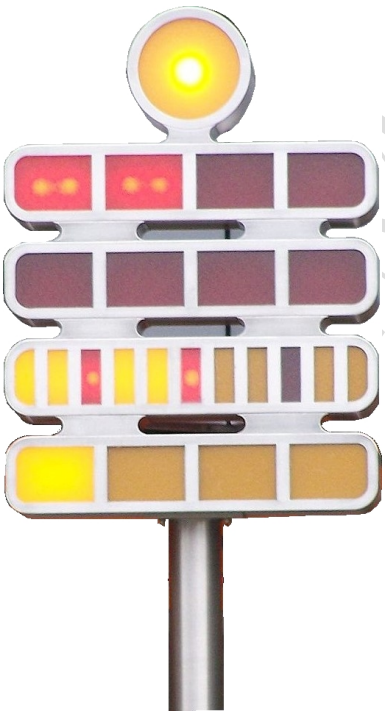
Université de Poitiers

Faculté des Sciences Fondamentales et Appliquées Université de Poitiers

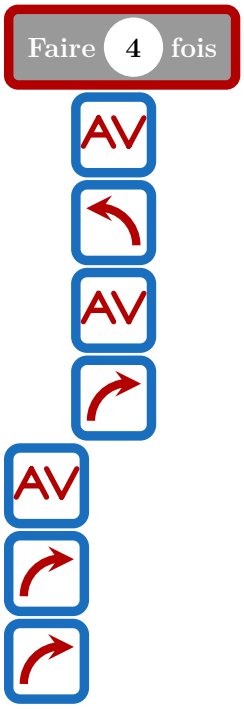
IREM Poitiers

Mathématiques vivantes au lycée
(Fascicule 1)

Algorithmique et Programmation en seconde



```
from turtle import *

for i in range(1,6):
    forward(10)
    right(90)
    formward(100)
    left(90)
    formward(100)
    left(90)
    formward(100)
    right(90)
forward(100)

main.loop()
```



Algorithmique débranchée

Algorithmique et arithmétique

IREM de Poitiers, Groupe Lycée

Septembre 2018

IREM&S Poitiers

Les mathématiques vivantes au lycée

Fascicule 1

ALGORITHMIQUE ET PROGRAMMATION EN SECONDE

Par le groupe Lycée de l'IREM de Poitiers

IREM&S Poitiers

Ce travail s'inscrit dans le cadre d'une recherche initiée par la commission Inter IREM Didactique.

IREM&S Poitiers

Préface : le mot des directeurs

Nous, vous, mathématiciens de métier, savons tous que les mathématiques ne sont pas immobiles, qu'elles évoluent avec le temps, mais aussi totalement en imbrication avec la société, avec les autres sciences, qu'elles soient humaines, sociales ou évidemment les sciences de la nature, et maintenant l'avènement de l'ère du numérique. Mais est-ce que nos élèves, leurs parents, ou l'ensemble de la société ont bien conscience de cette omniprésence bienveillante et bienfaisante des mathématiques ?

Certains objecteront que cette omniprésence serait une de ces *fake news* lancées par le lobby des mathématiciens qui souhaitent à toute force préserver leur pré carré [1], maintenir artificiellement l'importance de cette discipline dans la formation et la sélection, préserver leur métier de professeurs ou chercheurs tout juste bons à tyranniser les élèves ou à construire des concepts abscons... nous aimerions bien les convaincre de l'inanité de leur conception des mathématiques et de leur enseignement, mais la société ne laisse pas assez la parole à ses scientifiques pour que cela se passe aussi facilement ! Il est donc un chemin détourné que l'équipe lycée de l'IREM de Poitiers va prendre dans cet ouvrage : en passant par vous, collègues enseignants, formateurs et inspecteurs, nous pouvons essayer de montrer aux futurs citoyens, enfants de ces sceptiques, l'importance d'une formation mathématique ouverte et de qualité pour la compréhension du monde.

Depuis de nombreuses années les mathématiciens essayent de *vendre* leur science en passant par les jeux, la vulgarisation, des prophètes arachnophores, ... pour jouer un jeu de séduction, mais le vrai objectif n'est pas seulement dans la séduction : il faut aussi que le contenu scientifique « dur » passe dans le temps scolaire. C'est ce double objectif que les membres enseignants de l'équipe lycée de l'IREM de Poitiers suivent depuis de nombreuses années, en construisant une méthodologie fondée sur des approches didactiques dont ils vous parleront mieux que nous dans les pages qui suivent, mais surtout en la faisant vivre dans des *parcours* proposés aux élèves. Ces parcours mélangent mathématiques et questionnements non mathématiques, intéressement et investissement de l'élève, et mise en place de notions et résultats dans une progression pédagogique.
Certains de ces parcours et des articles présentés dans cet ouvrage ont eu une vie antérieure sur le site web de l'IREM de Poitiers avant d'évoluer, de vivre devant des élèves, et de revenir amendés, augmentés ou précisés.

Bonne lecture, et longue vie aux mathématiques.

Julien Michel, directeur de l'IREM de Poitiers 2012-2018

Youssef Barkatou, directeur de l'IREM de Poitiers 2018-...

1 Auraient-ils la notion correcte de carré sans les mathématiques ?

IREM de Poitiers

IREM&S Poitiers

Table des matières

IREM&S Poitiers

IREM&S Poitiers

Programmation en seconde : deux articles

De la notion d'algorithme depuis 2010 aux concepts de programmation en 2017, l'informatique s'est incluse dans l'enseignement des mathématiques au lycée. Cette implantation complète les structures ICN et ISN, toutes appelées à une prochaine transformation lors de la réforme du futur lycée en 2019, où sont prévus un enseignement commun obligatoire « Sciences Numériques et Technologie » en seconde, et une spécialité « Numérique et Sciences Informatiques » dans le cycle terminal.

Avec l'initiation à l'algorithmique depuis le cycle 1 (!) jusqu'au cycle 4 avec le désormais traditionnel exercice de brevet en Scratch, on peut dire que la boucle est bouclée et que l'installation de l'informatique est avérée dans l'enseignement secondaire en France.

Côté contenus enseignés, l'arrivée de l'algorithmique en classe de seconde en 2010 a chassé de manière incongrue l'arithmétique, terrain ô combien fécond et judicieux pour écrire des algorithmes.
Ce paradoxe est souligné par la présence importante de l'arithmétique dans les exemples du document officiel *Ressources pour le lycée*[2] (juin 2017) sur le site ministériel Eduscol.
Entre des professeurs férus d'informatique et d'autres hostiles à des contenus qui ne sont pas ceux pour lesquels ils sont devenus enseignants, les contenus d'informatique apparaîtront soit justifiés, soit déraisonnables.
Les problèmes techniques ne sont pas en reste : peut-on enseigner la programmation sur un écran de calculatrice graphique ? En classe entière ? Combien de « salles informatiques » par établissement ?

Les deux articles qui suivent concernent la classe de seconde.
Le premier article, basé sur des travaux de Laurent Signac[3] et Sylvie Alayrangues et Samuel Peltier[4], s'attaque à la didactique de l'algorithmique. Selon les auteurs de l'article, il ne suffit pas de se mettre sur une machine, observer la syntaxe sur des programmes déjà écrits et les « faire tourner » en tâtonnant pour avoir une idée de leur structure. Pour eux, il est utile de prendre un temps d'algorithmique « débranchée » pour comprendre ce que sont une boucle, une procédure ou une variable, et en quoi cela s'impose pour remplacer des instructions trop nombreuses ou moins performantes. L'intégration de ces contenus à une progression mathématique annuelle n'a pas été oubliée pour autant.
Dans le second article, c'est autour des durées et des objets pour indiquer le temps (montres, horloges..) qu'on traite l'algorithmique de la classe de Seconde et sa mise en œuvre sur machine. Côté mathématique, ce sont les attendus[5] d'arithmétique de fin de cycle 4 qui sont nécessaires. Une banque d'exercices variés est proposée, chacun d'entre eux ayant pour objectif l'écriture d'un programme en Python permettant de résoudre automatiquement le problème.

IREM&S Poitiers

2 http://cache.media.eduscol.education.fr/file/Mathematiques/73/3/Algorithmique_et_programmation_787733.pdf

3 Enseignant à l'université de Poitiers, que nous remercions chaleureusement pour ses relectures.

4 Maîtres de conférences en informatique à l'Université de Poitiers

5 http://cache.media.education.gouv.fr/file/CSP/00/0/Projet-ajustement-clarification-Programmes_maths-C_2-3-4-31_mai_2018_966000.pdf pages 28 (arithmétique) et 33 (algorithmique et programmation)

Partie A.

De l'algorithmique débranchée vers la programmation en seconde

IREM&S Poitiers

Dans un article nommé « The Algorithms of Our Lives »(2013), Lev Manovich demande : « *Comment le logiciel que nous utilisons influence ce que nous exprimons et imaginons ? Devons-nous accepter les décisions prises pour nous par des algorithmes si nous ne savons pas comment ils fonctionnent ? Qu'est-ce que cela implique d'être citoyen d'une société basée sur le logiciel ? Ces questions et d'autres tout aussi importantes attendent d'être analysées.* [1] »

L'informatique a pris une place importante dans notre vie : utilisation de logiciels, d'objets connectés ou programmés pour agir, navigation sur Internet... Peu nombreux sont ceux qui comprennent les outils mis en jeu dans ces diverses pratiques. On peut se poser la question de savoir si cela est vraiment nécessaire : souvent nous ne savons pas comment fonctionne notre voiture et pourtant cela ne nous empêche pas de la conduire. D'autre part, il serait illusoire de croire que tout un chacun peut (ou pourra) lire un programme informatique et le comprendre ce qui peut d'ailleurs paraître inutile tant l'informatique en général et les langages de programmation en particulier évoluent vite. Néanmoins, on peut penser qu'avoir quelques rudiments d'algorithmique (sans nécessairement parler de programmation) pourrait permettre aux citoyens que nous sommes de mieux comprendre notre environnement numérique et nous permettre de prendre des décisions plus éclairées.

1. Les programmes

Depuis la rentrée 2017, le programme de mathématiques de seconde donne une part importante à l'algorithmique et à la programmation. Déjà le programme du cycle 4 a permis aux élèves de travailler un certain nombre de notions du programme de seconde : les variables, les boucles (bornées ou pas) et les instructions conditionnelles.
Cette nouvelle partie s'intitule **algorithmique** et **programmation** dans les deux programmes. Mais, alors que celui de collège met l'accent sur la programmation en cycle 4, celui de seconde différencie bien algorithmique et programmation en leur attribuant la même importance comme le montrent les extraits suivants.

IREM de Poitiers

1 The Algorithms of Our Lives — Lev Manovich (2013)

Programme du collège (cycle 4 - thème E)

Au cycle 4, les élèves s'initient à la **programmation** évènementielle. Progressivement, ils développent de nouvelles compétences , en **programmant** des actions en parallèle, en utilisant la notion de variable informatique, en découvrant les boucles et les instructions conditionnelles qui complètent les structures de contrôle liées aux évènements.

Attendu de fin de cycle
Écrire, mettre au point et exécuter un **programme** simple.

Connaissances et compétences associées
Décomposer un problème en sous-problèmes afin de structurer un **programme** ; reconnaitre des schémas.
Écrire, mettre au point (tester, corriger) et exécuter un **programme** en réponse à un problème donné.
Écrire un **programme** dans lequel des actions sont déclenchées par des évènements extérieurs.
Programmer des scripts se déroulant en parallèle.

- Notions d'**algorithme** et de **programme**.
- Notion de variable informatique.
- Déclenchement d'une action par un évènement, séquences d'instructions, boucles, instructions conditionnelles.

Programme du lycée :

La **démarche algorithmique** est, depuis les origines, une composante essentielle de l'activité mathématique. Au cycle 4, en mathématiques et en technologie, les élèves ont appris à écrire, mettre au point et exécuter un **programme simple**. Ce qui est proposé dans ce programme est une consolidation des acquis du cycle 4 autour de deux idées essentielles :

- la notion de fonction d'une part, et
- la **programmation** comme production d'un texte dans un **langage informatique** d'autre part.

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

- à décrire des **algorithmes en langage naturel** ou dans un **langage de programmation** ;
- à en réaliser quelques-uns à l'aide d'un programme simple écrit dans un **langage de programmation** textuel ;
- à interpréter des **algorithmes** plus complexes.

Un **langage de programmation** simple d'usage est nécessaire pour l'écriture des programmes.[. . .]
L'**algorithmique** a une place naturelle dans tous les champs des mathématiques [. . .]
À l'occasion de l'écriture d'**algorithmes** et de **petits programmes**, il convient [. . .]

Le programme de collège suggère l'utilisation d'un système de programmation par blocs sans institutionnalisation de connaissances en algorithmique. En revanche celui du lycée impose de développer des compétences à la fois en algorithmique et en programmation à l'aide d'un langage textuel.

IREM&S Poitiers

La transition Collège - Lycée dans ce domaine ne va pas de soi. Les collégiens ayant utilisé un système de programmation par blocs (*Scratch* étant le plus répandu) n'en sont pas devenus pour autant des programmeurs experts. En effet la pratique de *Scratch* (ou de tout autre système de programmation par blocs) à ce niveau fait apparaitre des insuffisances :

- des structures limitées (par exemple, le compteur non apparent dans la boucle « Repeat » ne permet pas de l'utiliser facilement dans la boucle) ;
- des blocs masquant des structures algorithmiques élémentaires dont les élèves ne peuvent pas percevoir l'utilité, ni en comprendre le fonctionnement.

Il nous apparait donc important en début de seconde de travailler spécifiquement l'algorithmique avant d'aborder la traduction dans un langage textuel, afin de distinguer les difficultés relevant de l'algorithmique de celles liées au langage de programmation.

En algorithmique, on rencontre souvent des difficultés d'ordre mathématique :

- rigueur et structuration d'une démarche (d'un raisonnement)
- conceptualisation de la notion de boucle liée à la notion d'itération
- utilisation des booléens (dans les instructions conditionnelles).

En programmation, d'autres obstacles interviennent

- affectation et gestion des variables.
  - ▷ différence entre `:=` et `=` pour certains langages
- syntaxe propre à chaque langage
  - ▷ `for i in range (1,17)` pour une boucle de 1 à 16 (*Python*) ;
  - ▷ le signe `==` pour un test d'égalité (*Algobox*, *Python*,...) ;
  - ▷ axe des ordonnées orienté vers le bas de l'écran pour la gestion des pixels ;
  - ▷ importance et utilité des indentations dans les boucles (*Python*) [2].

D'autre part, les nouveaux programmes de mathématiques de seconde, accordant une importance nouvelle à « l'algorithmique et la programmation », précisent que ces démarches trouvent leur place dans toutes les parties du programme. En conséquence, les enseignants de mathématiques ne devraient pas avoir de problèmes pour les inclure dans leur progression... pourtant des difficultés apparaissent en pratique.

Une première d'ordre matériel
L'accès en salle informatique est encore limité dans la plupart des établissements.

D'autres d'ordre pédagogique

- L'écriture d'un algorithme n'est pas le but unique ; il s'agit aussi de l'implémenter dans un langage ce qui exige des compétences que tous les professeurs de mathématiques ne possèdent pas nécessairement.
- Motiver l'écriture d'algorithmes permettant d'accomplir des tâches complexes et/ou répétitives n'est pas aisé quand des outils, tels que tableurs et calculatrices, plus facile d'accès, ont déjà été utilisés par les élèves et sont tout aussi intéressants à travailler.

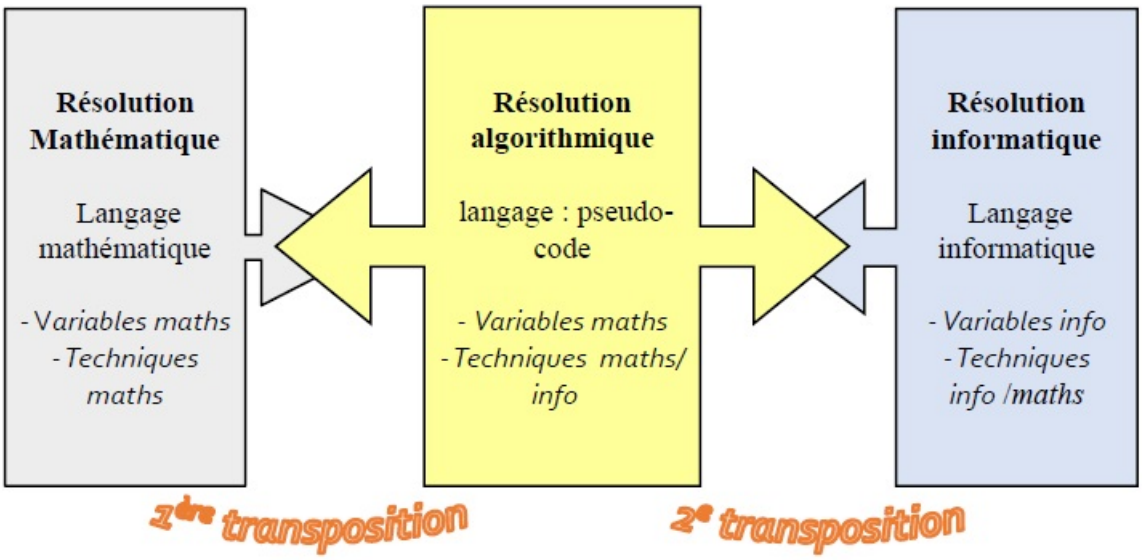
2 Dans *Python*, l'indentation n'est pas seulement un artifice graphique mais est nécessaire pour la structuration du programme.

IREM de Poitiers

2. Les avantages du débranché

Dans la suite nous prendrons pour définitions d'algorithme et programme celles que donne Simon Modeste dans sa thèse de doctorat en mathématiques [3]. Il définit un algorithme comme « une procédure de résolution de problèmes (s'appliquant à une famille d'instances du problème) et produisant, en un nombre fini d'étapes constructives, effectives, non ambiguës et organisées la réponse au problème ». Le programme est, quant à lui, « la traduction, l'implémentation d'un algorithme dans un langage de programmation afin de l'exécuter. »

On peut donc considérer que dans l'enseignement des Mathématiques, **l'algorithmique est la partie conceptuelle de la programmation informatique** ce que confirme Nathalie Briant [4] qui explique par le schéma ci-dessous l'ordonnancement des taches à accomplir et qui montre que la première transposition à faire ne peut être informatique. Si dans la résolution d'une famille de problèmes, le premier travail est mathématique, la recherche d'une technique de résolution (qui constituera l'algorithme) n'est pas encore un travail informatique à proprement parler.



Cette analyse faisant suite à la mise en évidence des difficultés répertoriées (de manière non exhaustive) dans la partie précédente, nous amène à penser qu'avant de placer les élèves de seconde face à un logiciel de programmation, il est souhaitable de leur faire travailler la nature même d'un algorithme. Les problèmes rencontrés lors de l'apprentissage de la syntaxe d'un langage de programmation risquent de masquer les difficultés d'ordre spécifiquement algorithmique notamment la conceptualisation de structures telles que affectation, répétition, boucle, condition. Une solution envisagée pour contourner cet obstacle est de se libérer de l'outil informatique et de commencer par un travail qualifié de « débranché » c'est-à-dire sans la moindre machine.

3 Voir *Enseigner l'algorithmique, pourquoi ? Quels apports pour l'apprentissage de la preuve ? Quelles nouvelles questions pour les mathématiques ?* [9]

4 Voir *Étude didactique de la reprise de l'algèbre par l'introduction de l'algorithmique au niveau de la classe de seconde du lycée français* [2]

À cet effet, un groupe de l'IREM de Poitiers a construit deux situations permettant un enseignement de l'algorithmique « débranchée » :

- l'une intégrée à un parcours [5] « Comment dessiner une figure sous contraintes » dans lequel sont aussi travaillées des notions purement géométriques et algébriques. Une des activités consistera à reproduire une figure sur un quadrillage représentant un écran informatique (composé de millions de pixels).
- l'autre menée parallèlement à un travail sur les fonctions.

Pour chacune de ces situations nous expliquerons nos choix, exposerons les observations faites lors de leur utilisation avec les élèves, dresserons un bilan et envisagerons d'éventuelles modifications.

3. Dessiner avec un jeu de cartes

a. Structuration

Nous avons imaginé un jeu de cartes [6] permettant de travailler la notion d'algorithme avec un nombre minimum d'instructions de base mais sans limitation du nombre de cartes.
Ce travail d'une durée de 3 ou 4 heures ne nécessite pas de salle informatique mais est sûrement plus productif et confortable pendant les heures en demi-classe où il est plus facile d'observer et de comprendre le comportement des élèves.
Par groupe de deux, ils sont amenés à écrire un algorithme (avec les cartes), qui sera lu et exécuté par un autre groupe d'élèves. Ainsi chaque production sera contrôlée par des pairs et entrainera des discussions.

Pour reproduire une figure, il est indispensable de faire une liste d'instructions claires, ordonnées et sans ambiguïté. Trois instructions élémentaires nous sont apparues nécessaires pour réaliser des constructions basiques (lettres, chiffres ou autres figures simples) sur un quadrillage : 

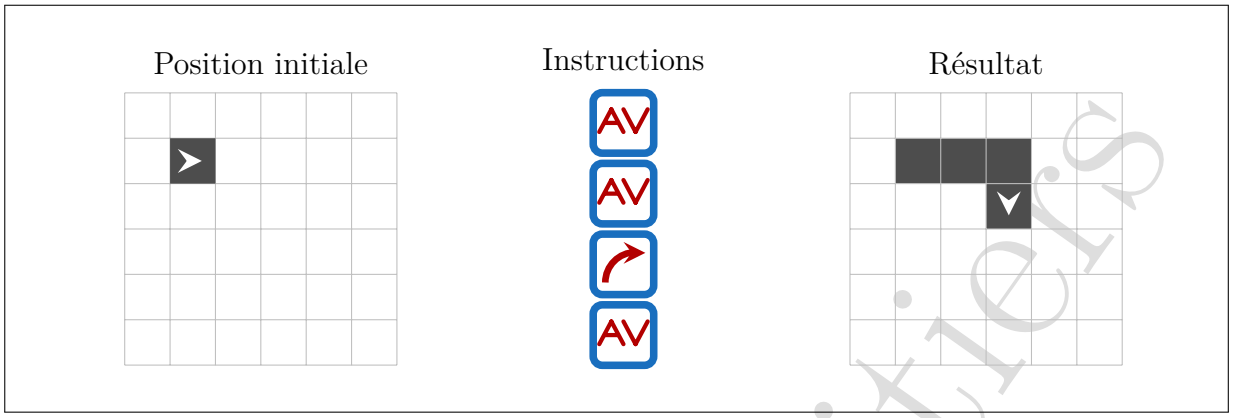
,  , et  .

IREM de Poitiers

5 Brochure de l'IREM de Poitiers 2011 : *Enseigner les Mathématiques en Seconde : Deux parcours sur la géométrie plane.*

6 Sur le site de l'IREM de Poitiers http://irem.univ-poitiers.fr/PF_brochures/dokebrochures/ figurent les planches de cartes imprimables.

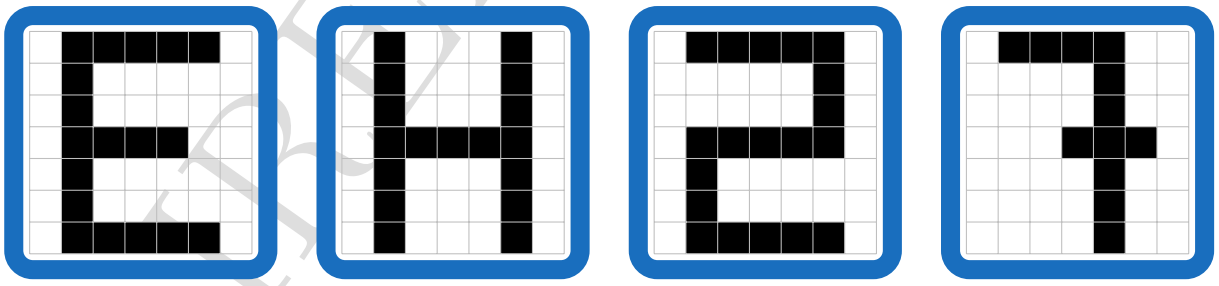
Pour que les élèves comprennent bien la signification de chacune des cartes, on leur propose une courte liste d'instructions et le dessin obtenu. Il est nécessaire d'accompagner la liste d'instructions d'une grille précisant la position initiale (lieu et orientation du curseur) de manière à signaler aux utilisateurs le point de départ du motif à reproduire.



Il s'agit pour les élèves de définir sans ambiguïté l'action représentée par chacune des trois cartes et l'ordre de lecture des instructions. Laissés seuls face à cet exemple, ils sont vite capables de définir chacune des trois cartes, mais quelques questions légitimes se posent sur le sens de lecture de la liste d'instructions.

- AV Avancer d'une case dans la direction du curseur en noircissant la case d'arrivée.
- Pivoter le curseur sur place d'un quart de tour dans le sens des aiguilles d'une montre.
- Pivoter le curseur sur place d'un quart de tour dans le sens inverse des aiguilles d'une montre.

Une fois les bases éclaircies, chaque groupe disposant d'un jeu de cartes comportant 25 cartes AV, 10 cartes et 10 cartes , va devoir lister et ordonner les instructions nécessaires pour la reproduction d'une figure qui leur sera donnée (lettre, chiffre, forme géométrique...) telles celles ci-dessous.



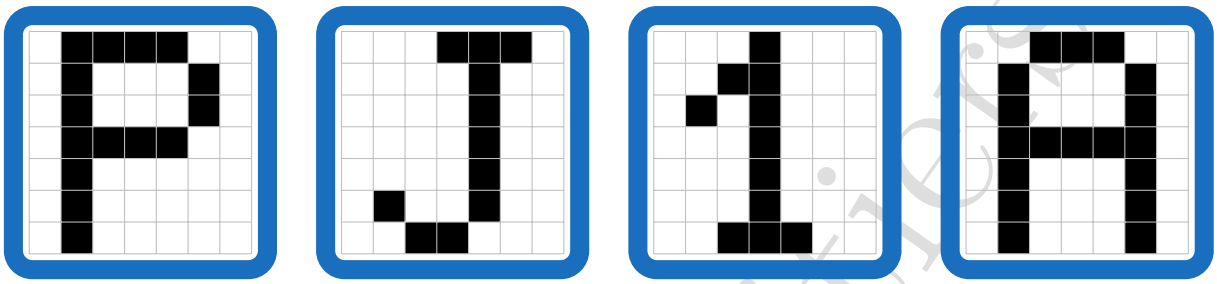
Les élèves décident seul de leur point de départ et de l'orientation de départ, mais ils devront en faire part au groupe qui devra lire et exécuter leur algorithme.

Une fois la liste de cartes posée sur la table, les groupes changent de place et tentent de suivre l'algorithme proposé par un autre groupe. Les élèves perçoivent alors l'importance d'être rigoureux. La validation est souvent rapide pour la plupart des groupes, mais une discussion intervient parfois entre certains en désaccord sur les dessins obtenus. On observe que les erreurs d'écriture d'algorithmes sont plus fréquentes que les erreurs de lecture.

IREM&S Poitiers

On peut, après ces deux activités, définir clairement un algorithme comme une liste ordonnée d'instructions simples et non ambiguës ayant pour but, dans cette première approche, de reproduire un dessin.

Certains groupes, plus rapides, ont essayé d'écrire un algorithme permettant de dessiner les lettres et chiffres ci-dessous (le choix de la position initiale du curseur est encore laissé aux élèves) :



Très vite, ces élèves se rendent compte de la nécessité de créer de nouvelles instructions. Certains proposent une carte « **crayon barré** » dans le but de pouvoir avancer sans dessiner. Mais on peut leur faire remarquer qu'il y a incompatibilité entre cette carte et l'instruction de base « avancer » 

qui demande clairement d'avancer en noircissant la case. Une autre proposition consiste à créer une carte pour avancer directement en diagonale vers la droite (ou vers la gauche). Certains groupent sont tentés de créer des cartes permettant d'avancer de plusieurs cases en même temps.

Il est intéressant de les laisser créer les cartes de leur choix, à condition d'être très clair sur leur effet, leur intérêt et leur compatibilité avec les trois cartes initiales. Tous les élèves n'ayant pas eu le temps d'aborder ces nouveaux dessins, les situations proposées dans la suite ne nécessiteront pas l'utilisation de ces nouvelles cartes.

b. Répétitions

Présentation de nouvelles figures aux élèves

Après avoir travaillé avec les cartes de base du jeu, le but à atteindre dans un second temps est la compréhension et l'apprentissage de la structure de boucle, très importante en informatique. On propose donc aux élèves de reprendre le même travail que précédemment avec des dessins comportant des motifs « répétitifs » tels que ceux ci-dessous :

Figure A

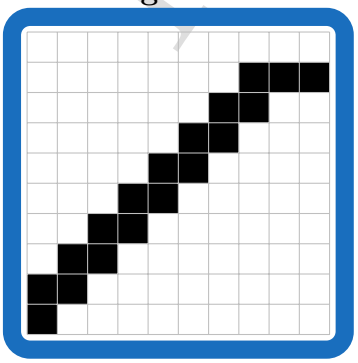


Figure B

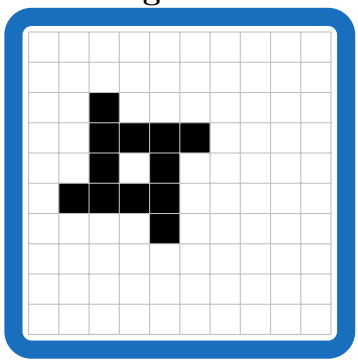
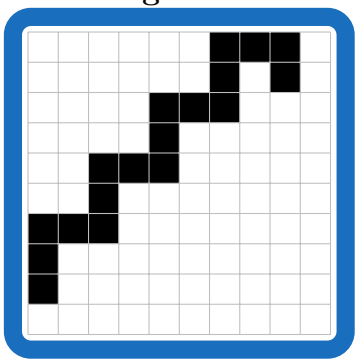
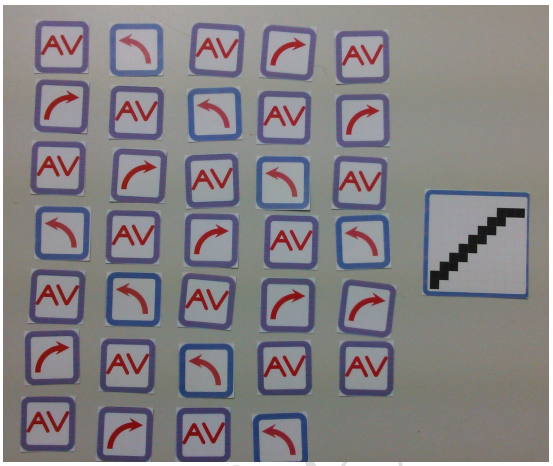


Figure C



IREM de Poitiers

Pour réaliser ces nouveaux dessins, les élèves sont amenés à utiliser un grand nombre de cartes, ce qui rend la lecture de l'algorithme fastidieuse (comme celui obtenu pour la figure A ci-contre).
Certains groupes sont même limités par le nombre de cartes dont ils disposent [7].
Le professeur va profiter de ces deux raisons pour rajouter des nouvelles cartes au jeu.



Introduction de trois cartes vierges pour de nouvelles instructions

En utilisant trois cartes vierges, les élèves vont devoir inventer des nouvelles instructions permettant d'obtenir un algorithme plus concis.
Dans les propositions faites par les élèves, il y a une certaine économie d'instructions par rapport à un algorithme n'utilisant que les trois cartes de base. On observe deux types de cartes créées :

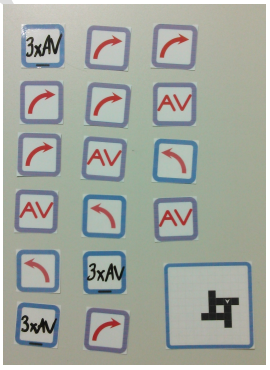
- celles qui s'apparentent à des procédures informatiques
- celles qui permettent de réitérer des instructions (les boucles).

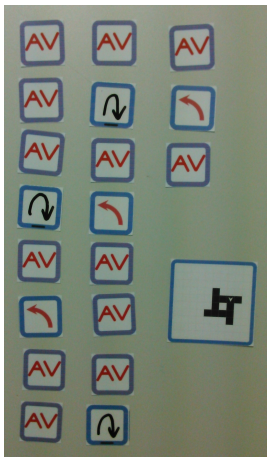
Cartes apparentées à une procédure

Des élèves ont ici créé deux cartes nouvelles pour l'algorithme permettant de reproduire la figure B .

- une carte $\boxed{3 \times \text{AV}}$ qui remplace donc trois cartes 
- une carte demi-tour qui remplace quant à elle deux cartes  ou deux cartes 

Dans les deux cas, le cahier des charges est respecté mais le gain de cartes reste modeste. Néanmoins l'idée sera reprise et développée plus loin (page 12).





IREM&S Poitiers

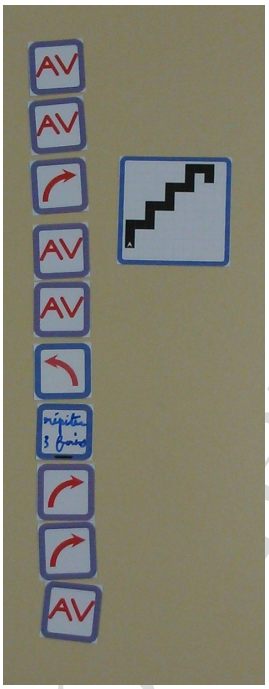
7 Voir le dessin d'un carré dans le fichier des dessins à reproduire sur le site de l'IREM http://irem.univ-poitiers.fr/portail/

Cartes définissant une boucle

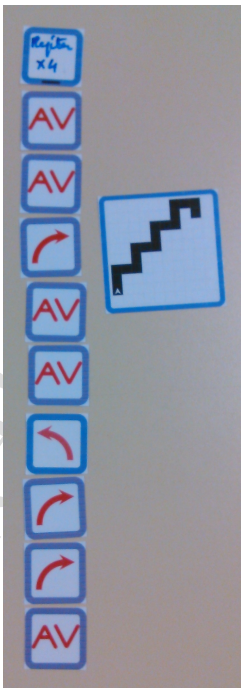
Dans cette nouvelle proposition pour la figure C, de nombreux élèves ont créé de manière naturelle une carte « Répéter … fois » permettant une importante économie de cartes. Mais tous ne l'utilisent pas de la même manière.

- Certains utilisent cette carte pour expliciter la volonté de réitérer une séquence d'instructions énoncée au préalable, manière de faire qui semble être la plus naturelle pour les élèves.
- D'autres, peut-être influencés par une utilisation de scratch au collège, l'utilisent avant une liste d'instructions à réitérer. Le sens du mot « répéter » n'est alors pas celui du sens commun (on ne répète que ce qui a été fait déjà une fois).

  Les élèves utilisent cette instruction dans le sens « faire … fois ».



avec la nouvelle carte :



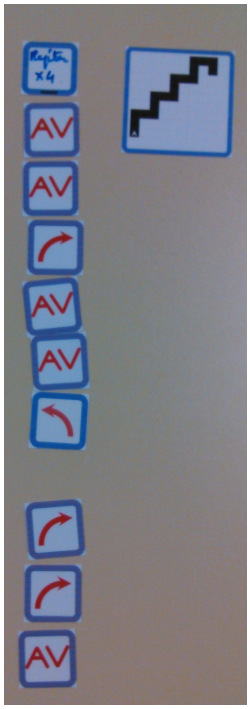
avec la nouvelle carte :

Un débat est lancé pour comprendre l'intérêt des diverses cartes proposées. Celles qui semblent emporter l'adhésion de nombreux élèves sont celles qui permettent de réitérer une liste d'instructions. Elles permettent en effet une économie de cartes parfois très importante.
La carte « Répéter … fois » est donc plébiscitée. Une ambiguïté subsiste toutefois sur les instructions à répéter. Les élèves qui **écrivent** l'algorithme ne sont pas toujours conscients de l'importance de trouver un moyen efficace et clair pour différencier ce qui est répété de ce qui ne doit pas l'être. Ceux qui **lisent** l'algorithme sont confrontés à cette question et l'exprime clairement aux « concepteurs des algorithmes ».

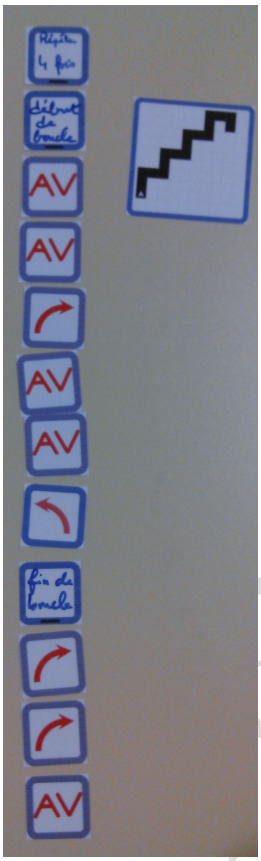
Pour lever cette ambiguïté, les propositions sont à nouveaux diverses :

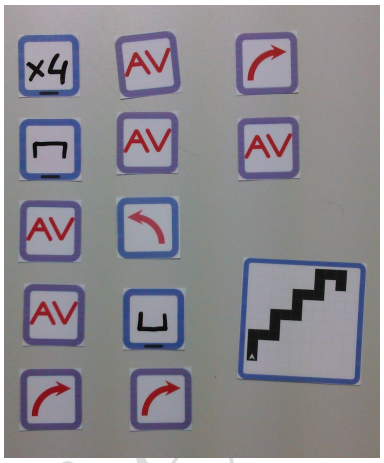
- Certains proposent de dessiner une longue accolade sur la table partant de la première instruction à la dernière devant être répétées. Cette solution ne peut évidemment pas être validée.
- D'autres écartent les cartes pour signifier un changement de « statut des instructions » dans la longue liste. (voir ci-dessous)
- Certains proposent alors des cartes de « début de boucle » et de « fin de boucle ». Cette dernière idée est conforme à de nombreux langages textuels (*Xcas*, *Scilab*, *Fortran*, *Maple* …).

IREM&S PONT



Dans cette proposition, les élèves insèrent tout simplement un espace pour séparer les cartes. Si cette idée est facile à comprendre avec le jeu de cartes, elle ne pourra pas être utilisée dans un langage informatique.





On voit dans ces deux nouvelles propositions, différentes cartes permettant de montrer le début, puis la fin de la liste d'instructions. Certains écrivent clairement « début de la boucle » (respectivement « fin de la boucle »), d'autres utilisent un symbole significatif.

Après discussion et pour remédier au problème de vocabulaire lié au mot « répéter », le professeur propose une nouvelle carte 

avec la possibilité d'écrire le nombre voulu à l'intérieur. Celle ci devra alors être placée avant les instructions à répéter.

Pour éviter d'avoir à construire des cartes particulières mais également pour se rapprocher des éléments syntaxiques du langage *Python*, le professeur propose, pour satisfaire à la seconde contrainte évoquée, d'effectuer un décalage vers la droite – dit indentation – pour toutes les instructions à répéter.

En prenant en compte ces deux propositions visant à lever les ambiguïtés constatées précédemment, les groupes vont modifier leurs algorithmes avec ces nouvelles cartes et contraintes de syntaxe [8].

c. Vers des procédures

Pour les figures ci-après, les élèves parlent assez vite d'un motif de base qu'il serait pratique de pouvoir répéter. Il s'agit alors de créer comme on l'a vu dans le paragraphe précédent (proposition d'une carte 3 × AV), une carte remplaçant une séquence d'instructions écrite à l'aide des cartes de base.

8 Voir annexe 2 (page 33) pour des exemples d'algorithmes.

Figure 1

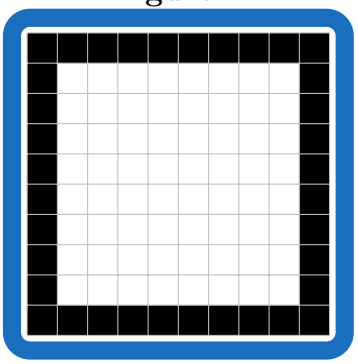


Figure 2

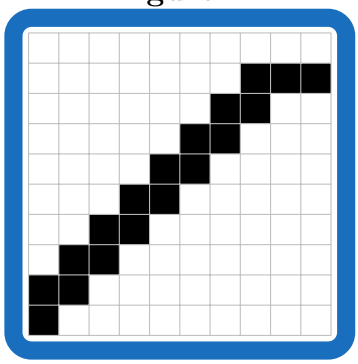
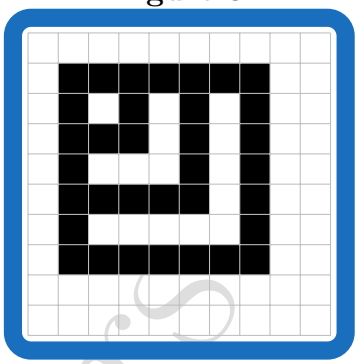


Figure 3



La figure 1 contient quatre lignes, dont l'algorithme de reproduction nécessite une écriture fastidieuse avec les cartes de base.
Les élèves fabriquent très vite l'algorithme (de ce que l'on appellera bientôt une procédure) permettant de construire la carte 

, certains avec les cartes de base, d'autres avec une boucle. L'algorithme de la première figure devient alors plus concis en utilisant cette nouvelle instruction.
On peut amener les élèves à réfléchir à une carte modulable utilisant un paramètre que l'utilisateur changera en fonction de ses besoins : 

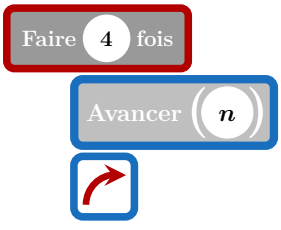
.

La figure 2 est un long zigzag et donc une succession « d'avancer / tourner ».
Les élèves ne sont pas toujours d'accord sur le motif de base[9] à répéter. L'écriture de l'algorithme leur permet de constater qu'une succession de quatre cartes se répètent.
La procédure permettant de créer la carte 

ne pose alors pas de problème et permet d'obtenir un algorithme concis pour construire la seconde figure.

Enfin, **la figure 3** contient trois carrés de différentes dimensions.
Si la décision du choix de la nouvelle carte à construire est rapide, sa mise en œuvre est un peu plus complexe. Pour éviter d'avoir à créer une carte pour chaque carré (de coté différent), l'idée proposée est de créer une carte 

[10] permettant de dessiner un carré de coté $n+1$. Celle-ci doit alors utiliser elle même une carte procédure créée au préalable, contenant un paramètre.

Définition de la carte « Avancer 9 fois » :	**Définition de la carte « Zigzag » :**	**Définition de la carte « Carré de coté $n+1$ » :**
		
		

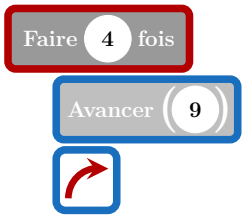
9 Plusieurs choix sont effectivement possibles.

10 Le paramètre de cette procédure ne représentera pas la longueur du côté du carré construit mais le nombre de fois où on avance pour dessiner un côté.

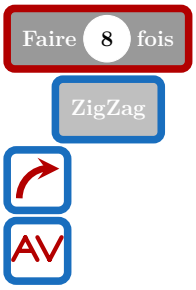
IREM de Poitiers

Avec ces nouvelles cartes, les algorithmes permettant de construire les figures 1, 2 et 3, sont simplifiés [11].

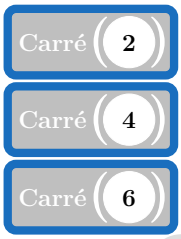
Algorithme figure 1 :



Algorithme figure 2 :



Algorithme figure 3 :



Ces nouvelles cartes créées devront plus tard être implémentées dans *Python*. Il faudra alors programmer des procédures afin de les utiliser dans les algorithmes de construction. Mais avant cela, le passage au langage de programmation devra être préparé : il suffit alors de donner la traduction de chacune des cartes dans le langage de programmation (quel qu'il soit).

Au préalable, une première évaluation [12] est possible (avec les cartes). Elle consiste en deux étapes :

- vérifier que les élèves savent lire , comprendre et suivre un algorithme,
- vérifier que les élèves savent écrire un algorithme permettant de construire une figure donnée.

IREM&S Poitiers

11 Dans le sens raccourcis surtout.
12 Voir page 25.

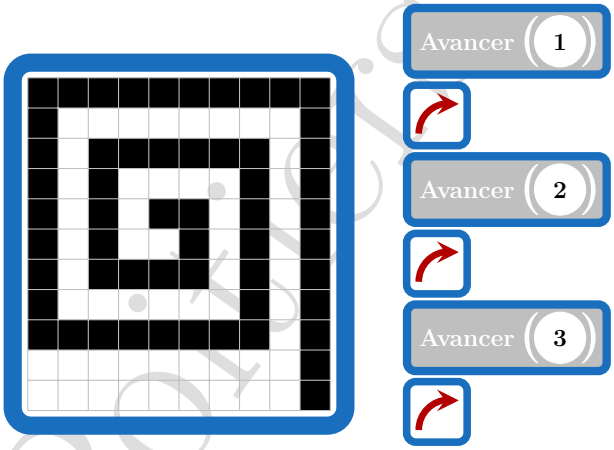
Les limites du jeu de cartes

Le jeu de cartes utilisé jusqu'alors permet de faire comprendre la structure d'un algorithme ainsi que la notion de séquence itérative. Pourtant, pour certains motifs, des difficultés subsistent.

Si l'on considère par exemple la spirale ci-contre, on peut mettre en évidence une des limites de l'utilisation du jeu de cartes. La construction de l'algorithme permettant de la reproduire avec les trois cartes de bases est fastidieuse, elle le reste même en utilisant la carte procédure 

. Les élèves comprennent assez facilement la structure itérative de ce motif avec une incrémentation d'une unité dans la carte avancer. Cependant, la syntaxe choisie lors de la définition de la carte 

ne facilite pas l'accès au compteur de la boucle [13].



On aurait pu créer une carte « Pour i allant de 1 à n » se rapprochant de la syntaxe informatique, mais celle-ci s'éloigne de la proposition naturelle des élèves. De plus la notion de variable n'ayant pas encore été abordée, l'utilisation du compteur i dans cette instruction serait une réelle difficulté à ce moment là. Un jeu de rôle, dans la partie **variables et affectations** page 18, sera l'occasion de revenir sur cette difficulté lors du passage à *Python*. Le plus important à ce moment de l'apprentissage est de consolider les structures de bases.

Mais avant d'aborder ces nouvelles difficultés, il faut passer de l'algorithmique débranchée à la programmation.

d. Premières traductions en *Python*

Le temps passé à utiliser le jeu de cartes a toujours eu pour but d'écrire des algorithmes à implémenter dans un langage textuel. Ce moment n'est alors qu'une simple traduction des instructions représentées jusqu'alors par des cartes. Si les élèves ont bien compris l'usage de ces cartes, le passage à un langage textuel tel que *Python* ne doit pas amener de difficultés importantes. Il faut bien-sûr donner la syntaxe de ce langage et la correspondance avec les cartes.

Remarquons que la traduction de la carte 

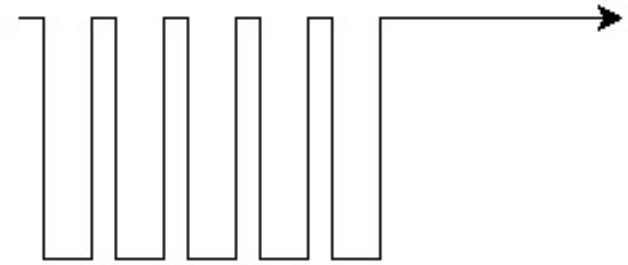
a un effet imperceptible sur un écran d'ordinateur du fait de la taille d'un pixel. Les élèves seront donc amenés à utiliser `forward(n)` traduction de la carte 

.

13 Comme avec *Scratch*, il reste la possibilité, d'utiliser une variable que l'on incrémente à chaque répétition de la boucle.

IREM&S Poitiers

Pour faire le lien entre les cartes connues et les instructions de *Python*, les élèves étudient pour commencer un algorithme, l'exécutent et repèrent l'effet de chacune des instructions utilisées :



```
from turtle import *

for i in range(1,6):
    forward(10)
    right(90)
    forward(100)
    left(90)
    forward(20)
    left(90)
    forward(100)
    right(90)
forward(100)

mainloop()
```

Il leur est facile de faire les correspondances suivantes :

Carte du jeu	Instruction correspondante dans *Python*
	`forward(1)`
	`left(90)`
	`right(90)`
	`forward(n)` [14]
	`for i in range(1,n+1) :`

Notons que l'utilisation de telles instructions dans *Python* nécessite, pour certains interfaces, d'appeler le module *turtle* avec la commande `from turtle import *`.

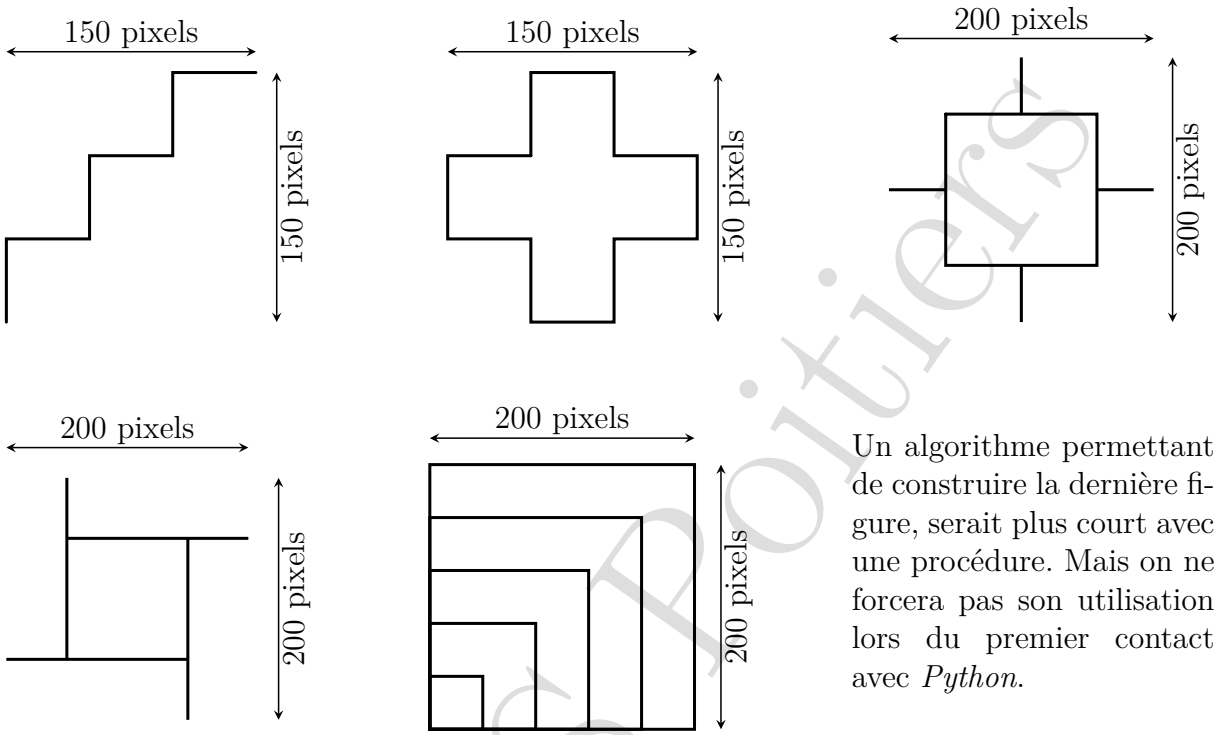
De plus, le curseur, représenté par une flèche, est par défaut placé au milieu de l'écran et pointe horizontalement vers la droite.

Tout langage de programmation a, ainsi, quelques « règles » implicites qu'il faut connaître. Celui-ci est très vite compris par les élèves, même s'il sera la cause de certaines erreurs vite corrigées.

14 L'instruction `forward(n)` de *Python* dessine une ligne de longueur n alors que la fonction  du jeu de cartes donne une ligne de longueur $n+1$ à cause du point de départ déjà noirci.

IREM&S Po

La deuxième étape consiste à écrire le code permettant de construire un dessin donné. Cette étape est plus compliquée puisque les élèves doivent, par eux même, réfléchir aux structures à utiliser ainsi qu'au point de départ de leur construction. Des dessins de complexités croissantes peuvent leur être proposés.



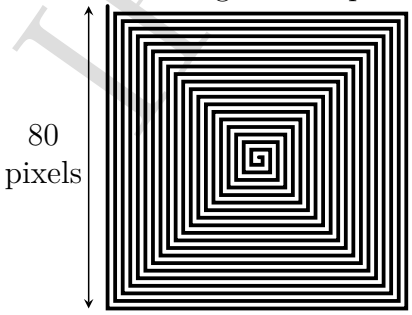
Un algorithme permettant de construire la dernière figure, serait plus court avec une procédure. Mais on ne forcera pas son utilisation lors du premier contact avec *Python*.

Selon le motif à reproduire, les difficultés sont diverses :

- Certains oublient l'orientation de départ de l'ordinateur mais corrige cette erreur après une première compilation (CTRL + E).
- D'autres ont du mal à choisir le point de départ de leur dessin et un motif de base à répéter. On pourra laisser le choix aux élèves (pour cette première utilisation du langage de programmation) de la conception de leur algorithme (avec ou sans structure répétitive). Sans sollicitations, les élèves tenteront assez vite de réfléchir à un moyen de raccourcir leurs algorithmes.

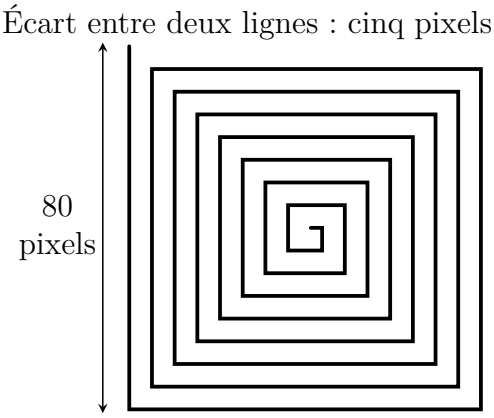
On peut demander aux élèves plus rapides et efficaces, de réfléchir à un algorithme permettant de construire les spirales dont l'écriture avec le jeu de cartes posait problème.

Écart entre deux lignes : un pixel




```
from turtle import*

for i in range (1, 81):
    forward(i)
    right(90)

mainloop()
```

IREM de Poitiers




```
from turtle import*

for i in range (1,27):
    forward(5*i)
    right(90)

mainloop()
```

4. Variables et affectations

Avec le travail réalisé « en débranché », les élèves se sont familiarisés avec la structure d'un algorithme. Mais seules des situations de construction de motifs leur ont été proposées. La notion de variable n'a donc jamais été évoquée. C'est, là encore, une difficulté qu'il ne faut pas négliger.

a. Analyse a priori

On entend parfois l'analogie entre variable informatique et « case mémoire » dans laquelle on stocke un objet (valeur, liste, ...). Cette étape est pertinente si l'on travaille avec des objets non mutables [15] (comme les nombres entiers ou flottants...).
Elle n'est plus valable lorsque les objets utilisés sont des listes par exemple, pour lesquels la notion d'adresse serait plus correcte.
Cependant, pour introduire les notions délicates de variable et surtout d'affectation, l'utilisation d'une enveloppe peut aider à s'avérer efficace pour surmonter les difficultés [16].

Exemple

En langage naturel :	En langage informatique (*Python*)
variable prend la valeur variable $\times 10$	variable = `variable` $\times 10$

Dans cet exemple, que cela soit en langage naturel ou en langage informatique, le « variable » de droite est la valeur prise dans la case mémoire pour faire un calcul dont le résultat sera rangé ensuite dans la « variable » de gauche représentant le nom de la case mémoire. Il faudra donc être bien clair sur la différence entre la valeur et le nom de la case mémoire dans laquelle elle va être rangée. La difficulté sera accentuée quand on voudra implémenter en *Python* un algorithme contenant des affectations puisque la commande « prend la valeur » sera remplacée par un signe =, qui a alors un statut tout à fait particulier, et différent de ceux du signe = en mathématiques connus des élèves (équation, définition, identité).

15 Un objet mutable est un objet dont on peut changer les propriétés une fois qu'il a été défini (comme une liste).

16 Par exemple dans l'instruction *a prend la valeur* $a+1$, les deux « a » n'ont pas le même statut : l'un faire référence à la variable, l'autre à la valeur contenue dans la variable.

APMEP

b. Un jeu de rôle pour comprendre le principe d'affectation

Pour permettre aux élèves de bien différencier ces deux aspects d'une variable, l'IREM de Poitiers propose d'utiliser un jeu de rôle qui aura pour but de faire comprendre la notion d'affectation informatique [17].
C'est à partir d'un nouveau parcours d'étude et de recherche ayant pour thème la modélisation, qu'une nouvelle situation est proposée aux élèves.

La situation

À la naissance de leur enfant, un couple de jeunes parents décide d'ouvrir un compte épargne, d'y déposer une somme initiale puis de l'abonder chaque année de 200 €. Ils désirent connaître la somme dont disposera l'enfant à six ans puis à sa majorité. Après des discussions rapides sur les taux d'intérêts actuels, le professeur propose un compte épargne rémunéré à 2% par an.

Gestion en classe

La demi-classe est divisée en trois groupes de six élèves. dans chaque groupe les élèves jouent chacun leur tour, le rôle d'un banquier chargé de calculer la nouvelle somme sur le compte épargne. On aura au préalable fait le point sur le moyen de faire les calculs correspondant à la transaction :

On multiplie d'abord la somme par 1,02 puis on ajoute 200.

Le professeur initialise le jeu en jouant le rôle des parents :

- il ouvre le compte en utilisant une enveloppe sur laquelle il écrit « Épargne », c'est la **variable** ;
- il écrit ensuite sur un papier la somme initiale qu'il place dans l'enveloppe, c'est la première opération d'**affectation**.

On choisit de ne pas dire au groupe classe le montant de la somme initialement mise dans l'enveloppe pour insister plus tard sur le fait qu'à chaque étape la valeur est perdue quand on la remplace. En réalité, les trois groupes ont la même somme initiale [18], afin que la correction et le bilan soit plus simples.

Chaque élève doit alors :

- ouvrir l'enveloppe, prendre le papier et lire la valeur écrite dessus,
  cela revient à prendre la valeur de la variable Epargne,
- calculer la nouvelle somme,
- et remettre un nouveau papier sur lequel figure le résultat obtenu,
  ce qui correspond a l'affectation d'une nouvelle valeur dans la variable en se substituant à la valeur précédente : Epargne $\leftarrow$ Epargne $\times 1{,}02 + 200$.

IREM de Poitiers

17 Comme le jeu de cartes utilisé précédemment pour travailler la notion d'algorithme.
18 1000 €

Le sixième élève banquier est chargé d'énoncer la somme disponible sur le compte épargne aux 6 ans de l'enfant. Tous sont conscients que les sommes intermédiaires disparaissent au fur et à mesure des calculs de chaque élève. Une fois le jeu de rôle terminé, un point est réalisé en grand groupe, sur les différentes étapes effectuées et une traduction possible dans *Python*.

Le parallèle entre l'expérimentation et le programme informatique est alors possible afin que les élèves implémentent l'algorithme dans l'ordinateur.

Algorithme	**Étapes**	**Traduction en *Python***
Ouvrir le compte et déposer 1000€	Initialisation	`epargne = 1000`
Faire 6 fois de suite Calculer la nouvelle somme après un an	Affectations	`for i in range(1,7):` `    epargne = epargne*1.02+200`
Regarder la somme disponible	Sortie	`print(epargne)`

Les élèves n'ont aucune difficulté pour déterminer la somme disponible sur le compte épargne à la majorité de l'enfant. Il leur suffit en effet d'effectuer un changement de la valeur maximale du paramètre de la boucle « for » dans le programme .

affectation.py

```
1 epargne=1000
2 for i in range (1,19):
3     epargne=epargne*1.02+20
4 print (epargne)
```

On peut, avant ou après le codage en *Python*, faire un bilan sur l'expérimentation effectuée et sur la correspondance de chaque étape avec l'informatique :

- L'enveloppe qui contient la somme sur le compte est appelée **variable** en informatique. On peut lui donner le nom que l'on veut.
- Une variable ne peut contenir qu'une seule **valeur** à la fois.
- **Affecter** une nouvelle valeur à une variable détruit la valeur précédente.

5. Introduire la boucle « While » et consolider la notion de variable

La situation

La même situation peut servir, dans un second temps à faire découvrir la boucle « tant que . . . ». Celle-ci permet de répéter une (ou des) instruction(s) un nombre de fois inconnu tant qu'une condition est respectée [19].

Avec les mêmes conditions bancaires que précédemment, les parents décident maintenant de retirer tout l'argent dès que la somme a dépassé 5 000 €.

On demande alors aux élèves de :

- chercher combien d'années il faudra attendre pour atteindre ces 5 000 € ;
- déterminer le temps nécessaire selon la somme initiale.

19 Les élèves n'ont jusqu'alors utilisé qu'une boucle « for » connaissant au préalable le nombre d'itérations à effectuer.

IREM&SPIRES

Gestion en classe

On laisse cette fois-ci, tous les élèves d'une demi-classe ensemble avec trois enveloppes de couleurs différentes (virtuellement trois comptes épargnes) et trois sommes initiales différentes. Chaque élève jouera le rôle d'un banquier et choisira un camarade à qui il transmettra l'enveloppe. Il leur est alors difficile de compter le nombre de « banquiers » qui voient passer « un compte ». Dès qu'un élève ouvre l'enveloppe et s'aperçoit que la somme souhaitée a été atteinte ou dépassée, l'enveloppe est mise de coté. À la fin de l'expérimentation, les élèves doivent répondre à la question initiale : « combien d'années ont été nécessaires pour atteindre 5000 € sur chacun des trois comptes ? »

Vu le nombre de transactions, ils ne peuvent pas être certains de la réponse. Ils sont alors amenés à comprendre la nécessité d'introduire une seconde variable représentée par une enveloppe « Année ». On reconduit alors cette expérimentation en liant chaque enveloppe Épargne à son enveloppe Année.

Avec les deux enveloppes Épargne et Année, chaque élève n'a accès qu'à une seule ligne des tableaux suivants.

Enveloppe rouge :
Somme initiale : 2 000€

Enveloppe Épargne	Enveloppe Année
2000	0
2240,00	1
2484,80	2
2734,50	3
...	...
...	...
4920,49	11
5218,90	12

Enveloppe verte :
Somme initiale : 1 200€

Enveloppe Épargne	Enveloppe Année
1200	0
1424,00	1
1652,48	2
1885,53	3
...	...
...	...
...	...
4778,16	14
5073,73	15

Enveloppe bleue :
Somme initiale : 600€

Enveloppe Épargne	Enveloppe Année
600	0
812,00	1
1028,24	2
1248,80	3
...	...
...	...
...	...
4842,56	17
5139,41	18

Le jeu terminé, on peut faire le point sur la structure algorithmique à utiliser puis sa traduction en *Python*. Les élèves proposent de manière assez naturelle une structure de boucle sous la forme **« faire... jusqu'à Epargne** $\geqslant 5000$ **»** ou **« répéter ... jusqu'à Epargne** $\geqslant 5000$ **»**. Cette syntaxe est moins générale que celle utilisée dans les langages informatiques et ne permet pas par exemple de faire tourner la boucle 0 fois. Le professeur doit alors indiquer la syntaxe de la boucle correspondante en *Python* : **« tant que *condition* faire *instructions* »**.

Il s'agit alors d'expliquer aux élèves que cette structure nécessite l'évaluation (« vrai » ou « faux ») de la condition à chaque passage dans la boucle. Ce qui mobilise une variable particulière dite de type booléen qui ne peut prendre que deux valeurs « vrai » ou « faux » selon que la condition est satisfaite ou non ; la boucle prend fin dès que la variable booléenne prend la valeur « faux ». Dans le cas présent on écrira donc : « tant que Epargne < 5000 alors instructions ». Il est intéressant de noter que la condition d'arrêt

DEV&S Poitiers

de cette boucle est la négation de celle des propositions faites par les élèves ce qui peut partiellement expliquer leurs difficultés lors de son utilisation.

Une fois ce problème éclairci, le professeur peut alors donner la syntaxe de la nouvelle instruction.

Algorithme	**Traduction en *Python***
Ouvrir le compte et déposer une somme Initialiser l'enveloppe Année	`Epargne = 2000` `Annee = 0`

Faire jusqu'à obtenir 5000€
Calculer la nouvelle somme après un an
Augmenter la variable Année de un an
	`while Epargne < 5000:`
	`Epargne = Epargne*1.02+200`
	`Annee = Annee+1`

Imprimer la somme disponible Imprimer le nombre d'années passées	`print(Epargne)` `print(Annee)`

Comme la première fois, le passage à *Python* ne pose pas de problème. La structure de boucle a déjà été bien expliquée et les élèves ne sont pas perturbés par la syntaxe du langage :

- le symbole « : » à la fin de la ligne « while » a déjà été rencontré lors de la boucle « for ».
- L'importance de l'indentation a déjà été discutée.

Néanmoins il faut insister sur ce deuxième point comme le montre l'exemple suivant :

L'élève a choisi (volontairement ou pas ?) de mettre les instructions « print » à l'intérieur de la boucle. Il se rend alors compte, à l'exécution que cela permet de voir toutes les sommes intermédiaires, ce qui n'avait pas été demandé dans la consigne.

```
#Modélisation d'une épargne

epargne=1200
annee=0
while epargne < 5000 :
    epargne=epargne*1.02+200
    annee=annee+1
    print(annee)
    print(epargne)
```

```
1
1424.0
2
1652.48
3
1885.5296
4
2123.240192
5
2365.7049958400003
6
2613.0190957568
7
2865.279477671936
8
3122.585067225375
...
13
4488.394261082518
14
4778.162146304168
15
5073.7253892302515
```

© EM&S Poitiers

6. Un jeu pour tout réinvestir

Dans toute utilisation d'un langage informatique textuel, il est difficile pour un élève de travailler directement sur machine. Pour comprendre la nécessité d'une boucle, d'une variable et de son utilisation, il est important de réfléchir à l'algorithme sur papier avant de se lancer sur la machine.

Afin de réinvestir tout ce qui a été étudié précédemment, on demande aux élèves d'écrire un programme *Python* permettant à un utilisateur de jouer contre l'ordinateur à un jeu que l'on appellera « le juste prix ».

Pour trouver un nombre entier compris entre 1 et 1000 choisi au hasard par l'ordinateur, le joueur devra faire des propositions successives en tenant compte des réponses de l'ordinateur du type « trop grand » ou « trop petit ». Pour bien comprendre les étapes importantes de ce jeu, l'idée est de faire jouer le rôle de l'ordinateur par un élève et le rôle de l'utilisateur par un autre. Un troisième élève observe le jeu de l'extérieur et doit être capable d'expliciter les différentes étapes, rôle difficile, car l'objectif n'est pas d'analyser la stratégie du joueur mais bien les étapes du jeu.

Les élèves énoncent facilement les étapes telles que le choix au hasard d'un nombre (étape 1), la proposition du joueur (étape 2) puis la réponse de l'ordinateur (étape 3) et la répétition des deux dernières. Cependant il est nécessaire de leur poser des questions sur le rapport entre l'étape 2 et l'étape 3. Les élèves découvrent alors la nécessité d'une condition permettant à l'ordinateur de donner une réponse plutôt que l'autre.

Comme pour le jeu précédent, les élèves voient naturellement une répétition du type « faire ... jusqu'à ce que ... », qu'ils ont déjà été amenés à transformer en une boucle « tant que ..... » et à réfléchir à la condition d'arrêt. Mais une telle structure ne permet pas la réponse de l'ordinateur à la proposition faite par le joueur.

Une fois les différentes étapes du jeu bien énoncées, les élèves sont chargés d'écrire en langage naturel l'algorithme permettant d'y jouer. Sollicité par les élèves, le professeur traduit alors les instructions qu'ils n'ont pas déjà rencontrées : cinq demandes sont ainsi formulées.

- Comment faire choisir un nombre au hasard par l'ordinateur ?
  L'instruction `random.randint(1,1000)` nécessite d'importer le module `random`. Cela ne gène pas les élèves qui ont déjà utilisé précédemment le module de construction `turtle`.
- Comment demander une valeur à l'utilisateur ?
  L'instruction `int(input("Donner un nombre :"))` se décompose en :
  - `input("Donner un nombre")` qui écrit la question sur l'écran, attend une réponse de l'utilisateur et retourne cette réponse sous forme d'une chaîne de caractères.
  - `int(.)` qui donne le statut d'entier à cette réponse.
- Le symbole « différent » pour la condition d'arrêt de la boucle : `!=`
- Comment introduire une condition ?
  On utilise l'instruction `if « condition »:` suivi d'un retour à la ligne avec une indentation

IREM&SPoitiers

- Comment faire afficher un résultat dans la fenêtre d'exécution [20] ?
  La commande `print(".")` permet d'afficher dans la fenêtre d'exécution.

Un exemple d'algorithme proposé par des élèves :

Algorithme	**Traduction en *Python***
Tirage d'un nombre au hasard	`Nombre = random.randint(1,1000)`

Répéter autant de fois que nécessaire
Demander une proposition
Comparer le nombre et la proposition
Réponse de l'ordinateur
	`while Proposition != Nombre:`
	`  Proposition = int(input("Donner un nombre"))`
	`  if Proposition > Nombre:`
	`  print("C'est moins")`
	`  if Proposition < Nombre:`
	`  print("C'est plus")`

Réponse finale de l'ordinateur	`print("C'est gagné")`

Une fois devant la machine, les élèves qui testent cet algorithme constatent l'échec lors de l'exécution.

JustePrix.py

```
1  #Jeu du juste prix
2
3  import random
4
5  Nombre=random.randint(1,1000)
6  while Proposition!=Nombre:
7      Proposition=int(input("Saisissez un nombre entre 1 et 1000 : "))
8      if Proposition>Nombre:
9          print ("C'est moins")
10     if Proposition<Nombre:
11         print ("C'est plus")
12 print ("Bravo, c'est gagné")
```

C'est en consultant la fenêtre d'exécution que les élèves peuvent comprendre qu'il y a un problème avec la variable « Proposition » que le programme ne reconnait pas.

Pour rentrer dans la boucle, et faire la comparaison avec la variable « Nombre », il faut initialiser avec n'importe quelle valeur entière la variable « Proposition » afin qu'il y ait quelque chose à comparer avec « Nombre ».

```
Traceback (most recent call last):
  File "<string>", line 420, in
 run_nodebug
  File "C:\Users\Philippe\Desktop\juste
 prix.py", line 6, in <module>
    while Proposition != Nombre :
NameError: name 'Proposition' is not
 defined
```

Une fois ce problème d'initialisation corrigé, le programme fonctionne. Les élèves peuvent alors essayer de le compléter pour compter le nombre de propositions faites avant de gagner. Il faudra alors utiliser une nouvelle variable, l'initialiser et l'incrémenter [21].

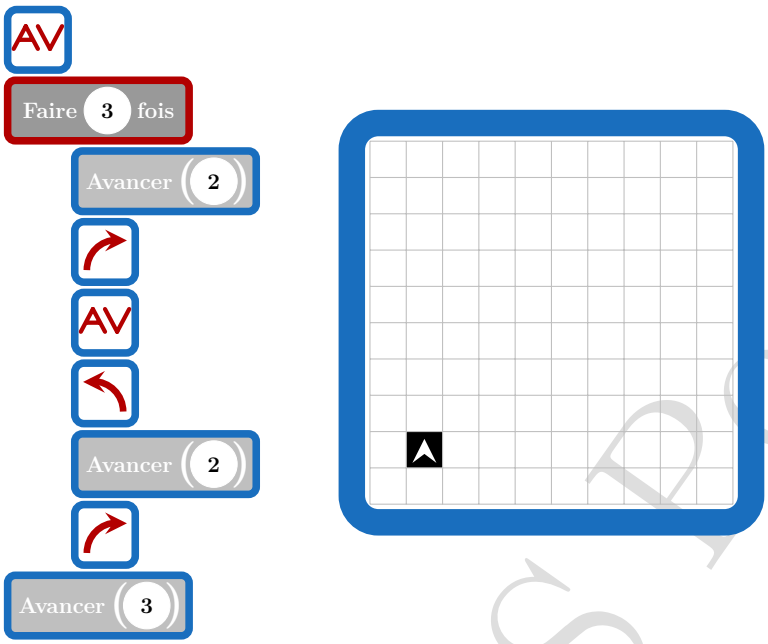
20 La fenêtre d'exécution est la fenêtre de dialogue entre la machine et l'utilisateur, il est appelé par exemple *shell* avec *Pyzo* ou *console* avec *EduPython*.

21 Ajouter 1.

7. Évaluations

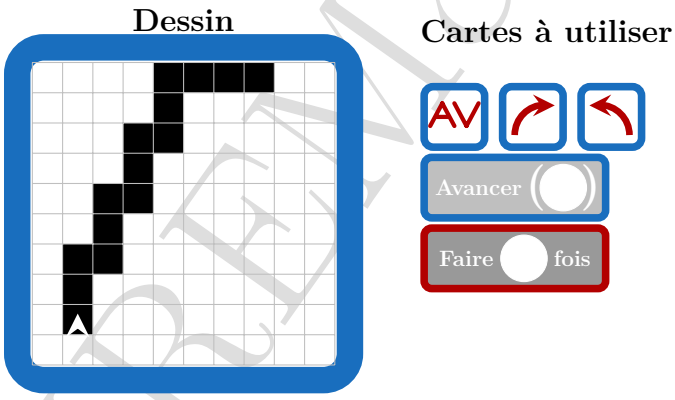
a. Évaluation avec les cartes

Dans un premier temps, il est demandé aux élèves de lire un algorithme et de l'effectuer en traçant la figure :



L'algorithme choisi n'est pas trivial.
Pourtant, cette première question ne pose pas de problème.
85% des élèves réussissent à produire le dessin image correspondant à l'algorithme donné. Les rares à fournir une image erronée se sont trompés de sens ou bien ont oublié que la carte « pivoter » ne noircit pas une case supplémentaire.

La seconde question consiste à demander aux élèves d'écrire un algorithme qui permettrait de reproduire un dessin en utilisant uniquement certaines cartes.



Les résultats sont moins concluants. Tous les élèves ont voulu utiliser une boucle mais n'ont pas toujours réussi à trouver un motif de base à répéter, ou n'ont pas utilisé l'indentation.

b. Évaluation sur le langage *Python*

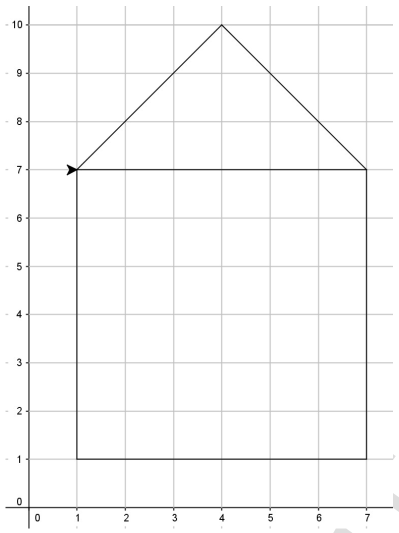
En devoir maison, sur papier

Après deux ou trois utilisations de *Python*, un nouveau type d'évaluation peut être proposée. Celle qui suit, donnée en devoir maison, demande aux élèves de comprendre puis d'écrire un algorithme. Les élèves pouvaient, en cas de besoin implémenter l'algorithme fourni sur un ordinateur pour les aider.

IREM&S Poitiers

Question 1 :
Écrire un algorithme, avec le langage *Python*, permettant de dessiner la figure complète ci-contre (composée d'un carré, d'un triangle équilatéral et d'un triangle rectangle isocèle) en partant du point de départ représentée par une flèche.
On considérera que la largeur d'un pixel sera représentée par un carreau.

. . . .
. . . .
. . . .
. . . .
. . . .
. . . .



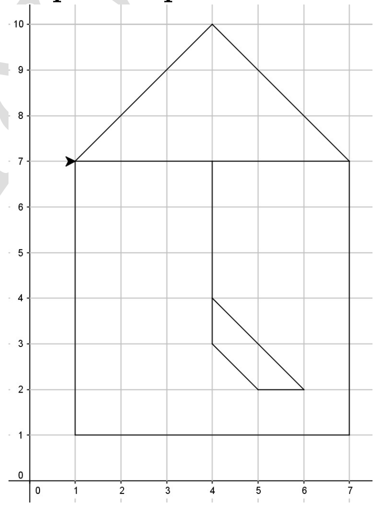
Question 2 :
Compléter la figure à l'aide de l'algorithme suivant, en partant de la même position initiale (le curseur) :
forward(3)
right(90)
forward(4)
left(45)
forward($\sqrt{2}$)
left(45)
forward(1)
left(135)
forward($2\sqrt{2}$)

Comme dans la précédente évaluation, la conception de l'algorithme est moins réussie que la partie lecture, celle-ci étant encore moins bien réussie que la partie lecture de l'évaluation avec les cartes. Mais les difficultés rencontrées par les élèves sont plus d'ordre géométrique (angles et longueurs) que d'ordre algorithmique.
Certains élèves confondent les longueurs $\sqrt{2}$ et 2 alors qu'un travail avait été fait plus tôt dans l'année sur la longueur de la diagonale d'un carré ($\sqrt{2}a$ pour un carré de coté a).

Réponse question 1 [22] **:**

for i in range(1,5) :
    forward(6)
    right(90)
left(45)
forward($3\sqrt{2}$)
right(90)
forward($3\sqrt{2}$)

Réponse question 2 :



IREM&S Poitiers

22 En *Python* le $3\sqrt{2}$ serait écrit `3*sqrt(2)`.

En classe sur papier puis sur ordinateur

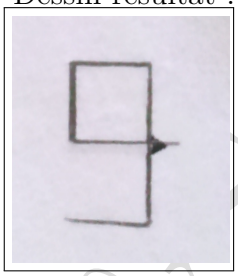
Pour évaluer une dernière fois cette formation à l'algorithmique et au langage *Python*, une autre activité est proposée aux élèves. D'abord sur papier seulement, puis à l'aide de l'ordinateur pour s'auto-corriger. Les vingt-trois élèves présents [23] doivent interpréter des algorithmes. C'est en effet une des attentes du programme qui demande d'entrainer les élèves à :

- décrire des algorithmes en langage naturel ou dans un langage de programmation ;
- en réaliser quelques-uns à l'aide d'un programme simple écrit dans un langage de programmation textuel ;
- interpréter des algorithmes plus complexes.

Algorithme 1

```
from turtle import *

left(180)
forward(50)
right(90)
forward(50)
right(90)
forward(50)
right(90)
forward(100)
right(90)
forward(50)

mainloop()
```

Dessin résultat :



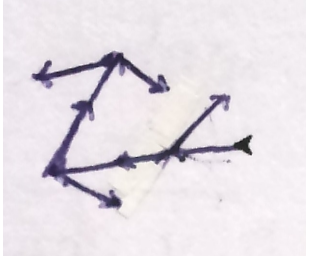
Algorithme 2

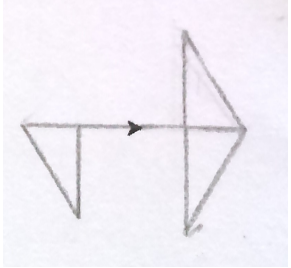
```
from turtle import*
for i in range (1, 4):
    forward(50)
    left(120)
    forward(50)
    left(120)
    forward(50)

mainloop()
```

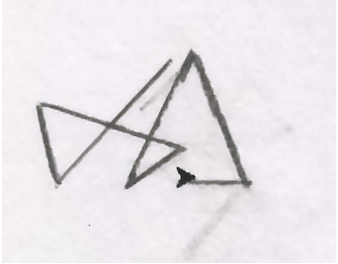
Dessin résultat :

Cette première étape consiste à construire des motifs comme les élèves le faisaient avec le jeu de cartes.

- Le premier algorithme ne pose aucun problème pour 17 élèves sur 23. Pour les autres, c'est encore un problème d'angle ou d'orientation qui fausse leur résultat.
- Le second algorithme est un peu moins réussi puisque 11 élèves sur 23 fournissent le bon logo. Parmi les autres, certains fournissent un dessin représentant un polygone (pentagone, hexagone, octogone). Certaines propositions enfin sont plus énigmatiques.







23 Travail proposé en fin d'année.

IREM&S Poitiers

Dans les deux algorithmes suivants, des variables interviennent. Les élèves sont invités à faire fonctionner les algorithmes, plusieurs fois si nécessaire.

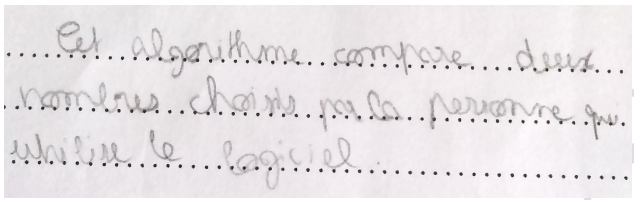
Algorithme 3 :

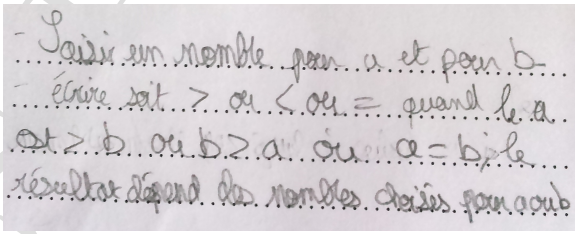
```
a=input("Saisissez un nombre : ")
a=int(a)
b=input("Saisissez un nombre : ")
b=int(b)
if a>b:
    print(a, " >", b)
elif a<b:
    print(b," >", a)
else:
    print(b," =", a)
```

Algorithme 4 :

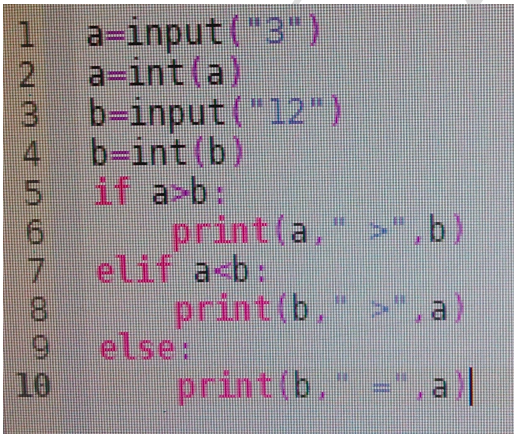
```
a=input("entrez un  nombre :")
a=int(a)
i=0
while i<11:
    print(a*i)
    i=i+1
```

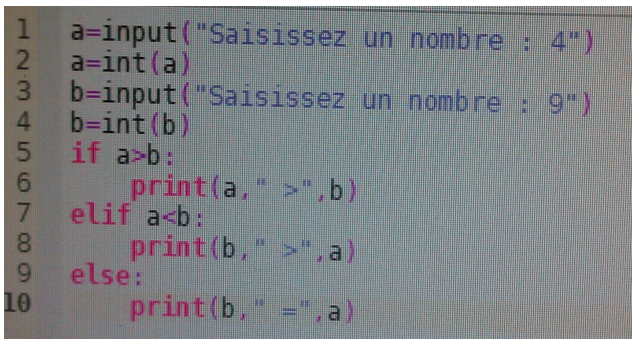
Le troisième algorithme est assez bien compris puisque dix-neuf élèves sur vingt-trois savent (plus ou moins bien) expliquer ce qu'il fait.





Les quatre autres n'arrivent pas à le comprendre lors d'une simple lecture, mais semblent de plus confondre l'algorithme et son utilisation. En effet, lors de son implémentation, ces élèves saisissent un nombre à la suite de la demande faite sans attendre la compilation. Il semble que les lignes « a=input("Saisissez un nombre : ") » et « a=int(a) » ne soient pas comprises.

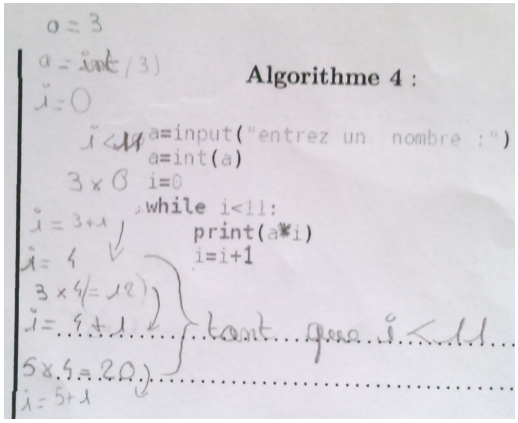




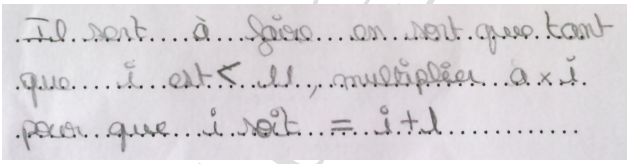
Le quatrième algorithme est le plus compliqué vu les difficultés qu'il cumule : boucle « Tant . . . que », utilisation du compteur dans la boucle et affectation. Volontairement écrit de cette manière peu pertinente car équivalent en réalité à une boucle finie, le pro-

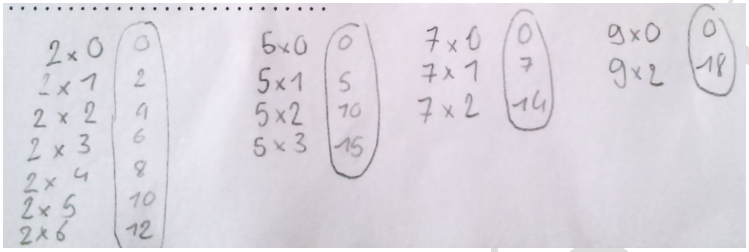
fesseur demande aux élèves l'ayant décrypté, de le réécrire avec une autre sorte de boucle (boucle « Pour i allant de 1 à … ».

Certains, suivant la suggestion du professeur, testent l'algorithme et écrivent le résultat de chacune étape.



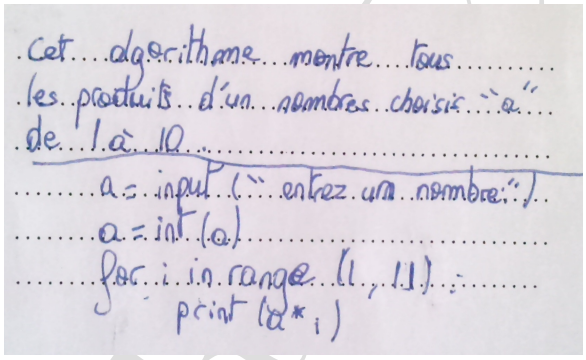
Le premier élève semble avoir compris que la condition d'arrêt de la boucle porte sur i. Pourtant son explication laisse penser qu'il ne comprend pas exactement ce que fait cet algorithme.

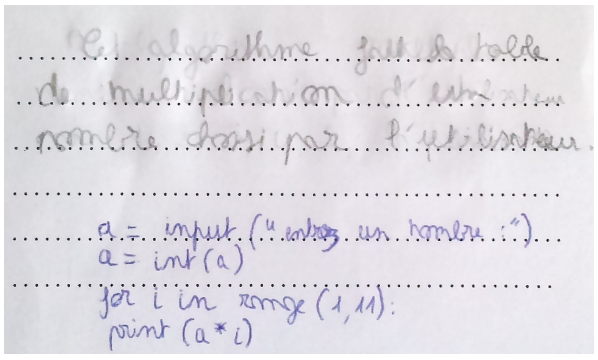




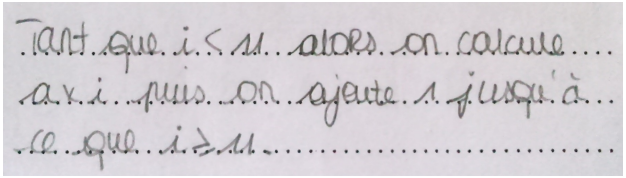
Le second élève, quant à lui, stoppe l'algorithme quand le résultat de la multiplication (dans la variable a) dépasse 11. Cet élève comprend donc la structure itérative, fait les bonnes affectations, mais confond les deux variables.

Cinq élèves sur vingt-trois ont parfaitement compris le but de l'algorithme, l'exprime tout à fait correctement (en parlant de table de multiplication) et proposent un changement de boucle parfaitement écrite.

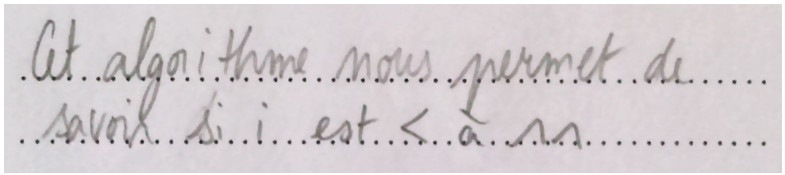


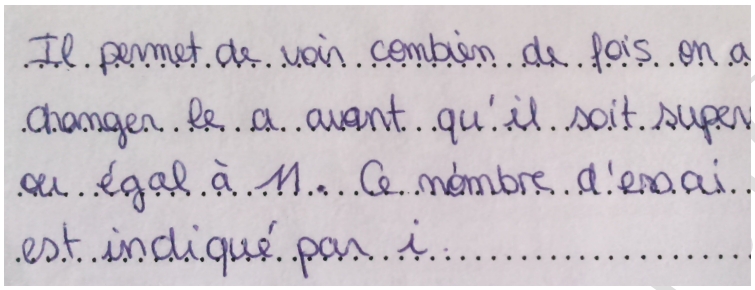


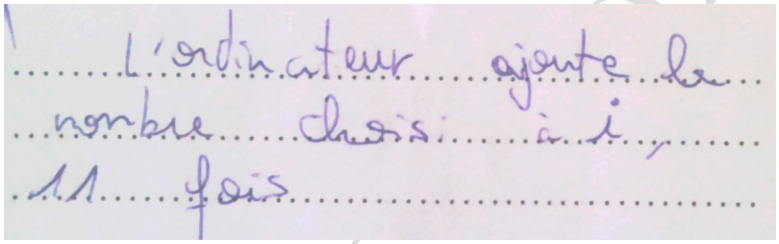
Onze autres se contentent de dire littéralement ce qu'ils lisent dans l'algorithme sans prendre le recul suffisant pour y voir l'écriture d'une table de multiplication.

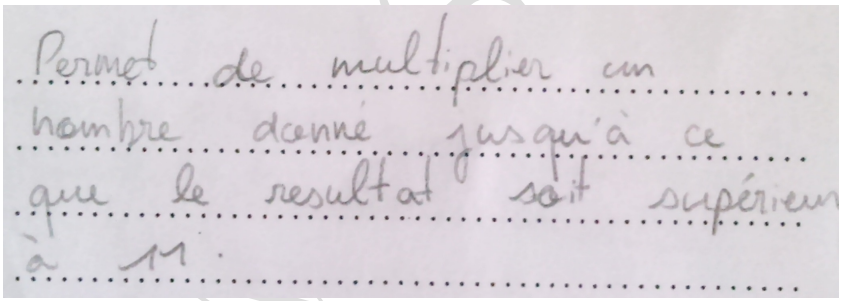


Les neuf derniers ne saisissent pas du tout le but de l'algorithme à la simple lecture. Ils ne le comprennent qu'à l'exécution après l'avoir implémenté.


Cet algorithme nous permet de savoir si i est < à n


Il permet de voir combien de fois on a changer le a avant qu'il soit supérieur ou égal à 11. Ce nombre d'essai est indiqué par i


L'ordinateur ajoute le nombre choisi à i, 11 fois


Permet de multiplier un nombre donné jusqu'à ce que le resultat soit supérieur à 11.

IREM

8. Conclusion

Cette initiation à l'algorithmique et à la programmation en langage *Python* aura nécessité une dizaine d'heures[24]. Elle aura permis de mettre en évidence les difficultés rencontrées par les élèves et ainsi de conforter un certain nombre d'a priori des expérimentateurs :

- La programmation dans un langage textuel n'est pas seulement affaire de codage. Plus que la connaissance de la syntaxe du langage il apparait indispensable de maitriser les différentes structures algorithmiques mises en jeu pour l'obtention du but visé par le programme.
- Un travail de réflexion sur papier s'avère nécessaire avant tout passage sur machine.

L'utilisation du jeu de cartes et la pratique des jeux de rôles ont donné aux élèves la possibilité, partant de leurs représentations et de leurs intuitions concernant les actions à mettre en place pour écrire des programmes, de se construire un savoir algorithmique bien ancré. L'écriture en langage textuel n'est plus alors qu'une simple « traduction ».

Les expérimentateurs ont aussi remarqué que même pour les élèves ayant travaillé la programmation avec Scratch au collège, ce travail de fond n'a pas été inutile. Néanmoins il faudra observer dans les années à venir comment évolueront les connaissances algorithmiques des collégiens pour adapter ce type d'initiation en début de classe de seconde.

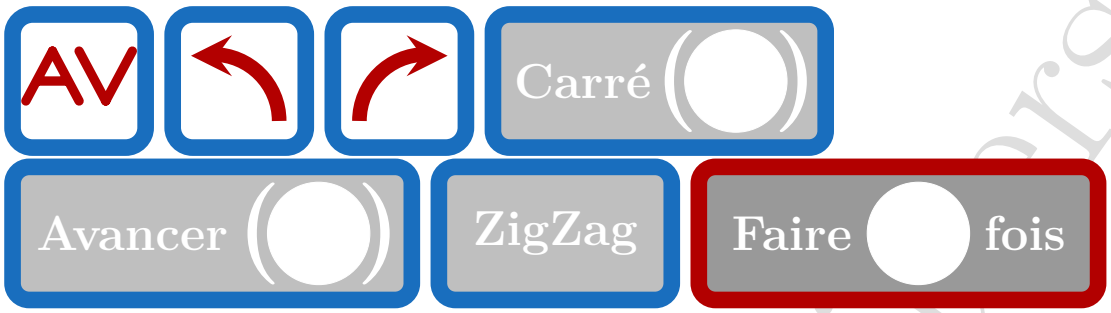
IREM&S Poitiers

24 Quatre ou cinq heures pour le jeu de cartes ; une heure pour le jeu de rôle ; quatre ou cinq heures devant les ordinateurs sur *Python*

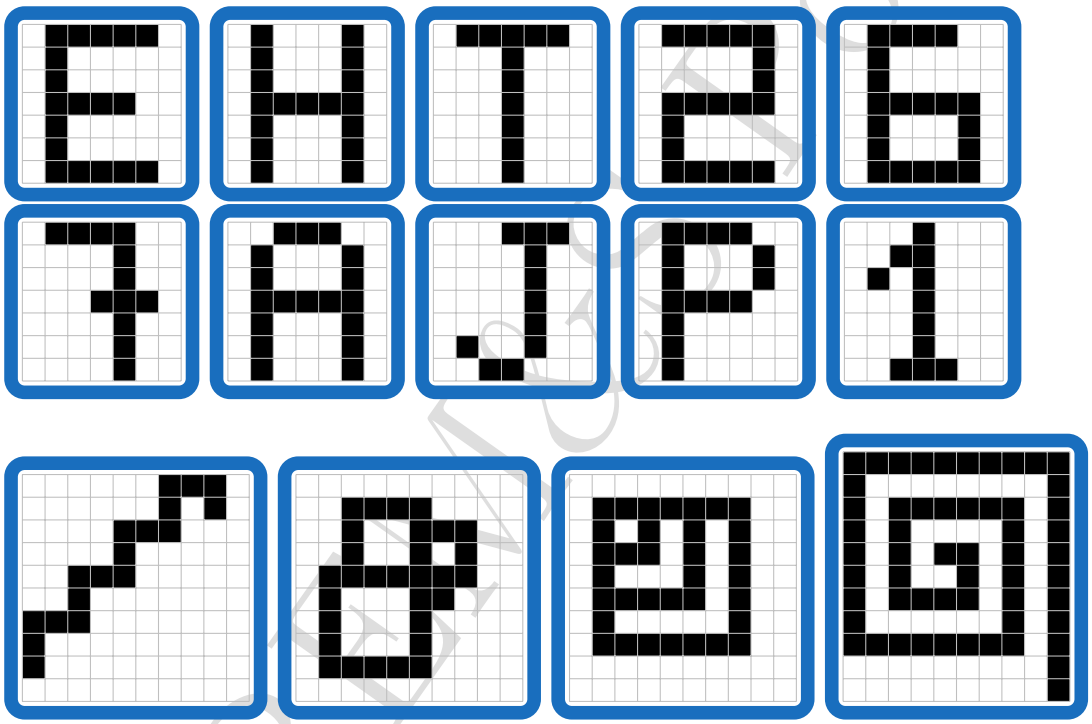
9. Annexes

a. Annexe 1 : exemples de cartes du jeu et de dessins

Exemple de cartes pour jouer [25] (ci-dessous à l'échelle $\frac{1}{2}$)



Exemple de dessins proposés aux élèves [26] (ci-dessous à l'échelle $\frac{2}{5}$)



IREM de Poitiers

25 Voir le fichier *Planches_de_cartes.pdf* qui contient le jeu de cartes imprimable est disponible sur le site de l'IREM de Poitiers http://irem.univ-poitiers.fr/PF_brochures/dokebrochures/.

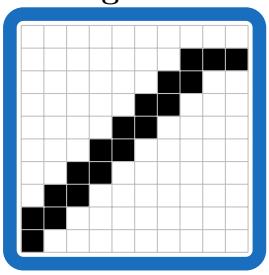
26 Voir le fichier *Dessins_structuration.pdf* qui contient les dessins proposés au élèves est disponible sur le site de l'IREM de Poitiers http://irem.univ-poitiers.fr/PF_brochures/dokebrochures/.

b. Annexe 2 : Répétitions

Dessins proposés aux élèves.

Exemples d'algorithmes avec les cartes et les indentations.

Figure 1



Exemple d'algorithme de construction

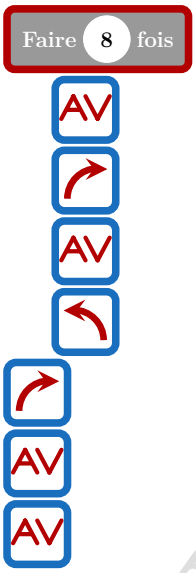
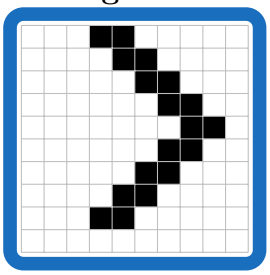


Figure 2



Exemple d'algorithme de construction

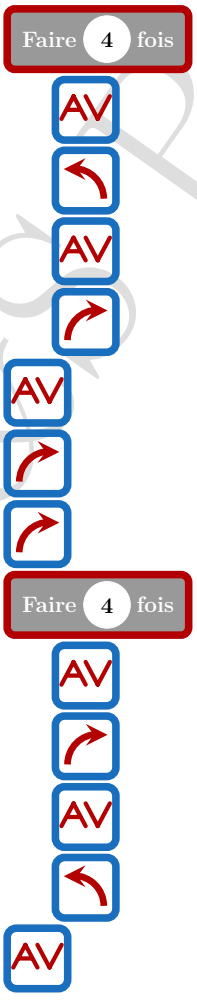
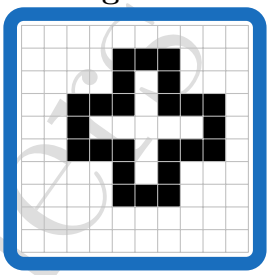
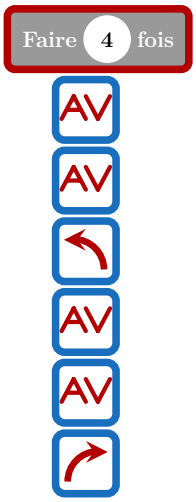


Figure 3



Exemple d'algorithme de construction



IREM&SPOITIERS

Bibliographie de la partie A

[1] Jacques ARSAC. “La didactique de l’informatique : un problème ouvert ?” In : *Colloque francophone sur la didactique de l’informatique*. http ://www.epi.asso.fr/association/dossiers/d07som.htm, sept. 1988.

[2] Nathalie BRIAND. *Etude didactique de la reprise de l’algèbre par l’introduction de l’algorithmique au niveau de la classe de seconde du lycée français*. 2015.

[9] Simon MODESTE. *Enseigner l’algorithme pour quoi ? Quelles nouvelles questions pour les mathématiques ? Quels apports pour l’apprentissage de la preuve ?. Histoire et perspectives sur les mathématiques*. 2012.

[10] Chi Thanh NGUYEN et Annie BESSOT. “La prise en compte des notions de boucle et de variable informatique dans l’enseignement des Mathématiques au lycée”. In : *Petit x* 62 (2003), p. 7-32.

[11] Janine ROGALSKI. “Alphabétisation informatique”. In : *Bulletin vert de l’APMEP* 347 (1985), p. 61-74.

[12] Janine ROGALSKI. “Enseignement des méthodes de programmation dans l’initiation à l’informatique”. In : *Colloque francophone sur la didactique de l’informatique*. http ://www.epi.asso.fr/association/dossiers/d07som.htm, sept. 1988.

[13] Janine Rogalski, Samurçay Renan et Jean-Michel HOC. “L’apprentissage des méthodes de programmation comme méthode de résolution de problème”. In : *Le Travail Humain* 51.4 (1988), p. 309-320.

[14] Laurent SIGNAC. https ://framagit.org/poitiers-infosansordi/fiches-activites.

IREM&S Poitiers

Partie B.

C'est l'heure de programmer : arithmétique et horloges du monde

IREM&S Poitiers

« *Calendriers et instruments de mesure ont été une source importante de problèmes mathématiques à résoudre, d'arithmétique comme de géométrie. Mais, même si l'heure et les unités de durée font partie de notre univers quotidien, savoir d'où viennent ces unités et comment calculer avec elles restent des questions largement sans réponses pour nos élèves. Une des caractéristiques de cet univers des durées est la coexistence de nombreuses unités d'usage courant qui amène à travailler naturellement multiples et diviseurs d'entiers, écritures fractionnaires et décimales, donc dans un contexte numérique.* [1] »

Le partage de la durée du jour a nécessité la construction d'objets permettant une mesure du temps de plus en plus précise [2]. On se propose dans cet article de traiter les contenus d'algorithmique et programmation de Seconde en étudiant divers instruments de mesure du temps telles l'horloge révolutionnaire et la montre binaire.

L'organisation retenue est celle d'un parcours balisé par deux études possédant ces points communs :

- la rencontre avec les contenus de seconde d'algorithmique et de programmation ;
- un sujet d'étude porteur de sens : les horloges, les formats et les unités de durée ;
- un thème mathématique : l'arithmétique conforme aux attendus de cycle 4 ;
- une banque d'exercices pour travailler les notions de programmation abordées dans les études, mais dans des contextes et sur des thèmes autres que les durées.

1. Les programmes d'arithmétique

Extrait [3] des attendus de fin de cycle 4

Comprendre et utiliser les notions de divisibilité et de nombres premiers
Connaissances ➢ Multiples et diviseurs ➢ Critères de divisibilité par 2, 3, 5, 9 ➢ Division euclidienne (quotient, reste) ➢ Définition d'un nombre premier ; liste des nombres premiers inférieurs ou égaux à 30 ➢ Fractions irréductibles Compétences associées ➢ Déterminer si un entier est ou n'est pas multiple ou diviseur d'un autre entier ➢ Déterminer les nombres premiers inférieurs ou égaux à 100 ➢ Utiliser les critères de divisibilité par 2, 3, 5, 9, 10 ➢ Déterminer les diviseurs d'un nombre à la main, à l'aide d'un tableur, d'une calculatrice ➢ Décomposer un nombre entier en produit de facteurs premiers (à la main ou à l'aide d'un logiciel) ➢ Simplifier une fraction pour la rendre irréductible ➢ Modéliser et résoudre des problèmes mettant en jeu la divisibilité (engrenages, conjonction de phénomènes, etc.)

1 IREM de Poitiers *Enseigner les mathématiques en 6ème : Les durées* p.6
2 *ibid.* p.76
3 http://cache.media.education.gouv.fr/file/CSP/00/0/Projet-ajustement-clarification-Programmes_ maths-C_ 2-3-4-31_ mai_ 2018_ 966000.pdf, p.28, p.35

Attendus de fin de cycle
▪ Écrire, mettre au point et exécuter un programme simple.

Écrire, mettre au point, exécuter un programme
Connaissances ➢ Notions d'algorithme et de programme ➢ Notion de variable informatique Compétences associées ➢ Écrire, mettre au point (tester, corriger) et exécuter un programme en réponse à un problème donné

2. Qu'est-ce qu'un enseignement par parcours ?

Un parcours d'étude et de recherche est une organisation particulière pour enseigner des notions mathématiques et dont l'origine théorique est due au didacticien Yves Chevallard [4]. Dans le groupe lycée de l'Irem de Poitiers, nous avons tenté des mises en œuvre pratiques de cette théorie, et considéré qu'il y avait parcours d'étude et de recherche lorsque :

- l'enseignement est motivé par une question, interne ou externe aux mathématiques ;
- une ou plusieurs enquêtes permettent aux élèves de rencontrer la question et de chercher un lien avec un fait de société, plus ou moins scientifique ;
- les séquences d'enseignement suivent un motif répété : études, puis synthèse de cours ou de méthodes, puis mise en pratique lors d'exercices ;
- les études ont l'avantage, par rapport à des activités classiques, d'être toutes en lien avec la question initiale.

Dans la pratique, il n'est pas toujours aisé de formuler une question, d'autant que les programmes scolaires ne sont pas écrits de cette manière ; on s'autorise donc parfois à penser que les notions sont abordées autour d'un même thème ; ainsi on pourra considérer :

- que le présent parcours est motivé par cette question : quels systèmes d'écriture de l'heure les Hommes ont-ils imaginés ?
- ou bien que le présent parcours regroupe des notions d'arithmétique et programmation sur le thème : montres, horloges et formats de durées.

IREM&S Poitiers

4 On trouvera, en annexes pages 57 à 59 de ce fascicule, des détails et des références sur ce qu'est un parcours (d'étude et de recherche).

3. Les grandes lignes du parcours

Ce parcours se décompose en deux études.

La première étude traite de la conversion heure normale - heure révolutionnaire

- Enquête préalable à l'étude 1 : le système métrique, l'heure révolutionnaire.
- Étude 1 : conversion heure normale - heure révolutionnaire
- Contenus travaillés lors de l'étude 1 :
  choisir ou déterminer le type d'une variable (entier, flottant ou chaîne de caractères) ; concevoir et écrire des affectations à des variables ; écrire une formule permettant un calcul combinant des variables.
- Contenus supplémentaires travaillés lors des exercices
  programmer une instruction conditionnelle ; programmer des fonctions simples, ayant un petit nombre d'arguments.

La deuxième étude porte sur le système binaire abordé par les montres binaires.

- Enquête préalable à l'étude 2 : le système binaire.
- Étude 2 : les montres binaires.
- Contenus travaillés lors de l'étude 2 :
  programmer une boucle non bornée.
- Contenus supplémentaires travaillés lors des exercices : *programmer une boucle bornée.*

4. Étude 1

a. Énoncé de l'étude

Étude 1 : l'heure révolutionnaire

Enquête :

- **Le système métrique** : Qu'est-ce que c'est ? Quand a-t-il été imposé ? Par qui ? Pourquoi ?
- **L'heure révolutionnaire** : Qu'est-ce que c'est ? Pourquoi l'heure révolutionnaire n'a-t-elle pas été adoptée ?

Partie 1

On notera 1 SR pour une seconde révolutionnaire et 1 s pour une seconde (« normale »)

1. Expliquer pourquoi 1 SR = 0,864 s.
2. Noter l'heure qu'il est à la seconde près (tout de suite maintenant) et la convertir en Heure Révolutionnaire.
3. Décrire une méthode qui permet de convertir une heure (en h, min, s) en Heure Révolutionnaire (en HR, MR, SR).
4. Automatiser cette méthode à l'aide du logiciel *Python*.

IREM&S Poitiers

Partie 2

1. Ecrire la conversion de 6HR 20 MR 12 SR en heure « normale ».
2. Décrire une méthode qui permet de convertir une Heure Révolutionnaire en heure « normale ».
3. Automatiser cette méthode à l'aide du logiciel *Python*.

b. Éléments de réponse à l'enquête [5]

Avant de commencer l'étude 1, on demande aux élèves une recherche sur l'heure révolutionnaire afin de se placer dans le contexte historique du changement d'unités à la Révolution française.

En classe, le professeur recueille les réponses des élèves sous la forme de son choix (padlet, diaporama, ...) et prend un quart d'heure pour les commenter. Lors du bilan, il faut mettre en valeur la relation : nombre de secondes révolutionnaires en un jour = nombre de secondes normales en un jour qui sera la base des conversions de cette étude 1.
On emploiera le mot *normal* au lieu de *sexagésimal* par commodité de langage auprès des élèves.

L'heure révolutionnaire, un peu d'histoire

Durant la Première République, le temps décimal fut officiellement introduit en France par le décret du 4 Frimaire de l'An II (24 novembre 1793) : le jour, de minuit à minuit, est divisé en dix parties ou heures décimales, chaque partie en dix autres, ainsi de suite jusqu'à la plus petite portion commensurable de la durée. La centième partie de l'heure est appelée minute décimale ; la centième partie de la minute est appelée seconde décimale. La journée commençant à minuit, à midi il était donc 5 heures. À fin de la journée, à minuit, il était 10 heures. De nombreuses montres décimales furent construites à l'époque, devenues aujourd'hui des pièces de musée, car déjà en 1795 le temps décimal fut aboli en France, dix ans avant le calendrier révolutionnaire.

La raison principale pour cet échec semble tenir au fait que le nombre dix n'est pas divisible par quatre sans reste. Ainsi sur un cadran, deux des quatre points cardinaux n'étaient même pas marqués par un grand trait : ils correspondent aux minutes 25 et 75. Ce défaut semble avoir été déterminant dans l'abandon du temps en décimal, malgré le succès du système métrique décimal dans les poids et mesures. Un autre facteur déterminant était la nécessité en effet de modifier non seulement les cadrans de toutes les horloges, pendules et montres existantes mais aussi une partie des rouages internes des mécanismes. Or à l'époque ces rouages étaient fabriqués à la main : la modification était matériellement mais aussi financièrement inenvisageable. De plus, l'heure en soi n'est pas une denrée marchande : on "donne" l'heure, on ne la vend pas, contrairement, par exemple, à un temps de travail. Or, à l'époque, on ne facturait pas un travail au temps passé. Cette habitude viendra par la suite et jusque bien avant dans le XXe siècle, dans certaines entreprises, on a utilisé des chronomètres décimaux pour évaluer le temps passé à une tâche, en heures et dixièmes d'heure.

5 IREM de Poitiers *Enseigner les mathématiques* en 6ème : Les durées (p.58)

IREM de Poitiers

c. Éléments de réponse à la partie 1

1. D'une part 1 jour $= 24 \times 60 \times 60 = 86400$ s et d'autre part 1 jour $= 10 \times 100 \times 100 =$ 100000 SR. Donc 100 000 SR = 86 400 s d'où 1 SR = 0,864 s.
2. 16 h 33 min 14 s = ...s = ...SR = ...HR ...MR ...SR
3. Convertir l'heure en s puis en SR.
   Diviser par 10 000 pour avoir le nombre de HR.
   Diviser le reste de SR par 100 pour avoir le nombre de MR.
   En déduire le nombre de SR restant.
4. Programme en *Python* :

```
# ETAPE 0 : entrer l'heure normale
heures=int(input('heure :'))
minutes=int(input('minutes :'))
secondes=int(input('secondes :'))

# ETAPE 1 : conversion de h min s en s
totalsecondes=3600*heures+60*minutes+secondes

# ETAPE 2 : conversion de s en SR
totalSR=round(totalsecondes*100000/86400)

# ETAPE 3 : conversion de SR vers HR MR SR
HR=floor(totalSR/10000)   # nb de HR

reste=totalSR-10000*HR  # rab de SR après avoir enlevé les HR

MR=floor(reste/100)       # nb de MR

SR=reste-100*MR   # nb de SR

# ETAPE 4 : affichage
print(heures,'h',minutes,'min',secondes,'s','=',totalSR,'SR','=',HR,
'HR',MR,'MR',SR,'SR')
```

Remarques sur le rôle possible du professeur lors de l'écriture du programme.

- Si les élèves rencontrent le logiciel *Python* pour la première fois alors le professeur peut écrire le programme avec eux en le projetant.
- L'instruction « round » arrondit à l'entier le plus proche. Cela permet de travailler uniquement avec des entiers.
- L'instruction « floor » fournit la partie entière du nombre.
- L'instruction « int » signifie que la valeur considérée est convertie en entier.
- Les instructions « reste de la division euclidienne de a par b » (`a%b`) et « quotient de la division euclidienne de a par b » (`a//b`) existent déjà dans *Python*.

5. Étude 2

a. Énoncé de l'étude

Étude 2 Les montres binaires

Enquête : Qu'est-ce que le système binaire ? Où est-il utilisé ?

Questions

1. Quelle heure affichent ces montres ?


2. Colorier la montre de gauche ci-dessous pour qu'elle affiche 17h30.
   Noter l'heure qu'il est à la minute près (tout de suite maintenant) et constater avec émerveillement qu'il est possible de colorier la montre de droite pour qu'elle affiche cette heure.


3. Décrire une méthode pour convertir une heure normale en heure binaire.
4. Automatiser cette méthode sur *Python*.

IREM des Pays de la Loire

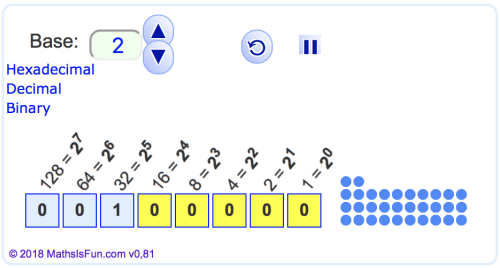
b. Éléments de réponse à l'enquête

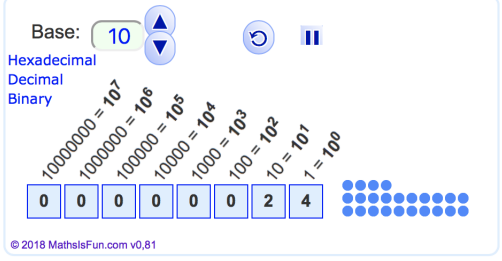
Avant l'étude 2, on propose aux élèves une enquête pour se familiariser avec l'écriture binaire des entiers. L'objectif de cette enquête est que les élèves comprennent où le système binaire intervient et comment lire la valeur d'un entier écrit en base 2.

« *L'arithmétique binaire (plus simplement le calcul binaire) est utilisée par les systèmes électroniques les plus courants (calculatrices, ordinateurs, etc.) car les deux chiffres 0 et 1 s'y traduisent par la tension ou le passage d'un courant. Par exemple, le 0 peut être représenté par l'état bas (tension ou courant nul) et 1 par l'état haut tension qui existe, courant qui passe*[6] ».

Les codes barres et les QR codes font partie des systèmes les plus familiers à utiliser le binaire. Les codes barres servent à référencer des objets et les QR codes permettent d'afficher des textes via un lien optique.

L'écriture d'un entier dépend du choix de la base, les applets ci-dessous peuvent aider à la compréhension de la base 2[7] .





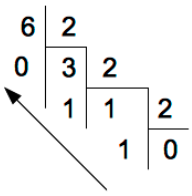
IREM Poitiers

6 https://fr.wikipedia.org/wiki/Système_binaire
7 https://www.mathsisfun.com/binary-number-system.html

c. Éléments de réponse aux questions

3. On effectue la division euclidienne du nombre d'heures par 2. On écrit le reste puis on effectue la division euclidienne du quotient par 2. On écrit le reste à gauche du reste précédent et on continue jusqu'à ce que le quotient soit nul.

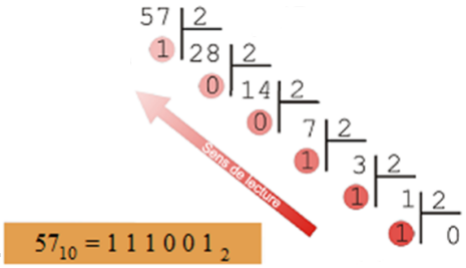
Exemple de conversion pour le nombre d'heures : h = 6



Justification de l'écriture des restes de droite à gauche :

$$\begin{aligned} 6 &= 3\times 2+0 \\ &= (1\times 2+1)\times 2+0 \\ &= ((0\times 2+1)\times 2+1)\times 2+0 \\ &= (0\times 2+1)\times 2\times 2+1\times 2+0 \\ &= 0\times 2\times 2\times 2+1\times 2\times 2+1\times 2+0 \\ &= \mathbf{0}\times 2^3+\mathbf{1}\times 2^2+\mathbf{1}\times 2^1+\mathbf{0} \end{aligned}$$

L'écriture binaire de 6 est donc 110. Les lampes allumées sont donc celles du 4 et du 2. Cet exemple pourra persuader les élèves de la validité de la méthode.
Voici par exemple la conversion en binaire de 57 minutes [8].



4. Voici deux programmes possibles en *Python*. Le premier met en œuvre la méthode de la question 3, le second est une version améliorée utilisant une liste.

```
#Conversion en base 2 avec les restes arrivant dans l'ordre contraire
def Base2(a):
    while a>0 :
        r=reste(a,2)
        a=quotient(a,2)
        print(r)

#Conversion en base 2
def Base2Liste(a):
    L=[]                    # L est la liste vide qui recevra les restes successifs
    while a>0 :
        r=reste(a,2)
        a=quotient(a,2)
        L.append(r)         # Ajouter la valeur de r à la liste L
    L.reverse()             # Inverser l'ordre des restes de la liste L à la fin
    return(L)                                                    de la boucle
```

8 http://pedagogie.ac-limoges.fr/sti_si/accueil/FichesConnaissances/Sequence2SSi/co/ConvDecimalBinaire.html

6. Banque d'exercices

Les programmes en *Python* de certains exercices figurent sur la plate-forme d'accompagnement des brochures en ligne.

a. Banque d'exercices faisant suite à l'étude 1

Exercice 1

Dans l'étude 1, on remarque que les calculs font intervenir à plusieurs reprises le quotient et le reste d'une division euclidienne. On propose d'écrire ces procédures dans le langage *Python* :
Voici la procédure quotient : l'écrire, la tester sur un exemple.

```
from lycee import *

def quotient(a,b):
    q=floor(a/b)
    return(q)
```

Écrire une procédure « reste » puis une procédure « division euclidienne ».

Exercice 2 : test de divisibilité

1. 147 est-il divisible par 7 ?
2. Programmer sur *Python* un test de divisibilité.

Exercice 3 : qui dira 20 ? [9]

Le jeu de « Qui dira 20 ? » se joue à deux de la façon suivante : Chaque joueur choisit un nombre, soit 1 soit 2, et parle à son tour en ajoutant le nombre choisi à la somme précédente. Le premier des 2 joueurs à dire 20 gagne.
Les deux joueurs peuvent choisir le même nombre.
On appelle 2 le pas du jeu. C'est le nombre le plus élevé que le joueur peut choisir.
On appelle 20 la borne du jeu. C'est le nombre à atteindre par le gagnant.

1. Comment peut-on être sûr de gagner ?
2. Écrire un algorithme qui permet de déterminer comment gagner selon le pas et la borne choisis pour jouer.

```
# Qui dira 20 ?
def Nombre_debut(Pas,Borne):
    if reste(Borne,Pas)==0:
        print("Il faut commencer par", Pas)
    else:
        print("Il faut commencer par",reste(Borne,Pas))
```

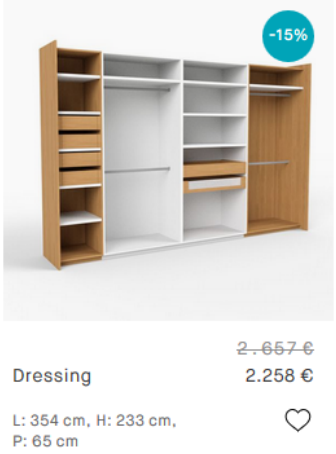
9 http://guy-brousseau.com/587/diaporama-3-lingenierie-des-situations-mathematiques/

IREM&S Poitiers

Exercice 4 : conversions d'unités de longueur américaines

Le système américain de mesure de longueur repose sur le pouce (inch), le pied (foot), le yard et le mile.

Unités	Divisions	Equivalents dans le système international
1 inch		2,54 cm
1 foot	12 inches	0,3048 m
1 yard	3 feet	0,9144 m
1 mile	5280 feet	1609,344 m



1. Convertir les dimensions de l'armoire en feet/inches. Peut-on installer cette armoire dans la chambre ?
2. Automatiser une méthode de conversion du système français vers le système américain en modifiant le programme de l'étude 1 et le tester sur votre taille.

Exercice 5 : monnaie des sorciers

– On parie trente-sept Gallions, quinze Mornilles et trois Noises, dit Fred en rassemblant son argent avec George, que l'Irlande va gagner, mais que ce sera Viktor Krum qui attrapera le Vif d'or. Et on ajoute même une baguette farceuse.
– Vous n'allez pas montrer à Mr Verpey des idioties pareilles, s'indigna Percy.
Mais Verpey ne semblait pas trouver que la fausse baguette magique était une idiotie. Au contraire, son visage juvénile brilla d'excitation lorsque Fred la lui tendit. Quand il la vit se transformer, avec un cri aigu, en un poulet en caoutchouc, Verpey éclata d'un rire tonitruant.
– Excellent ! Ça fait des années que je n'en avais pas vu d'aussi bien imitée. Je vous l'achète cinq Gallions ![10]

Il y a dix-sept Mornilles d'argent dans un Gallions d'or et vingt-neuf Noises de bronze dans une Mornille.[11]

10 *Harry Potter et la Coupe de Feu*, J.K. Rowling
11 *Harry Potter à l'École des Sorciers*, J.K. Rowling

IREM & Pariers

On estime que 1 Noise vaut environ 0,02 €.

1. Quelle somme, en euro, les jumeaux Weasley ont-ils misée ?
2. **(a)** Quelle somme en monnaie de sorcier représentent 200 € ?
   (b) Automatiser une méthode de conversion du système moldu vers le système des sorciers en modifiant le programme de l'étude 1.
3. Harry fait ses courses de rentrée sur le chemin de Traverse. Il achète un sac de piquants de Noueux, une plume, 7 poignées d'yeux de scarabées, 90 g de foie de dragon, une boîte de fioles en cristal. Quel est le nombre minimal de pièces qu'Harry peut utiliser pour payer ses achats ?

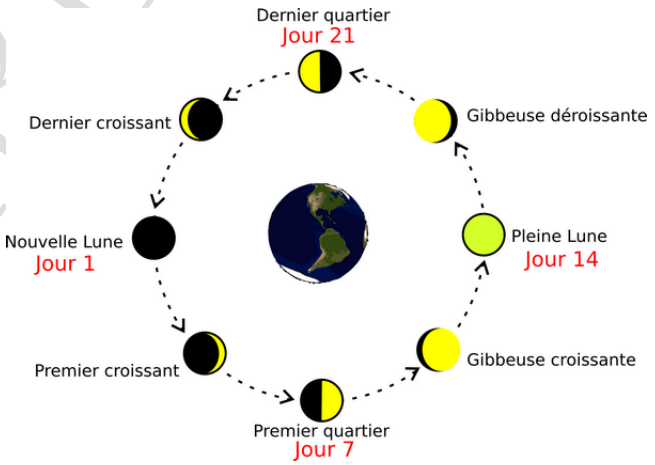
Sac de piquants de Noueux	6 Mornilles
Foie de dragon	17 Mornilles les 30 g
Yeux de scarabée	5 Noises la poignée
Boîte de fioles en cristal	2 Gallions et 11 Mornilles
Plume	15 Mornilles et 2 Noises

Exercice 6 : la date de Pâques

La définition précise du jour de Pâques fut établie en 325 par le concile de Nicée. Les Pères de l'Eglise réunis par l'Empereur Constantin la fixèrent ainsi :
« Pâques est le dimanche qui suit le 14^e jour le la Lune qui atteint cet âge le 21 mars ou immédiatement après ». [12]
Autrement dit, cette fête doit être célébrée le premier dimanche après la première lune suivant l'équinoxe de printemps. [13]
Résumons :

- on se place le 21/03 qui est l'équinoxe (même durée pour le jour et la nuit) de printemps ;
- on considère la première pleine lune PL qui vient le 21/03 ou après ;
- on considère le premier dimanche D qui vient après cette pleine lune ;
- ce dimanche est le dimanche de Pâques !

Rappelons que selon le cycle lunaire, il y a 28 jours entre deux pleines lunes :



12 https://fr.wikipedia.org/wiki/Histoire_du_calcul_de_la_date_de_Paques
13 http://dictionnaire.sensagent.leparisien.fr/Calcul de la date de Paques/fr-fr/

IPEM&S Poitiers

Partie B.

1. Repérer sur cette frise la date « minimale » de Pâques en indiquant la première pleine lune PL et le premier dimanche qui suit D. La première graduation indique le 20 mars.

2. Plus difficile, comment repérer la date « maximale » de Pâques ?
   Si PL tombe le jour de l'équinoxe, quelle sera la date la plus tardive de Pâques ?
   Sinon, de combien de jours au maximum peut-on retarder PL à compter du 21/03 ?
   Enfin, le dimanche de Pâques doit suivre la PL. A quelle date au plus tard peut donc tomber ce dimanche ? La réponse est la date « maximale » de Pâques.

3. Voici l'algorithme du mathématicien Carl Friederich Gauss (1707 – 1783) pour les années postérieures à 1582 (année de réforme du calendrier julien en calendrier grégorien) [14].
   Les Pâques grégoriennes en calendrier grégorien sont le :
   - H = (22 + d + e) mars
     ou le Q = (H – 31) = (d + e - 9) avril ;
   - Si d = 29 et e = 6, remplacer le 26 avril par le 19 avril ;
   - Si d = 28, si e = 6
     et si RESTE (11 M + 11) / 30 < 19,
     remplacer le 25 avril par le 18 avril.

   Vérifier avec cet algorithme que la date de Pâques en 2018 était le 1er avril.
   Écrire un programme qui renvoie le jour et le mois de la date de Pâques à partir de la saisie de l'année.

Dividende	Diviseur	Quotient	Reste
Année	19		*a*
Année	4		*b*
Année	7		*c*
Année	100	*k*	
13 + 8 *k*	25	*p*	
k	4	*q*	
15 - *p* + *k* - *q*	30		*M*
4 + *k* - *q*	7		*N*
19 *a* + *M*	30		*d*
2 *b* + 4 *c* + 6 *d* + *N*	7		*e*

IREM de Poitiers

14 https://fr.wikipedia.org/wiki/Calcul_de_la_date_de_Pâques_selon_la_méthode_de_Gauss

Exercice 7 : le numéro ISSN [15]

- Les publications en série, comme les journaux et les périodiques, sont identifiées par un numéro ISSN.
  Plus de 2 millions d'ISSN sont déjà attribués.

INTERNATIONAL STANDARD SERIAL NUMBER INTERNATIONAL CENTRE

ouest france Rennes

- Le numéro ISSN comporte 8 caractères. Ainsi, ISSN 0999 – 2138 est associé au journal Ouest-France Rennes . Alors que les 7 premiers chiffres caractérisent la publication, le 8ème est calculé : c'est la ***clé de contrôle***.

- ***Comment calculer la clé de contrôle*** du numéro ISSN a b c d – e f g ?
  - on calcule la somme : S = 8a +7b + 6c +5d + 4e + 3f + 2g ;
  - on calcule le reste r de la division euclidienne de S par 11 ;
  - lorsque r $\neq$ 1, la clé de contrôle est égale à 11 – r ;
    lorsque r = 1, la clé de contrôle est notée X (pour dix).
- Pourquoi le nom ***clé de contrôle*** ?
  Parce que la formule de la somme S permet de détecter des erreurs de saisie au moment où on référence les journaux et périodiques dans une base de données ; par exemple, si on fait l'erreur de permuter deux chiffres, alors l'ordinateur enverra un message d'erreur.

1. Vérifier que la clé de contrôle pour « Ouest-France Rennes » est bien le 8.
2. Calculer la clé de contrôle du numéro ISSN commençant par 1032 – 105...
3. Vérifier que le numéro ISSN 0385 – 210 possède la même clé que le précédent.
4. Le troisième chiffre du numéro ISSN d'un journal est illisible. On le note c.
   Le numéro se présente alors sous la forme ISSN 24c6 – 3014.
   Retrouver la valeur de c.
5. Écrire un programme qui renvoie la clé d'un numéro ISSN à partir de la saisie des 7 premiers chiffres.

```
from lycee import *

def quotient(a,b):
    q=floor(a/b)
    return(q)

def reste(a,b):
    r=a-b*floor(a/b)
    return(r)

def ISSN(a,b,c,d,e,f,g):
    S = 8*a +7*b + 6*c + 5*d + 4*e + 3*f + 2*g
    r=reste(S,11)
    if r==1:
        print(X)
    else:
        return(11-r)
```

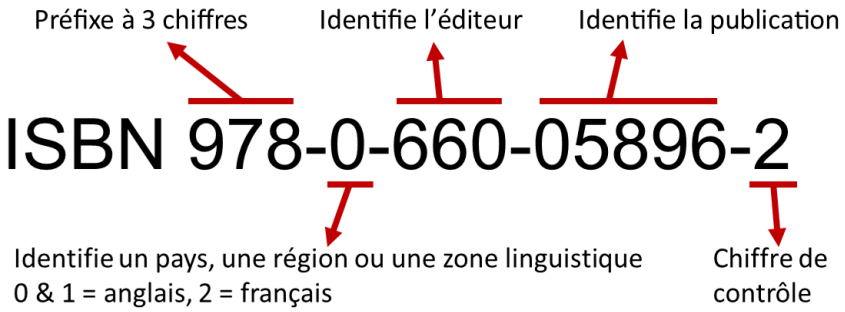
15 http://www.issn.org/

TI Nes Poitiers

Exercice 8 : le numéro ISBN [16]

International ISBN Agency

- Le code ISBN (International Standard Book Number, Numéro International Normalisé du Livre) permet de référencer chaque monographie à support imprimé comme un livre, ou multimédia comme un DVD dans des bases de données informatiques (bibliothèques, maisons d'édition,...)



- Le numéro ISBN a été créé en 1965 par Gordon Foster, professeur de statistiques du Trinity College de Dublin. Ce numéro possède désormais 13 chiffres, répartis en 5 segments. On considèrera ici des numéros ISBN commençant par 978 ou 979, qui sont les deux préfixes attribués à des livres. On s'occupera donc des 10 chiffres de droite, répartis en 4 groupes séparés par des tirets. Le 10ème chiffre est appelé ***clé de contrôle***.
- Comment calculer la clé de contrôle du numéro ISBN a – b c d – e f g h i ?
  - on calcule la somme : S = a +2b + 3c + 4d + 5e + 6f + 7g + 8h + 9i ;
  - on calcule le reste r de la division euclidienne de S par 11 ;
  - si r $\neq$ 10 alors ce reste r est la clé ;
    si r = 10 alors la clé est notée X (pour dix).

1. Compléter les codes suivants par leur clé :
0 – 713 – 66020 – .... 2 – 742 – 70008 – .... 0 – 691 – 05729 – ....

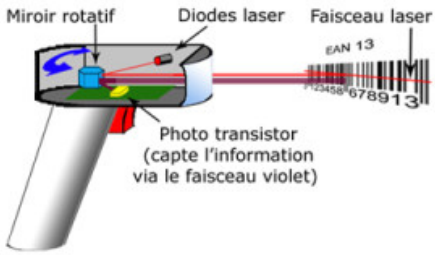
2. Un bibliothécaire saisit le code ISBN 2-700-31999-7.
Le logiciel lui indique alors qu'il a commis une erreur. Comment l'erreur a-t-elle été détectée ? Retrouver la bonne clé.

3. Le bibliothécaire reçoit un autre message d'erreur en saisissant ISBN 2 – 85368 – 313 – 2.
Corriger son erreur, sachant qu'elle porte seulement sur le chiffre de gauche.

4. Écrire un programme qui renvoie la clé d'un code–barres à partir de la saisie des 12 premiers chiffres.

16 https://www.isbn-international.org/

Exercice 9 : le numéro EAN 13 du code barres [17]



- Composé de 13 chiffres, le code EAN 13 est le code de distribution commerciale le plus utilisé au monde. Il figure sur tous les produits commercialisés. Le code barres EAN 13 repose en partie sur le numéro ISSN (pour les publications en séries) ou le numéro ISBN (pour les monographies : livres, DVD...)
- Par exemple, pour les publications en série :
  - les 3 premiers chiffres correspondent à un préfixe (977) ;
  - les 7 chiffres suivants correspondent à l'ISSN sans le tiret et sans la clé de contrôle ;
  - les 11ème et 12ème chiffres correspondent à un code de prix ;
  - le 13ème caractère est une clé de contrôle (CD : control digit)
- Comment calculer la ***clé de contrôle*** du code barres abc–defg-hij–kl ?
  - on fait la somme : S = a+3b+c+3d+e+3f+g+3h+i+3j+k+3l ;
  - on calcule le reste r de la division euclidienne de S par 10 ;
  - lorsque r $\neq$ 0, la clé de contrôle est égale à 10 – r
    lorsque r = 0, la clé de contrôle est 0.

1. Vérifier que le code ci-dessous est un code–barres valide.

2. Déterminer la clef de contrôle (notée ?) de ce code :

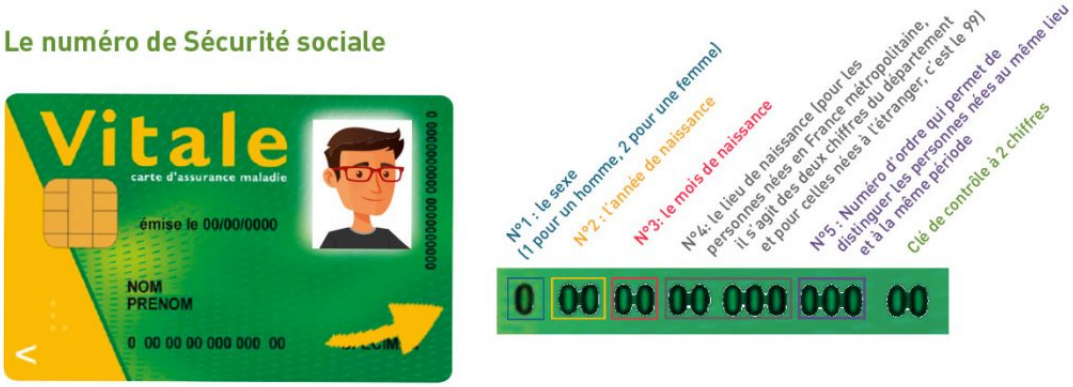
3. Déterminer le chiffre manquant du code 977 – 2 ? 370 – 417 – 67.

4. Ecrire un programme qui renvoie la clé d'un code barres à partir de la saisie des 7 premiers chiffres.

17 http://lecteurcodebarre.com

IREM & Poitiers

Exercice 10 : le numéro INSEE



- La carte Vitale est une carte à puce délivrée à l'âge de 16 ans qui permet de justifier de ses droits médicaux, et simplifie la gestion des remboursements. L'adhésion à la Sécurité Sociale d'un individu né en France commence par son immatriculation par un numéro généré par l'INSEE (Institut national de la statistique et des études économiques).
  Composé de 15 chiffres, le numéro INSEE d'un individu commence par 13 chiffres dont le sens est donné sur l'image ci-dessus. Les deux derniers chiffres sont ceux de la clé de contrôle.
- Comment calculer la ***clé de contrôle*** du numéro INSEE :
  a – b c – d e – f g – h i j – k l m ?
  - on considère le nombre entier à 13 chiffres : a b c d e f g h i j k l m ;
  - on calcule le reste r de la division euclidienne de cet entier par 97 ;
  - la clé de contrôle est égale à 97 − r .

1. Ségolène a pour numéro INSEE, sans la clef, le numéro A = 2 88 07 33 183 048
   (a) A quelle saison est-elle née ? Quelle est la préfecture du département où elle est née ?
   (b) Combien de personnes nées la même année que Ségolène peut-on référencer ?
   (c) Calculer la clé avec votre calculatrice ? Si vous obtenez l'entier 86 alors elle se trompe . . .Quelle pourrait être la cause de cette erreur ?

Pour pallier les risques précédents, on va soit faire quelques calculs préalables pour déterminer la clé de contrôle avec un nombre plus petit que A, soit utiliser un ordinateur et pas une calculatrice.

2. Usage de calculs arithmétiques :
La division euclidienne de A par 10^6 est : $A = 10^6\ q + r$ où q est le quotient et r le reste.
   (a) Ecrire la division euclidienne de 10^6 par 97.
   (b) Démontrer que $A = 97 \times 10\ 309\ q + 378\ 343 + r$.
   (c) Déterminer alors la clef Insee de Ségolène.
   (d) Vérifiez la clé de votre numéro Insee ou de celui d'une personne de votre entourage.

3. Usage d'un programme :
Ecrire un programme qui renvoie la clé d'un numéro INSEE à partir de la saisie des 13 premiers chiffres.
Vérifiez la clé de votre numéro Insee ou de celui d'une personne de votre entourage.

IREM&S Poitiers

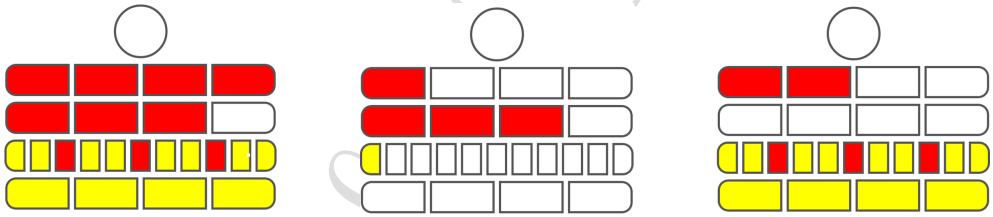
b. Banque d'exercices suite à l'étude 2

Exercice 11 : horloge de Berlin [18]

La Mengenlehreuhr « horloge de la théorie des ensembles » fut la première horloge publique faisant usage de lumières pour afficher l'heure. Elle est citée dans le livre *Guiness des records*. Construite en 1975, elle est installée à Berlin, d'abord sur le Kurfürstendamm (les Champs-Elysées de Berlin) puis dans la Budapesterstrasse. L'heure se lit sur les quatre rangées de lampes. Le disque jaune bat la seconde.

1. Sur la photo de droite, il est 10h31. Expliquer pourquoi.

2. Quelle heure indiquent ces horloges [19] ?



3. Colorier ces horloges pour qu'elles indiquent : 14h42 et 19h37.

4. Modifier le programme de l'étude 2 pour convertir une heure normale en heure berlinoise.

IREM de Poitiers

18 https://www.thevintagenews.com/2017/02/08/mengenlehreuhr-the-berlin-clock-that-shows-time-with-the-help-of-light-signals-and-a-mathematical-theory/

19 https://dodona.ugent.be/en/exercises/527398301/

Exercice 12 : l'origine du code ASCII

Le 10 mars 1876, le Dr Graham Bell met au point le téléphone, une invention révolutionnaire qui permet de faire circuler de l'information vocale dans des lignes métalliques [...]. Cela permit l'essor des téléscripteurs, machines permettant le codage et le décodage des caractères grâce au code Baudot (les caractères étaient alors définis sur 5 bits, il y avait donc 32 caractères uniquement)[20]

Un bit est une unité d'information ; on peut considérer qu'un bit prend l'une des deux valeurs 0 ou 1. D'après le texte ci-dessus, les caractères étaient définis sur 5 bits comme 11010 , ce qui représente l'entier $16 + 8 + 2 = 26$. Ce nombre correspondait à un caractère spécial, donc à une touche du téléscripteur.

1. **(a)** Ecrire les entiers de un à dix avec 5 bits.
   (b) Combien de caractères peut-on coder avec 5 bits ?
2. Le code ASCII (American Standard Code for Informations Interchange) a permis par la suite d'étendre le codage à 7 bits. On voit sur l'image ci-contre la correspondance entre des caractères et leur code ASCII (écrit ici en décimal et hexadécimal)
   (a) Peut-on coder l'entier 200 sur 7 bits ? Pourquoi ?
   (b) Combien de caractères peut-on coder avec 7 bits ?

Decimal	Hex	Char
64	40	@
65	41	A
66	42	B
67	43	C
68	44	D
69	45	E
70	46	F
71	47	G
72	48	H
73	49	I
74	4A	J
75	4B	K
76	4C	L
77	4D	M
78	4E	N
79	4F	O
80	50	P
81	51	Q
82	52	R
83	53	S
84	54	T
85	55	U
86	56	V
87	57	W
88	58	X
89	59	Y
90	5A	Z

3. Évolution dans les années 1960
   Le code ASCII a été mis au point pour la langue anglaise, il ne contient donc pas de caractères accentués, ni de caractères spécifiques à une langue. Pour coder ce type de caractère il faut recourir à un autre code. Le code ASCII a donc été étendu à 8 bits (= 1 byte = un octet) pour pouvoir coder plus de caractères.
   (a) Combien vaut l'entier représenté par l'octet 10100000 ?
   (b) Combien de caractères peut-on coder sur un octet ?
   (c) Quel est le plus grand entier que l'on puisse écrire avec un octet ?
   (d) Combien de bits faut-il pour coder mille caractères ? Deux mille caractères ? Et 16 384 caractères ?

IREM de Poitiers

20 http://www.commentcamarche.net/contents/93-code-ascii
http://infoindustrielle.free.fr/Images/Telescripteur_ edison.jpg

Exercice 13 : usage du code ASCII - Chiffrement de César [21]

Le code César (ou chiffre de César) est un chiffrement par décalage parmi les plus simples et les plus connus, il consiste en un décalage fixe des lettres de l'alphabet. Par exemple, si on choisit de décaler de 3 lettres, la lettre A sera codée par la lettre D.

1. Coder le message suivant à l'aide d'un décalage de 4 lettres :
   HEURE REVOLUTIONNAIRE.
2. Décoder le mot suivant qui a été codé par un décalage de 3 lettres : EDUBFHQWUH
3. Si le message est plus long, coder et décoder à la main prend beaucoup de temps. On va donc utiliser un programme pour faire ce travail.
   Pour coder les lettres sur l'ordinateur, on utilise le code ASCII (voir exercice 12). Les capitales de A à Z sont codées par les entiers de 65 à 90. L'espace est codée par l'entier 32.
   En *Python*, la fonction *chr* renvoie la lettre dont on a entré le code ASCII [22]. Par exemple, chr(65) renvoie A. La fonction *ord* renvoie le code ASCII de la lettre entrée. Par exemple, ord(B) renvoie l'entier 66.
   (a) Écrire sur *Python* une procédure appelée « Décalage » ayant en arguments une liste L de nombres entiers et un entier n correspondant au décalage choisi, et qui ajoute n à chaque élément de la liste.
   (b) Écrire sur *Python* une procédure appelée « Character » ayant en argument une liste L de nombres entiers et qui applique la fonction *chr* à chaque élément de L.
   (c) Utiliser la procédure ci-dessous pour coder le mot de votre choix.

```
#Coder un mot avec un pas de César à choisir
mot=input("mot à coder : ")
n=demande("décalage : ")
L=[]
for i in range(len(mot)):
    L.append(ord(mot[i]))
Y=Character(DécalageCesar(L,n))
X="".join(Y)                #Transforme une liste en chaîne de caractères
print(X)
```

où sont appelées les procédures « DécalageCesar » et « Character »

```
#Décaler des codes ASCII avec
# le pas de César choisi
def DécalageCesar(L,n):
    for i in range(len(L)):
        if L[i]!=32 :
            L[i]=L[i]+n
            if L[i]>90 :
                L[i]=L[i]-26
    return(L)

#Décaler des nombres avec le pas de César choisi
def Décalage(L,n):
    for i in range(len(L)):
            L[i]=L[i]+n
    return(L)
```

```
#Transcrire une liste de codes
# ASCII en liste de lettres
def Character(L):
    L2=[]
    for i in range(len(L)):
        L2.append(chr(L[i]))
    return(L2)
```

21 https://www.dcode.fr/chiffre-cesar

22 En fait Python utilise l'Unicode et non l'ASCII. Les deux tables coïncident sur les 128 premiers caractères, mais l'Unicode en contient plusieurs milliers. *chr()* renvoie le code Unicode.

Exercice 14

1. Écrire un programme pour déterminer la liste des diviseurs d'un entier.
2. Écrire un programme pour tester si un entier est premier ou non.

```
#Liste des diviseurs d'un entier
def Diviseurs(n) :
    L=[]
    for i in range(1,n+1):
        if n%i==0 :
            L.append(i)
        else :
            i=i+1
    return(L)

#Test de primalité d'un entier
def Primalite(n):
    for i in range (2,floor(sqrt(n))+1):
        if n%i==0 :
            return False
        else :
            i=i+1
    return True
```

Exercice 15 : les nombres amiables

Comme on demandait à Pythagore ce qu'est un ami, il aurait répondu : « *Celui qui est l'autre moi-même, comme sont 220 et 284* ».
Voyons pourquoi 220 et 284 sont des amis :

1. Faire la liste de tous les diviseurs de 220 et de tous les diviseurs de 284.
2. Faire la somme des diviseurs de 220, sauf 220.
3. Faire la somme des diviseurs de 284, sauf 284. Conclusion ?

IREM&S Poitiers

Partie C.

Annexes

IREM&S Poitiers

L'enseignement des mathématiques par parcours

De nombreux rapports et études déplorent depuis la fin des années 1990 la perte de sens et le désintérêt au collège et au lycée pour les sciences en général et les mathématiques en particulier.

L'IREM de Poitiers a orienté ses recherches depuis 2005 afin de redonner du sens aux mathématiques enseignées du cycle 3 à la Terminale, pour montrer que cette discipline n'est pas qu'un passage obligé de la réussite scolaire en France mais qu'elle permet, en interaction avec d'autres sciences, de se questionner sur le monde qui nous entoure.

La mise en œuvre des recherches a été collective, en collaboration avec les autres IREM et l'INRP – devenu Ifé. Ces structures ont fourni un cadre institutionnel à nos travaux.

La base théorique est celle des recherches en didactique des mathématiques de Yves Chevallard [1], et notamment la notion de parcours d'étude et de recherche (PER).

Le groupe collège et le groupe lycée de l'IREM de Poitiers ont conçu des parcours qui sont une alternative possible aux traditionnels chapitres des manuels scolaires. Un parcours a pour origine une question, interne ou externe aux mathématiques, et auxquelles les notions enseignées en classe doivent contribuer à répondre.

Cela permet aux élèves, nous l'espérons, de mieux comprendre pourquoi on fait des mathématiques.

Pour bâtir ces parcours, nous avons effectué des recherches :

- *historiques* (d'où viennent les notions ? quel est le contexte de leur développement ?)
- *écologique* (dans quels domaines de la vie courante et des sciences les voit-on apparaître ?)
- *didactique* (comment les transmettre ? quelle organisation en classe de mathématiques ?)

Cette notion de parcours [2] s'est déclinée à l'IREM de Poitiers dans les directions suivantes :

- enseignement par les grandeurs pour le groupe collège : structuration du programme et des notions autour de l'étude des grandeurs usuelles : longueurs, aires, volumes, angles, durées, prix, populations...
- parcours sur la géométrie, l'analyse, les statistiques et probabilités pour le groupe lycée, avec des contextualisations en lien avec diverses sciences telles la démographie, l'histoire, l'astronomie, l'optique, l'acoustique...

IREM&S Poitiers

1 Voir la bibliographie page 58 où sont regroupées quelques références à textes de Yves Chevallard.

2 Voir les bibliographies pages 58 et 59 pour davantage de précisions sur la notion et des exemples de parours

Bibliographie : textes de Yves Chevallard

Textes disponibles au format pdf sur le site http://yves.chevallard.free.fr/. Pour chacun, quelques paragraphes ayant guidé nos réflexions sont indiqués ci-desous.

[15] ***Sur l'attractivité des disciplines enseignées à l'École***.
"Quel avenir pour les mathématiques au collège et au lycée ? Les mathématiques dans la cité" Chap. 1 à 5. In : *Conférences de la Famille mathématique.* IUFM d'Aix-Marseille. Mar. 2009.

[16] ***Sur l'isolement des mathématiques dans l'enseignement secondaire***.
"Quel avenir pour l'enseignement des mathématiques ?" Chap. 1,2,5. In : *Colloque L'enseignement des mathématiques du collège au premier cycle de l'université.* IUFM de Lorraine. Oct. 2003.

[17] ***Sur l'utilité d'un programme scolaire et le rapport des citoyens au savoir***.
"Des programmes, oui. Mais pour quoi faire ? Vers une réforme fondamentale de l'enseignement" Chap. 1,4. In : *Présentation devant le comité scientifique de la Conférence nationale sur l'enseignement des mathématiques.* IFÉ (ENS-Lyon). Jan. 2012.

[18] ***Sur la nécessité pour le professeur de mathématiques de visiter d'autres sciences***.
"La notion de PER : problèmes et avancées" Chap. 6. IUFM de Toulouse. avril 2009.

[19] ***Sur les moments d'études, en classe, d'une question mathématique***.
"Passé et présent de la théorie anthropologique du didactique" Chap. 6. In : *Conférence dans le cadre du premier congrès international sur la théorie anthropologique du didactique.* Baeza (Espagne). Oct. 2005.

[20] ***Sur les notions d'activités et parcours d'étude et de recherche***.
"La place des mathématiques vivantes dans l'éducation secondaire : transposition didactique des mathématiques et nouvelle épistémologie scolaire" Chap. 6.1 à 6.5. In : *Conférence donnée à la 3e Université d'été Animath.* Saint-Flour. août 2004.

IREM de Poitiers

Bibliographie : publications du groupe lycée de l'IREM de Poitiers

[3] Nathalie Chevalarias. "De l'assimilation d'une théorie didactique à sa mise en oeuvre dans les classes : L'exemple des parcours d'Étude et de recherche". In : *Petit x* 94 (2014), p. 5-25.

[4] Dominique Gaud et Nicolas Minet. "Parcours d'étude et de recherche en géométrie pour la classe de seconde". In : *Petit x* 79 (2009), p. 49-71.

[5] Groupe lycée IREM de Poitiers. *Enseigner les mathématiques en première S : trois parcours sur l'analyse et le géométrie analytique*. 2014.

[6] Groupe lycée IREM de Poitiers. *Enseigner les mathématiques en seconde : deux parcours sur la géométrie plane*. 2011.

[7] Groupe lycée IREM de Poitiers. *Enseigner les mathématiques en seconde : quatre parcours sur les statistiques et probabilités*. 2015.

[8] Groupe lycée IREM de Poitiers. *Enseigner les mathématiques en seconde : trois parcours sur les fonctions*. 2011.

IREM&S Poitiers

Partie D.

Pour expérimenter

Si notre démarche vous intéresse et si vous voulez l'expérimenter, nous mettons à disposition des acheteurs de la brochure certains documents sous forme numérique.

A cette fin, nous avons ouvert un espace réservé [1] sur le site de l'IREM de Poitiers.
Vous y trouverez des documents figurant dans les brochures et d'autres que nous utilisons dans nos classes :

- des diaporamas ;
- des ressources TICE ;
- des énoncés des études ;
- des cours ;
- des banques d'exercices ;
- des évaluations ;
- une sitographie avec hyperliens.

Contactez le secrétariat de l'IREM de Poitiers :

Bâtiment de mathématiques H3,
SP2MI Futuroscope
Boulevard Marie et Pierre Curie TSA 61 125
86 073 POITIERS Cedex 9
Tél : (+33) 05 49 45 38 77
Mél : secrirem@math.univ-poitiers.fr
Site : http://irem2.univ-poitiers.fr/portail/

Bonne expérimentation !

1 Appelé *Doke-brochures* et dont l'adresse est http://irem.univ-poitiers.fr/PF_brochures/ dokebrochures/ et dont le code QR est ci-dessus.

IREM&S Poitiers

IREM&S Poitiers

Auteurs : Aubin Joséphine, Chevalarias Nathalie, Chauvin Philippe, De Ligt Frédéric, Dhérissard Sébastien, Gaud Dominique, Grillet Marc, Jussiaume Loïc, Jutand Olivier, Kirch Cyrille, Lebot Bertrand, Mesnier Walter, Michel Julien, Minet Nicolas.

Titre : Mathématiques vivantes au lycée

Fascicule 1 – Algorithmique et programmation en Seconde

Public concerné :

Professeurs de lycée, enseignants en formation initiale, formateurs d'enseignants

Date : Septembre 2018

Mots clés : Algorithmique, arithmétique, carte, Chevallard, débranché, étude, horloge, parcours, programmation, Python, seconde.

Résumé : Les travaux de recherche de l'IREM de Poitiers s'appuient sur la notion de parcours d'étude et de recherche (PER) développée par Yves Chevallard. L'objectif est de donner du sens aux mathématiques enseignées en mettant en prise, autant que faire se peut, les objets d'enseignement du cours de mathématiques avec des faits de société ou des questions scientifiques.
Avec cette collection ***Mathématiques vivantes au lycée***, l'IREM de Poitiers propose des fascicules qui sont chacun constitués d'articles de deux types :

- réflexions didactiques ou pédagogiques sur un sujet mathématique précis ;
- exemples de parcours expérimentés en classe de lycée général ou technologique avec des énoncés pour les élèves (études, banque d'exercices, trame de cours...) et des indications pour la gestion en classe.

Ce premier fascicule a pour objet l'enseignement de l'algorithmique, qui a fait son entrée dans les programmes de lycée depuis plusieurs années. Il propose deux articles au niveau seconde :

- *Algorithmique débranchée : du jeu de cartes vers le logiciel* Python
  On propose ici de travailler la notion d'algorithme par l'intermédiaire d'un jeu de cartes puis d'un jeu de rôle. L'implémentation dans le langage de programmation ne se fera qu'après et ne sera alors qu'une simple traduction des algorithmes travaillés au préalable.
- *C'est l'heure de programmer : arithmétique et horloges du monde*
  On propose ici d'aborder les contenus d'algorithmique et de programmation dans un PER dont le but est d'étudier divers instruments de mesure du temps telles l'horloge révolutionnaire et la montre binaire.

Des ressources en ligne accompagnent chaque fascicule, sur une plate-forme accessible via le site de l'IREM de Poitiers.

IREM de Poitiers

IREM : Bâtiment de mathématiques H3,
SP2MI Futuroscope
Boulevard Marie et Pierre Curie TSA 61 125
86 073 POITIERS Cedex 9
Tél : (+33) 05 49 45 38 77
Mél : secrirem@math.univ-poitiers.fr
Site : http://irem2.univ-poitiers.fr/portail/

ISBN : 978-2-85954-100-2
EAN : 9782859541002