Algorithmique avec Algobox

Fiche 2

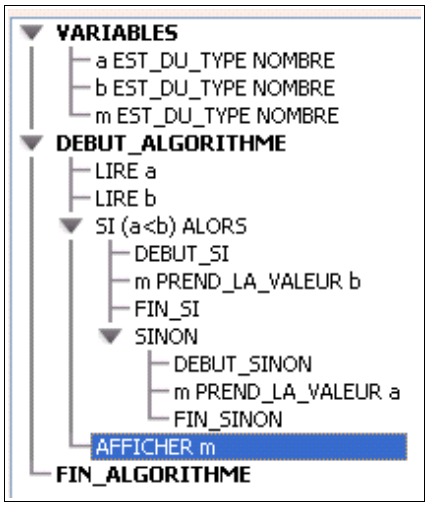
Cette fiche est la suite directe de la première.

1. Instructions conditionnelles :

1.1. Reprise de la fiche 1 :

Lecture d'un algorithme :

ORDINATEUR INTERDIT : Après d'éventuels essais papier crayon, indiquer ce que fait cet algorithme.



Après avoir lu les données a et b, cet algorithme affiche m qui représente : .....

1.2. Équipe sportive :

Pour organiser des rencontres sportives, un moniteur doit connaître l'âge des enfants, puis constituer des équipes homogènes. Parmi les moins de 16 ans et les plus de 6 ans, les catégories sont : "Poussin" de 6 à 7 ans, "Pupille" de 8 à 9, "Minime" de 10 à 11 et "Cadet" après 12 ans.

L'activité consiste à créer un algorithme qui classe l'enfant dans la catégorie de son âge.

Suggestion : commencer par :

SI âge > 12
ALORS Afficher "....."
SINON .....
FINSI

Remarque : il faut bien comprendre que, dans tout algorithme, les instructions sont examinées dans l'ordre chronologique où elles sont rencontrées. La condition "ALORS" étant remplie, la condition "SINON" n'est même pas examinée.

Suggestion : Faire un rapide organigramme avec les instructions Si, Alors, Sinon, Finsi décalées.

1.3. Moyennes en mathématiques :

Exercice : Demander 5 notes, calculer la moyenne et attribuer la mention correspondante :

- Si moyenne $\geqslant$ 16, mention "Très bien"
- Si moyenne $\geqslant$ 14, mention "Bien"
- Si moyenne $\geqslant$ 12, mention "Assez bien"
- Si moyenne $\geqslant$ 10, mention "Passable"
- Si moyenne $\geqslant$ 8, "Admis oral du deuxième groupe"
- Sinon "Recalé"

1.4. Coefficient directeur :

Connaissant les coordonnées de deux points A et B **distincts** du plan, on veut créer un programme qui donne le coefficient directeur de la droite (AB). On le nommera COEF1.

Rappels mathématiques : **ORDINATEUR INTERDIT** : ***activité papier – crayon*** : A (-1;3) et B (3;1). Calculer le coefficient directeur de la droite (AB).

Réponse :

Même question pour la droite (CD) avec C(2;-3) et D(2;5)
Réponse :

S'il y a deux types de réponses possible, c'est qu'il y a un choix et donc une instruction conditionnelle :
SI ALORS SINON

Quelles sont les données ? (*instructions LIRE*) :

Sur lesquelles doit se porter le test ?

Écrire l'algorithme qui, en fonction des coordonnées des points A et B, donne le coefficient directeur de la droite (AB)
Lancer quatre fois cet algorithme pour remplir la deuxième ligne du tableau du § 1.5. page suivante.

1.5. Équation réduite d'une droite du plan :

Rappels mathématiques : ORDINATEUR INTERDIT : activité papier – crayon : A (-1;3) et B (3;1). Le coefficient directeur de la droite (AB) a été calculé au paragraphe précédent.
Calculer l'équation réduite de la droite (AB) :

Compléter l'algorithme précédent pour obtenir l'équation réduite de la droite (AB).
Lancer quatre fois cet algorithme pour remplir la troisième ligne du tableau page suivante.

Points	A(-2;1) et B(3;4)	C(0;2) et D(3;3)	E(4;7) et F(4;-5)	G(125;732) et H(-458;2009)
Coefficients directeurs				
Équation réduite de la droite				

1.6. Exercices d'évaluation :

1.6.1. Chaussée mouillée ou pas ?

Reprendre l'algorithme sur la distance de sécurité (fiche 1) : un véhicule doit respecter une distance minimale avec le véhicule qui le précède, afin d'avoir le temps de freiner avant une collision. Ce temps correspond à celui de la perception puis de la réaction du conducteur, ainsi que des possibilités de freinage du véhicule. Ce temps est fonction de la vitesse du véhicule.

Des études statistiques ont montré que cette distance peut être calculée par la formule : D = 8 + 0,2 v + 0,003 v^2 , où v est en kmh^{-1} et D en mètres. Les mêmes études ont montré que cette distance doit être majorée de 40 % si la chaussée est mouillée. Créer alors un nouvel algorithme qui nous donne cette distance en fonction de la vitesse suivant l'état de la chaussée, avec ou sans eau. Compléter alors le tableau suivant :

Vitesse	50	90	110,5	130
Distance de sécurité sur chaussée mouillée				

1.6.2. Contrat simple d'assurance :

Une compagnie d'assurance automobile propose à ses clients trois familles de tarifs identifiables par une couleur, du moins au plus onéreux : tarifs bleu, orange et rouge.

Le tarif dépend de la situation du conducteur :

- Un conducteur de moins de 25 ans se voit attribuer le tarif rouge s'il a été responsable d'un accident, orange sinon.
- Un conducteur de plus de 25 ans bénéficie du tarif bleu s'il n'est à l'origine d'aucun accident, du tarif orange sinon.

ORDINATEUR INTERDIT : Écrire l'algorithme (donc uniquement papier crayon) permettant de saisir les données nécessaires (sans contrôle de saisie) et de traiter ce problème. Un organigramme peut faire gagner beaucoup de temps !

Appeler le professeur pour qu'il contrôle votre algorithme.

1.6.3. Contrat d'assurance (pour les experts) :

Une compagnie d'assurance automobile propose à ses clients quatre familles de tarifs identifiables par une couleur, du moins au plus onéreux : tarifs bleu, vert, orange et rouge.

Le tarif dépend de la situation du conducteur :

- Un conducteur de moins de 25 ans et titulaire du permis depuis moins de deux ans se voit attribuer le tarif rouge, si toutefois il n'a jamais été responsable d'un accident. Sinon, la compagnie refuse de l'assurer.
- Un conducteur de moins de 25 ans et titulaire du permis depuis plus de deux ans, ou de plus de 25 ans mais titulaire du permis depuis moins de deux ans a le droit au tarif orange s'il n'a jamais provoqué d'accident, au tarif rouge pour un accident, sinon il est refusé.
- Un conducteur de plus de 25 ans et titulaire du permis depuis plus de deux ans bénéficie du tarif vert s'il n'est à l'origine d'aucun accident, du tarif orange pour un accident, du tarif rouge pour deux accidents, et refusé au-delà.

De plus, pour encourager la fidélité des clients acceptés, la compagnie propose un contrat de la couleur immédiatement la plus avantageuse s'il est entré dans la maison depuis plus d'un an.

Écrire l'algorithme (donc uniquement papier) permettant de saisir les données nécessaires (sans contrôle de saisie) et de traiter ce problème. Avant de se lancer à corps perdu dans cet exercice, on pourra réfléchir un peu et s'apercevoir qu'il est plus simple qu'il n'en a l'air (cela s'appelle faire une analyse préalable du problème !)

1.6.4. Un tour de magie :

Un magicien demande à un spectateur de penser à un nombre et de l'écrire sur une ardoise. Il l'invite à cacher cette ardoise le temps du numéro. Il lui demande d'ajouter 3 puis de multiplier cette somme par le nombre auquel il a pensé au départ. Il insiste : ne pas oublier ce résultat, puis calculer le carré du nombre de départ. Enfin il demande de soustraire ce résultat du précédent. Au spectateur un peu hagard après tous ces calculs, le magicien demande de dire à haute voix le résultat final. Instantanément le magicien annonce le nombre pensé déclenchant une salve d'applaudissements alors que le spectateur brandit son ardoise en preuve.

Concevoir un algorithme avec **AlgoBox** qui calcule le nombre d'arrivée en connaissant le nombre de départ.

Expérimenter avec plusieurs nombres de départ.

Comment fait le magicien?

1.6.5. Un autre tour de magie (pour les « experts ») ?

Un magicien demande à un spectateur d'effectuer les calculs suivants, **sans les lui dire**.
Prendre sa pointure (en nombre entier), la multiplier par deux.
Au résultat ajouter 5 et multiplier le nouveau résultat par 50.
Au dernier nombre trouvé, ajouter son âge (en nombre entier).
Le magicien demande de dire à haute voix le résultat final. Instantanément le magicien annonce la pointure et l'âge du spectateur.

Faire une étude mathématique de la situation.
Puis concevoir un algorithme qui permet au magicien d'annoncer la pointure et l'âge.

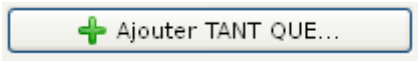
2. Boucles :

2.1. Introduction aux boucles :

Il est possible de demander à l'ordinateur (ou à la calculatrice) de répéter une même tâche autant de fois que l'on veut.
On utilise alors **une boucle**.
Avec Algobox, la syntaxe pour une boucle peut prendre deux formes :

Lorsque l'on connait le nombre de départ et celui d'arrivée (fin de la boucle).

Exemple :
```
Pour i de 1 jusqu'à 20
    | traitement
FinPour
```



Lorsque le nombre d'arrivée (fin de la boucle) est testé par le programme.

Exemple :
```
Tant que N <= 20
    | traitement
FinTantQue
```

2.2. Boucle Pour...De....A.... :

Exemple : Les parents de Léa versent 100 € sur un livret à sa naissance, puis versent 20 € chaque mois sur ce livret. On veut écrire un algorithme donnant la somme S sur ce livret au bout d'un certain nombre N de mois.

Déclaration des variables	*N, i, S*	*N* nombre de mois *i* variable pour incrémenter *S* somme de départ
Entrées d'initialisation	*Lire N* *Lire S*	Nombre de boucles Initialisation de la somme
Traitement	*Pour i allant de 1 jusqu'à N* *S prend la valeur S + 20* *FinPour*	Début de la boucle : i prend la valeur 1 On remplace S par S + 20 i prend la valeur 2 On remplace S (valeur précédente) par S + 20 ….. Dernière valeur de i : N
Sortie	*Afficher S*	Affichage de la somme obtenue au bout de N mois

Saisir ce programme sur l'ordinateur et l'exécuter afin de compléter le tableau suivant :

Somme de départ	100			125			150		
Nombre de mois	6	12	18	6	12	18	6	12	18
Somme obtenue									

2.3. Savoir lire un algorithme :

On considère l'algorithme suivant :

Demander N entier naturel Donner à A la valeur 1 Donner à B la valeur 1 Pour *i* variant de 1 à N Donner à A la valeur 4A Donner à B la valeur B + 4 FinPour Afficher A et B	Résultats obtenus : Si N = 2, alors A = et B = Si N = 4, alors A = et B =

2.4. Exercices :

2.4.1.

Écrire un algorithme qui demande un nombre de départ et qui affiche ensuite les dix nombres suivants.

Par exemple, si l'utilisateur entre le nombre 17, le programme affichera les nombres de 18 à 27.

Votre algorithme :

2.4.2.

Écrire un algorithme qui demande un nombre et qui calcule la somme des entiers consécutifs de 1 jusqu'à ce nombre. Par exemple, si on entre 5, l'algorithme doit calculer 1+2+3+4+5

Votre algorithme	Résultats obtenus :
	Somme des 10 premiers nombres entiers : Somme des 15 premiers nombres entiers :

2.4.3.

Écrire un algorithme qui demande un nombre entier N supérieur ou égal à 1 et qui calcule la somme des N premiers nombres impairs et qui affiche cette somme.

Votre algorithme	Résultats obtenus :
	Si N = 5, somme obtenue : Si N = 12, somme obtenue :

2.4.4. Écrire un algorithme qui demande un nombre de départ et qui ensuite écrit la table de multiplication de ce nombre (de 1 à 9) présentée comme suit (cas où l'utilisateur entre le nombre 7) :

Votre algorithme

$7 \times 1 = 7$
$7 \times 2 = 14$

2.5. Évaluations :

2.5.1. Fonction : On considère la fonction f définie par $f(x) = x^2 - 3 \times x - 7$

Écrire l'algorithme qui calcule les images des nombres donnés dans le tableau suivant :
Les résultats seront recopiés à 10^{-3} près.

x	5	$\sqrt{(5)}$	π	$\frac{1+\sqrt{(7)}}{1-3\times\pi}$
f (x)				

2.5.2. Points alignés :

A, B et C étant trois points du plan, définis par leurs coordonnées, on veut tester s'ils sont alignés.

Écrire l'algorithme qui a partir des coordonnées des trois points répond par « Oui » ou « Non » à la question : « A, B et C sont-ils alignés ? »

Tester l'algorithme avec les données suivantes :

Coordonnées des points	A $\begin{pmatrix}-2\\3\end{pmatrix}$ B $\begin{pmatrix}4\\1\end{pmatrix}$ C $\begin{pmatrix}1,5\\1,8\end{pmatrix}$	A $\begin{pmatrix}-2\\-2\end{pmatrix}$ B $\begin{pmatrix}1\\-1\end{pmatrix}$ C $\begin{pmatrix}31\\9\end{pmatrix}$	A $\begin{pmatrix}1\\-4\end{pmatrix}$ B $\begin{pmatrix}100\\62\end{pmatrix}$ C $\begin{pmatrix}-2,6\\-6,4\end{pmatrix}$
Points alignés ?			

2.5.3. Triangle isocèle :

A, B et C étant trois points non alignés du plan, définis par leurs coordonnées, on veut tester si le triangle est isocèle en A.

Écrire l'algorithme qui, à partir des coordonnées des trois points ,répond par « Oui » ou « Non » à la question : « Le triangle est-il isocèle en A ? » ? »

Tester l'algorithme avec les données suivantes :

Coordonnées des points	A $\begin{pmatrix}1\\2\end{pmatrix}$ B $\begin{pmatrix}-2\\3\end{pmatrix}$ C $\begin{pmatrix}4\\5\end{pmatrix}$	A $\begin{pmatrix}-2\\0\end{pmatrix}$ B $\begin{pmatrix}2\\0\end{pmatrix}$ C $\begin{pmatrix}0\\2*\sqrt{(3)}\end{pmatrix}$	A $\begin{pmatrix}-2\\0\end{pmatrix}$ B $\begin{pmatrix}2\\0\end{pmatrix}$ C $\begin{pmatrix}0\\3,46\end{pmatrix}$
Triangle isocèle en A ?			

2.5.4. Boule de cristal :

Cet algorithme est destiné à prédire l'avenir, et il doit être infaillible !

Il lira au clavier l'heure et les minutes, et il affichera l'heure qu'il sera une minute plus tard. Par exemple, si l'utilisateur tape 21 puis 32, l'algorithme doit répondre : « Dans une minute, il sera 21 heure(s) 33 ".

N.B. On suppose que l'utilisateur entre une heure valide. Pas besoin donc de la vérifier.

2.5.5. Élections (devoir maison) :

Les élections législatives, en Guignolerie Septentrionale, obéissent à la règle suivante :

- Lorsque l'un des candidats obtient plus de 50 % des suffrages, il est élu dès le premier tour.
- En cas de deuxième tour, peuvent participer uniquement les candidats ayant obtenu au moins 12,5 % des voix au premier tour.

Vous devez écrire un algorithme qui permette la saisie des scores de quatre candidats au premier tour. Cet algorithme traitera ensuite le candidat numéro 1 (et uniquement lui) : il dira s'il est élu, battu, s'il se trouve en ballotage favorable (il participe au second tour en étant arrivé en tête du premier tour) ou défavorable (il participe au second tour sans avoir été en tête au premier tour).

Scores des candidats en %	A 60 B 15 C 13 D 12	A 38 B 25 C 32 D 4	5A 45 B 25 C 12 D 18	A 8 B 25 C 32 D 35
Résultat du candidat A				

3. Quelques possibilités supplémentaires du langage d'Algobox:

3.1. Conjonctions « OU » ; « ET » :

Il est possible de combiner plusieurs conditions avec ET et OU :
La condition à écrire pour vérifier que x est strictement compris entre 1 et 5 est :
`x>1 ET x<5`
La condition à écrire pour vérifier que x est égal à 3 OU à 5 est : `x==3 OU x==5`

ATTENTION : il faut un espace avant et après la conjonction

3.2. Exemple avec OU :

L'heure de l'exercice 2.5.4. : on veut tester si l'heure saisie est bien conforme : heure entre 0 inclus et 24 exclu et pour les minutes entre 0 inclus et 60 exclu.

Voici un exemple :

```
TANT_QUE ((minute<0) OU (minute>=60)) FAIRE
    DEBUT_TANT_QUE
    AFFICHER "Mauvaise saisie des minutes"
    LIRE minute
    FIN_TANT_QUE
```

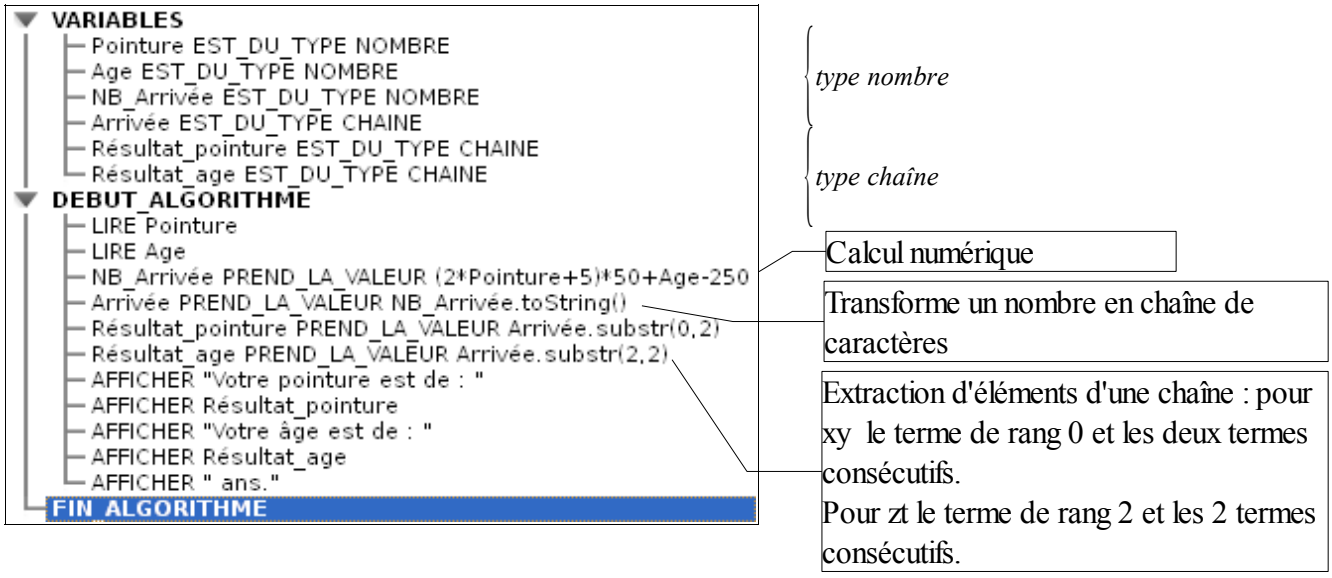
3.3. Variable de type chaîne :

On peut avoir besoin de manipuler des données non numériques. Nous donnons des exemples où le résultat est de la forme « Oui » ou « Non ». La variable contenant ce résultat n'est pas de type « nombre » mais de type « Chaîne ». La syntaxe à employer est simple, le tutoriel d'Algobox est là aussi très précis :

Opérations avec les chaînes :

- *Le contenu d'une chaîne doit être encadré par des guillemets :*
  Exemple : `a` prend la valeur `"bonjour"` (`a` étant une variable du type `chaine`)
- *Il est possible d'ajouter (concaténer) des chaînes :*
  Exemple : `b` prend la valeur `a+"bonjour"` (`a` et `b` étant des variables du type `CHAINE`)
- *Il est possible d'extraire le contenu d'une chaîne avec l'instruction*
  `chaîne.substr(position_premier_caractère_à_extraire,nombre_de_caractères_à_extraire)`.
  Attention : la premier caractère a pour position 0 (et pas 1)
  Exemple : `b` prend la valeur `a.substr(4,2)` (`b` sera alors formé des 5ème et 6ème caractères de `a` ; `a` et `b` étant des variables du type `CHAINE`)
- *Un nombre peut-être transformé en chaîne avec l'instruction `nombre.toString()`*
  Exemple : `machaine` prend la valeur `nb.toString()` (`nb` étant une variable du type `NOMBRE` et `machaine` étant une variable du type `CHAINE`)

Dans l'activité 1.6.4. « un tour de magie », nous manipulons une chaîne de façon plus élaborée. D'un résultat de calculs (numérique donc) de la forme xyzt, il s'agit d'extraire xy d'un côté et zt de l'autre.



3.4. Le hasard :

Algobox permet aussi de générer des nombres aléatoires.

Les syntaxes sont indiquées dans le § 3.2. page 2 de la fiche 1

Que fait l'algorithme ci-contre ?

```
VARIABLES
    nbface EST_DU_TYPE NOMBRE
    nbpile EST_DU_TYPE NOMBRE
    i EST_DU_TYPE NOMBRE
    lancer EST_DU_TYPE NOMBRE
DEBUT_ALGORITHME
    nbface PREND_LA_VALEUR 0
    nbpile PREND_LA_VALEUR 0
    POUR i ALLANT_DE 1 A 1000
        DEBUT_POUR
        lancer PREND_LA_VALEUR floor(random()*2+1)
        SI (lancer==1) ALORS
            DEBUT_SI
            nbface PREND_LA_VALEUR nbface+1
            FIN_SI
            SINON
                DEBUT_SINON
                nbpile PREND_LA_VALEUR nbpile+1
                FIN_SINON
        FIN_POUR
    AFFICHER "On a obtenu "
    AFFICHER nbface
    AFFICHER " fois face et "
    AFFICHER nbpile
    AFFICHER " fois pile."
FIN_ALGORITHME
```

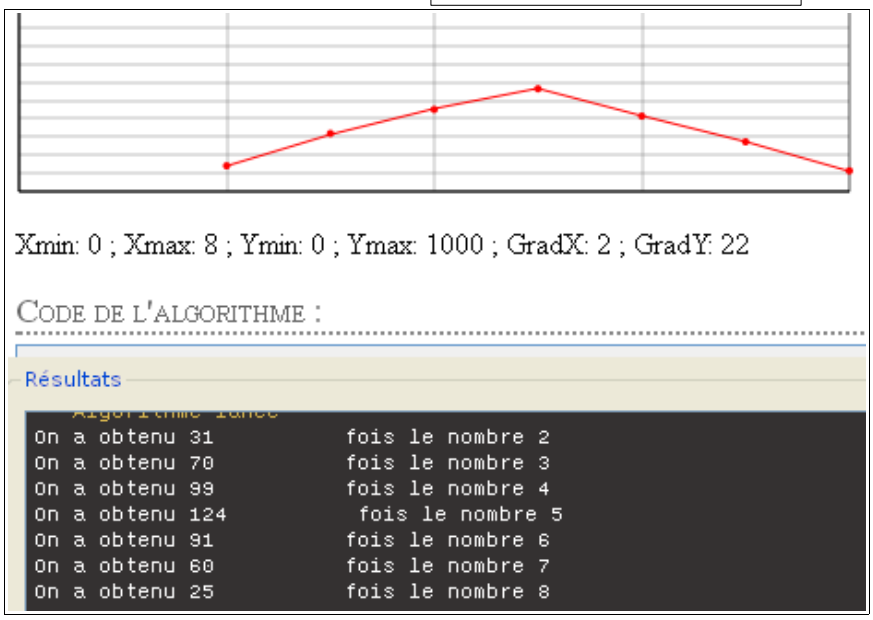
Activité : On lance simultanément deux dés tétraèdriques dont les faces sont numérotées 1, 2, 3 et 4. On s'intéresse à la somme des chiffres situés sur la base des deux tétraèdres.
Quelles sont les valeurs possibles pour cette somme ?

On veut créer un algorithme qui simule N lancers de ces deux dés, N étant un nombre à saisir dans l'algorithme. Pour gagner du temps et uniformiser un peu vos algorithmes, on donne les variables :

```
VARIABLES
    i EST_DU_TYPE NOMBRE
    N EST_DU_TYPE NOMBRE
    Lancer1 EST_DU_TYPE NOMBRE
    Lancer2 EST_DU_TYPE NOMBRE
    Somme EST_DU_TYPE NOMBRE
    NB2 EST_DU_TYPE NOMBRE
    NB3 EST_DU_TYPE NOMBRE
    NB4 EST_DU_TYPE NOMBRE
    NB5 EST_DU_TYPE NOMBRE
    NB6 EST_DU_TYPE NOMBRE
    NB7 EST_DU_TYPE NOMBRE
    NB8 EST_DU_TYPE NOMBRE
```

Suggestion pour ceux qui sont en avance : faire une représentation graphique des résultats obtenus comme celle donnée en exemple avec N = _____

A vous de trouver ce nombre à l'aide de l'image ci-dessous !



Fiche professeur

. Résultats attendus :

Coefficients directeurs
Équation réduite
Pages 2 et 3 § 1.4. et 1.5.

Points	A(-2;1) et B(3;4)	C(0;2) et D(3;3)	E(4;7) et F(4;-5)	G(125;732) et H(-458;2009)
Coefficients directeurs	0,6	0,33333...	Pas de coefficient	-2,19
Équation réduite de la droite	y = 0,6 x + 2,2	y = 0,33.. x +2	x = 4	y = -2,19 x + 1 0005,7993

1.6.
page 3

Vitesse	50	90	110,5	130
Distance de sécurité sur chaussée mouillée	35,7	70,42	93,42	118,58

Page 5
§ 2.2.

Somme de départ	100			125			150		
Nombre de mois	6	12	18	6	12	18	6	12	18
Somme obtenue	220	340	460	245	365	485	270	390	510

Page 5
2.3.

Si N = 2, alors A = 16 et B = 9
Si N = 4, alors A = 256 et B = 17

Somme des entiers consécutifs de 1 à N Page 6 § 2.4..2	N = 10 S = 55
	N =15 S = 120

Somme des N premiers nombres impairs Page 6 § 2.4..3	N = 5 S = 25
	N =10 S = 100

Page 6 2.5. Évaluations :

2.5.1.

x	5	$\sqrt{(5)}$	π	$\frac{1+\sqrt{(7)}}{1-3\times\pi}$
f (x)	3	-8,708 203 9	-6,555 173 6	-5,514 510 2

Page 7

2.5.2.

Coordonnées des points	A $\begin{pmatrix}-2\\3\end{pmatrix}$ B $\begin{pmatrix}4\\1\end{pmatrix}$ C $\begin{pmatrix}1,5\\1,8\end{pmatrix}$	A $\begin{pmatrix}-2\\-2\end{pmatrix}$ B $\begin{pmatrix}1\\-1\end{pmatrix}$ C $\begin{pmatrix}31\\9\end{pmatrix}$	A $\begin{pmatrix}1\\-4\end{pmatrix}$ B $\begin{pmatrix}100\\62\end{pmatrix}$ C $\begin{pmatrix}-2,6\\-6,4\end{pmatrix}$
Points alignés ?	Non	Oui	Oui

2.5.3.

Coordonnées des points	A $\begin{pmatrix}1\\2\end{pmatrix}$ B $\begin{pmatrix}-2\\3\end{pmatrix}$ C $\begin{pmatrix}4\\5\end{pmatrix}$	A $\begin{pmatrix}-2\\0\end{pmatrix}$ B $\begin{pmatrix}2\\0\end{pmatrix}$ C $\begin{pmatrix}0\\2*\sqrt{(3)}\end{pmatrix}$	A $\begin{pmatrix}-2\\0\end{pmatrix}$ B $\begin{pmatrix}2\\0\end{pmatrix}$ C $\begin{pmatrix}0\\3,46\end{pmatrix}$
Triangle isocèle en A ?	Non	Oui ATTENTION : limite du procédé : le triangle reste isocèle avec C $\begin{pmatrix}0\\2*\sqrt{(3)}+10^{-12}\end{pmatrix}$	Non

Remarque : attention à la gestion des nombres ! Le tutoriel d'Algobox est très précis à ce sujet :

Dans AlgoBox, comme avec tous les langages de programmation, la représentation interne des nombres qui ne sont ni des entiers, ni des décimaux simples (du style 0.05) peut engendrer des problèmes de précision et d'arrondis. Une égalité mathématique peut donc se retrouver non vérifiée lors de l'exécution réelle d'un algorithme.

L'utilisateur doit être conscient de ces problèmes de précision dès qu'il manipule des nombres non entiers. *La prudence est notamment de mise lors de l'utilisation d'un bloc* `TANT_QUE` *: une condition ne prenant pas en compte les problèmes de précision en informatique peut engendrer une boucle infinie (qui sera stoppée par le mécanisme de sécurité interne d'AlgoBox). Il convient donc de tenir compte de ces limites de précision lors de la conception de certains algorithmes.*

Page 7 § 2.5.4. Boule de cristal :

L'énoncé posé tel quel est un parfait exemple du fonctionnement à l'implicite, fonctionnement que nous employons tous, de façon plus ou moins consciente. Que signifie exactement le N.B. :

On suppose que l'utilisateur entre une heure valide. Pas besoin donc de la vérifier.

Par *heure* l'élève doit-il comprendre heure et minute, auquel cas il n'a pas à tester la validité des deux saisies (heure et minute) ou bien doit-il comprendre que seule l'heure est considérée comme correctement validée et il doit tester la validité de la saisie des minutes ?

Nous proposons un fichier Algobox pour la deuxième possibilité : avec test de validité des minutes qui inclus le cas où l'élève saisi 59 minutes.

Page 7 § 2.5.5. Élections :

Scores des candidats en %	A 60 B 15 C 13 D 12	A 38 B 25 C 32 D 4	5A 45 B 25 C 12 D 18	A 8 B 25 C 32 D 35
Résultat du candidat A	Élu	Ballotage défavorable	Ballotage favorable	Battu