'étape 3 puis à une étape n.

b) Quelle est l'aire du triangle créé à l'étape 1, à l'étape 2, à l'étape 3 puis à une étape n.

c) Donner une formulation de l'aire de la partie blanchie à l'étape 1, à l'étape 2, à l'étape 3 puis à une étape n.

2) a) Présenter un algorithme en langage naturel puis appliqué à un logiciel (de type Algobox ou ...) permettant de demander à l'utilisateur un entier n puis de calculer la somme des aires des triangles créés jusqu'à cette étape n.

b) Modifier le programme précédent afin de déterminer la valeur de n qui permet d'obtenir une aire blanchie au moins égale à 90% de celle du triangle de la figure initiale.

3) Prolongement sous geoplan ou géogébra.

Construire une macro qui, étant donné un triangle ABC permet de construire 4 triangles comme le procédé précédent l'indique. Le triangle central devra être d'une couleur différente des trois qui l'entourent. Utiliser cette macro pour construire le triangle de Sierpinski à l'étape 4.

Exercice 9 : Adaptation tableur / algorithme

100 € ont été déposés sur un compte d'épargne en janvier 2000. Le détenteur de ce compte a préparé une feuille de calcul de tableur pour visualiser l'évolution de son épargne. Le taux d'intérêt du compte est de 3% par an (intérêts cumulés).

	A	B
1	Taux (%)	3
2	Année	Épargne (€)
3	2000	100
4	2001	103
5	2002	106,09

1. Justifier les résultats de la colonne B.
2. Quelles formules de tableur sont nécessaires pour remplir ce tableau ?
3. Cette feuille de calcul peut se traduire sous forme d'algorithme, dont les entrées sont le taux t et le montant m des économies en 2000, et la sortie le montant de l'épargne une certaine année, ou l'année où l'on obtient un certain montant.
   a) Écrire l'algorithme permettant d'obtenir le montant de l'épargne au bout de 10 ans.
   b) Écrire l'algorithme permettant d'obtenir l'année où le montant de l'épargne atteint 200€.

Exercice 10 : Approximation de π

1. Soit n un nombre entier strictement supérieur à 2. Déterminer la longueur de la hauteur d'un triangle OAB isocèle en O, tel que $OA=OB=1$, et $\widehat{AOB}=2\pi/n$.
2. Dans un repère orthonormé, on donne $A(1;0)$. Quelles sont les coordonnées du point B du cercle trigonométrique, tel que $(\overrightarrow{OA};\overrightarrow{OB})=2\pi/n$? Même question avec le point C du cercle tel que $(\overrightarrow{OB};\overrightarrow{OC})=2\pi/n$.
3. Créer un algorithme demandant une valeur pour n, puis traçant le polygone régulier à n côtés de centre O, inscrit dans le cercle trigonométrique, et dont un sommet est A. Quelle est l'aire de ce polygone ?

Exercice 11 : que fait cet algorithme ?

```
VARIABLES
  x EST_DU_TYPE NOMBRE
  y EST_DU_TYPE NOMBRE
  i EST_DU_TYPE NOMBRE
  n EST_DU_TYPE NOMBRE
  c EST_DU_TYPE NOMBRE
  p EST_DU_TYPE NOMBRE
DEBUT_ALGORITHME
  LIRE n
  c PREND_LA_VALEUR 0
  POUR i ALLANT_DE 1 A n
    DEBUT_POUR
    x PREND_LA_VALEUR random()
    y PREND_LA_VALEUR random()
    SI (x*x+y*y<1) ALORS
      DEBUT_SI
      c PREND_LA_VALEUR c+1
      FIN_SI
    FIN_POUR
  p PREND_LA_VALEUR 4*c/n
  AFFICHER p
FIN_ALGORITHME
```

De quel nombre s'approche-t-on ?

On pourra s'aider d'une figure.

L'approximation est-elle bonne ?