L'ALGORITHMIQUE : UN ATOUT POUR L'ENSEIGNEMENT DES MATHÉMATIQUES AU LYCÉE ?

Conférence IREM Strasbourg, le 27 janvier 2016

Nathalie BRIANT ,PRAG-Docteur

ESPE de Montpellier

Laboratoire LIRDEF, Université de Montpellier

Plan de l'exposé

Deux parties :

- Partie 1 : quelques bases
  - Définitions d'un algorithme, de l'algorithmique et de la programmation
  - Émergence d'une pensée algorithmique dans la résolution de certains types de problèmes mathématiques utilisant les TICE
- Partie 2 : deux exemples de progression en 2nde
  - Utilisation de l'algorithmique pour la compréhension de concepts algébriques
  - Utilisation de l'algorithmique pour la compréhension du concept de dichotomie

Introduction

- Entretien avec trois professeurs expérimentés (plus de 15 ans en lycée)

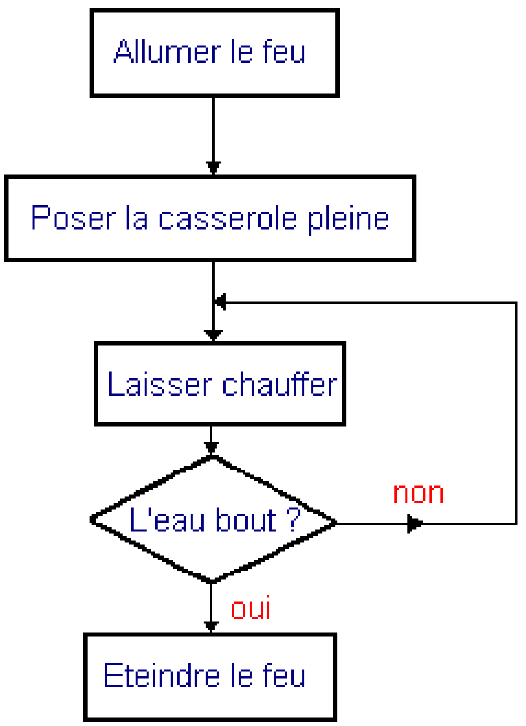
- *« Le problème, c'est que je me pose la question de l'utilité pour les gens qui ne vont pas être des scientifiques, d'autant plus que l'outil informatique, plus on l'utilise, moins on se sert de programmation, puisqu'on utilise déjà des choses qui sont toutes programmées. Et je ne vois pas l'intérêt de programmer, mis à part pour les scientifiques. Le commun des mortels, le technicien va utiliser des logiciels qui sont déjà construits. »*

- *« Quand je pense au temps que ça nécessite… je dois dire que je fais des séances d'algorithmique dans un but purement… je me soumets au programme, point final. Franchement, si je savais qu'on ne me reprocherait pas de ne pas le faire, je ne le ferais pas… Je pense en maths qu'il y a d'autres parties du programme qui me paraissent intéressantes, il y a d'autres démarches plus fructueuses que ça… la réflexion, je préfèrerais l'avoir sur autre chose… »*

- *« C'est un problème de société, c'est lié à la sur-dimension de la machine dans leur vie, le portable, les mp3, tout ça… je dois dire que très peu d'élèves réfléchissent d'abord sur les maths avant d'aller sur la machine …»*

Partie 2
Dichotomie
Algèbre
Partie 1
Pensée algo
Définitions

Dans le monde, il y a 10 catégories de personnes : celles qui savent compter en binaire et celles qui ne savent pas.



Organigramme pour faire bouillir de l'eau.

ALGORITHME, ALGORITHMIQUE ET PROGRAMMATION

La recette des crêpes

Le site le plus simple pour faire la pâte à crêpes !

Bienvenue sur le site le plus simple pour faire la pâte à crêpe! Cette recette facile de pâte à crêpe se transmet de bouches à oreilles pour votre plus grand plaisir! Temps de préparation : 10mn.

Pâte à crêpe pour 15 crêpes: 50g de beurre, 4 oeufs, 2 cuillères à café de sucre, 1 pincée de sel, 250g de farine, 1/2 litre de lait.

Mettre l'ensemble des ingrédients dans un récipient sauf le beurre.

Mélanger avec un fouet jusqu'à obtenir de la pâte liquide et sans grumeaux.

Ajouter les 50g de beurre fondu: fondre au micro-onde, ça va plus vite! Pour plus de goût, ajouter de la fleur d'oranger ou du rhum...

Si possible, laisser reposer, puis étaler une dose de pâte dans une poêle chaude préalablement graissée.

Laisser cuire à feu doux...

...puis retourner pour laisser cuire l'autre côté.

Vous pouvez déguster vos crêpes! Bon appétit !

L'« algorithme » des crêpes

Ingrédients : beurre, œufs, sachet de sucre vanillé, farine, lait, sel

Récipients : saladier, verre mesureur, poêle, assiette

***Opérations de base : mettre dans un récipient**, mélanger, attendre pendant … minutes, retourner, laisser cuire pendant … minutes*

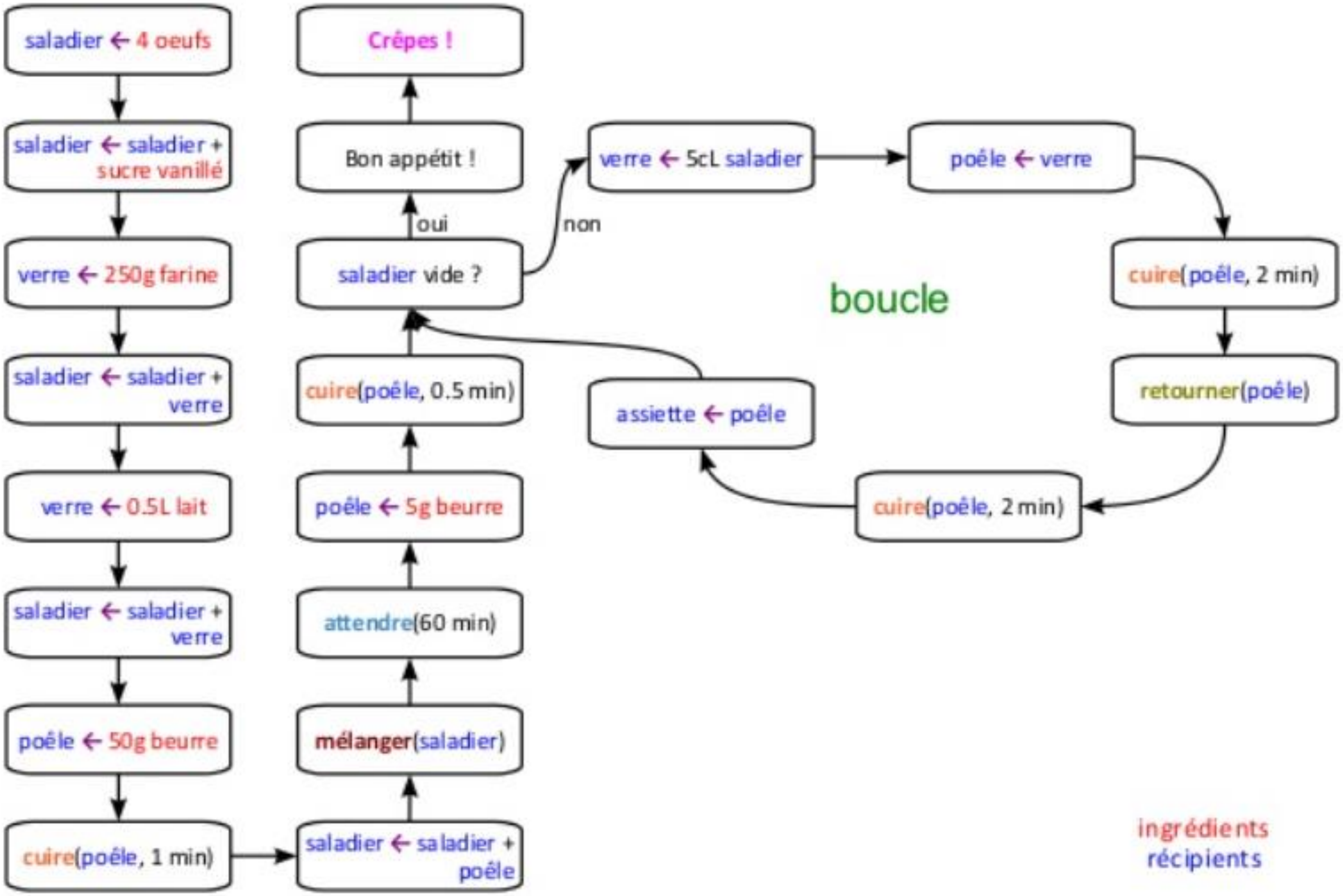
Algorithme des crêpes :

Mettre **4 œufs** dans le saladier
Mettre **1 sachet de sucre vanillé** dans le saladier
Mettre **250 g de farine** dans le verre mesureur
Mettre le contenu du verre mesureur dans le saladier
Mettre **0,5 L de lait** dans le verre mesureur
Mettre le contenu du verre mesureur dans le saladier
Mettre **50 g de beurre** dans la poêle
Laisser cuire la poêle pendant **1** minute
Mettre le contenu de la poêle dans le saladier
Mélanger le contenu du saladier
Attendre pendant **60** minutes

Mettre **5 g de beurre** dans la poêle
Laisser cuire la poêle pendant **0,5** minute
Tant que le saladier n'est pas vide :

- Mettre **5 cL** du saladier dans le verre mesureur
- Mettre le contenu du verre mesureur dans la poêle
- Laisser cuire la poêle pendant **2** minutes
- Retourner le contenu de la poêle
- Laisser cuire la poêle pendant **2** minutes
- Mettre le contenu de la poêle dans l'assiette

Organigramme de la pâte à crêpes



A quoi sert un algorithme ?

- À décrire les **étapes** de résolution d'un problème :

→ de façon structurée , ordonnée et non ambiguë

→ à partir d'opérations de base

→ indépendamment d'un langage de programmation

"étapes" aussi appelées "pas de l'algorithme"

A quoi sert un algorithme ?

- À décrire les étapes de **résolution d'un problème** :

→ de façon structurée, ordonnée et non ambiguë

→ à partir d'opérations de base

→ indépendamment d'un langage de programmation

Un algorithme est une procédure de résolution de problème.
Les données du problème en entrée
Le résultat de sa résolution en sortie

A quoi sert un algorithme ?

- décrire les étapes de résolution d'un problème :

→ **de façon structurée , ordonnée et non ambiguë**

→ à partir d'opérations de base

→ indépendamment d'un langage de programmation

Processus qui s'exécute pas à pas

L'action à chaque pas est strictement déterminée par l'algorithme,
les données d'entrée et les résultats obtenus dans les étapes précédentes.

A quoi sert un algorithme ?

- À décrire les étapes de résolution d'un problème :

→ de façon structurée, ordonnée et non ambiguë

→ **à partir d'opérations de base**

→ indépendamment d'un langage de programmation

Opérations de base = « Actions élémentaires » compréhensibles et maîtrisées par l'exécutant (homme ou machine)

A quoi sert un algorithme ?

- À décrire les étapes de résolution d'un problème :

→ de façon structurée, ordonnée et non ambiguë

→ à partir d'opérations de base

→ **indépendamment d'un langage de programmation**

Compréhensible sans apprendre un langage de programmation

Adapté pour des problèmes qui se traitent sans ordinateur

Différence entre algorithme et programme

« Un algorithme est une procédure de résolution de problème, s'appliquant à une famille d'instances du problème et produisant, en un nombre fini d'étapes constructives, effectives, non-ambigües et organisées, la réponse au problème pour toute instance de cette famille. » (Modeste, 2012)

- Un algorithme n'est pas un programme.
- Un algorithme est une description abstraite des étapes conduisant à la solution d'un problème.

Algorithme = partie conceptuelle d'un programme
Programme = implémentation (i.e. réalisation) de l'algorithme, dans un langage de programmation et sur un système particuliers.

Exprimer un algorithme dans un langage de programmation a pour but de le rendre exécutable par une machine numérique (prog 2012, spé info en TS)

Différence entre algorithmique et programmation

Introduction au cours d'algorithmique-programmation de l'Ecole Polytechnique (Potier et Werner, 2013) :

- « L'objet de l'**algorithmique** est de comprendre si l'on peut résoudre tel ou tel problème par le calcul, et si oui, de quelle manière, et à quel prix en termes de temps et de mémoire. Cette discipline est essentiellement indépendante du choix d'une machine ou d'un langage de programmation particuliers.

- La **programmation** consiste plus spécifiquement à exploiter un langage de programmation particulier pour organiser les programmes de façon simple, élégante et robuste. »

Algorithmes : preuves de correction, de terminaison - Complexité

- **Correction d'un algorithme**

→ donne-t-il le résultat attendu ?

→ quelles que soient les données d'entrée ?

En pratique :
- vérifier sur les "cas de base"
- vérifier sur des exemples aléatoires

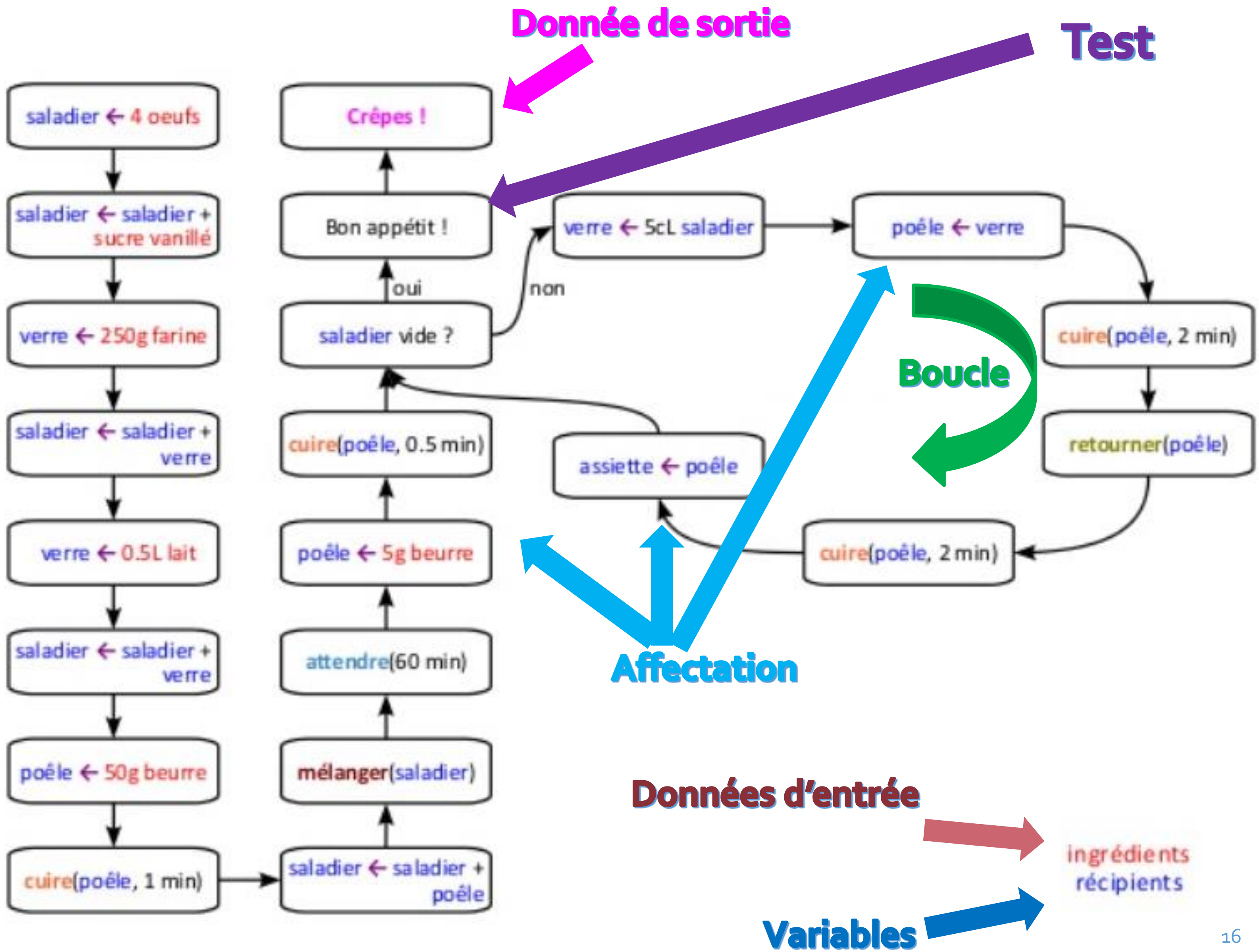
- **Preuve de terminaison et complexité d'un algorithme**

→ l'algorithme se termine-t-il ?

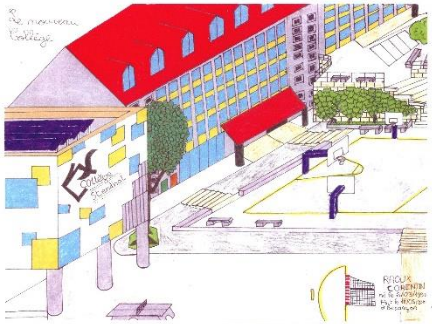
→ en combien de temps ? → Complexité

Théorie de la complexité :
→ nombre d'opérations en fonction de la taille du problème, dans le pire cas
→ prouver qu'on ne peut pas utiliser moins d'opérations pour résoudre le problème, dans le pire cas

Les divers constituants d'un algorithme

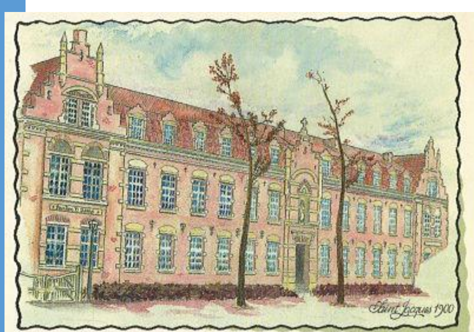


Ce que l'on trouve dans les programmes institutionnels à propos de l'algorithmique



Collège 2016

Classes de 5^{e} à 3^{e} *En 5ème, 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 événements. En 3ème, ils abordent la gestion des objets, en leur faisant échanger des messages.*



Lycée 2009

- **Classe de 2^{nde}** *Familiariser les élèves avec les grands principes d'organisation d'un algorithme : gestion des entrées-sorties, affectation d'une valeur et mise en forme d'un calcul.*

- **Classe de S, ES, L, STL, STI2D ($1^{ère}$ et T)**

-Instructions élémentaires (affectation, calcul, entrée, sortie) : *écrire une formule permettant un calcul ; un programme calculant et donnant la valeur d'une fonction ; les instructions d'entrées et sorties nécessaires au traitement.*

-Boucle et itérateur, instruction conditionnelle : *programmer un calcul itératif, une instruction conditionnelle, une fin de boucle conditionnelle.*

ÉMERGENCE D'UNE PENSÉE ALGORITHMIQUE DANS LA RÉSOLUTION DE CERTAINS TYPES DE PROBLÈMES MATHÉMATIQUES UTILISANT LES TICE

COMPRENDRE CETTE ÉMERGENCE

Comprendre ce que nous désignons par « **pensée algorithmique** » au travers de l'exemple suivant :

Créer un algorithme portant sur la simplification de la racine carrée d'un entier naturel noté N, sous la forme $a\sqrt{b}$ *avec a et b entiers et b le plus petit possible.*

Comparaison d'algorithmes : écrire $\sqrt{N}$ sous la forme $a\sqrt{b}$

Exemple de N = 72

Algorithme 1

- Effectuer la décomposition de *N* en facteurs premiers :
  $N = p_1^{\alpha_1} p_2^{\alpha_2} \dots p_k^{\propto_k}$
- *a* et *b* valent 1
- Pour tout entier i compris entre 1 et *k* :
  - Si $\propto_i$ est pair, *a* contient le facteur $p_i^{\frac{\propto_i}{2}}$
  - Si $\propto_i$ est impair, *a* contient le facteur $p_i^{\frac{\propto_i - 1}{2}}$ et *b* contient le facteur p_i
- En sortie, *a* et b sont égaux au produit des facteurs définis ci-dessus

N = $3^2 \times 2^3$

a= 3

a=3×2
b =2

$\boldsymbol{\sqrt{N} = 6\sqrt{2}}$

Algorithme 2

- Pour chaque entier *i* compris entre 1 et Ent($\sqrt{N}$) :
  - Tester si la division euclidienne de *N* par *i*² donne un reste nul
  - Si c'est le cas, affecter à *a* la valeur de *i*
  - Si ce n'est pas le cas, *a* garde sa valeur
  - Passer à la valeur suivante de *i*
- Calculer la valeur de *b* = *N*/*a*²

$\boldsymbol{\sqrt{N} \approx 8}$

***i* = 8**

***a* = 6**

***b* = 72/36 = 2**

$\boldsymbol{\sqrt{N} = 6\sqrt{2}}$

Exemples de simplification de racines carrées sous Algobox

Soit N un entier naturel.

Pour chaque entier I compris entre 1 et Ent($\sqrt{N}$) :

- Tester si la division de N par I^2 donne un reste nul ;
- Si c'est le cas, affecter à a la valeur de I ;
- Si ce n'est pas le cas, passer à la valeur suivante de I.

Calculer la valeur de b : N/a^2

Afficher

racine(N) = a*racine(b)

Algorithme de simplification de $\sqrt{N}$ sous la forme $a\sqrt{b}$ où $(a, b) \in N^2$ et où b est le plus petit possible.

```
VARIABLES
  N EST_DU_TYPE NOMBRE
  I EST_DU_TYPE NOMBRE
  a EST_DU_TYPE NOMBRE
  b EST_DU_TYPE NOMBRE
DEBUT_ALGORITHME
  LIRE N
  POUR I ALLANT_DE 1 A floor(sqrt(N))
    DEBUT_POUR
    SI (N%(I*I)==0) ALORS
      DEBUT_SI
      a PREND_LA_VALEUR I
      FIN_SI
    FIN_POUR
  b PREND_LA_VALEUR N/(a*a)
  AFFICHER a
  AFFICHER "*racine("
  AFFICHER b
  AFFICHER ")"
FIN_ALGORITHME
```

Programme correspondant à l'algorithme ci-contre sous Algobox

N = 120

```
***Algorithme lancé***
2*racine(30)
***Algorithme terminé***
```

N = 256

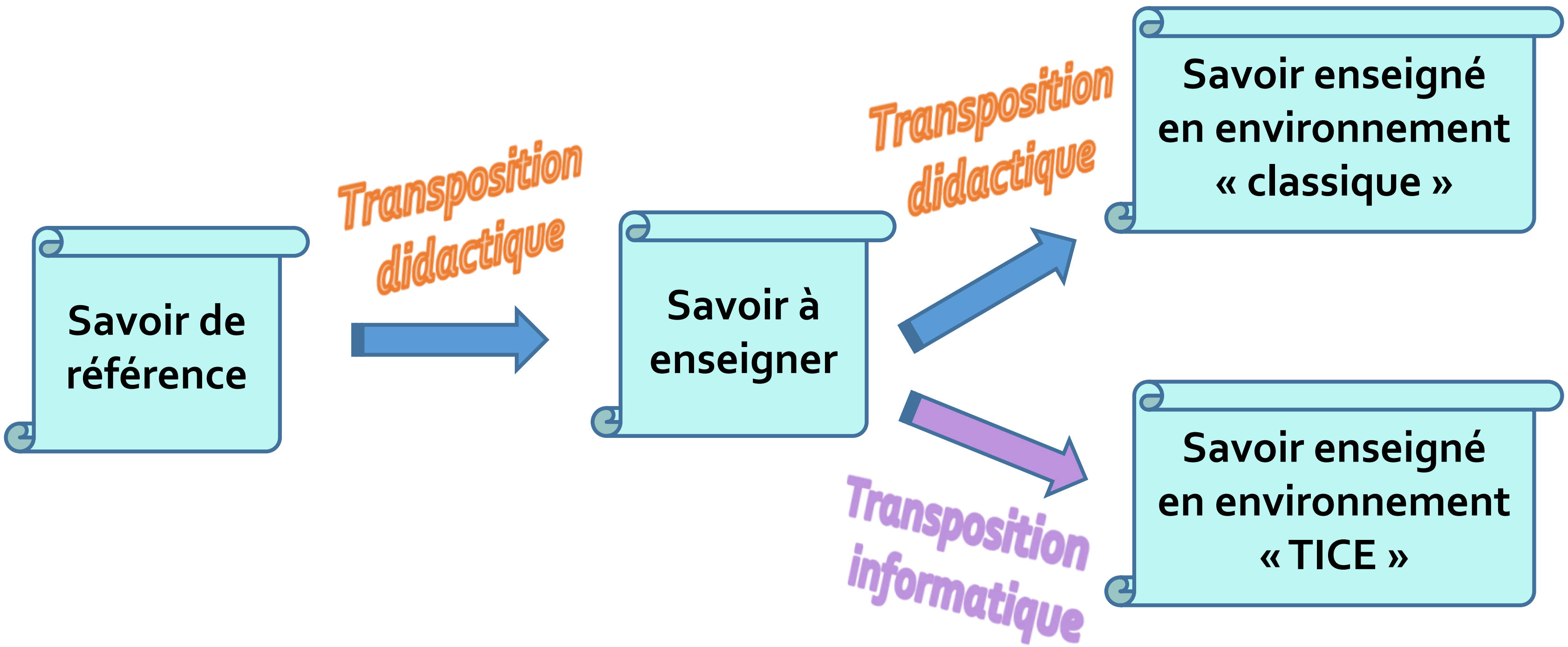
```
***Algorithme lancé***
16*racine(1)
***Algorithme terminé***
```

N = 1789

```
***Algorithme lancé***
1*racine(1789)
***Algorithme terminé***
```

Résultats obtenus par le programme pour trois valeurs particulières de N

Une adaptation du concept de transposition didactique aux TICE

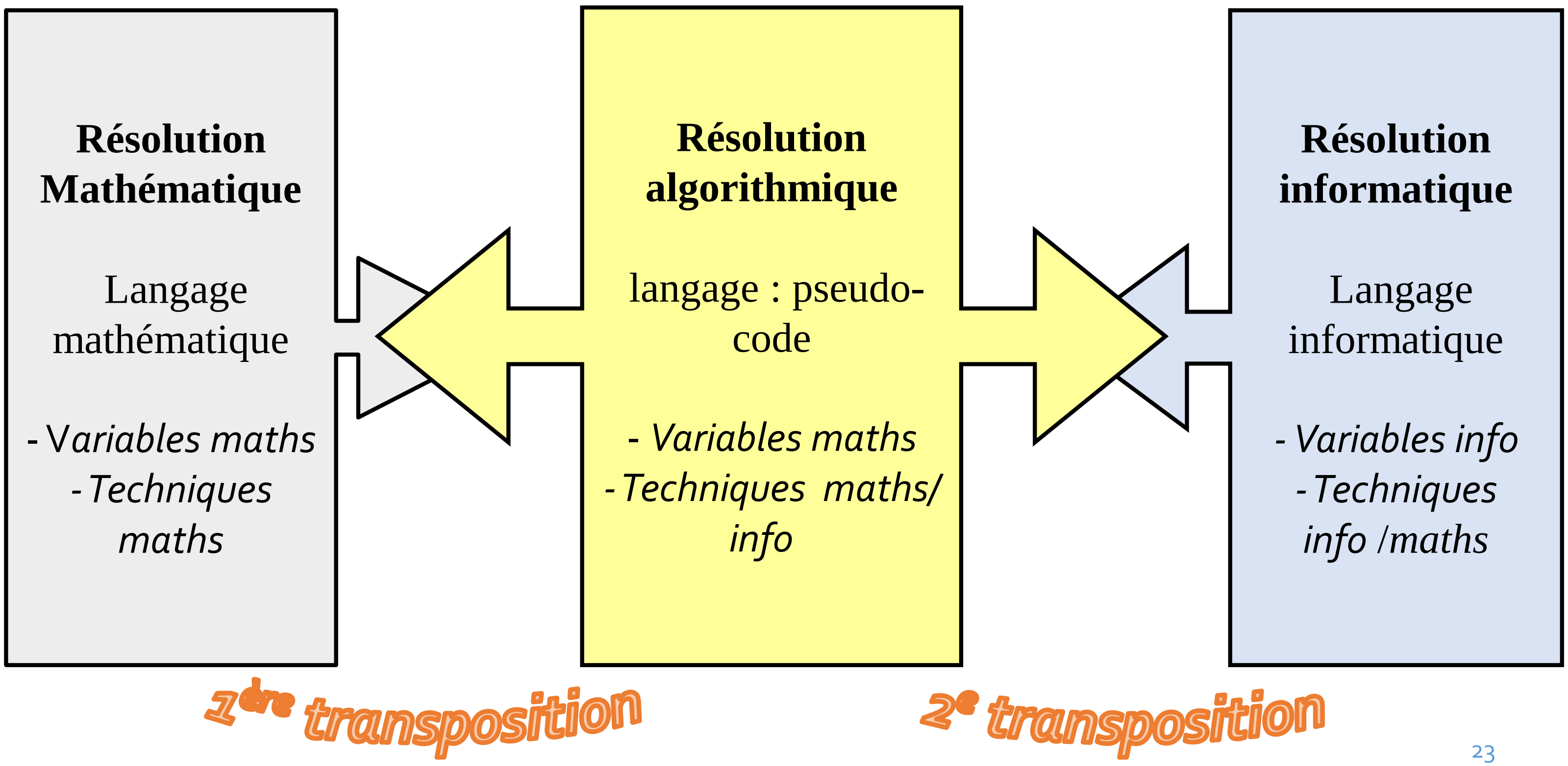


Transposition didactique : Chevallard (1982)

Transposition informatique : Balacheff (1994)

Aux contraintes de la transposition didactique s'ajoutent, ou plutôt se combinent, celles de modélisation et d'implémentation informatiques. (Balacheff, 1994)

Modèle de double transposition pour effectuer la tâche : « concevoir un programme pour résoudre un problème mathématique »



Émergence d’une pensée algorithmique

- Définition de la pensée algorithmique (Modeste, 2012 ; Hart, 1998) :

La pensée algorithmique serait alors une façon d’aborder un problème en essayant de systématiser sa résolution, de se questionner sur la façon dont des algorithmes pourraient ou non le résoudre. (p.47)

- Une pensée algorithmique amène à une *généralisation* d’une tâche donnée, en s’intéressant au type de tâches sous-jacent, c’est-à-dire en ne considérant pas le problème pour des valeurs numériques données, mais en envisagent sa résolution de manière plus générale.

- Pour effectuer le type de tâches « concevoir un programme pour résoudre un type de problèmes mathématiques », le sujet va devoir apprendre à se décentrer de sa posture d’individu pour se placer dans la position de tenir compte de ce que sait faire la machine.

Partie 2

Dichotomie

Algèbre

Partie 1

Pensée algo

Définitions

EXEMPLE D'UNE PROGRESSION EN 2^{NDE} : UTILISATION DE L'ALGORITHMIQUE POUR LA COMPRÉHENSION DE CONCEPTS ALGÉBRIQUES

Utilisation de l'algorithmique pour la compréhension de concepts algébriques

Objectifs :

- Illustrer le modèle de la double transposition développé dans la première partie
- Montrer comment le détour par une pensée algorithmique a permis de développer et d'asseoir des concepts algébriques gravitant autour de la notion d'équation.

→ L'expérimentation menée porte sur les équations polynomiales de degré 1 et 2 et leur résolution algébrique. Le public est constitué d'élèves de différentes classes de seconde du lycée.

Résoudre dans **R** : $2(x-1) + 5x = 3x + 4 - 2(x+1)$

$$2x - 2 + 5x = 3x + 4 - 2(x+1)$$

$7x - 2$	$x + 2$
$7x - 2 = 0$	$x + 2 = 0$
$7x = 2$	$x = -2$
$x = \frac{2}{7}$	

Confusion
ax + b = cx + d
et
(ax+b)(cx+d) = 0

Résoudre dans **R** : $x^2 + 6x + 9 = 0$

$$x^2 + 6x = -9$$

$$x^2 + x = -\frac{9}{6}$$

$$x^3 = -\frac{9}{6}$$

Confusion techniques résolutions équations premier et second : « isoler » x dans un membre

Résoudre dans **R** : $(3x + 1)^2 - 4 = 0$

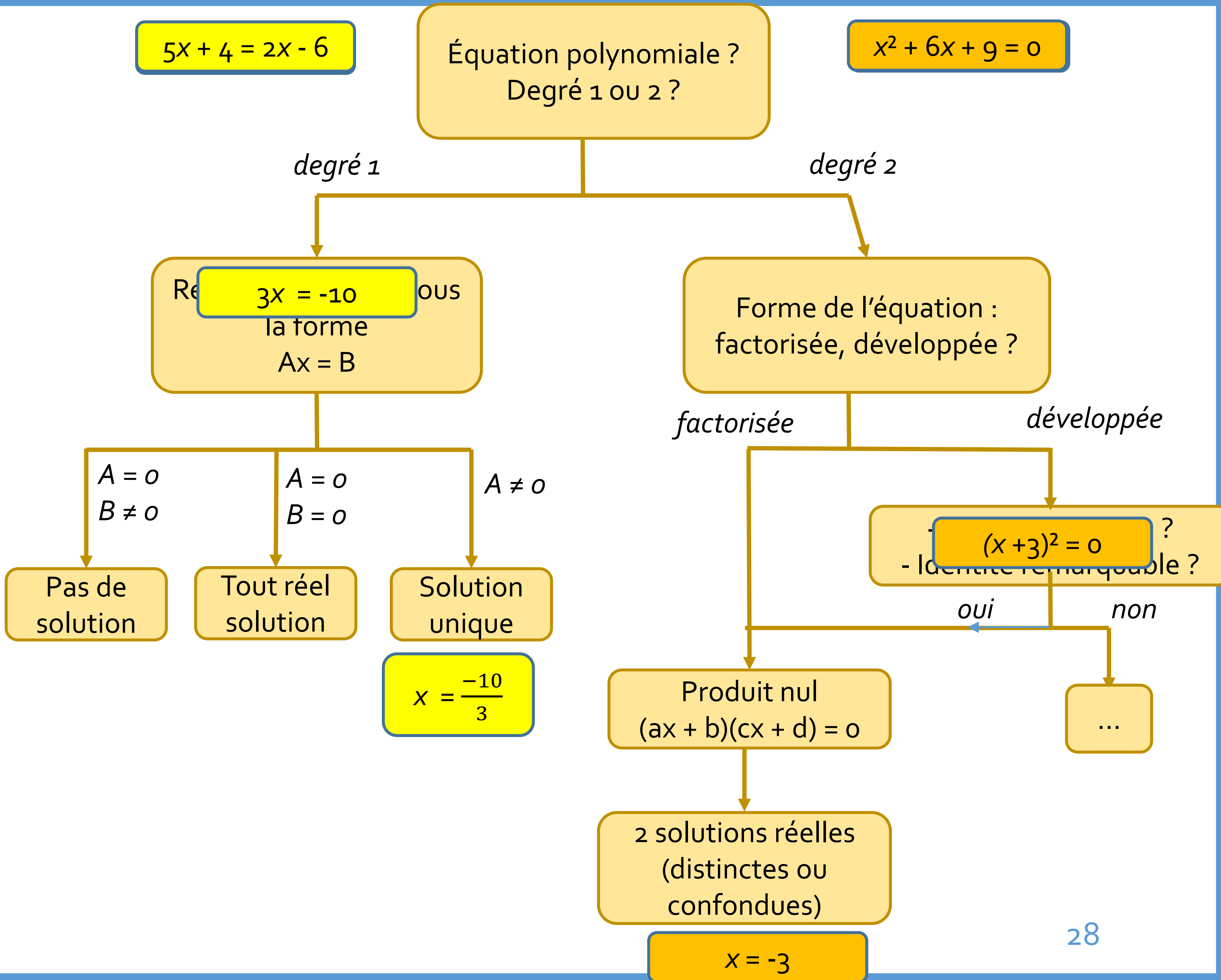
$$9x^2 + 6x + 3 - 4 = 0$$

$$9x^2 + 6x = -1$$

$$3x + \sqrt{6}\,x = -1$$

$$3\sqrt{6}\,x = -1$$

$$x = -\frac{1}{3\sqrt{6}}$$



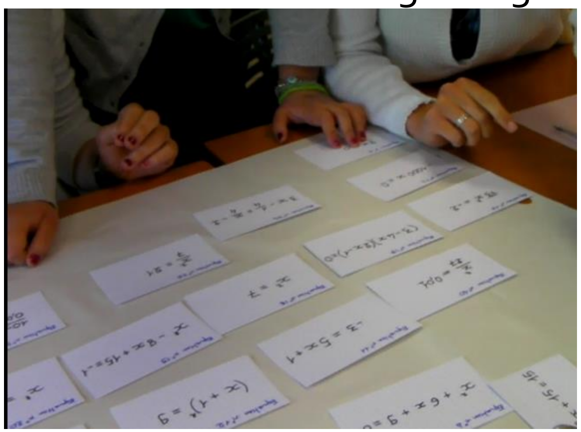
5x + 4 = 2x - 6
Équation polynomiale ?
Degré 1 ou 2 ?
x² + 6x + 9 = 0
degré 1
degré 2
3x = -10
la forme
Ax = B
Forme de l'équation :
factorisée, développée ?
A = 0
B ≠ 0
A = 0
B = 0
A ≠ 0
factorisée
développée
(x +3)² = 0
Pas de solution
Tout réel solution
Solution unique
x = −10/3
oui
non
Produit nul
(ax + b)(cx + d) = 0
…
2 solutions réelles (distinctes ou confondues)
x = -3

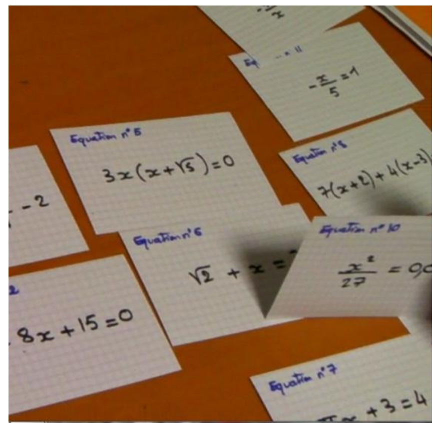
Une ingénierie composée de trois situations non indépendantes

<table>
<tr><th></th><th><u>Situation 1</u></th><th>Situation 2</th><th>Situation 3</th></tr>
<tr><td>Contenu</td><td>Déterminer une classification d'équations polynomiales de degré 1 ou 2 (se ramenant à $x^2 = a$ ou se ramenant au 1^{er} degré)</td><td>Déterminer un/des algorithmes/ programmes permettant d'automatiser la résolution d'équations de degré 1</td><td>Déterminer un/des algorithmes/ programmes permettant d'automatiser la résolution de certaines équations de degré 2</td></tr>
<tr><td>Fonction/ Objectif</td><td>Catégorisation d'équations polynomiales de degré 1 ou 2.
Émergence que le degré de l'équation influe sur la technique de résolution</td><td colspan="2">- Modélisation des équations
- Détermination de techniques de résolution pour les types d'équations reconnues
→ Algorithmiser les techniques de résolution des équations de degré 1 et 2</td></tr>
<tr><td>Objets algo /prog</td><td>Néant</td><td colspan="2">Variable informatique ; affectation de variable ; étapes d'un programme de calcul ; instructions d'entrée/sortie
Test conditionnel « si … alors… sinon »
Mise en œuvre d'un algorithme complexe, modification d'un algorithme existant</td></tr>
</table>

Situation n°1

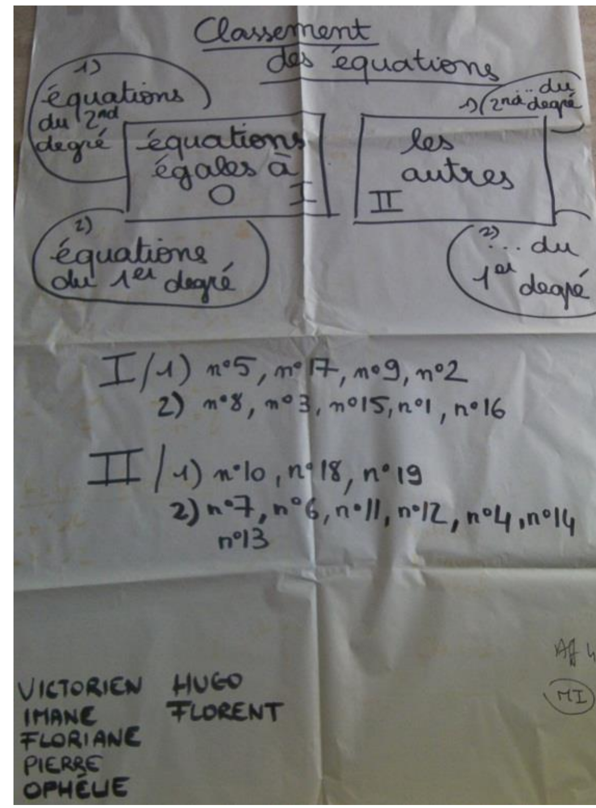
- Organisation : Travail en groupes de 3 à 4 élèves
- Tâche à effectuer : ***Les élèves doivent proposer une classification d'une vingtaine d'équations (choisies) du premier et du second degré, factorisées ou non, réduites ou non.***
- Matériel : des équations sur cartons déplaçables, une affiche pour collecter les classifications
- Restitution en grand groupe avec débat sur les différentes classifications.





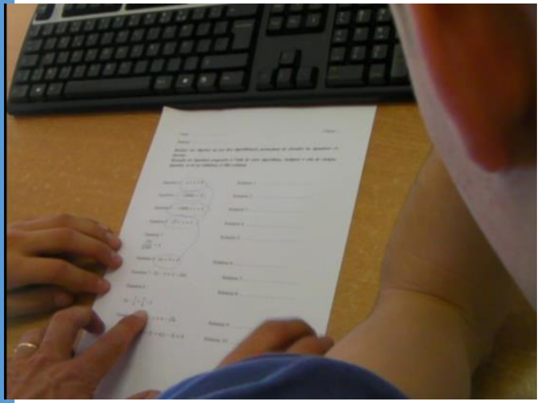
Phase de recherche par groupe

Phase de mise en commun

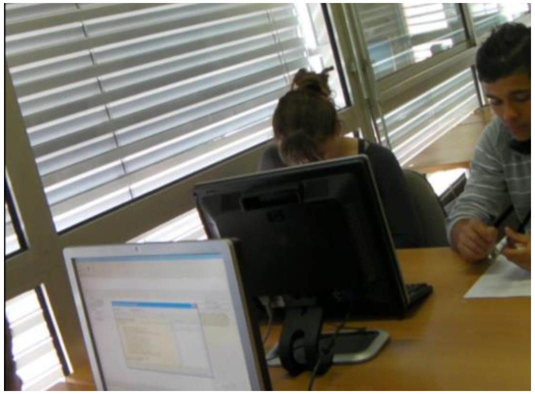


Un exemple d'affiche

Situation n°2



- Organisation : Travail en salle informatique en binômes
- Énoncé comportant une liste d'une dizaine d'équations du 1er degré



*Équation 1 :	$x + 3 = 0$	Équation 7 :	$\frac{7}{2}x + 3 = \frac{2}{3}$
*Équation 2 :	$2x - 3 = 4$	Équation 8 :	$\frac{7}{2}x + \frac{2}{5} = \frac{8}{3} + \frac{1}{2}x$
*Équation 3 :	$3 - 2x = -2$	Équation 9 :	$3 = 2x + 1$
Équation 4 :	$2 + x = 5x$	Équation 10 :	$3x + 2 = 5 + 3x$
Équation 5 :	$2x + 3 = 3x + 1$	Équation 11 :	$1{,}8x - 3 = 2{,}5x + 7{,}4$
Équation 6 :	$8 - x = \sqrt{2}$		

- Réaliser un algorithme permettant de résoudre les 3 premières équations, **sans les transformer au préalable**.
- Signaler par une étoile les équations similaires. Faire fonctionner l'algorithme pour ces équations.
- Comment peut-on résoudre les équations restantes avec un autre algorithme ? Le construire et les résoudre à l'aide de celui-ci.

Exemple de fiche de travail distribuée aux élèves

But pour l'élève et enjeu d'enseignement

- **But pour l'élève** : réalisation d'un programme qui résolve « automatiquement » une liste d'équations

Équation 1 :	$x + 3 = 0$	Équation 7 :	$\frac{7}{2}x + 3 = \frac{2}{3}$
Équation 2 :	$2x - 3 = 4$	Équation 8 :	$\frac{7}{2}x + \frac{2}{5} = \frac{8}{3} + \frac{1}{2}x$
Équation 3 :	$3 - 2x = -2$	Équation 9 :	$3 = 2x + 1$
Équation 4 :	$2 + x = 5x$	Équation 10 :	$3x + 2 = 5 + 3x$
Équation 5 :	$2x + 3 = 3x + 1$	Équation 11 :	$1{,}8x - 3 = 2{,}5x + 7{,}4$
Équation 6 :	$8 - x = \sqrt{2}$		

- **Enjeu d'enseignement** : reprise du concept d'équation et des objets qui gravitent autour de ce concept.
- La situation offre un travail de ***modélisation*** des équations → détermination d'une équation *paramétrée* qui couvre tous les cas de figure d'une liste d'équations donnée.
- Un *modèle* pour les trois premières équations de la liste *: ax + b = c ; un modèle* pour toutes les équations : *ax + b = cx + d*

Double *transposition* pour la conception d'un algorithme/ programme de résolution d'équations du type ax + b = cx + d

1ère transposition **2e transposition**

Résolution Mathématique

$$ax + b = cx + d$$

$$\cancel{ax - cx = d - b}$$

$$\cancel{(a - c)x = d - b}$$

- Si $a \neq c$

$$x = \frac{d - b}{a - c}$$

- Si a = c et $b \neq d$

Pas de solution

- Si a = c et $b = d$

Tout réel est solution

Résolution algorithmique

Données en entrée : a, b, c, d

- Si $a \neq c$

Donnée en sortie : $\frac{d-b}{a-c}$

- Si a = c et $b \neq d$

Message en sortie : Pas de sol.

- Si a = c et $b = d$

Message en sortie : Tout réel sol.

Résolution informatique

```
VARIABLES
  a EST_DU_TYPE NOMBRE
  b EST_DU_TYPE NOMBRE
  c EST_DU_TYPE NOMBRE
  d EST_DU_TYPE NOMBRE
  x EST_DU_TYPE NOMBRE
DEBUT_ALGORITHME
  LIRE a
  LIRE b
  LIRE c
  LIRE d
  SI (a!=c) ALORS
    DEBUT_SI
    x PREND_LA_VALEUR (d-b)/(a-c)
    AFFICHER x
    FIN_SI
    SINON
      DEBUT_SINON
      SI (b!=d) ALORS
        DEBUT_SI
        AFFICHER "pas de solution"
        FIN_SI
        SINON
          DEBUT_SINON
          AFFICHER "tout nombre est solution"
          FIN_SINON
      FIN_SINON
FIN_ALGORITHME
```

Pensée algébrique
Pensée algorithmique

Pensée algébrique

Définition au sens de Radford (2006 et 2008)

Caractérisée par deux grands principes :

- possibilité de nommer des quantités indéterminées ou inconnues (dans des registres variés comme la langue, des schémas ou du symbolisme divers)
- possibilité de raisonner sur ces quantités comme si elles étaient connues.

Pensée algorithmique

Définition au sens de Modeste (2012)

- en tant que pensée mathématique parmi d'autres (intra-mathématique) :
  - elle est une approche particulière de certains types de problèmes mathématiques
- en tant que pensée extra-mathématique (informatique) :
  - questionne l'efficacité des algorithmes qu'elle produit → *complexité*
  - utilise la notion de variable informatique → *affectation*

Bilan de l'expérimentation : entrée des élèves ***dans*** une pensée algébrique ***et dans*** une pensée algorithmique

Equation 8 : $\underset{a}{\frac{7}{2}x} + \underset{b}{\frac{2}{5}} = \underset{d}{\frac{8}{3}} + \underset{c}{\frac{1}{2}x}$

– Equation 9 : $3 = 2x + 1$

Equation 10 : $\underset{a}{3x} + \underset{b}{2} = \underset{d}{5} + \underset{c}{3x}$

Equation 11 : $\underset{a}{1{,}8x} - \underset{b}{3} = \underset{c}{2{,}5x} + \underset{d}{7{,}4}$

Production de l'élève Marine considérant des équations du premier degré sous une forme générique

1er modèle

$ax + b = c$

$x = \frac{(c-b)}{a}$

2nd modèle

Si $(a-c) \neq 0$: $ax + b = cx + d$

$x = \frac{(d-b)}{(a-c)}$

Si $(a-c) = 0$

Production de l'élève Julie considérant la modélisation des équations

- → Utilisation en actes de *paramètres* pour l'écriture sous une appellation unique des différentes équations données.
- → Capacité à dégager une structure générale algébrique

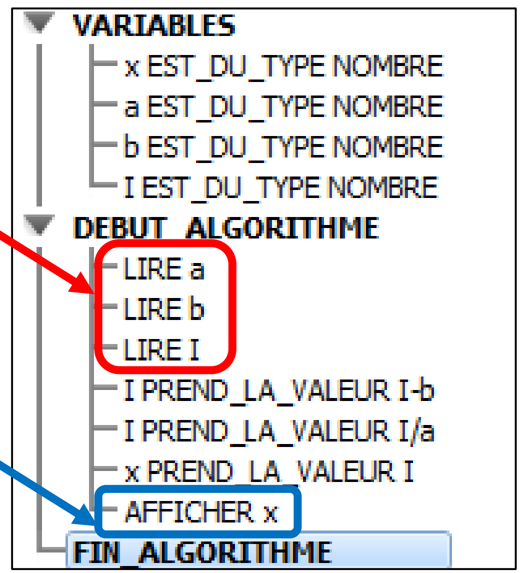
- → Capacité à généraliser la technique de résolution d'une équation du premier degré.
- → Entrée dans la capacité à envisager des cas particuliers

Bilan de l’expérimentation : entrée des élèves dans une pensée algébrique *via* une pensée algorithmique

→ *L’introduction de l’algorithmique et de la programmation favorise l’enseignement et l’apprentissage de concepts algébriques : Indicateurs dans les apprentissages des élèves.*

Paramètres :
données d’entrée
à faire lire à l’ordinateur

Inconnue :
donnée de sortie
du programme



Production de l élève Thomas :
Résolution de ax + b = I

Différentiation du rôle
des lettres :
Nouveau *registre* pour
les objets d’une
équation

Bilan de l'expérimentation : entrée des élèves dans une pensée algébrique ***via*** une pensée algorithmique

→ *L'introduction de l'algorithmique et de la programmation favorise l'enseignement et l'apprentissage de concepts algébriques : témoignage d'un enseignant*

Page suivante

Extrait de l'entretien avec le professeur Éric

- **Professeur Éric :** J'ai trouvé ça vraiment porteur, parce qu'ils ont réussi, à la fin des séances, à repérer de suite quel type d'équations c'était, qui est *a*, qui est *b*, etc. Alors que souvent, quand je fais cours avec les seconde, on fait ça petit à petit, et je sais que c'est pas ancré, ça a toujours été un problème. Par rapport à ce que je fais d'habitude, ça c'est largement ancré.
- **Chercheur :** Tu dirais que ce qu'on a gagné ici dans l'apprentissage de l'algèbre, c'est la compréhension des paramètres ?
- **Professeur Éric** : Exactement, oui. Et par rapport aux équations, ils se posent la question de quel type d'équations c'est, et une fois qu'on a relevé le type d'équations, ils savent à peu près faire la démarche, je dis à peu près parce que c'est pas complètement ancré. Mais ceci dit, avant il y avait vraiment un gros problème, quand il y avait des nombres à virgule ou des racines carrées. Alors que maintenant, grâce aux paramètres justement, et avec l'algorithmique, ça a permis de cataloguer et de se dire par exemple « ça, c'est pareil que $ax + b$ » et ils y arrivent. D'ailleurs j'ai remarqué que certains élèves passent par $ax + b$, quand le *a* et le *b* sont compliqués, finalement ils cherchent la formule [*pour déterminer la solution de l'équation*] …
- **Chercheur :** Tu veux dire qu'ils remplacent les valeurs numériques par *a* et *b* ?
- **Professeur Éric :** Oui, quand *a* et *b* sont des racines carrées… Pour quelques élèves, pas tous, j'ai remarqué qu'ils faisaient ça. Ils cherchent la formule générale et ensuite ils remplacent par les valeurs numériques.

Progression complète : situation 1

	Détails des contenus	Commentaires
En amont de la situation n°1	Premiers concepts d'algorithmique (actions élémentaires, structure d'un algorithme, structures alternatives avec condition) et de programmation (variables informatiques, instructions, tests) avec prise en main d'un logiciel de programmation (apprentissage du langage spécifique).	Plusieurs séances sont nécessaires pour la mise en place de ces premiers concepts. Les enseignants expérimentateurs en avaient réalisé 2 ou 3, ce qui ne semble pas suffisant, au vu des résultats.
Situation n°1	Faire classer des équations comportant un mélange d'équations polynomiales du premier et du second degré, sous des formes diverses (factorisées, développées,…).	Pour avancer plus rapidement vers une classification des équations permettant de faire émerger des techniques de résolution, la consigne précise que le critère de classification porte « sur la façon de les résoudre » et que le verbe « classer » est pris dans le sens de former des catégories et non pas d'ordonnancer.

Progression complète : situation 2

	Détails des contenus	Commentaires
Situation n°2	- Faire résoudre en environnement papier-crayon quelques équations du premier degré données sous des formes diverses. - Faire concevoir un algorithme pour résoudre des équations du premier degré (choisies parmi les précédentes), données sous la forme ax + b = c, où a, b c sont des nombres déterminés. Déterminer la structure de cet algorithme puis le programmer. - Institutionnaliser le concept de paramètre. - Faire concevoir un algorithme pour résoudre des équations du premier degré, données sous la forme ax + b = cx + d où a, b, c, d sont des nombres déterminés. Déterminer la structure de cet algorithme puis le programmer. - Institutionnaliser sur la forme générique d'une équation du premier degré (ax + b = cx + d) et sa résolution. Faire apparaître que la forme ax + b = c en est un cas particulier. - Proposer des équations du premier degré qu'il faut transformer au préalable (en environnement papier-crayon) pour appliquer le programme de résolution des équations de la forme ax + b = cx +d.	La résolution des équations en environnement papier-crayon permet des contrôles et des rétroactions : - contrôler si le programme fonctionne et le valider ; - contrôler si les résolutions « à la main » sont justes ; - comparer les résultats obtenus et interpréter les différences (valeur exacte « à la main » et approchée « à la machine » par exemple). Le second algorithme est une complexification du premier. La proposition d'une équation vérifiant a = c est proposée de manière à obliger l'élève à considérer le test a = c et à ajouter une structure alternative à l'algorithme. L'objectif est de montrer ici que moyennement quelques transformations élémentaires (développement, réduction), toute équation du premier degré s'écrit sous cette même forme.

Progression complète : situations 2 et 3

	Détails des contenus	Commentaires
Situation n°2bis	Proposer un mélange d'équations du premier et du second degré et demander de résoudre celles qu'il est possible de résoudre avec le programme de la situation n°2 (de résolution des équations de la forme $ax + b = cx + d$).	Reprise de la situation n°1 sur la catégorisation : approfondir les concepts travaillés lors de la première situation et faire le lien avec la résolution de ces équations, en faisant émerger qu'une même technique de résolution est applicable pour toutes (quelle que soit la nature des coefficients). Différencier 1^{er} et 2nd degré.
Situation n°3 (début)	- Faire résoudre en environnement papier-crayon des équations du second degré données sous les formes $(ax + b)(cx + d) = 0$ et $x^2 = a$ ou s'y ramenant par des transformations simples. - Faire concevoir un algorithme pour résoudre des équations du second degré, données sous la forme $(ax + b)(cx + d) = 0$, où a, b c, d sont des nombres déterminés. Déterminer la structure de cet algorithme puis le programmer.	La résolution en environnement papier-crayon a le même objectif de contrôle et de rétroaction que dans la situation n°2. Par transformations simples, nous entendons celles qui ne font pas appel à la recherche de la forme canonique (ce n'est pas au programme de seconde). Les équations des listes seront proposées sous les formes $(ax + b)(cx + d) = 0$ ou $x^2 = a$, ou s'y ramenant moyennant une factorisation par application d'une identité remarquable ou la recherche d'un facteur commun.

Progression complète : situation 3

	Détails des contenus	Commentaires
Situation n°3 (fin)	- Proposer une liste d'équations du second degré et demander de résoudre celles qu'il est possible de résoudre avec le programme précédent. - Faire concevoir un algorithme pour résoudre des équations du second degré, données sous la forme $x^2 = a$ où a est un nombre déterminé. Déterminer la structure de cet algorithme puis le programmer. - Institutionnaliser sur la résolution des équations de la forme $(ax + b)(cx + d) = 0$ en faisant apparaître que la forme $x^2 = a$ en est un cas particulier lorsque a est positif ou nul. - Proposer une liste d'équations du second degré et demander de résoudre celles qu'il est possible de résoudre avec les deux programmes précédents.	Les équations des listes seront proposées sous les formes $(ax + b)(cx +d) = 0$ ou $x^2 = a$, ou s'y ramenant moyennant une factorisation par application d'une identité remarquable ou la recherche d'un facteur commun. L'institutionnalisation permet de montrer la technologie sous-jacente aux deux techniques de résolution et d'englober ces techniques On montrera quelques équations du second du second degré qui possèdent des solutions réelles et dont on ne connaît pas de technique pour les factoriser systématiquement : peut déboucher sur la forme canonique, en actes.

EXEMPLE D'UNE PROGRESSION EN 2^NDE : UTILISATION DE L'ALGORITHMIQUE POUR LA COMPRÉHENSION DU CONCEPT DE DICHOTOMIE

La place de l'algorithmique dans le programme de seconde

Extraits du programme de seconde (2009)

Organisation du programme

Le programme est divisé en trois parties,

- Fonctions
- Géométrie
- Statistiques et probabilités

Les capacités attendues dans le domaine de l'algorithmique d'une part et du raisonnement d'autre part, sont transversales et doivent être développées à l'intérieur de chacune des trois parties. Des activités de type algorithmique possibles sont signalées dans les différentes parties du programme et précédées du symbole ◇.

Domaine ***Fonctions***

Expressions algébriques Transformations d'expressions algébriques en vue d'une résolution de problème.	• Associer à un problème une expression algébrique. • Identifier la forme la plus adéquate (développée, factorisée) d'une expression en vue de la résolution du problème donné. • Développer, factoriser des expressions polynomiales simples ; transformer des expressions rationnelles simples.
Équations Résolution graphique et algébrique d'équations.	• Mettre un problème en équation. • Résoudre une équation se ramenant au premier degré. ◇ Encadrer une racine d'une équation grâce à un algorithme de dichotomie.

La place de l'algorithmique dans le programme de seconde

Extraits du programme de seconde (2009)

Domaine ***Géométrie***

Configurations du plan Triangles, quadrilatères, cercles.	Pour résoudre des problèmes : • Utiliser les propriétés des triangles, des quadrilatères, des cercles. • Utiliser les propriétés des symétries axiale ou centrale.	Les activités des élèves prennent appui sur les propriétés étudiées au collège et peuvent s'enrichir des apports de la géométrie repérée. ◇ Le cadre de la géométrie repérée offre la possibilité de traduire numériquement des propriétés géométriques et permet de résoudre certains problèmes par la mise en œuvre d'algorithmes simples.

Domaine ***Statistiques et Probabilités***

Échantillonnage Notion d'échantillon. Intervalle de fluctuation d'une fréquence au seuil de 95%*.	• Concevoir, mettre en œuvre et exploiter des simulations de situations concrètes à l'aide du tableur ou d'une calculatrice.	Un échantillon de taille n est constitué des résultats de n répétitions indépendantes de la même expérience. À l'occasion de la mise en place d'une simulation, on peut : • utiliser les fonctions logiques d'un tableur ou d'une calculatrice, ◇ mettre en place des instructions conditionnelles dans un algorithme.
Réalisation d'une simulation.	• Exploiter et faire une analyse critique d'un résultat d'échantillonnage.	

Objectif général de l’expérimentation menée

« Les différentes activités trouvées sur ce sujet nous ont permis de voir ce que nous ne voulions pas faire, à savoir donner l'algorithme et demander aux élèves à quoi il sert. »

Mémoire de Master 2 MEEF- Parcours Maths (2015). C. Porte & M. Revet. *Comment amener l'algorithme de dichotomie pour la résolution d'équation en classe de seconde générale ?* ESPE de Montpellier

L'expérimentation menée sur la dichotomie en 4 situations

	Situation 1	Situation 2	Situation 3	Situation 4
Contenu	Écrire un algorithme de calcul pour définir une fonction	Déterminer une fonction affine par morceaux à partir d'un problème de pourcentage	Deviner un nombre entier compris entre 1 et 100. L'ordinateur joue successivement les deux rôles (celui qui sait puis celui qui devine)	Écrire l'algorithme de résolution d'équation par dichotomie pour une fonction reprise d'un DM : $f(x) = x^2 + (5-\sqrt{2})x + 5\sqrt{2}$
Fonction / Objectif	Découvrir l'algorithmique et le logiciel Algobox Lien entre algèbre/analyse et algorithmique	Lier résolution graphique/ algébrique d'une équation et algorithmique	Découvrir l'algorithme de dichotomie. Montrer comment il accélère la réussite du jeu (avant transfert vers la résolution d'équation)	Comprendre l'intérêt de l'algorithme de dichotomie pour encadrer une racine d'une équation.
Objets algo/ prog	Variable informatique ; affectation de variable ; étapes d'un programme de calcul ; instructions d'entrée/sortie	Test conditionnel « si … alors… sinon »	Instructions itératives et utilisation d'une boucle « tant que »	Mise en œuvre d'un algorithme complexe

Situation 1 (06/01/15) (1h en demi-groupe)

1. Quelles sont les variables informatiques utilisées dans l'algorithme ci-contre ?

....

ENTRÉE *Saisir n*
TRAITEMENT
a prend la valeur n+4
*a prend la valeur a*n*
a prend la valeur a+4
SORTIE *afficher a*

2. Remplir le tableau de suivi des variables (appelé aussi tableau d'étapes) ci-dessous avec le nombre **2** en entrée.

Variables	*n*	*a*
Ligne 1 (ENTREE)	2	0
Ligne 3 (Instruction : « a prend la valeur *n*+4 »)	2	6
Ligne 4 (Instruction : « a prend la valeur *a*n* »)	2	12
Ligne 5 (Instruction : « a prend la valeur *a*+4 »)	2	16
Ligne 6 (SORTIE), le programme affiche :	....	

3. Refaire un nouveau tableau sur votre compte-rendu de TP pour recommencer avec le nombre – 6.

9. ☐ Entrer le programme ci-contre en utilisant l'aide.

10. ☐ Tester votre algorithme avec le nombre 2

11. ☐ Le résultat est-il le même que celui trouvé avec le tableau de suivi des variables ?

12. ☐ En utilisant le mode pas à pas (voir aide) et en observant bien ce qu'affiche Algobox, recommencer avec -6 .

13. ☐ Expliquer en quelques mots ce que fait le mode pas à pas.

```
VARIABLES
  n EST_DU_TYPE NOMBRE
  a EST_DU_TYPE NOMBRE
DEBUT_ALGORITHME
  LIRE n
  a PREND_LA_VALEUR n+4
  a PREND_LA_VALEUR a*n
  a PREND_LA_VALEUR a+4
  AFFICHER a
FIN_ALGORITHME
```

16. ☐ Ecrire l'expression obtenue si on applique le programme de calcul à un nombre quelconque que l'on appellera x .

17. ☐ Transformer l'expression obtenue pour démontrer votre conjecture.

Algorithmique/programmation

Découverte de :
- Variable informatique
- Affectation de variable
- Étapes d'un programme de calcul
- Instructions d'entrée/sortie
- Différence entre algorithme et langage de programmation

Mathématiques

- Reprise de la notion de fonction
- Identité remarquable
- Développement
- Factorisation

Source : TP7 p.42 de Math'x 2nde

Situation 2 (20/01/15) (2 séances d'1 h en ½ groupe)

1. Lors des soldes, un magasin affiche la promotion suivante : «Pour tout achat de moins de 100€, profitez de 20% de réduction. Puis, à partir de 100€ d'achat, bénéficiez de 65% de réduction sur la part de l'achat au-delà de 100€. ».
   a) Calculer le prix après réduction d'un article valant initialement 70 €, démontrer qu'un article valant initialement 120 € sera vendu 87 € après réduction.
   b) On note x le prix avant réduction et $p(x)$ le prix après réduction.
   - Pour $x \leq 100$,déterminer $p(x)$ en fonction de x .
   - Démontrer que, si $x > 100$, on a : $p(x) = 0{,}35\,x + 45$.

c) On considère l'algorithme ci-contre.
- Quel est le rôle de cet algorithme ?
- Compléter les pointillés
- Faire fonctionner cet algorithme en remplissant le tableau de suivi des variables ci-dessous, pour X=60 et donner le résultat obtenu.

```
(1) VARIABLES ................................
(2)     Saisir X
(3) Si  X ≤ 100
(4)     alors  Affecter à P la valeur 0,8 X
(5)     sinon  Affecter à P la valeur 0,35 X + 45
(6) FinSi
(7) Afficher P
```

Variables	.....	.....
Ligne 2 (ENTREE)		
Ligne 3-4-5-6 : Test conditionnel Si...Alors...Sinon....		
Ligne 7 (SORTIE), le programme affiche :	....	

d) Refaire un nouveau tableau sur votre compte-rendu de TP pour recommencer avec le nombre 150 .

4. ☐ Programmer l'algorithme sous Algobox (aide du TP précédent.
5. ☐ A l'aide de l'algorithme, établir le tableau de valeurs de la fonction p pour $x \in [0\,;200]$ avec un pas de 20.
6. ☐ Sauvegarder le fichier (Fichier\Sauver) .
7. ☐ Représenter graphiquement la courbe de la fonction p .
8. ☐ A l'aide de la représentation graphique, sachant qu'un client a payé 108€, conjecturer le prix de l'article avant réduction. Faire de même avec 72€.
9. ☐ Retrouver vos résultats par le calcul.

Algorithmique

Reprise de :
- Variable informatique
- Affectation de variable
- Étapes d'un programme de calcul
- Instructions d'entrée/sortie

Découverte de :
- Test conditionnel si alors sinon

Mathématiques

- Fonctions affines par morceaux
- Pourcentages
- Mise en équation d'un problème
- Résolution graphique et algébrique d'une équation

C- *Modifier l'algorithme*

Après une semaine de promotion, la direction du magasin décide d'une troisième tranche de réduction : A partir de 300€ d'achat, la promotion sera de 90% de réduction sur la part de l'achat au-delà de 300€.

Source : Manuel Repère Chapitre 2 p 48

Situation 3 (24/02/15) (1h en demi-groupe)

2de 14	TP - Jeu du nombre caché	2014/2015

Partie A

Léa choisit un nombre compris entre 1 et 100, Brahim doit le deviner.
Brahim fait des propositions et Léa répond "trop grand", "trop petit" ou "gagné".
Le jeu s'arrête lorsque Brahim a trouvé le nombre.

✍ Faire fonctionner le jeu, en binôme, en notant le déroulement du jeu dans un tableau.

✍ Quelles tactiques de jeu adoptez vous pour proposer un nombre en fonction de la réponse donnée (trop grand, trop petit).

On décide de remplacer Léa par un ordinateur.
On note N le nombre à trouver et R les propositions.
Un algorithme à faire exécuter à l'ordinateur dans ce cas pourrait être :

```
N←— Nombre aléatoire compris entre 1 et 100.
R←—0
Tant que R≠N Faire
    | Lire(R)
    | Si R>N Alors
    |     Afficher("Trop grand")
    | Si R<N Alors
    |     Afficher("Trop petit")
Afficher("Gagné")
```

Partie B

On décide de remplacer Brahim par un ordinateur.
Léa répond R = 0 si trouvé
R = (1) si le nombre proposé est trop grand
et R= (-1) si le nombre proposé est trop petit

Tactique de l'ordinateur : l'ordinateur propose **un nombre C au hasard** dans l'intervalle des possibilités.

✍ Ecrire l'algorithme de ce jeu.
Programmer l'algorithme sous Algobox

Tactique de l'ordinateur : l'ordinateur propose **un nombre C proche du milieu** de l'intervalle des possibilités.

✍ Ecrire l'algorithme de ce jeu.
Programmer l'algorithme sous Algobox

Partie C

✍ Tester les algorithmes de ce jeu, avec les deux tactiques de recherche : nombre au hasard dans l'intervalle et recherche **dichotomique.**

Combien d'essais sont-ils nécessaires dans chacun des cas ?

Vos conclusions en terme de performance ?

Algorithmique

Reprise de :
- Instructions d'entrée/sortie
- Test conditionnel si alors sinon

Découverte de :
- Boucle tant que

Source : publication de Mai 2010 de Henri ROLAND IREM Aix-Marseille : http://www.irem.univ-mrs.fr/IMG/pdf/dichotomie.pdf

Devoir maison

(entre les situations 3 et 4, vacances de Paques)

Exercice IV :

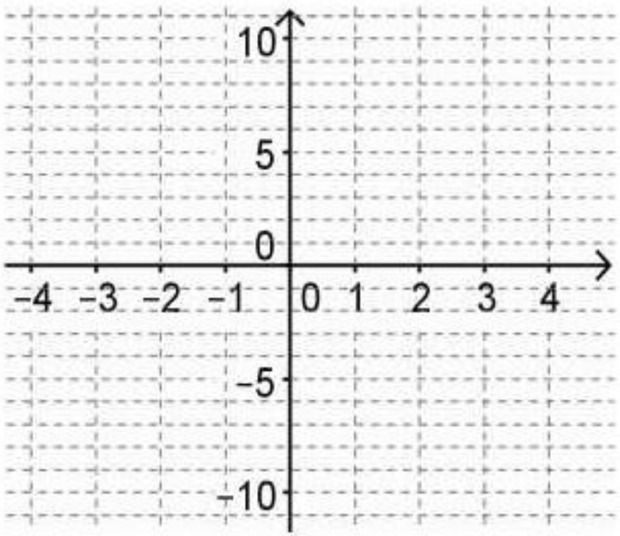
Soit la fonction *f* de courbe représentative C_f définie sur $\mathbb{R}$ par :

$$f(x)=x^2+(5-\sqrt{(2)})x-5\sqrt{(2)}$$

1) A l'aide de votre calculatrice,
en affichant le graphe de la fonction (choisir judicieusement la fenêtre d'affichage) et en vous aidant de la table des valeurs,

tracer ci-contre le plus précisément possible, une représentation graphique de la fonction *f* sur l'intervalle [-4;4].

(on ne demande pas de justifier)



***2)** Pour $x\in[1;2]$ compléter le tableau de valeurs ci-dessous : (on donnera des valeurs approchées à 10^{-2} près), (on ne demande pas de justifier)*

x	1	1,1	1,2	1,3	1,4	1,5	1,6	1,7	1,8	1,9	2
$f(x)$											

3) On admet dans cette question que l'équation $f(x) = 0$ a une unique solution sur l'intervalle [-4;4].
En vous aidant des questions 2 et 3, donner un encadrement au dixième près de la solution de l'équation $f(x) = 0$ (vous expliquerez votre démarche)

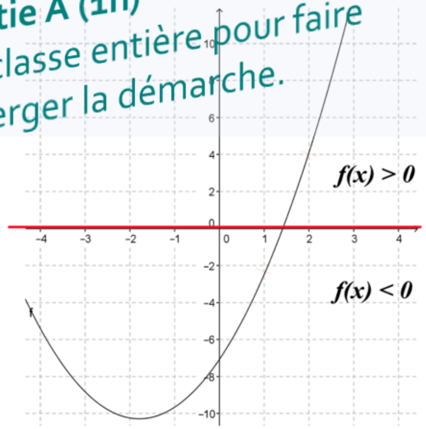
Situation 4 (28/04/15) (2 heures)

2de 14 Lycée Feuillade **Algorithme de résolution d'équation** 2014/2015

PARTIE A

Partie A (1h)
En classe entière pour faire émerger la démarche.

Soit la fonction f de courbe représentative C_f définie sur ℝ par : $f(x)=x^2+(5-\sqrt{2})x-5\sqrt{2}$



Ci-contre une représentation graphique de la fonction f sur l'intervalle [-4;4].

Ci-dessous le tableau de valeurs pour $x\in[1;2]$

x	1	1,1	1,2	1,3	1,4	1,5	1,6	1,7	1,8	1,9	2
$f(x)$	-2,49	-1,92	-1,33	-0,72	-0,09	0,56	1,23	1,91	2,62	3,35	4,1

On admet dans cette question que l'équation $f(x) = 0$ a une unique solution sur l'intervalle [-4;4].
$1,4 < x < 1,5$ est un encadrement au dixième près de la solution de l'équation $f(x) = 0$.

- Comment l'ordinateur pourrait chercher des encadrements plus précis ?

PARTIE B

Parties B et C (1h)
En demi-groupe pour la mise en œuvre.

- Écrire l'algorithme de cette recherche d'encadrement d'une solution d'équation.

PARTIE C

- Saisir le programme correspondant sous Algobox et le tester pour différentes valeurs.

Algorithmique

- Variable informatique
- Affectation de variable
- Étapes d'un programme de calcul
- Instructions d'entrée/sortie
- Test conditionnel si alors sinon
- Boucle tant que
- Modification, adaptation et complexification d'un algorithme existant

Mathématiques

- Fonctions polynomiale de d°2
- Résolution graphique d'équations
- Tableau de valeurs
- Monotonie d'une fonction
- Valeur exacte et approchées de la solution d'une équation
- Principe de dichotomie

Algorithmes de la situation 3
Partie A : l'ordinateur fait deviner le nombre auquel il a pensé

N = Valeur aléatoire choisie par l'ordinateur à deviner

Boucle Tant que

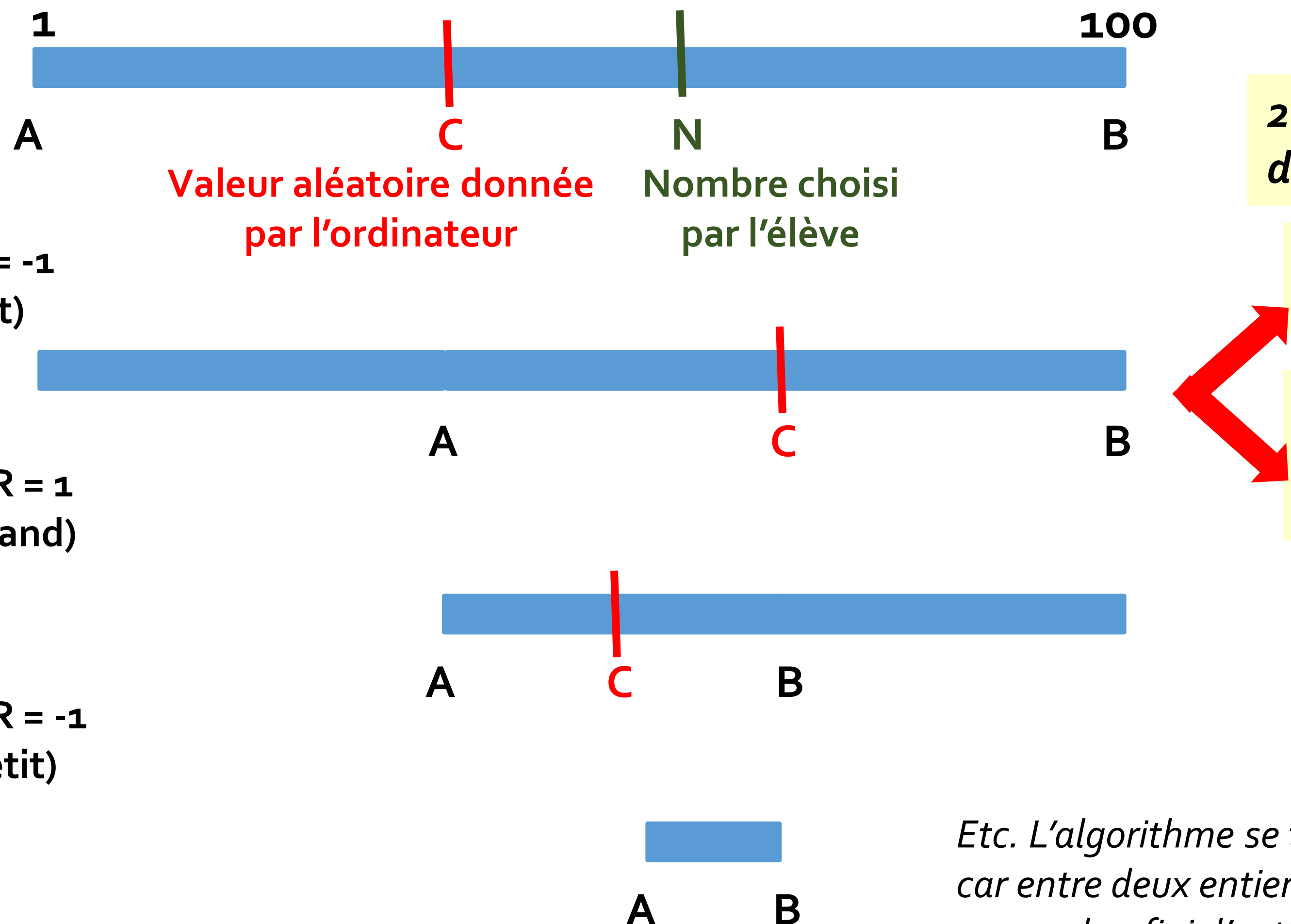
R = Valeur entrée par l'élève

Test de comparaison de R par rapport à N

```
VARIABLES
  R EST_DU_TYPE NOMBRE
  N EST_DU_TYPE NOMBRE
DEBUT_ALGORITHME
  N PREND_LA_VALEUR ALGOBOX_ALEA_ENT(1,100)
  R PREND_LA_VALEUR 0
  TANT_QUE (R != N) FAIRE
    DEBUT_TANT_QUE
    LIRE R
    SI (R>N) ALORS
      DEBUT_SI
      AFFICHER "trop grand"
      FIN_SI
    SI (R<N) ALORS
      DEBUT_SI
      AFFICHER "trop petit"
      FIN_SI
    FIN_TANT_QUE
  AFFICHER "Gagné !"
FIN_ALGORITHME
```

[Programme]

Algorithmes de la situation 3
Partie B : l'ordinateur devine le nombre auquel l'élève a pensé



Élève : R = -1 (trop petit)

Élève : R = 1 (trop grand)

Élève : R = -1 (trop petit)

2 types d'algorithmes :

$$C = \mathrm{Ent}\,\frac{A+B}{2}$$

C = Valeur aléatoire donnée par l'ordinateur

Etc. L'algorithme se terminera car entre deux entiers, il y a un nombre fini d'entiers

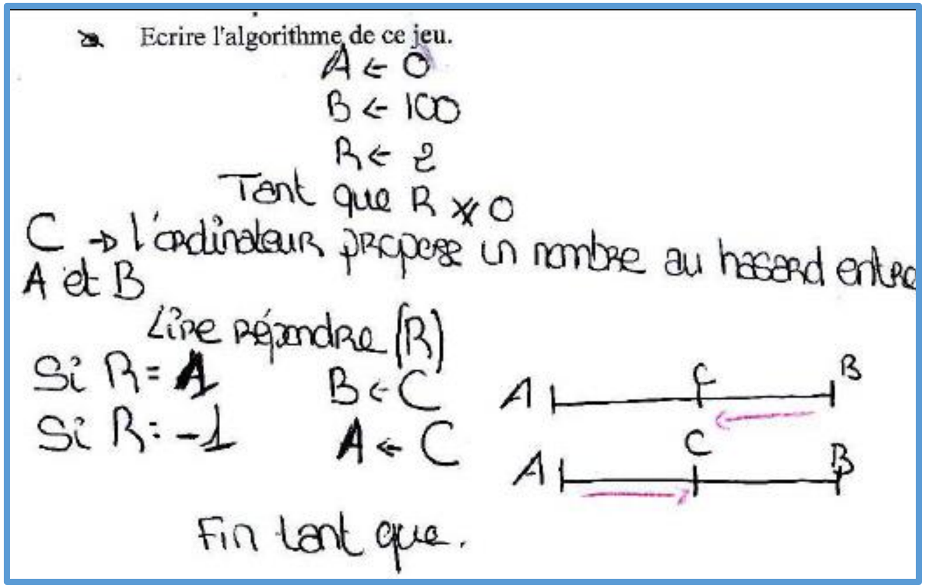
Algorithmes de la situation 3 Partie B : l'ordinateur devine le nombre auquel l'élève a pensé

```
VARIABLES
  R EST_DU_TYPE NOMBRE
  C EST_DU_TYPE NOMBRE
  A EST_DU_TYPE NOMBRE
  B EST_DU_TYPE NOMBRE
DEBUT_ALGORITHME
  C PREND_LA_VALEUR ALGOBOX_ALEA_ENT(1,100)
  AFFICHER C
  R PREND_LA_VALEUR 1000
  A PREND_LA_VALEUR 1
  B PREND_LA_VALEUR 100
  TANT_QUE (R!=0) FAIRE
    DEBUT_TANT_QUE
    LIRE R
    SI (R==1) ALORS
      DEBUT_SI
      B PREND_LA_VALEUR C-1
      C PREND_LA_VALEUR floor((A+B)/2)
      FIN_SI
    SI (R==-1) ALORS
      DEBUT_SI
      A PREND_LA_VALEUR C+1
      C PREND_LA_VALEUR floor((A+B)/2)
      FIN_SI
    AFFICHER C
    FIN_TANT_QUE
FIN_ALGORITHME
```

[Programme]

Quelques résultats pour la situation 3 : l'écriture de l'algorithme

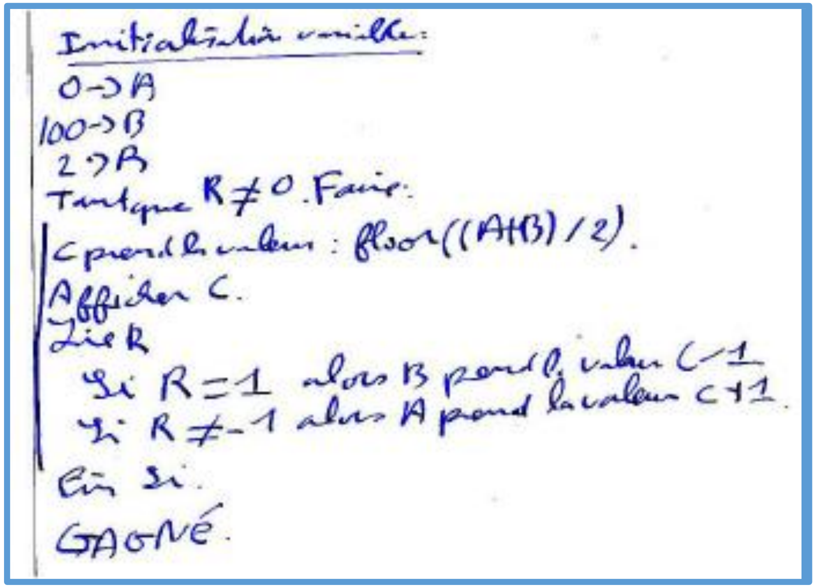
- **Algorithme proposant un nombre au hasard dans l'intervalle des possibilités (10 binômes sur 15)**



Ecrire l'algorithme de ce jeu.

A ← 0
B ← 100
R ← 2
Tant que R ≠ 0
C → l'ordinateur propose un nombre au hasard entre A et B
Lire réponse (R)
Si R = 1 B ← C
Si R = -1 A ← C
Fin tant que.

- **Algorithme proposant un nombre au milieu de l'intervalle des possibilités (8 binômes sur 15)**



Initialiser variables :
0 -> A
100 -> B
2 -> R
Tant que R ≠ 0 Faire.
C prend la valeur : floor((A+B)/2).
Afficher C.
Lire R
Si R = 1 alors B prend la valeur C-1
Si R ≠ -1 alors A prend la valeur C+1.
Fin Si.
GAGNÉ.

Quelques résultats pour la situation 3 : la « magie » de l'ordinateur

→ Pas de modification des bornes de l'intervalle de recherche, erreur faite par 3 binômes sur 15

→ Dialogue du binôme avec l'enseignante :

- Élève : *Pourquoi l'ordinateur ne donne pas un nombre plus petit alors qu'il y a bien marqué « trop petit » ?*
- Prof : *Mais comment l'ordinateur peut le savoir ?*
- Élève : *On lui a dit, « trop petit » !*
- Prof : *Vous avez demandé à l'ordinateur d'afficher « trop petit » à l'écran, mais à aucun moment cette donnée n'est utilisée par le programme pour modifier les bornes de l'intervalle de recherche…*

Difficulté récurrente des « actions élémentaires » : qu'est-ce qui est compréhensible par la machine ?

Code de l'algorithme

```
VARIABLES
  C EST_DU_TYPE NOMBRE
  R EST_DU_TYPE NOMBRE
  A EST_DU_TYPE NOMBRE
  B EST_DU_TYPE NOMBRE
DEBUT_ALGORITHME
  R PREND_LA_VALEUR 2
  A PREND_LA_VALEUR 0
  B PREND_LA_VALEUR 100
  C PREND_LA_VALEUR 0
  TANT_QUE (R!=0) FAIRE
    DEBUT_TANT_QUE
    C PREND_LA_VALEUR ALGOBOX_ALEA_ENT(A,B)
    AFFICHER C
    LIRE R
    SI (R==1) ALORS
      DEBUT_SI
      AFFICHER "trop grand "
      FIN_SI
    SI (R==-1) ALORS
      DEBUT_SI
      AFFICHER "trop petit"
      FIN_SI
    SI (R==0) ALORS
      DEBUT_SI
      AFFICHER "trouvé"
      FIN_SI
    FIN_TANT_QUE
FIN_ALGORITHME
```

Quelques résultats pour la situation 3 : comparaison des algorithmes « nombre au hasard » à l'algorithme « dichotomie »

- Il s'agit de comparer les deux algorithmes en terme de performance et de déterminer, s'il existe, le nombre d'essais maximum avec la recherche dichotomique.
- Les deux tactiques de jeu sont testées et les 8 groupes ont constaté que la recherche dichotomique est la plus performante en terme de nombre de tentatives pour trouver.

Résultat obtenu pour un (seul) binôme :

1er algo :

83 → 5 essais.	72 → 9 essais
36 → 6 essais	92 → 6 essais.
48 → 7 essais	12 → 8 essais

2ème algo :

22 → 6 essais.	82 → 6 essais.
8 → 5 essais.	46 → 4 essais
67 → 3 essais.	73 → 5 essais.

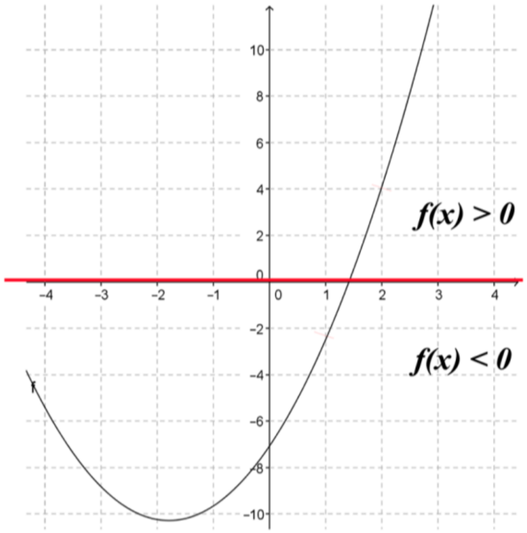
Le 2ème algorythme est plus performant

Par dichotomie, le nombre maximum d'essais est 7 car 2^6 trop petit → $64 < 100$.
$2^{\boxed{7}}$ ça marche → $128 > 100$.

Quelques résultats pour la situation 4 : partie A- Recherche d'une méthode d'encadrement d'une racine d'une équation

<u>PARTIE A</u>

Soit la fonction f de courbe représentative C_f définie sur $\mathbb{R}$ par : $f(x) = x^2 + (5 - \sqrt{(2)})x - 5\sqrt{(2)}$



Ci-contre une représentation graphique de la fonction f sur l'intervalle [-4;4].

Ci-dessous le tableau de valeurs pour $x \in [1;2]$

x	1	1,1	1,2	1,3	1,4	1,5	1,6	1,7	1,8	1,9	2
$f(x)$	-2,49	-1,92	-1,33	-0,72	-0,09	0,56	1,23	1,91	2,62	3,35	4,1

On admet dans cette question que l'équation $f(x) = 0$ a une unique solution sur l'intervalle [-4;4].
$1{,}4 < x < 1{,}5$ est un encadrement au dixième près de la solution de l'équation $f(x) = 0$.

✍ Comment l'ordinateur pourrait chercher des encadrements plus précis ?

Quelques résultats pour la situation 4 : partie A- Recherche d'une méthode d'encadrement d'une racine d'une équation

Quelques extraits du dialogue prof/élèves :

- E : *On fait un zoom sur la courbe pour voir les valeurs plus précises quand la courbe traverse l'axe des x, avec GeoGebra c'est facile à faire.*
- E : *Avec GeoGebra, on peut aussi utiliser le calcul formel pour résoudre et on a directement la réponse.*
- Prof *: C'est vrai, je n'avais pas précisé comment nous allions utiliser l'ordinateur. On souhaite écrire un algorithme puis le programmer dans Algobox. »*
- E *: Entre 1 et 2 on a fait une table de valeurs. On refait une table de valeurs entre 1,4 et 1,5 et on regarde quand ça change de signe.*
- E : *Pour deviner le nombre caché, il fallait aussi changer les bornes de l'intervalle pour en avoir un plus petit.*
- E : *On divise l'intervalle en deux et on garde le morceau où ça change de signe, et après on recommence.*
- Prof : *Comment effectuer le test pour savoir si les bornes de l'intervalle ont des images de signes opposés ?*
- E *: On teste si f(a) est positive et f(b) négative et aussi le contraire*
- Prof *: Pourriez-vous trouver un test qui regrouperait ces deux cas ?*
- E : *on divise l'un par l'autre et on regarde le signe .*
- Prof : *Et comment faire pour savoir quand il faudrait arrêter la recherche et afficher l'encadrement trouvé ?*
- E *: On divise l'intervalle un nombre de fois suffisant.*
- E *: On divise l'intervalle jusqu'à obtenir la précision qu'on veut.*

Quelques résultats pour la situation 4 : partie A- Recherche d'une méthode d'encadrement d'une racine d'une équation

Un exemple de production d'élève sur la méthode de dichotomie :

Les comditioms sur la fonction qui doivent être remplies sont : - elle doit passer par l'axe des absisses

- elle doit être strictement momotome (croissante ou décroissante)

- une borne d'un signe et une borne de l'autre. f(a) signe contraire à f(b)

Divise l'interval en 2

- On prend le milieu (exemple milieu de a - b)

- Il va falloir tester et garder les intervals où il y a les 2 signes contraires.

ex :

a			C (milieu)			b
~	~	~	~	~	~	~
- . . .	- . .	-0,09	0,56	~~1,23~~	~~1,91~~	~~.~~

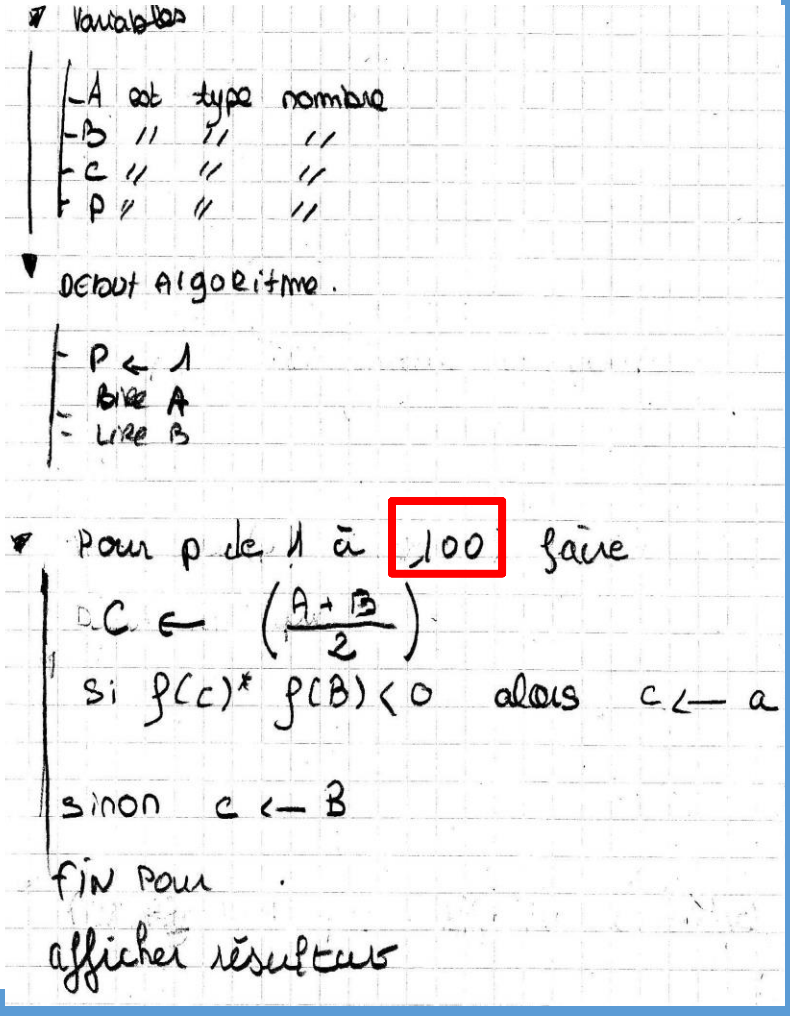
On fait cela jusqu'à s'arrêter

Quelques résultats pour la situation 4 : partie B- Recherche et programmation de l’algorithme de dichotomie

Commentaires des auteurs du mémoire : *Arrivé à ce stade l'hétérogénéité de la classe est à nouveau ressortie avec une grande disparité de compréhension allant de ceux qui ne savent pas du tout quoi faire à ceux qui ont à peine eu besoin d'un coup de pouce pour finir.*

- **Algorithme avec une boucle « pour »**

Correspond à 2^{100} divisions de l’intervalle [A ; B]


Variables
- A est type nombre
- B " " "
- C " " "
- P " " "

Debut Algoritme.
- P ← 1
- Lire A
- Lire B

Pour p de 1 à $\boxed{100}$ faire
C ← $\left(\frac{A+B}{2}\right)$
Si f(c)* f(B) < 0 alors c ← a
sinon c ← B
fin Pour.
afficher résultats

Voir le [programme]()

Quelques résultats pour la situation 4 : partie B- Recherche et programmation de l'algorithme de dichotomie

- Algorithme avec une boucle « tant que »

La précision est fixe

Partie B :

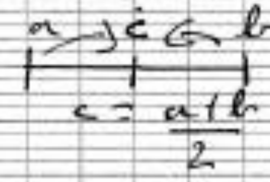
Saisir les variables :

a : 1ère borne de l'intervalle.

b : 2ème borne de l'intervalle.

c : (a+b)/2.

precision : précision atteinte $\boxed{p = 0{,}0001}$




```
* Tant que (b-a) > precision
      c = (a+b)/2

          Si  F1(c) x F1(b) > 0
            alors b <- c.
            Sinon a <- c
          Fin Si
    Fin tant que.

  Afficher a.
  Afficher b.
```

Voir le [programme]()

CONCLUSION

Conclusion relative à l'expérimentation sur la résolution d'équations

- **La pensée algorithmique comme vecteur pour revisiter des objets de l'algèbre, les généraliser, mais aussi les matérialiser**

Rapprochement avec un *micro-monde* dont Capponi et Laborde (1995) donnent la définition suivante :

> *Un micro-monde est une création d'un monde de réalités artificielles fournissant un modèle (au sens des logiciens) d'une théorie. Ce monde comporte des objets sur lesquels on peut agir grâce à des actions, on peut aussi créer de nouveaux objets. (p. 265).*

Application de ce concept au logiciel de programmation Algobox : possibilité de créer des représentations d'un objet théorique et d'agir sur ces représentations.

→ L'algorithmique et la programmation forment un micro-monde pour le domaine algébrique, où il est possible d'explorer et d'expérimenter sur des objets de l'algèbre. Ceci renforce une entrée dans la pensée algébrique.

Conclusion relative à l'expérimentation sur la dichotomie

- **L'algorithme de dichotomie est accessible, dans sa conception, à des élèves de seconde**

Une progression sur une séquence étalée dans le temps permet d'amener les élèves à concevoir eux-mêmes l'algorithme de dichotomie pour encadrer une solution d'une équation.

La classe de seconde est une classe de détermination et il ne s'agit pas d'y former des programmeurs mais de faire en sorte que les mathématiques et l'algorithmique soient au service d'activités de résolution de problèmes pour les sciences. (Document ressources, Classe de seconde, algorithmique)

→ L'algorithmique et la programmation se présentent comme un moyen de trouver rapidement une solution approchée d'une équation par la dichotomie. Elles se trouvent au service de la résolution d'un problème mathématique.

Conclusion relative aux deux expérimentations menées

- **Les contraintes et difficultés rencontrées**

→ *Pour l'algorithme de dichotomie :*

Nécessité d'aborder l'ensemble des instructions et des objectifs communs aux trois années de lycée pour le concevoir (boucle tant que, test conditionnel, …) . Difficile d'envisager une progression de la seconde à la terminale de ce point de vue.

→ *Pour les deux expérimentations*

- Nécessité d'une instrumentation relativement poussée : On ne peut pas faire l'économie de la nécessaire genèse instrumentale, au sens de Rabardel, pour obtenir des résultats probants. Du temps est nécessaire …

- Difficultés rencontrées par les enseignants pour intégrer l'algorithmique et la programmation dans leur enseignement des mathématiques. Ces professeurs sont autodidactes. Nécessité d'un *équipement praxéologique* de la profession, comportant à la fois des savoirs de référence sur l'algorithmique, la programmation et la logique mais aussi des savoirs pour enseigner ces nouvelles disciplines.

Conclusion générale

- **La pensée algorithmique, comme pensée intra mathématique mais aussi extra mathématique**
- La résolution algorithmique et son écriture dans un environnement informatique nécessitent une *transposition*.
- La complexité de cette transposition vient :
  - de la *non-congruence* entre les environnements papier-crayon et informatisé
  - d'une *pensée algorithmique* qui n'est pas contenue entièrement dans la *pensée mathématique*

→ Ajouter un point de vue informatique (donc extra-mathématique) permet de prendre en considération une dimension qui n'existe pas en environnement mathématique usuel papier-crayon, et qui induit un mode de pensée différent.

→ Considérer cette pensée comme spécifique est un élément didactique important pour mieux comprendre la transposition qui se produit lors de la recherche d'algorithmes pour résoudre un problème.

Références bibliographiques 1/2

- Artigue, M. (1997). Le logiciel Dérive comme révélateur de phénomènes didactiques liés à l'utilisation d'environnements informatiques pour l'apprentissage. *Educational Studies in Mathetatics*, *33*(2), 133-169.
- Artigue, M. (2005). L'intelligence du calcul. Dans *Actes de l'Université d'été de Saint-Flour, France : le calcul sous toutes ses formes.*
- Artigue, M. et Haspekian, M. (2007). L'intégration de technologies professionnelles à l'enseignement dans une perspective instrumentale : le cas des tableurs. In Baron, Guin et Trouche (Eds), *Environnements informatisés et ressources numériques pour l'apprentissage.* p. 37-63. Paris : Hermès.
- Balacheff, N. (1994). Didactique et intelligence artificielle. *Recherches en didactique des mathématiques*, *14*(1), 9-42.
- Bosch, M. et Chevallard, Y. (1999). La sensibilité de l'activité mathématique aux ostensifs : objet d'étude et problématique. *Recherches en didactique des mathématiques.* *19*(1), 77-123. http://yves.chevallard.free.fr/spip/spip/IMG/pdf/Sensibilite_aux_ostensifs.pdf
- Briant, N. (2013). *Étude didactique de la reprise de l'algèbre par l'introduction de l'algorithmique au niveau de la classe de seconde du lycée français.* (Thèse de doctorat, université Montpellier 2).
- Capponi B. et Laborde C. (1995). *Cabri-classe, apprendre la géométrie avec un logiciel.* Archimède : Grenoble.
- Chevallard Y. (1985). *La transposition didactique*. Grenoble : La pensée sauvage.
- Chevallard, Y. (1989). Le passage de l'arithmétique à l'algèbre dans l'enseignement des mathématiques au collège - Deuxième partie. Perspectives curriculaires : la notion de modélisation. *Petit x* n°19, 43-75.
- Chevallard, Y. (1999). L'analyse des pratiques enseignantes en théorie anthropologique du didactique. *Recherches en Didactique des Mathématiques*, *19*(2), 221-266.
- Duval, R. (1993). Registres de représentation sémiotique et fonctionnement cognitif de la pensée. *Annales de didactique et de sciences cognitives.* *5*. 37-65. IREM de Strasbourg
- Duval, R. (1995). *Sémiosis et pensée humaine : Registres sémiotiques et apprentissages intellectuels.* Berne : Peter Lang.
- Douady, R. (1986). Jeux de cadres et Dialectique outil-objet. *Recherches en didactique des mathématiques*, *7*(2), 5-31.

Références bibliographiques 2/2

- Hart, E. W. (1998). Algorithmic Problem Solving in Discrete Mathematics. In L. J. Morrow & M. J. Kenney (Eds.), The teaching and learning of Algorithm in school mathematics, 1998 NCTM Yearbook (p. 251-267). Reston, VA : National Council of Teachers of Mathematics.
- Haspekian, M. (2005). *Intégration d'outils informatiques dans l'enseignement des mathématiques. Étude du cas des tableurs.* (Thèse de doctorat, université Paris 7).
- Laborde, C. (2003). *The design of curriculum with technology : Lessons from projects based on dynamic geometry environments.* Reaction to A. Cuoco & P. Goldenberg's presentation "CAS and curriculum : Real improvement or déjà vu all over again ?" CAME Symposium, Reims.
- MEN (2008). Ministère de l'éducation nationale. Programmes du collège. Programmes de l'enseignement de mathématiques. BO spécial n° 6 du 28 août 2008. Paris.
- MEN (2009) MINISTÈRE DE L'ÉDUCATION NATIONALE. PROGRAMME DE MATHÉMATIQUES DE LA CLASSE DE SECONDE GÉNÉRALE ET TECHNOLOGIQUE. BULLETIN OFFICIEL N° 30 DU 23 JUILLET 2009.
- Modeste, S. (2012). *Enseigner l'algorithme pour quoi ? Quelles nouvelles questions pour les mathématiques ? Quels apports pour l'apprentissage de la preuve ?* (Thèse de doctorat, Université de Grenoble).
- Porte, C. et Revet, M. (2015*). Comment amener l'algorithme de dichotomie pour la résolution d'équation en classe de seconde générale ?* (Mémoire de Master 2 MEEF, ESPE de Montpellier)
- Rabardel, P. (1995). *Les hommes et les technologies. Approche cognitive des instruments contemporains*. Paris : Armand Colin.
- Radford, L. (2006). Algebraic Thinking and the Generalization of Patterns: A Semiotic Perspective. In S. Alatorre, J. L. Cortina, M. Sáiz, A. Méndez (Eds.), *Proceedings of the 28th Conference of the International Group for the Psychology of Mathematics Education, North American Chapter*, Mérida: Universidad Pedagógica Nacional, November 9 – 12, Vol. 1, pp. 2-21.
- Radford, L. (2008). Iconicity and Contraction: A Semiotic Investigation of Forms of Algebraic Generalizations of Patterns In Different Contexts. *ZDM – The International Journal on Mathematics Education*. DOI 10.1007/s11858-007-0061-0.