Chloé Ubéra
IREM d'Aquitaine
Alex Esbelin
IREM de Clermont-Ferrand

L'algorithmique au lycée, ... et alors

Algorithmique entre programmation et démarche mathématique

La conception unanime de l'algorithmique en fait un outil pour la programmation ; son introduction ne peut alors être naturelle que dans un projet de programmation. Or l'informatique est en dehors (à côté, mais en dehors) des mathématiques. Ainsi l'algorithmique ne peut apparaître que plaquée sur l'enseignement des mathématiques. Mais il existe une alternative ....

1 Introduction

1.1 Problématique.

L'introduction de l'algorithmique dans les programmes du secondaire releve de l'injonction institutionnelle. En l'absence d'une culture algorithmique répandue chez les enseignants de mathématiques, les modalités de son enseignement dépendent essentiellement de la formulation de cette 'injonction. La situation est différente par exemple pour la logique ou les nombres complexes.

1.2 Modalités

Comme les participants à cet atelier, les lecteurs sont invités à s'interroger sur les problèmes avant de lire leur analyse,
⋆ soit en essayant de les résoudre,
⋆ soit en essayant de décrire la stratégie envisagée spontanément pour le résoudre (pas de stratégie est une réponse possible).

2 Points de départ

2.1 Comment évaluer la production d'un algorithme par un élève ? Que veut-on que les élèves sachent ?

Le problème suivant suit une organisation pourrait être celle d'un problème de bac [1]. Contrairement à la partie "mathématique", seulement ébauchée, la partie "algorithmique" est donnée en détail. La question posée aux participants ne concerne pas le problème, mais des réponses (imaginaires) d'élèves qui font suite.

Résumé de la première partie mathématique
On considère la fonction f définie sur $[-2;1]$ par $f(x) = 2x^3 - 3x^2 - 6x$. Le tableau de variations de f sur $[-2;1]$ ci-contre est donné aux élèves ou déterminé par eux.
On veut donner une valeur approchée de x_0.

x	-2		x_0		1
f		↗	$f(x_0)$	↘	
	-16				-7

1. La situation utilisée a été proposée par Henri Rolland, de l'IREM d'Aix-Marsille avec un tout autre objectif.

Partie algorithmique

1) Prenons $a = -2$ et $b = 1$. On divise l'intervalle $[a; b]$ en trois parties de même longueur avec $u = -1$ et $v = 0$. Calculer $f(u)$ et $f(v)$. Ces résultats permettent-ils de dire lequel des trois intervalles $[a; u]$, $[u; v]$ ou $[v; b]$ contient x_0 ?

2) Reprendre la méthode initiée au 1) en remplaçant $[a; b]$ par l'intervalle trouvé à la question précédente et obtenir un nouvel encadrement de x_0.

3) Si on répète encore deux fois la méthode, quelle sera alors la longueur de l'encadrement de x_0 obtenu ?

4) En s'appuyant sur les résultats précédents, écrire un algorithme permettant de calculer un encadrement de x_0 à une précision donnée.

Question 1 : évaluer les quatre exemples de réponses suivantes

Réponse 1

Je pars de l'intervalle $[a; b]$.
Je le découpe en trois parties égales avec u et v.
Je calcule $f(u)$ et $f(v)$.
si $f(u) < f(v)$, je remplace a par u ;
si $f(u) > f(v)$, je remplace b par v ;
si $f(u) = f(v)$, je remplace a par u et b par v ;
Je recommence jusqu'à ce que $a - b$ soit inférieur à la précision voulue.

Réponse 2

A, B, E et f sont donnés.
Tant que B-A $\geq$ E
    H $\leftarrow$ B-A/3
    U$\leftarrow$A+H
    V$\leftarrow$B-H
    Si f(U)$\leq$ f(V) alors A$\leftarrow$U
    Si f(U)$\geq$ f(V) alors B$\leftarrow$V
Afficher(A,B)

****

Réponse 3

Je pars de l'intervalle $[a; b]$.
Je calcule $u = \frac{2a+b}{3}$ et $v = \frac{a+2b}{3}$.
tant que $u - v \geq e$:
je calcule $f(u)$ et $f(v)$;
si $f(u) < f(v)$, je remplace a par u ;
si $f(u) > f(v)$, je remplace b par v ;
si $f(u) = f(v)$, je remplace a par u et b par v.
Dès que $u - v < e$, j'écris $[u; v]$

Réponse 4

On découpe en trois parties égales l'intervalle $[a; b]$ qui sont $[a; u]$, $[u; v]$ et $[v; b]$. Si $f(u) < f(v)$, on remplace a par u ; si $f(u) > f(v)$, on remplace b par v ; sinon, on remplace a par u et b par v. On recommence autant de fois que nécessaire.

$\star$ Remarque à propos de la réponse 1 : les instructions *Je ... découpe [l'intervalle] en trois parties égales avec u et v* et *Je recommence jusqu'à ce que $a - b$ soit inférieur à la précision voulue* ne sont pas précises.

$\star$ Remarque à propos de la réponse 2 : les variables ne sont pas typée, or l'opération "/3" concerne des nombres flottant alors qu'un typage dynamique ou naturel ferait de A et B des entiers.

Conclusion : *notre hypothèse (qui mériterait d'être approfondie) est que la majorité des enseignants évalueront la réponse 2 comme la meilleure, et que cet évaluation met en évidence la conception suivante :*

<u>La</u> fonction des algorithmes est d'être des outils utiles pour la programmation

2.2 Partons maintenant de la certitude suivante :

<u>Une</u> fonction des algorithmes est d'être des outils utiles pour la programmation

Cette affirmation relève de l'évidence si on accepte les deux points suivants :
$\star$ ce n'est ni une définition, ni une caractérisation ;
$\star$ ce n'est pas nécessairement la seule fonction.

2. La remarque faite par un participant sur la disponibilité d'une méthode alternative simple (détermination de la valeur qui annule la fonction dérivée polynomiale de degré 2) n'invalide pas ce problème : il n'est pas proposé comme situation d'apprentissage, et particulièrement comme situation permettant de montrer l'intérêt d'une démarche algorithmique. L'exploitation pédagogique de cette remarque n'entre pas dans le champ de ce travail.

Supposons (provisoirement) hypothèse suivante : c'est la seule

Sous cette hypothèse, on peut prendre l'introduction des algorithmes dans les programmes de mathématiques comme le cheval de Troie de l'informatique (au sens de sciences du numérique) qui chercherait en rentrer dans la place de l'enseignement secondaire qui lui est fermée. Evisageons maintenant deux sous-hypothèses suivant que cette place lui serait (enfin) ouverte :

Cas 1 : *que faire dans le cas où le cursus prévoit un enseignement informatique ?* C'est le cas en IUT, dans les départements informatique. On relèvera (partie XXX) comment cette fonction y est l'un des fils directeurs de l'organisation de l'enseignement. Ce sera peut-être le cas dans l'avenir au lycée. La seule question importante dans ce cas est celle de la formation ou à tout le moins du recrutement des professeurs d'informatique. Et la question subsidiaire de savoir ce qu'on en fait en maths.

Projet de Programme Pédagogique National
DUT Informatique 2013
Version 6.4

Champs disciplinaires, unités d'enseignements
Les champs disciplinaires "informatique" (environ 50% des enseignements)
Le champ disciplinaire **"Algorithmique - Programmation - Langages"** couvre l'ensemble du spectre de l'activité de développement de logiciel. Outre la présentation des bases théoriques de la construction d'un programme (algorithmique, décomposition de problèmes en sous-problèmes, mécanismes de validation),...
Les champs disciplinaires de culture scientifique, sociale et humaine (environ50% des enseignements)
Le champ disciplinaire « Mathématiques », support théorique des technologies de l'information et de la communication, apporte les connaissances reliées au domaine informatique : l'arithmétique pour la théorie de la cryptographie, l'algèbre linéaire pour la théorie du codage, l'analyse et la géométrie pour le traitement des signaux et des images, les probabilités et les statistiques pour l'informatique de gestion et le traitement des données sans oublier les graphes, langages et les grammaires pour la théorie des langages et l'étude des réseaux. Globalement, ces enseignements participent aussi au développement de l'aptitude à l'expression et à la communication scientifique ainsi que de l'aptitude à la formalisation et à la modélisation.

Conclusion : *notre hypothèse est que, comme c'est le cas dans les DUT informatique, l'apparition de l'informatique chasse l'algorithmique du domaine officiel des mathématiques*

Cas 2 : *que faire dans le cas où le cursus ne prévoit pas d'enseignement informatique ?* C'est le cas au lycée actuellement.

Mais d'abord : cherchons une autre fonction si il en est une.

3 Concepts de l'algorithmique et de sa didactique

3.1 Qu'est-ce qu'un algorithme ?

3.1.1 Un peu d'histoire

Entre les XIIème et XVème siècle, le mot désigne les techniques opératoires sur les entiers et les décimaux héritées des arabes par opposition à celles héritées des romains, pronées par les abacistes. Bien que cet aspect conduit à un débat important au primaire (à propos de la question suivante : quel sont les avantages respectifs de l'usage des calculatrices et de l'usage des techniques classiques dans les apprentissages des opérations), nous ne l'abordons pas (remarquer que la question ne se pose plus depuis des décennies en ce qui concerne la notion de racine carrée !).

Au XIXème et au début du XXème, le terme est utilisé dans le champ philosophique dans le cadre d'étude sur les différentes formes de pensée [3].

Bien sûr, des algorithmes existent depuis bien avant ces deux périodes, mais le terme lui même ne semble pas avoir été utilisé avant le XX ème siècle dans d'autres acceptions que les deux citées ci-dessus. Il n'apparaît pas par exemple dans l'article de Tarry concernant le problème ci-dessous [4].

3.1.2 Le problème du labyrinthe :

Les gardes de Minos vous ont conduit à l'intérieur du labyrinthe et vous y abandonnent !
Question 2 : Montrer qu'il est possible de sortir !
Démontrer le Théorème : il existe un chemin qui conduit de tout point d'un labyrinthe vers la sortie.

Quelques réponses possibles :

Réponse 1 : suivre au retour le chemin de l'aller. Comparer avec l'argument "le labyrinthe est modélisé par un graphe connexe et la définition de la connexité entraine la conclusion".

Réponse 2 : suivre le fil d'Ariane.

Réponse 3 : suivre les déplacements suivi pour l'aller en inversant les directions et l'ordre.

Réponse 4 : la méthode de Tarry :

> G. Tarry, "Le problème des labyrinthes", Nouvelles Annales de Mathématiques, ser. 3, 14 (1895), 187-190.
> Tout labyrinthe peut être parcouru en une seule course, en passant deux fois en sens contraire par chacune des allées, sans qu'il soit nécessaire d'en connaître le plan.
> Pour résoudre ce problème, il suffit d'observer cette règle unique :
> Ne reprendre l'allée initiale qui a conduit à un carrefour pour la première fois que lorsqu'on ne peut pas faire autrement.
> Nous ferons d'abord quelques remarques.....
> En suivant cette marche pratique, un voyageur perdu dans un labyrinthe ou dans des catacombes, retrouvera forcément l'entrée avant d'avoir parcouru toutes les allées et sans passer plus de deux fois par la même allée.
> Ce qui démontre qu'un labyrinthe n'est jamais inextricable, et que le meilleur fil d'Ariane est le fil de raisonnement.

Bilan :
⋆ une méthode ne peut être retenue comme algorithmique si elle utilise une connaissance globale de la structure de donnée ;
⋆ la question des moyens disponibles dépasse la seule remarques précédente : peut on écrire sur les murs ? a-t'on pu écrire sur les murs ?

On cherche à mettre en évidence que si la description de ce que doit être un algorithme n'est pas facile, le consensus pour savoir parmi les méthodes précédentes lesquelles sont algorithmiques est facile à atteindre.

---

3. *Un algorithme conventionnel - qui d'ailleurs n'a de sens que rapporté au langage - n'exprimera jamais que la nature sans l'homme. Il n'y a donc pas à la rigueur de signes conventionnels, simple notation d'une pensée pure et claire pour elle-même, il n'y a que des paroles dans lesquelles se contracte l'histoire de toute une langue, et qui accomplissent la communication sans aucune garantie, au milieu d'incroyables hasards linguistiques.* M. Merleau-Ponty, Phénoménologie de la perception,1945, p. 219.ou *Mais le cerveau, nous l'avons assez vu, possède une activité propre ; il peut opérer des constructions, à base d'images ou d'algorithmes, à l'aide desquelles il relie entre elles les sensations et les transforme en « objets ». Il est parfaitement vrai que la perception, conformément à la thèse de Lachelier et de M. Brunschvicg, suppose déjà une « réflexion », du même ordre que celle qui fait la science.* R. Ruyer, Esquisse d'une philosophie de la structure,1930, p. 218.

4. On peut trouver dans la littérature mathématqiue des XVIIIème et XIXème de nombreuxautres exemples de ce type.

3.1.3 L'algorithmique et l'informatique

On raconte que les autorités militaires américaines ont évalué le coût de la réalisation et du test de programmes à conduit à un montant exhorbitant : l'algorithmique a pu apparaître comme une façon d'augmenter la productivité des informaticiens. [5]

On trouvera facilement de nombreuses variantes comme les *commandements* de Mac Naughton :

- ⋆ Un algorithme doit pouvoir être écrit dans un langage de programmation ;
- ⋆ Son exécution dépend d'une donnée d'entrée et produit une donnée de sortie
- ⋆ Il exécute les unes après les autres un nombre fini étapes ;
- ⋆ Son déroulement à chaque état dépend exclusivement de son état.

Bilan : *la description de la notion d'algorithme par les informaticiens vise à encadrer une pratique intégrée au travail de conception de programme.*

3.2 de la *notion* au *concept* [6]

Lors du deuxième congrès international des mathématiciens, à Paris en 1900, David Hilbert présente une liste de problèmes parmi lesquels *Entscheidung der Lösbarkeit einer diophantischen Gleichung. Eine diophantische Gleichung mit irgendwelchen Unbekannten und mit ganzen rationalen Zahlkoefficienten sei vorgelegt : man soll ein Verfahren angeben, nach welchen sich mittels einer endlichen Anzahl von Operationen entscheiden lässt, ob die Gleichung in ganzen rationalen Zahlen lösbar ist.* [7]

Pour en arriver à un théorème énonçant l'inexistence d'une telle méthode, il aura fallu donner un statut mathématique à la notion de méthode. C'est ce qu'a fait par exemple Turing en définissant ses fameuses machines, sous une forme comme suit :

> Une machine de Turing est un septuplet $(Q, \Gamma, B, \Sigma, q_0, \delta, F)$ où
> Q est un ensemble fini d'états ;
> Γ est l'alphabet de travail des symboles de la bande ;
> $B \in \Gamma$ est un symbole particulier (dit blanc) ;
> Σ est l'alphabet des symboles en entrée ($\Sigma \subseteq \Gamma - \{B\}$) ;
> $q_0 \in Q$ est l'état initial ;
> $\delta : Q \times \Gamma \to Q \times \Gamma \times \{\leftarrow, \rightarrow\}$ est la fonction de transition ;
> $F \subseteq Q$ est l'ensemble des états acceptants (ou finaux, terminaux).

Nous n'écrivons pas tout : il s'agit ici seulement de montrer qu'une telle définition appartient bien au champ des mathématiques.

Conclusion : *tant qu'il s'agit de produire des modes de résolution d'un problème (de la même façon de résolution des équation polynomiales), il y a facilement consensus sur ce qui relève de la* **méthode** *et ce qui relève de* **l'algorithme**. *La nécessité d'une* **définition** *n'apparaît que lorsqu'il s'agit de démontrer une inexistence.*

3.3 Y-a-t'il des concepts spécifiques à l'informatique ?

Alice veut s'identifier chaque fois qu'elle allume Bob son ordinateur.

Elle a donc sauvegardé un mot de passe, et chaque fois qu'elle allume son ordinateur, elle écrit son mot de passe et Bob vérifie en comparant avec celui qu'elle a sauvegardé.

5. **des références manquent ici : consulter un article ultérieur.**
6. Cette distinction est utilisée par exemple dans Chabert et al. "histoire d'algorithmes"
7. De la possibilité de résoudre une équation diophantienne. Soit la donnée d'une équation diophantienne à un nombre quelconque d'inconnues et à coefficients entiers rationnels : on doit trouver une méthode par laquelle, au moyen d'un nombre fini d'opérations, on pourra décider si l'équation est résoluble en nombres entiers rationnels

Mais Charles le pirate peut lire le contenu de la mémoire. **Question : comment empêcher qu'il ne découvre puis utilise le mot de passe ?**



Réponse : on utilise une fonction f difficilement inversible : la mémoire de l'ordinateur contient un algorithme de calcul de f et l'image par f du mot de passe **a** d'Alice. Quand quelqu'un veut s'identifier avec un mot de passe **b**, Bob utilise l'agorithme pour calculer $f(\mathbf{b})$ et compare avec la liste des autorisations. Ce procédé est efficace si la connaissance de l'algorithme de calcul d'une image par f et celle de $f(\mathbf{a})$ ne permet pas de déterminer **a**.



Alice veut pouvoir communiquer avec Bob à distance. Charles le pirate apprend comment intercepter les messages. **Question : Comment faire pour que les informations qu'il découvre ne lui permettent pas de se faire passer pour Alice ?**

Question : comment deux interlocuteurs peuvent se mettre d'accord sur un code secret de telle manière qu'un piratage soit repéré ?

Question : comment deux interlocuteurs peuvent se mettre d'accord sur un code secret par des échanges entièrement publics ?

Pour la réponse à ces questions, il faudra se reporter à l'article qui développera ce travail !

Conclusion : *il existe des concepts spécifiques à l'informatique : parmi ceux-ci, les* **fonctions difficilement inversibles**, *qui utilise les concepts* **rapidité d'exécution d'un programme** *vs* **complexité en temps d'une algorithme**.

3.4 Y-a-t'il des concepts partagés entre mathématique et informatique ?

On veut commencer à préciser ici les relations entre l'algorithmique et (de part et d'autre) les sciences du numérique et les mathématiques, (en particulier spécifier autant que possible les problèmes, concepts et méthodes relevant de chacune de ces disciplines). Euh... en fait surtout du côté des mathématiques. Hors de tout contexte d'enseignement, la tomographie médicale conduit, par un chemin qui sera décrit en détail dans un article ultérieur, à un problème de tomographie discrète comme le suivant

Dans la grille suivante, on a compté le nombre de pixels noirs par horizontales et par verticales

1					
2					
3					
1					
	1	3	1	2	

compléter la grille

Une fois le problème résolu dans le cas de figure proposé, (et même avant), on se convainc aisément qu'un algorithme exhaustif, qui envisage toutes le grilles possibles et vérifie si leurs projections tomographiques conviennent, permet de résoudre ce problème.

Question : comment représenter toutes les grilles ?

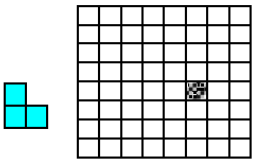
Réponse : une façon de représenter ces grilles consiste à représenter chacune par un entier écrit en binaire, de longueur 16, chaque chiffre représentant la coloration noir/blanc de chacune des cases : par exemple 1000000000001000 représente la grille ayant un pixel noir en haut à gauche et un pixel noir en bas à gauche...

Conclusion : *l'identification et le codage d'une structure de donnée est un exemple de problème partagé entre informatique et mathématique ; plus généralement aussi, la représentation des structures finies. La difficulté pour un mathématicien de répondre spontanément à cette question montre la spécificité de certains concepts des sciences du numérique.*

3.5 Démarche algorithmique

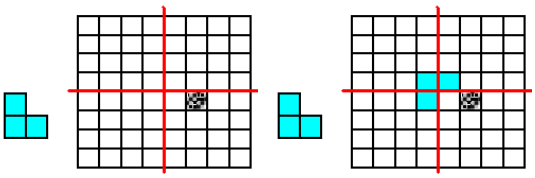
3.5.1 Et tu recommences

Question : résoudre le problème suivant :
Etant donné un carré de taille 2^n, n quelconque avec un « trou » d'une case, pour quelles positions du trou est-il pavable par des triminos coudés ? Ci-contre, un cas particulier [a]



a. D'après D. Grenier et C. Payan ; voir par ex *Deux exemples de situations de recherche pour la classe*

Réponse :



On découpe le carré en quatre carrés de même taille. En posant un trimino sur les trois petits carrés qui ne comportent pas le trou, on obtient quatre petits carrés comportant un trou. Ansi, si il est possible de résoudre le problème pour des carrés de taille 2^n, il est possible de le résoudre le problème pour des carrés de taille 2^{n+1}.

L'algorithme évoqué ci-dessus réoud le problème (démontre le théorème) de manière complète et acceptable par un mathématicien. Pourtant, la réalisation d'un programme requiert une technicité assez importante.

Conclusion : *une démarche algorithmique peut se développer dans un cadre mathématique sans qu'il soit nécessaire de recourir à l'informatique pour la justifier, l'illustrer, l'institutionnaliser, ...*

C'est la méthode de diviser pour régner, récursive, ... : pour ranger un paquet de copies dans l'ordre alphabétique, j'en prend quatre que je range dans l'ordre alphabétique, puis quatre autres, puis je fusionne les deux tas. Et ainsi de suite...

Les deux autres éléments clés de la démarche algorithmique que nous avons identifiés à ce jour :

3.5.2 Algorithmes exhaustifs

Les algorithmes cités au début de la partie précédente sont tous des algorithmes exhaustifs. Un algorithme exhaustif permet la résolution d'un problème de recherche d'informations sur la partie d'un ensemble donné définie par une contrainte sur les éléments de cet ensemble. Il est possible de le décliner sous des variantes adaptées à l'information cherchée :

⋆ existe-t'il des éléments x de X satisfaisant $P(x)$?

> Pour chaque élément x de X, tester si $P(x)$;
> Dès qu'un test est positif, répondre "il existe des éléments x de X satisfaisant $P(x)$" ;
> Si aucun test n'est positif, répondre le contraire.

⋆ combien d'éléments x de X satisfont $P(x)$?

Pour chaque élément x de X, tester si $P(x)$;
Quand un test est positif, incrémenter le compteur des réponses de 1 ;

La question à envisager ici est la suivante :

Rédiger un algorithme exhaustif relève-t'il des mathématiques ?

On pourrait d'abord considérer la question de savoir ce qu'est *rédiger un algorithme exhaustif*? Si nous nous sommes bien compris nous même, il ne reste plus qu'à

⋆ écrire une fonction qui teste si $P(x)$;

⋆ énumérer tous les éléments de X.

D'un point de vue mathématique, ceci ne pose pas de problème. En revanche, l'énumération effective des objets considérés, dans un projet de programmation, peut être difficile : on a vu de telles structures de données plus haut. On peut aussi penser à *énumérer tous les graphes à 12 sommets*, *énumérer tous les arbres bianires de profondeur 6*, ...

3.5.3 Systématiser le tâtonnement

C'est la méthode du juste prix, du balayage, de la dichotomie. On trouvera dans un article à venir la relation d'une démarche d'investigation menée sur le tableur colorcalc dans une classe de ZEP en 1990, où les élèves mettenet en oeuvre spontanéement une telle démarche dans la recherche d'un encadrement de $\sqrt{32}$.

4 *Exemples de situations d'enseignement*

On va maintenant utiliser les outils mis en place pour analyser des situations d'enseignement. Ces trois situations données ici sans commentaires sont destinée à provoquer une analyse contradictoire.

4.1 Première situation

Etude d'une suite
On considère la suite (un) définie pour tout entier strictement positif par :

$$u_1 = 0 \text{ et } u_{n+1} = u_n + ln(n) - 5 \text{ epourtout } n \geq 1$$

1) Ecrire un algorithme qui calcule et affiche les dix premiers termes de la suite.

$u_1 = 0,000$	$u_4 = -13,208$	$u_7 = -23,421$
$u_2 = -5,000$	$u_5 = -16,822$	$u_8 = -26,475$
$u_3 = -9,307$	$u_6 = -20,213$	$u_9 = -29,395$

Que peut-on conjecturer sur le sens de variation et la limite de la suite u_n ?
2) Modifier l'algorithme pour qu'il affiche les termes de la suite. Que penser des conjectures faites au 1).
3) Démontrer qu'à partir d'un certain rang la suite est croissante.
4) Ecrire un algorithme qui renvoie le rang k à partir duquel la suite u_n dépasse N. Que peut-on conjecturer sur la limite ?
5) Démontrer les conjectures.

4.2 Deuxième situation

Un jeu de loterie
Un organisateur de jeux propose « Le double tirage », jeu de son invention qui se déroule de la manière suivante : ⋆ le joueur commence à donner une mise fixée à 3 euros.
⋆ il fait ensuite tourner une roue comme celle dessinée dans le schéma ci-dessous. S'il tombe dans la zone bleue, il tire une boule dans l'urne 1 ; si c'est dans la zone verte, il tire une boule dans l'urne 2 ; si c'est dans la zone jaune, il gagne 20 euros.
⋆ l'urne 1 contient huit boules noires et deux boules rouges ; l'urne 2 contient sept boules noires et trois boules rouges. Si le joueur tire une boule rouge, il gagne 5 euros, sinon il perd sa mise.

Partie A : On commence par vouloir simuler le jeu à l'aide d'un algorithme.
1) Que fait cet algorithme :

```
VARIABLES
roue EST_DU_TYPE NOMBRE
DEBUT_ALGORITHME
roue PREND_LA_VALEUR random()
SI (roue<(205/360)) ALORS
DEBUT_SI
AFFICHER "Piochez dans l'urne 1"
FIN_SI
SINON
DEBUT_SINON
SI (roue<(340/360)) ALORS
DEBUT_SI
AFFICHER "Piochez dans l'urne 2"
FIN_SI
SINON
DEBUT_SINON
AFFICHER "Félicitations ! Vous avez gagné 20 euros"
FIN_SINON
FIN_SINON
FIN_ALGORITHME
```

2) Le modifier pour qu'il simule une partie et affiche le gain algébrique.
3) Ecrire l'algorithme.
4) Le tester pour plusieurs parties.
5) Modifier l'algorithme pour qu'il simule un nombre donné de parties et affiche le gain moyen.
6) A l'aide d'un arbre de probabilité, calculer la probabilité de gagner 5 euros, celle de gagner 20 euros puis celle de perdre sa mise. En déduire le gain moyen.

4.3 Troisième situation : recherche exhaustive

Variations sur une activité de mathématiques incluant l'algorithmique autour du calcul de $d(n, a, b, c)$

• Première version

Déterminer le nombre de triplets (x, y, z) d'entiers positifs tels que $ax + by + cz = n$, où a, b, c et n sont des entiers strictement positifs.

• Deuxième version

⋆ **Première étape** On suppose que $a = 13$, $b = 17$, $c = 25$ et $n = 39470$. A l'aide du tableur, trouver le nombre de triplets (x, y, z) d'entiers positifs tels que $ax + by + cz = n$.

⋆ **Deuxième étape**

On suppose que $a = 134937$, $b = 173238$, $c = 374227$ et $n = 100300401$. Trouver le nombre de triplets (x, y, z) d'entiers positifs tels que $ax + by + cz = n$.

• Troisième version

On veut rechercher le nombre de triplets (x, y, z) d'entiers positifs tels que $ax + by + cz = n$, où a, b, c et n sont des entiers strictement positifs.

⋆ **Première variante**

Première étape

Pour s'entrainer, on recherche le nombre de couples (x, y) d'entiers positifs tels que $ax + by = n$, où a, b et n sont des entiers strictement positifs.

On envisage d'utiliser l'algorithme suivant :

```
Entrée: entiers poisitifs a, b, c et n
   Pour i entier
      Pour j entier
         si a × i + b × j = n alors r := r + 1
```

Que penser de cette proposition ?

Deuxième étape

Adapter l'algorithme à la recherche du nombre de triplets (x, y, z) d'entiers positifs tels que $ax + by + cz = n$, où a, b, c et n sont des entiers strictement positifs.

⋆ **Deuxième variante**

Première étape

Pour s'entrainer, on recherche le nombre de couples (x, y) d'entiers positifs tels que $ax + by = n$, où a, b et n sont des entiers strictement positifs.

On envisage d'utiliser l'algorithme suivant :

```
Entrée: entiers poisitifs a, b, c et n
   Pour i entier de 1 à n
      Pour j entier de 1 à n
         si a × i + b × j = n alors r := r + 1
```

Que penser de cette proposition ?

Deuxième étape

Adapter l'algorithme à la recherche du nombre de triplets (x, y, z) d'entiers positifs tels que $ax + by + cz = n$, où a, b, c et n sont des entiers strictement positifs.

⋆ **Troisième variante**

Première étape

Pour s'entrainer, on recherche le nombre de couples (x, y) d'entiers positifs tels que $ax + by = n$, où a, b et n sont des entiers strictement positifs.

On envisage d'utiliser l'algorithme suivant :

```
Entrée: entiers poisitifs a, b, c et n
   Pour i entier de 1 à n/a
      Pour j entier de 1 à n/b
         si a × i + b × j = n alors r := r + 1
```

Que penser de cette proposition ?

Deuxième étape

Adapter l'algorithme à la recherche du nombre de triplets (x, y, z) d'entiers positifs tels que $ax + by + cz = n$, où a, b, c et n sont des entiers strictement positifs.