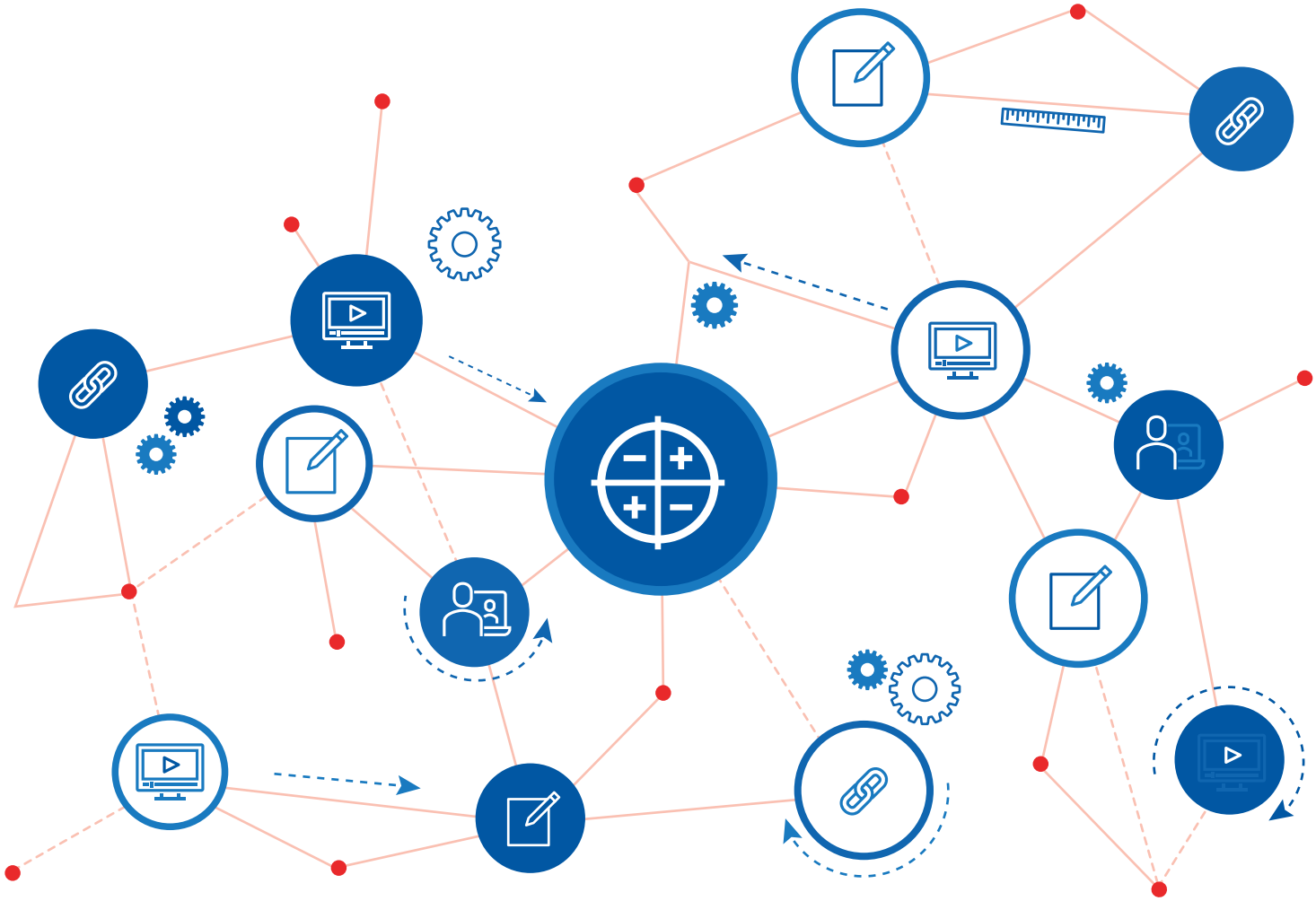
T³ FRANCE

Teachers Teaching with Technology™



Matthieu Carbon et Jean-Baptiste Civet

Mathématiques
Activités avec la calculatrice graphique
Lycée technologique – Classes de 1ère

www.t3france.fr

python™

Activités mathématiques en classe de première technologique

Table des matières

1. Avant-propos. 2
2. Mise à jour système 3
3. Equations et inéquations du 1er degré 4
4. Entrainement aux automatismes 6
5. Coefficient multiplicateur 8
6. Tracer et cadrer une fonction 10
7. Racines d'un trinôme du 2nd degré 12
8. Second degré : maximum et intersection 14
9. Trinôme du second degré : image, racine et signes 16
10. Fonction $x \mapsto ax^2 + b$. Résolution de système 18
11. Equations et inéquations du second degré 20
12. Equation réduite de droite 22
13. Fonction du second degré et Python 24
14. Algorithme de seuil pour une suite 26
15. Taux d'une variation d'une fonction 28
16. Algorithme de balayage pour la recherche d'un extremum 30
17. Suite quelconque 32
18. Suite arithmétique 34
19. Suite géométrique 36
20. Simuler le lancer de 3 dés 38
21. Intérêts composés 40
22. Générer des listes pour le 421 42
23. Simulation de la somme de 2 dés 44
24. Représentation de la somme de 2 dés 46
25. Comparaison de deux populations 48
26. Polynôme de degré 3 50
27. Nombre dérivé et tangente 52
28. Combien de 6 ? 54
29. Répétition d'expériences de Bernoulli 56
30. Algorithme de calcul de moyenne et d'écart type 58
31. STI2D/STL - Fonction trigonométrique 60
32. STI2D/STL - Produit scalaire 62

AVANT PROPOS

Ce livret est destiné aux enseignants et à leurs élèves suivant un cursus en voie technologique.

Chaque fiche a été rédigée selon le modèle suivant : un exercice type est proposé. Il peut faire partie des "classiques" présents dans tous les manuels ou bien encore s'inspirer de l'ancienne banque des E3C de la voie technologique. L'énoncé peut ainsi être proposé, en l'état, par l'enseignant à ses élèves. La deuxième partie de la fiche consiste en la résolution, à l'aide de la TI-82 Advanced Edition Python ou de la TI-83 Premium CE Edition Python (ou Python Adapter) de l'exercice. Plusieurs méthodes de résolutions de l'exercice peuvent être proposées. Par exemple à l'aide des fonctionnalités de représentation graphique et de calculs ou bien à l'aide de courts scripts Python.

Il a été décidé d'intégrer une progressivité dans les fiches proposées, tenant systématiquement sur deux pages. Cependant certains exercices peuvent parfois se prolonger sur deux fiches. C'est un choix volontaire pour permettre dans une première activité de manipuler, d'explorer la situation proposée. Puis dans une deuxième activité de mener l'exploitation mathématique pour la résolution de la situation proposée.

De même, par exemple, concernant l'enseignement de l'algorithmique, il a été décidé de l'intégrer progressivement dans les activités en commençant par proposer deux méthodes de résolutions pour s'approprier l'environnement de développement et comprendre l'enjeu de l'algorithmie (Analyse et Suite notamment) jusqu'à finir notamment dans le cas de la fluctuation d'échantillonnage par des activités uniquement centrées sur des manipulations et résolutions en Python. Certaines activités sont donc liées chronologiquement pour faire l'acquisition de certaines connaissances pratiques. On commence par le lancer d'un ou plusieurs dés avant de passer dans une autre activité à la répétition des lancers puis à l'observation des fluctuations jusqu'à la répétition d'expériences de Bernoulli.

L'ensemble des fiches proposées couvrent les thèmes de l'Analyse, des Statistiques et Probabilités, de l'Algorithmie ainsi que certaines parties des séries STI2D/STL.

Mises à jour système et compléments logiciels

Pour réaliser les fiches proposées dans cet ouvrage, il est nécessaire de disposer d'une calculatrice Texas Instruments, permettant également de programmer en langage Python, cela bien que ce ne soit pas l'objectif premier de ce livret.

Cela concerne les modèles suivants :

- TI-82 Advanced Édition Python
- TI-83 Premium CE + Python Adapter
- TI-83 Premium CE Édition Python

Pour les deux derniers modèles, il convient que le système d'exploitation de la calculatrice soit à jour.

L'installation des mises à jour calculatrice est très facile : il suffit de télécharger un fichier à l'adresse indiquée ci-dessous puis de copier le fichier téléchargé dans la machine à l'aide du logiciel ad hoc (à savoir le logiciel TI-Connect CE prévu à cet effet).

Mise à jour des calculatrices TI-83 Premium CE	

Equations & inéquations du 1er degré

Enoncé

Un gaz est parfait lorsque ses molécules n'interagissent pas entre elles, en dehors des chocs survenant lorsqu'elles se rencontrent. La loi des gaz parfaits s'écrit sous la forme :

$$PV = nRT$$

- P est la pression du gaz, exprimé en Pascal (Pa) ;
- V est le volume occupé par le gaz, exprimé en mètres cubes (m^3) ;
- n est la quantité de matière, exprimée en moles (mol) ;
- R est la constante universelle des gaz parfaits : $R \approx 8{,}31\ J.K^{-1}.mol^{-1}$;
- T est la température, exprimée en Kelvin (K).

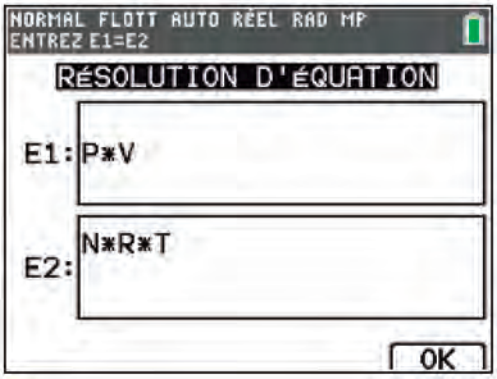
1. A) On donne $P = 1{,}1 \times 10^5\ Pa$; $V = 0{,}002\ m^3$ et $T = 323\ K$ $(50°C)$. Déterminer n.
   B) On donne $P = 2 \times 10^5\ Pa$; $V = 0{,}002\ m^3$ et $n = 0{,}16\ mol$. Déterminer T.
2. Dans cette question, $V = 0{,}017\ m^3$ et $n = 0{,}84\ mol$. Pour quelle(s) valeur(s) de T a-t-on P qui dépasse $1{,}2 \times 10^5\ Pa$?

1. Résolution d'équations

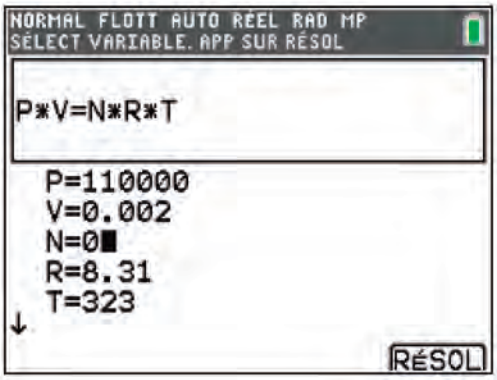
Afin de déterminer les réponses à ces deux équations du premier degré à une inconnue, nous allons utiliser le solveur d'équations de la calculatrice.

- A l'aide de la touche [résol], on exécute la commande **1:Résoudre…**

- Dans le 1er bloc **E1**, nous entrons l'expression $P * V$; dans le bloc **E2**, nous saisissons l'expression $N * R * T$. Puis on valide par **Ok**, à l'aide la touche [graphe].



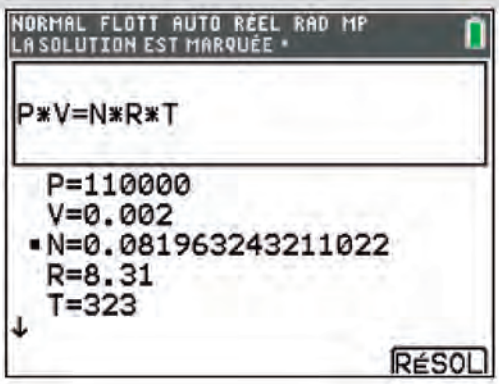
- Le nouvel écran nous rappelle l'égalité que nous avons entrée précédemment, puis nous propose d'entrer les valeurs des différentes variables.



- On commence donc par entrer $P = 1{,}1 \times 10^5$, $V = 0{,}002$, $R = 8{,}31$ et $T = 323$. On place alors le curseur devant la variable recherchée, ici n. Enfin, on lance la résolution à l'aide de **Résol** (touche [graphe]).

Equations & inéquations du 1er degré

- On obtient ainsi $n \approx 0{,}082\ mol$. Notez qu'un carré noir apparaît devant la variable que nous avons recherchée.



- On procède de la même façon pour la seconde partie de cette question et on obtient $T \approx 300{,}84\ K\ \approx 27{,}84\ °C$

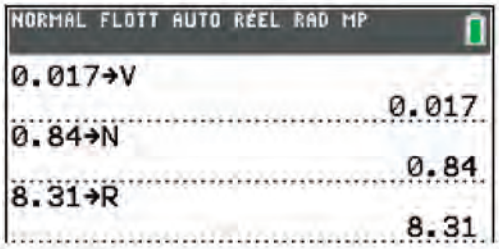
Nous pouvons à présent passer à la résolution proprement dite :

A) $n = \frac{PV}{RT} = \frac{1{,}1\times10^5\times0{,}002}{8{,}31\times323} \approx 0{,}082\ mol$ B) $T = \frac{PV}{nR} = \frac{2\times10^5\times0{,}002}{8{,}31\times0{,}16} \approx 300{,}84\ K$

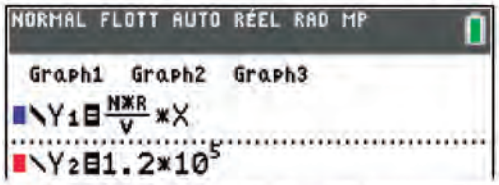
2. Résolution de l'inéquation

$P = \frac{nR}{V} \times T$ ce qui va nous permettre de représenter graphiquement la pression en fonction de la température, à l'aide d'une fonction linéaire.

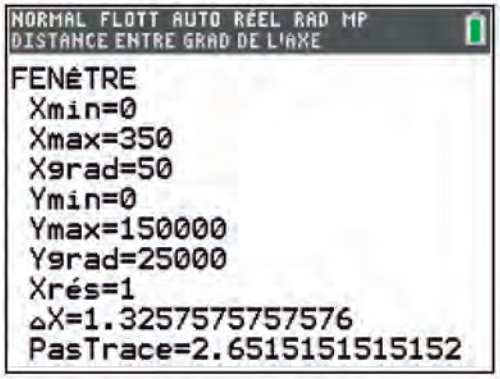
- On stocke les valeurs de V, n et R à l'aide la touche [sto→].



- Dans le menu [f(x)], on saisit notre fonction linéaire dans **Y1**. Attention, c'est la variable X (touche [X,T,θ,n]) qui va jouer le rôle de la température. On entre également la fonction constante égale à $1{,}2 \times 10^5$ dans **Y2**.

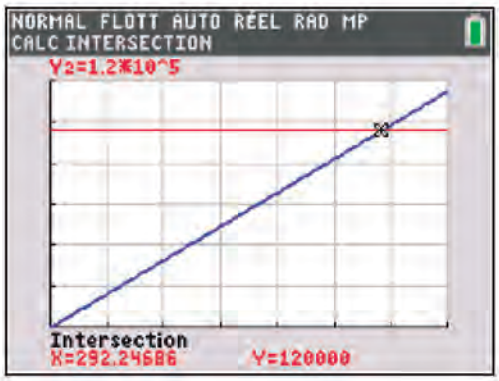


- Après avoir consulté le tableau de valeurs de nos fonctions (et à l'aide de la fiche TRACER-CADRER-FCT), on configure notre fenêtre graphique, dans le menu [fenêtre].



- A l'aide de la touche [graphe], on observe nos fonctions. Reste à déterminer l'abscisse du point d'intersection de ces dernières.

- Dans le menu [2nde] [trace] (calculs 14), on sélectionne la commande **5:intersection**. On appuie successivement 3 fois sur [entrer] : une 1ère fois pour sélectionner **Y1**, une 2nde pour **Y2** et une dernière pour la valeur initiale. On obtient $T \approx 292{,}25\ K$.



Résolvons l'inéquation :

$$P > 1{,}2 \times 10^5 \Leftrightarrow \frac{nR}{V} \times T > 1{,}2 \times 10^5 \Leftrightarrow T > \frac{1{,}2 \times 10^5 \times 0{,}017}{0{,}84 \times 8{,}31} \approx 292{,}25\ K$$

La pression du gaz dépassera ainsi $1{,}2 \times 10^5\ Pa$ lorsque la température dépassera environ $292{,}25\ K$, soit environ $19{,}25\ °C$.

Entrainement aux automatismes

Enoncé

La calculatrice va être utilisée pour vérifier son travail dans le cadre d'une session d'entrainement sur les compétences liées aux automatismes de calculs.

1. Calculer la fraction irréductible égale à $\frac{18}{25} \times \frac{5}{3}$

   Calculer la fraction irréductible égale à $\frac{6\times10^2}{3\times10^4}$

   Calculer la fraction irréductible égale à $\left(\frac{2}{3}\right)^2 \times \left(\frac{1}{3}\right)^2$

   Écrire sous la forme 3^n, avec n entier naturel, le nombre $(3^5 \times 3^{-3})^4$

2. La distance entre le Soleil et Pluton est de $5906{,}38 \times 10^6$ km. Convertir en mètres et donner la réponse sous la forme de l'écriture scientifique.

3. On donne la relation : $\frac{1}{R} = \frac{1}{R_1} + \frac{1}{R_2}$. Que vaut R si $R_1 = 5\ \Omega$ et $R_2 = 20\ \Omega$?

1. Calculs fractionnaires et puissances

La calculatrice est capable de gérer les résultats sous forme fractionnaire.

On pourrait dans un premier temps être tenté d'utiliser la touche [÷] pour saisir l'opération. On obtient alors 1,2. Vous remarquerez que le curseur a changé de forme. L'appui sur la touche [◆▶] permet alors de passer de l'écriture décimale du résultat à une écriture fractionnaire (et vice versa en appuyant de nouveau sur cette touche).

Cependant, il est possible de forcer l'affichage fractionnaire à l'aide de la touche [a b/c], ce qui va permettre de saisir directement les fractions dans nos calculs et d'obtenir par défaut un résultat fractionnaire.

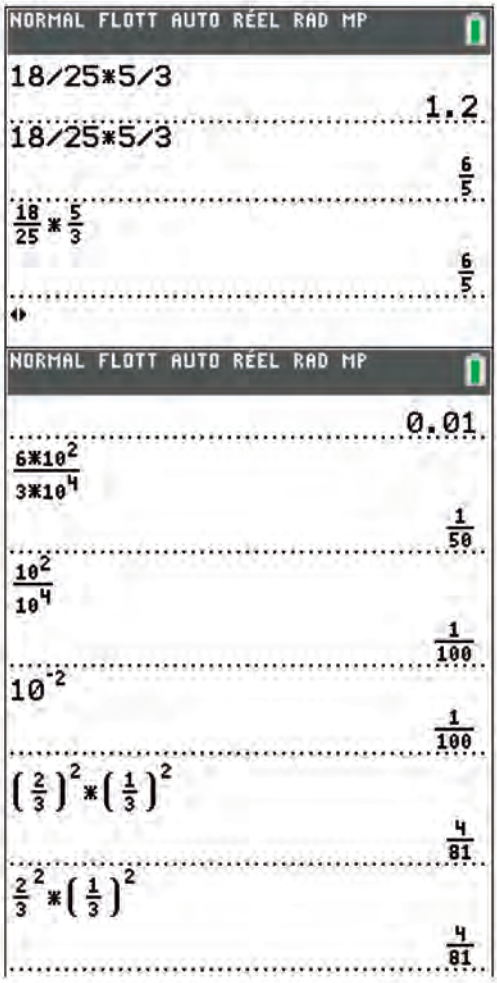
Pour saisir des écritures sous forme de puissance, on utilise [x²] lorsque l'exposant vaut 2. Sinon il faut utiliser la touche [^]. Le curseur indiquant la position de saisie se surélève et on peut taper l'exposant désiré. On sort de la saisie de l'exposant à l'aide de [›].

On n'oublie pas d'utiliser la touche [(-)] lors de la saisie d'exposants négatifs.

On vérifie ainsi que $\frac{10^2}{10^4} = 10^{-2}$

Pour élever les fractions au carré, il est préférable de les mettre entre parenthèses même si elles ne sont pas nécessaire car, par défaut, l'exposant placé à l'extérieur de la fraction porte bien sur le numérateur et le dénominateur. Cela reste un point de vigilance malgré tout et il vaut mieux prendre l'habitude d'utiliser les parenthèses lors de la saisie.

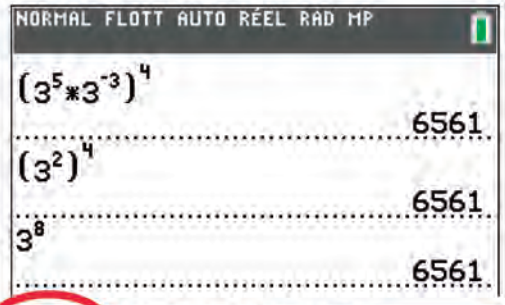
$\frac{18}{25} \times \frac{5}{3} = \frac{6}{5}$ puis $\frac{6\times10^2}{3\times10^4} = \frac{1}{50}$ et enfin $\left(\frac{2}{3}\right)^2 \times \left(\frac{1}{3}\right)^2 = \frac{4}{81}$



Entrainement aux automatismes

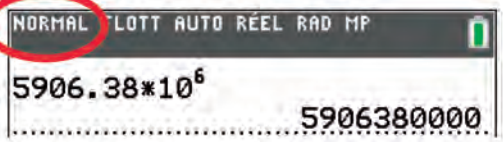
La calculatrice n'étant pas une calculatrice formelle, elle n'affiche pas forcément toutes les écritures mathématiques sous la forme attendue.

Cependant un travail de comparaison permet de vérifier chacune des étapes de nos calculs.



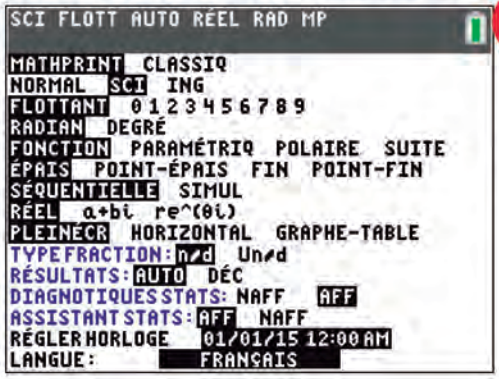
2. Ecriture scientifique

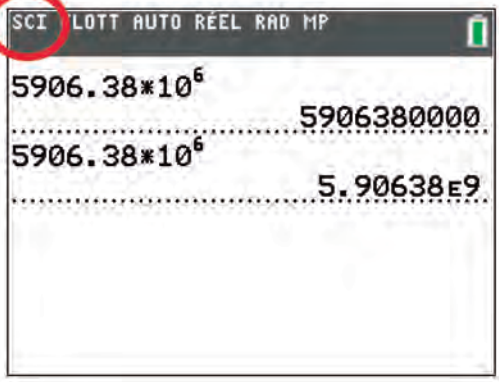
Par défaut, la calculatrice est en mode **NORMAL** mais il est possible de forcer l'affichage sous la forme scientifique. Pour cela on appuie sur [mode] puis on sélectionne le mode **SCI.** On fera de même pour revenir en mode **NORMAL.**

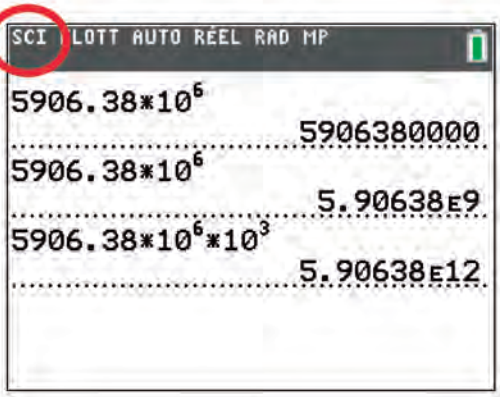


Ainsi `5.90638E9` km correspond à $5{,}90638 \times 10^9$ km, ce qui est bien la forme attendue.

La distance entre le Soleil et Pluton est donc de $5{,}90638 \times 10^{12}\ m$







3. Calculs par substitution

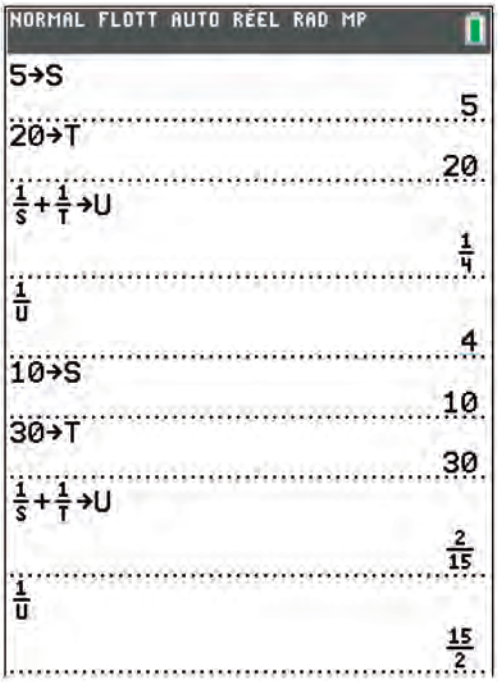
Il est possible avec la calculatrice de travailler par substitution. Le nom des variables ne doivent comporter qu'un caractère et sans indice.

Dans notre cas, il sera nécessaire de faire une correspondance entre R_1 et R_2 proposés dans l'énoncé et, par exemple **S** et **T** dans la calculatrice. La variable **U** nous servira à stocker l'inverse de R.

On affecte une valeur à une variable à l'aide de [sto→] et on saisit des « lettres » à l'aide de la combinaison [alpha] + la touche de la lettre de son choix (pour la lette **S** : [ln]).

Ainsi, dans notre exercice, on obtient que R vaut 4 Ω.

Si on affecte de nouvelle valeur à R_1 et R_2, il suffit de rappeler nos calculs à l'aide de l'historique de la calculatrice après avoir affecté nos nouvelles valeurs. Ainsi si $R_1 = 10\ \Omega$ et $R_2 = 30\ \Omega$ alors R vaut $7.5\ \Omega$.



Coefficient Multiplicateur

Enoncé

Un commerçant souhaite modifier le prix des articles qu'il vend dans son magasin.

Les informations sont récapitulées dans le tableau suivant :

Articles	Valeur initiale	Evolution	Coefficient Multiplicateur	Valeur Finale	Coefficient Réciproque	Evolution Réciproque
PC Portable	500	+15%				
Téléphone 1	242	−10%				
Câble USB	15	+30%				
Tablette	345	+15%				
Téléphone 2	99	−20%				
Console de jeux	199	−5%				

1. Calculer l'ensemble des coefficients multiplicateurs correspondants aux modifications souhaitées.
2. Calculer l'ensemble des valeurs finales.
3. Calculer l'ensemble des coefficients multiplicateurs réciproques qui permettraient de retrouver les valeurs initiales.
4. Calculer l'ensemble des taux d'évolution réciproque en pourcentage.

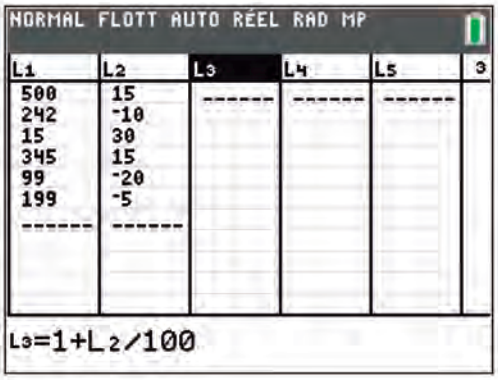
1. Coefficients multiplicateurs

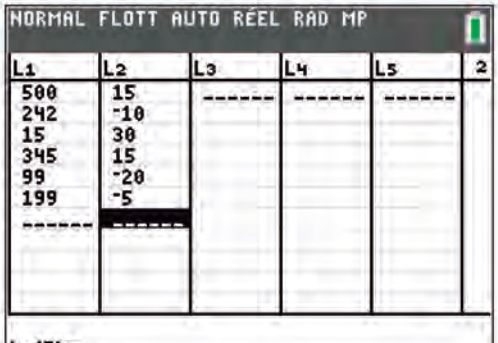
On commence par saisir dans l'éditeur de listes (stats puis **Modifier…**) l'ensemble des valeurs initiales données dans l'exercice.

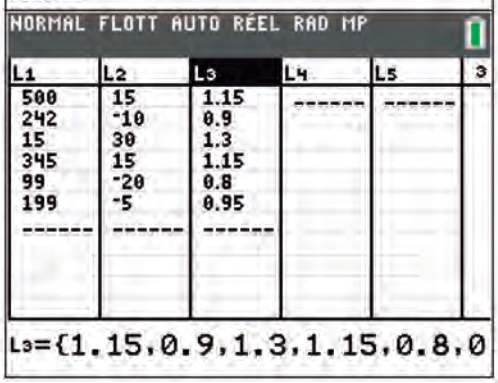
Le coefficient multiplicateur s'obtient à partir de la liste L2, en saisissant dans l'entête de L3, la formule **1+L2/100**

Pour cela, à l'aide des flèches de la calculatrice, on se positionne sur L3, on valide une première fois puis on saisit la formule souhaitée comme indiqué dans la capture d'écran.

Lorsqu'on valide, on obtient l'ensemble des valeurs souhaitées.







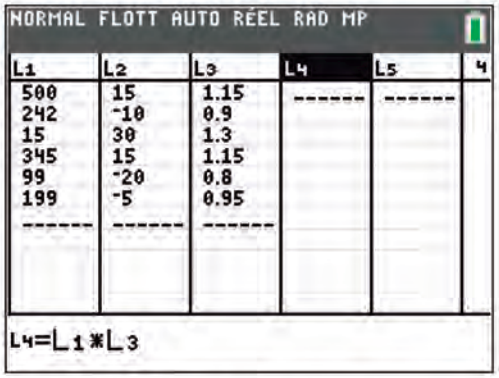
Coefficient Multiplicateur

Les noms de listes L1, L2, L3, L4, L5 et L6 peuvent s'obtenir immédiatement lorsqu'on saisit une formule, respectivement, par la combinaison des touches [2nde] + [1] ou [2] ou [3] ou [4] ou [5] ou [6]

2. Valeurs finales

On poursuit le travail de saisie, cette fois-ci dans L4 avec la formule L2*L3.

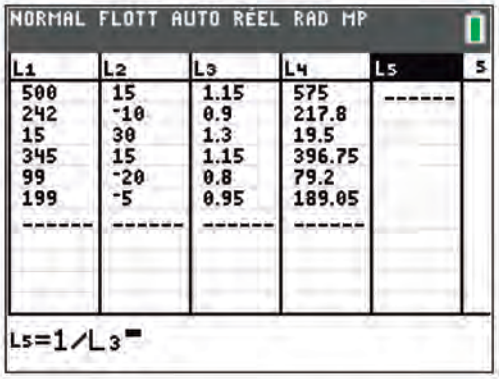
En effet, il s'agit de demander à la calculatrice de prendre chacune des valeurs contenues dans L2 et de les multiplier par le coefficient multiplicateur correspondant dans L3. La calculatrice comprend qu'elle doit fonctionner terme à terme.



3. Coefficients réciproques

On poursuit le travail dans L5 avec la formule 1/L3.

En effet, on cherche le coefficient multiplicateur réciproque qui est l'inverse du coefficient multiplicateur. On demande donc à la calculatrice d'inverser chacun des éléments de la liste L3.



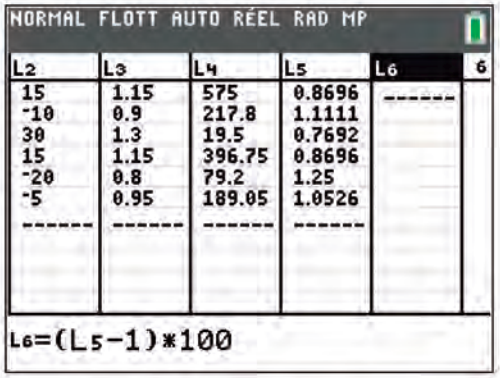
4. Taux réciproques

On souhaite maintenant obtenir le taux, exprimé en pourcentage, qui permettrait de revenir aux prix initiaux.

Augmenter ou diminuer une quantité de p %, c'est multiplier cette quantité par $CM = 1 + \frac{p}{100}$ où p représente le taux en pourcentage (positif pour une augmentation et négatif pour une réduction).

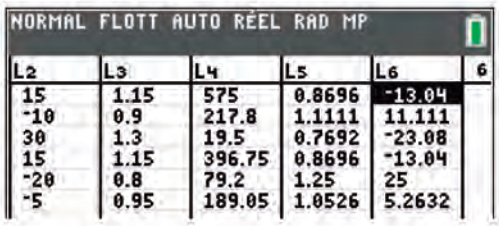
On en déduit que $p = (CM - 1) \times 100$.

Ainsi, on saisit dans L6, la formule (L5−1)*100

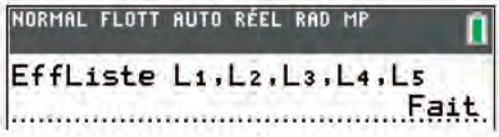


Cet exercice est l'occasion de retrouver que pour compenser une augmentation de 15%, il ne faut pas appliquer une réduction de 15% mais de 13 % (en arrondissant, bien sûr)

De même, on compense une baisse de 20% avec une augmentation de 25%



A la fin de l'exercice, vous pouvez effacer l'ensemble de votre travail à l'aide de la commande EffListe (accessible via la touche [stats]) et en complétant par le nom des listes à effacer.



Tracer & cadrer une fonction

Enoncé

On considère la fonction polynomiale de degré 3, définie sur $\mathbb{R}$ par :

$$f(x) = 2x^3 + 3x^2 - 72x + 35$$

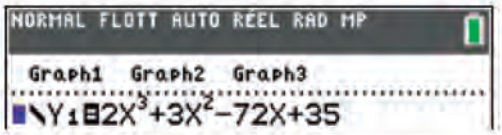
1. A l'aide de la calculatrice, compléter le tableau de valeurs suivant :

x	-8	-7	-6	-5	-4	-3	-2	-1	0	1	2	3	4	5	6
$f(x)$															

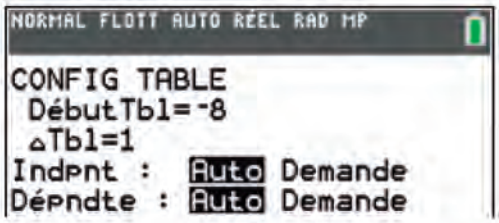
2. Tracer, sur papier millimétré, la courbe représentant la fonction f, sur l'intervalle $[-8\,;6]$.
3. Conjecturer, à l'aide de votre graphique :
   a. Le tableau de signes de f sur $[-8\,;6]$.
   b. Le tableau de variations de f sur $[-8\,;6]$.

1. Tableau de valeurs

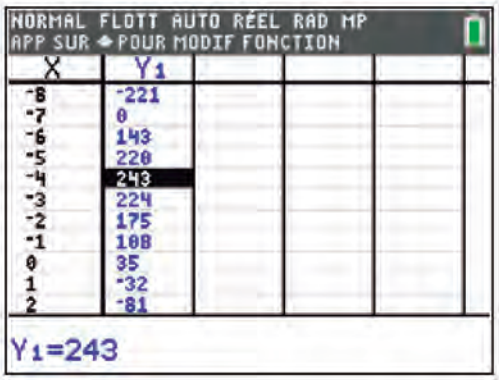
Dans le menu [f(x)], on commence par saisir l'expression de $f(x)$ dans **Y1**.



A l'aide des touches [2nde] [fenêtre] (déf table f2), nous entrons dans le menu de configuration du tableau de valeurs pour :



- Initialiser le tableau à **DébutTbl=-8** ;
- Définir le pas à **ΔTbl=1** ;
- Utilise le mode automatique (par défaut).



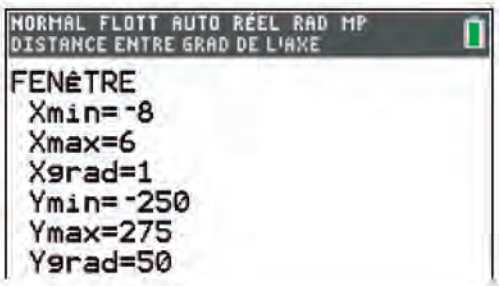
Remarque : pour obtenir d'autres valeurs de **X**, il est possible de paramétrer **Indpt** sur **Demande**. Cela permet de saisir la valeur de **X** que l'on souhaite et d'obtenir l'image correspondante dans le tableau de valeurs.

A l'aide des touches [2nde] + [graphe] (table f5), nous pouvons à présent consulter le tableau de valeurs de notre fonction. Les touches directionnelles permettent de se déplacer dans le tableau.

2. Représentation graphique

Les valeurs du tableau précédent vont nous permettre de renseigner la configuration de la fenêtre graphique de la calculatrice, accessible via la touche [fenêtre]

- **Xmin=-8** , **Xmax=6** et la graduation des abscisses **Xgrad=1** sont déterminées par l'intervalle de l'énoncé.
- **Ymin=-250** , **Xmax=275** et la graduation des ordonnées **Ygrad=50** sont déterminées à l'aide du tableau de valeurs.



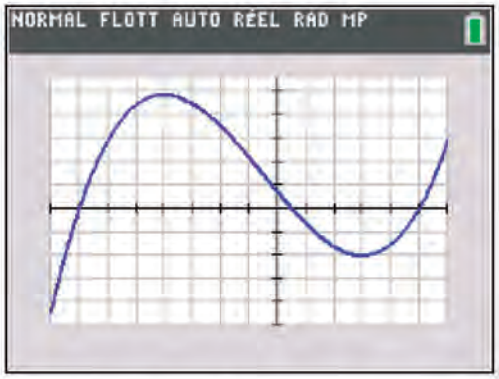
Il ne reste plus qu'à appuyer sur la touche [graphe] de la calculatrice pour observer la courbe ci-contre.

Tracer & cadrer une fonction

Ce travail de cadrage à la calculatrice est d'une grande aide pour le tracé « à la main » demandé. Il permet notamment de déterminer les échelles des abscisses et des ordonnées (en imaginant que l'écran est une feuille de papier millimétrée de format A4), mais également de placer ces axes intelligemment, afin que la courbe remplisse un maximum d'espace sur la feuille et que le tracé soit le plus précis possible.

Remarque :

Dans le menu [zoom], la commande **0:AjustZoom** peut également permettre un cadrage rapide et efficace.

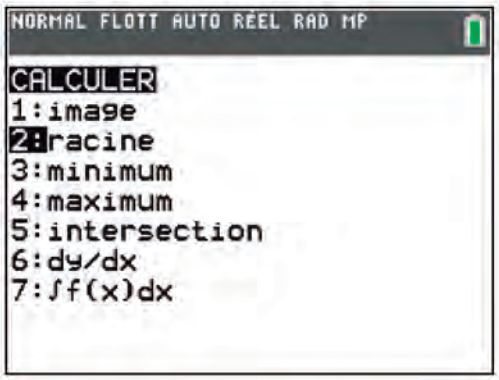


3. Conjectures graphiques

a. Tableau de signes

Grâce au graphique, on peut déjà conjecturer que les racines de f sont au nombre de 3. Nous allons voir comment déterminer une valeur approchée précise de la racine comprise entre 0 et 1 (cette méthode étant la même pour déterminer les 2 autres).

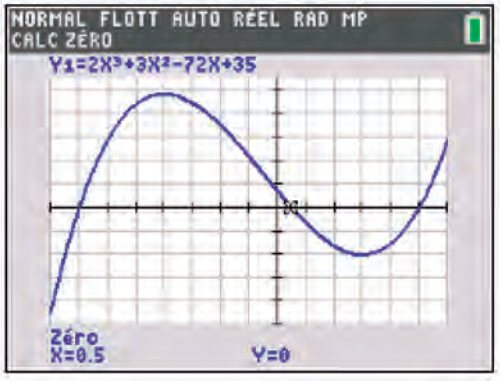
A l'aide des touches [2nde] [trace] (calculs f4), on commence par se rendre dans le menu **calculs**. On sélectionne la commande **2:racine** et on valide par la touche [entrer]. Une fois dans la fenêtre graphique, on place le curseur à gauche de notre racine et on appuie sur [entrer]. De même, en se plaçant à droite de la racine. On place enfin le curseur près de notre racine et on valide une dernière fois par [entrer]. On lit alors **X=0.5**. On obtient ainsi, après obtention des 2 autres racines :

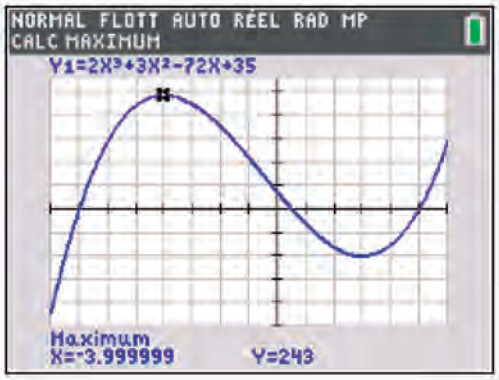


x	-8		-7		$\frac{1}{2}$		5		6
$f(x)$		$-$	0	$+$	0	$-$	0	$+$	

b. Tableau de variations

Pour les variations, il nous faut obtenir les coordonnées des extrema locaux de notre polynôme. Toujours dans le menu **calculs** ([2nde] [trace] (calculs f4)), on trouve les commandes **3:minimum** et **4:maximum**. La méthode est la même que pour la racine : on se place à gauche de notre extremum, puis à droite et enfin près de ce dernier. On valide chaque étape à l'aide de la touche [entrer]. On obtient finalement :





x	-8		-4		3		6
$f(x)$	-221	↗	243	↘	-100	↗	143

Thème : Analyse

Racines d'un trinôme du 2nd degré

TI-82 Advanced Edition Python

TI-83 Premium CE Edition Python

Enoncé

On considère le trinôme du 2nd degré suivant, défini sur $\mathbb{R}$: $f(x) = -2x^2 + 6x + 8$

1. A l'aide de la calculatrice, conjecturer une racine négative de f. Démontrer cette conjecture.
2. Calculer l'image de 4 par f. En déduire la forme factorisée de $f(x)$.
3. Expliquer pourquoi le maximum de la fonction f sur $\mathbb{R}$ est atteint pour $x = 1{,}5$.

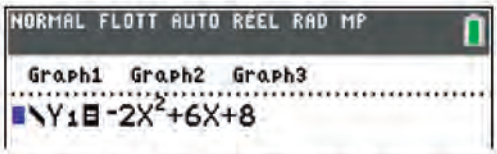
1. Racine négative de f

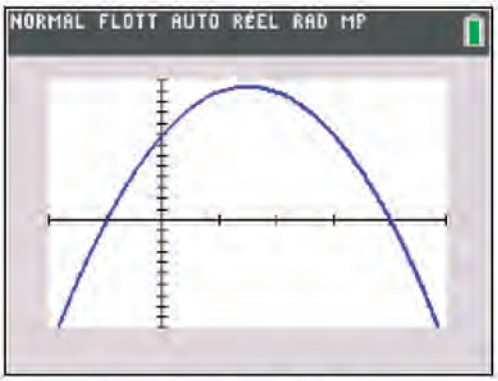
La première étape consiste à tracer la représentation graphique du trinôme.

- Dans le menu graph stats f1 [f(x)], on entre la fonction dans Y_1.
- On cadre la parabole, à l'aide du menu déf table f2 [fenêtre].
  Pour plus d'informations sur le cadrage d'un graphique, vous pouvez consulter la fiche 04-TRACER-CADRER UNE FONCTION.
- On peut déjà observer que ce trinôme semble admettre deux racines réelles distinctes, une négative et l'autre positive.

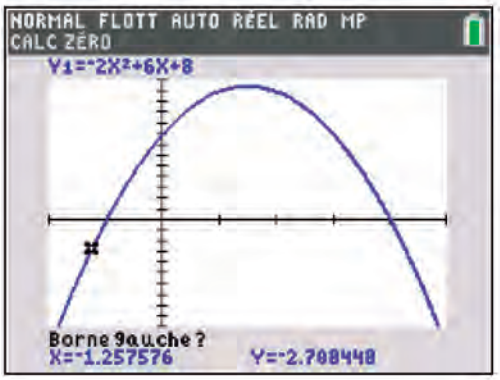
A présent, nous allons utiliser la calculatrice pour localiser la racine négative dont parle l'énoncé.

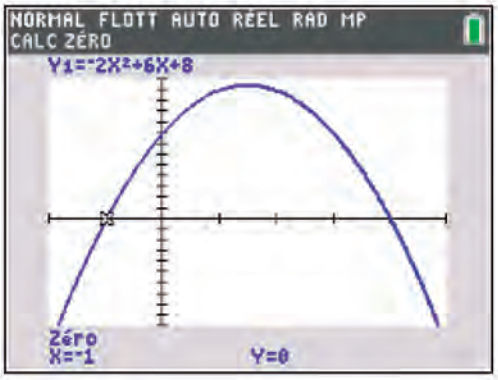
- Dans le menu [2nde] calcule f4 [trace], on sélectionne **2:racine** et on valide à l'aide de la touche [entrer].





- Désormais, on doit saisir à la calculatrice un intervalle dans lequel elle va rechercher une racine de notre trinôme. Rappelons ici que nous cherchons la valeur de la racine négative du trinôme.
  - On commence par définir la borne de gauche de notre intervalle. On se place donc à l'aide des flèches de direction à une abscisse de valeur inférieure à celle de la racine recherchée.
  - On fait de même pour la borne de droite demandée par la calculatrice.
  - Pour la 3ème et dernière étape, la calculatrice nous demande une valeur initiale. Cette étape est nécessaire car l'algorithme utilisé par la calculatrice est celui de Newton. Néanmoins, dans l'immense majorité des cas, il suffira d'appuyer sur [entrer] une dernière fois.
  - On lit la valeur de la racine, ici -1. Attention, cette valeur peut être approchée : il convient de valider la conjecture par un calcul.





$$f(-1) = -2 \times (-1)^2 + 6 \times (-1) + 8 = -2 - 6 + 8 = 0$$

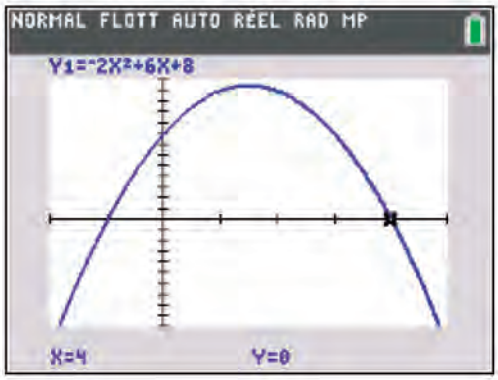
Racines d'un trinôme du 2nd degré

2. Seconde racine

Avant de faire le calcul, on peut déterminer l'image de 4 par la fonction f à l'aide de la calculatrice.

Dans le menu [2nde] [trace] (calculs f4), on sélectionne la commande **1:image**.

A l'invite de la calculatrice, dans le bandeau **X=**, on entre la valeur 4 et on valide par [entrer].



Un calcul nous permet alors de valider cette conjecture :

$$f(4) = -2 \times 4^2 + 6 \times 4 + 8 = -32 + 24 + 8 = 0$$

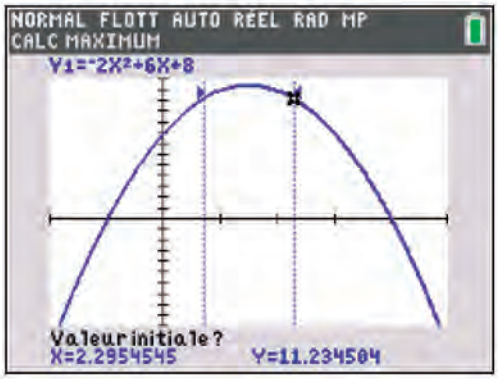
La seconde racine de f est donc 4, ce qui nous permet d'obtenir la forme factorisée du trinôme f (en faisant bien attention au signe de la racine négative) : $f(x) = -2(x+1)(x-4)$

3. Abscisse du maximum

Pour déterminer les coordonnées du sommet de la parabole, nous allons utiliser la commande **4:maximum**, disponible dans le menu [2nde] [trace] (calculs f4).

Cette commande fonctionne exactement sur le même principe que la commande **2:racine**, vue précédemment dans cette fiche.

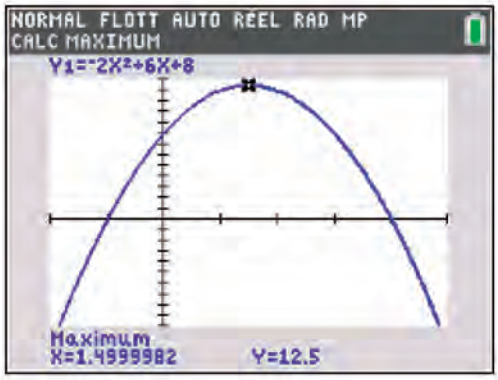
- On commence par définir la borne de gauche de notre intervalle. On se place donc à l'aide des flèches de direction à une abscisse de valeur inférieure à celle du maximum recherché.
- On fait de même pour la borne de droite demandée par la calculatrice.
- Pour la 3ème et dernière étape, la calculatrice nous demande une valeur initiale. Cette étape existe, de par l'algorithme utilisé par la calculatrice. Toutefois, étant donné que l'on construit généralement un intervalle où l'on ne rencontre pas de problème mathématique majeur, il suffira d'appuyer une dernière fois sur la touche [entrer].



- On lit la valeur de l'abscisse du sommet de la parabole, ici 1,5. Attention, cette valeur peut être approchée : il convient de valider la conjecture par un calcul.

  On rappelle que cette valeur est la moyenne des 2 racines du trinôme :

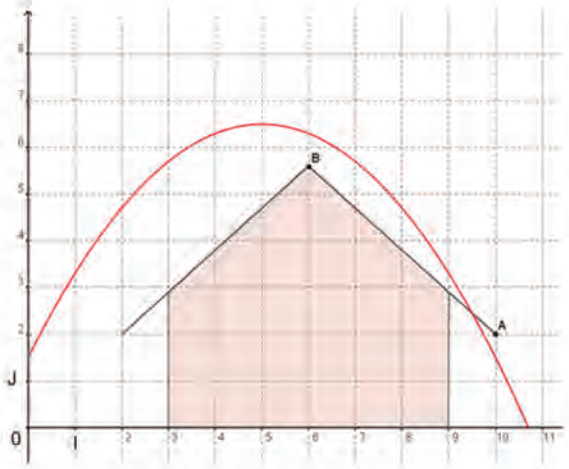
$$x = \frac{-1+4}{2} = \frac{3}{2} = 1{,}5$$



2nd degré : maximum & intersection

Enoncé

Durant une balade en forêt, un enfant se fabrique un arc et des flèches. Il s'intéresse à la trajectoire d'une de ses flèches. L'enfant décide de tirer sa flèche par-dessus un hangar désaffecté. La trajectoire est une portion de la courbe représentative de la fonction f située dans le quart plan rapporté au repère (O, I, J) ci-contre et définie pour tout réel x, par $f(x) = -0{,}2x^2 + 2x + 1{,}5$.



Une unité graphique correspond à 1 mètre dans la réalité.

1. a. De quelle hauteur, en mètre, la flèche est-elle tirée ? Justifier.

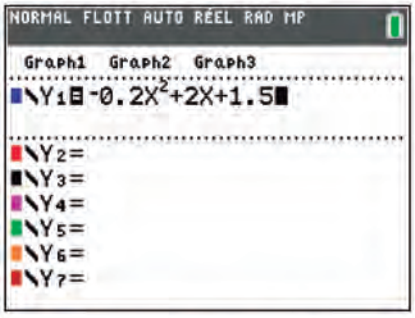
   b. Quelle hauteur maximale, en mètre, atteint-elle ? Justifier.

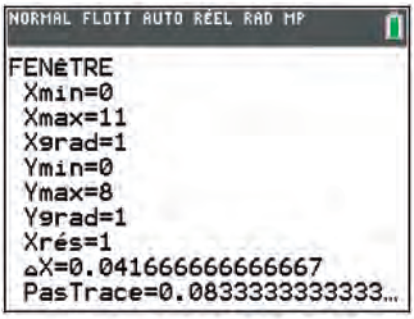
Une équation de la droite (AB) est $y = -0{,}9x + 11$. Démontrer que pour tout réel x, on a : $f(x) - y = -0{,}2(x-5)(x-9{,}5)$

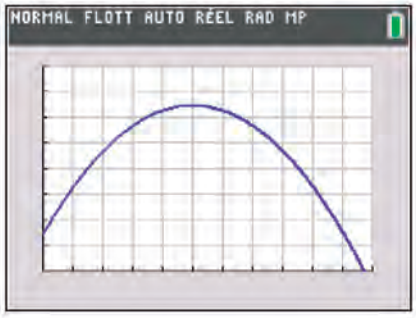
2. Quelles sont les coordonnées exactes du point d'impact sur le toit ?

Tracé initial

On commence par utiliser le menu [f(x)] pour entrer la fonction dans Y_1. Le graphique de l'énoncé nous permet de paramétrer rapidement une fenêtre adaptée à notre tracé, à l'aide du menu [fenêtre]. La touche [graphe] nous permet finalement d'obtenir la parabole de l'énoncé, ce qui va nous permettre de conjecturer les différentes réponses de l'exercice.

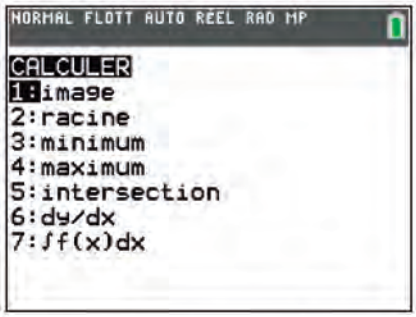


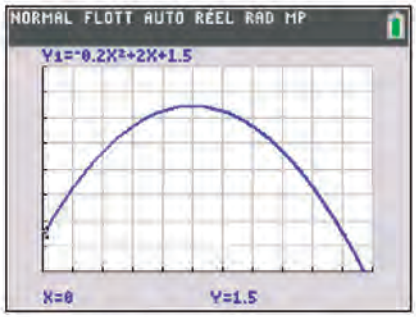


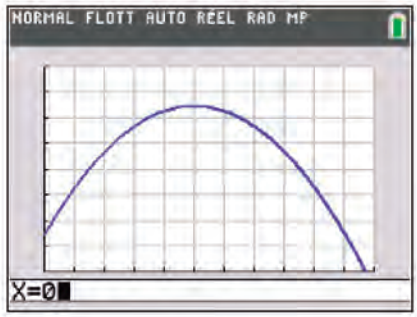


1.a. Image d'un réel par une fonction

Dans le menu [2nde] [trace] (calculs f4), on sélectionne **1:image**, puis on entre la valeur **X=0** et on valide par [entrer].







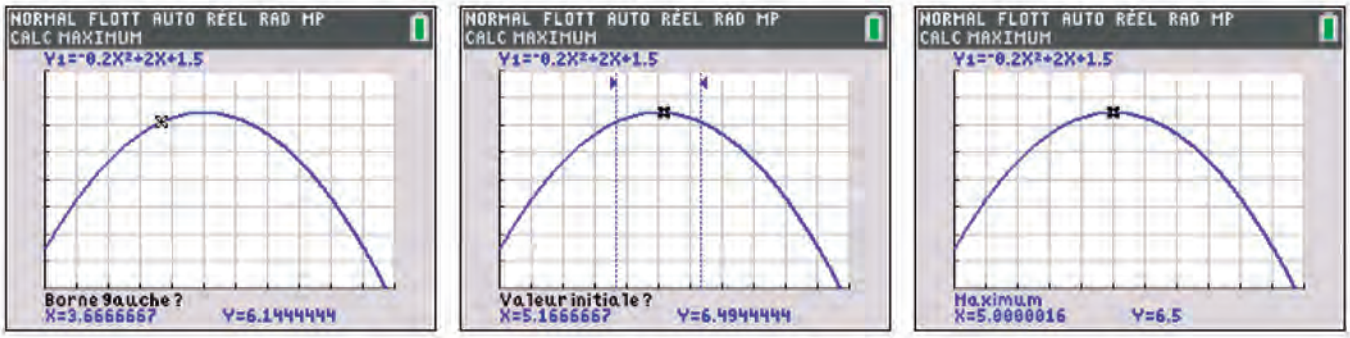
On valide ensuite cette conjecture par un calcul détaillé : $f(0) = -0{,}2 \times 0^2 + 2 \times 0 + 1{,}5 = 1{,}5$.

La flèche est donc tirée d'une hauteur de 1 mètre 50.

2nd degré : maximum & intersection

1.b. Valeur maximale d'une fonction sur un intervalle

Dans le menu [2nde] [trace] (calculs f4), on sélectionne **4:maximum**. De retour sur le graphique, on place le curseur à gauche du maximum et on appuie sur [entrer]. On se place ensuite à droite du maximum et on valide avec la touche [entrer]. On place enfin notre curseur près du maximum recherché et on appuie une troisième et dernière fois sur [entrer].



L'abscisse du sommet S de la parabole est donné par la formule $-\frac{b}{2a}$. On a donc ici : $x_S = -\frac{2}{2\times(-0,2)} = 5$.

De plus, $y_S = f(5) = -0,2 \times 5^2 + 2 \times 5 + 1,5 = -5 + 10 + 1,5 = 6,5$. La hauteur maximale est donc de 6 mètres 50.

2. Justifier une égalité

D'une part : $f(x) - y = -0,2x^2 + 2x + 1,5 - (-0,9x + 11) = -0,2x^2 + 2x + 1,5 + 0,9x - 11 = -0,2x^2 + 2,9x - 9,5$

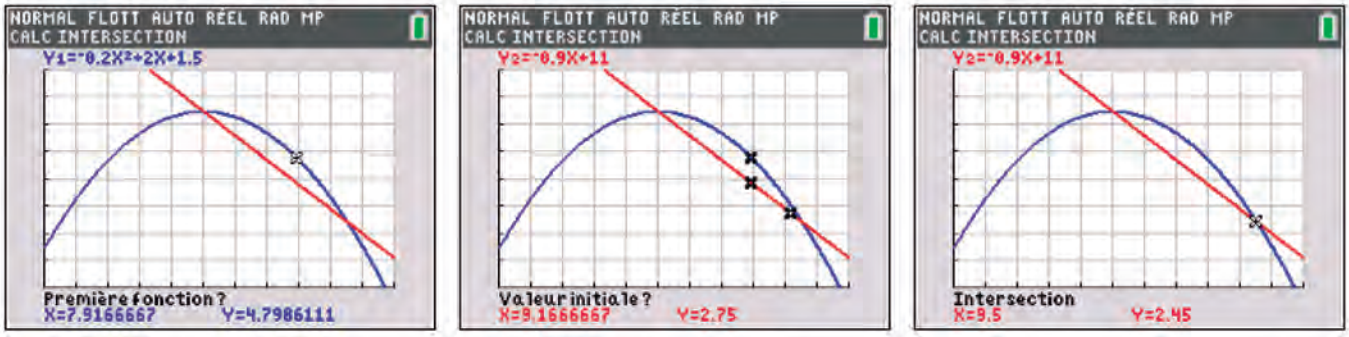
D'autre part : $-0,2(x-5)(x-9,5) = -0,2(x^2 - 9,5x - 5x + 47,5) = -0,2(x^2 - 14,5x + 47,5) = -0,2x^2 + 2,9x - 9,5$

L'égalité est ainsi justifiée, pour tout réel x.

3. Intersection de deux fonctions

Dans le menu [f(x)], on entre l'équation de la droite (AB) dans **Y2**.

Dans le menu [2nde] [trace] (calculs f4), on sélectionne **5:intersection**. De retour au graphique, on valide le choix de **Y1** par la touche [entrer], celui de **Y2** par [entrer] et enfin on se place près du point d'intersection recherché et on valide une troisième et dernière fois par [entrer].



L'intersection I est justifiée par la résolution de l'équation $f(x) - y = 0$.

La question 2. nous permet donc d'écrire $-0,2(x-5)(x-9,5) = 0 \iff x = 5$ ou $x = 9,5$.

Etant donné que nous recherchons une solution appartenant à l'intervalle $[6\,;10]$, on obtient donc $x_I = 9,5$.

De plus, $y_I = f(9,5) = -0,2 \times 9,5^2 + 2 \times 9,5 + 1,5 = -18,05 + 19 + 1,5 = 2,45$. Finalement, on a bien : $I(9,5\,;2,45)$.

Thème : Analyse

Trinôme du 2nd degré : image, racine & signes

TI-82 Advanced Edition Python

TI-83 Premium CE Edition Python

Enoncé

On considère la fonction f définie sur l'intervalle $[-4\,;4]$ par $f(x) = x^2 - 2x - 3$.

1. Calculer l'image de -1 par f.
2. Démontrer que 3 est solution de l'équation $f(x) = 0$.
3. A l'aide des 2 questions précédentes, donner une forme factorisée de $f(x)$.
4. En déduire le tableau de signes détaillé de f sur l'intervalle $[-4\,;4]$.

1. Image

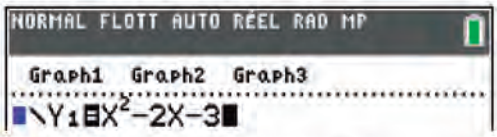
On commence par renseigner l'expression de la fonction f dans **Y1** en appuyant sur [f(x)].

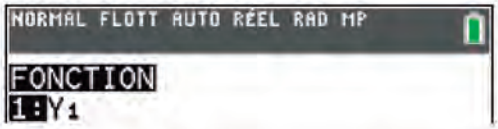
Dans l'écran de calcul ([2nde] [mode] (quitter)), on saisit l'expression **Y1(-1)**. **Y1** est disponible via la touche [var] , dans le menu **VAR Y**, en sélectionnant la commande **1:Fonction...**, puis **1:Y1**.

On valide la conjecture obtenue par un calcul détaillé :

$f(-1) = (-1)^2 - 2 \times (-1) - 3 = 1 + 2 - 3 = 0$

On peut déduire de ce calcul que -1 est une racine de f.





2. Racine

La calculatrice possède une application permettant de déterminer les racines des fonctions polynomiales, accessible via la touche [résol] et la commande **2:PlySmlt2**, suivie de **1:RACINES D'UN POLYNÔME**.

Une fois dans l'application (le nom apparaît dans le bandeau grisé supérieur de l'écran), on vérifie que le degré 2 est bien sélectionné, puis on valide par **SUIV.** (touche **f5**).

On saisit alors les coefficients de notre trinôme et on valide par **RÉSOL** (touche **f5**). Attention à bien renseigner les signes des coefficients !

Rappelons ici l'utilisation de la touche [X,T,θ,n] pour obtenir notre variable **X**.

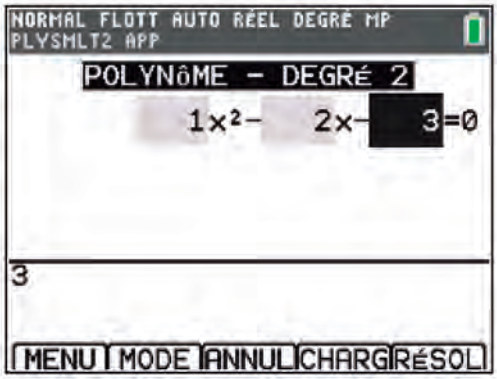
La calculatrice nous donne alors les 2 racines de la fonction f : -1 et 3.

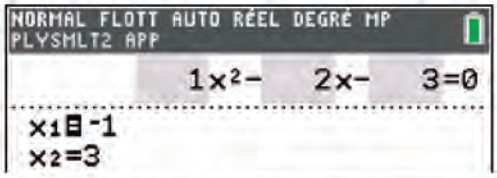
Reste à vérifier que 3 est bien une racine de f :

$f(3) = 3^2 - 2 \times 3 - 3 = 9 - 6 - 3 = 0$

Remarques :

- A l'aide de l'onglet **STO** (touche **f4**), on peut stocker les coefficients et/ou les racines dans une liste, mais également le polynôme dans une fonction **Y=**, ce qui nous évite de saisir 2 fois l'expression de la fonction.
- On quitte l'application à l'aide des touches [2nde] [mode] (quitter) et la commande **6:QUITTER APP**.





Trinôme du 2nd degré : image, racine & signes

3. Forme factorisée

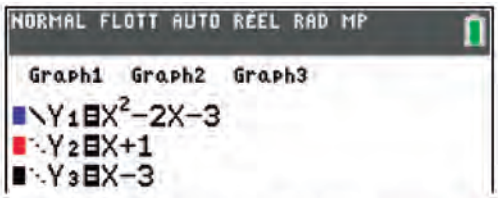
L'expression factorisée d'un trinôme du 2nd degré est de la forme :

$a(x-x_1)(x-x_2)$ où x_1 et x_2 sont les racines du trinôme. On déduit des 2 premières questions que :

$$f(x) = 1\big(x-(-1)\big)(x-3) = (x+1)(x-3)$$

4. Tableau de signes

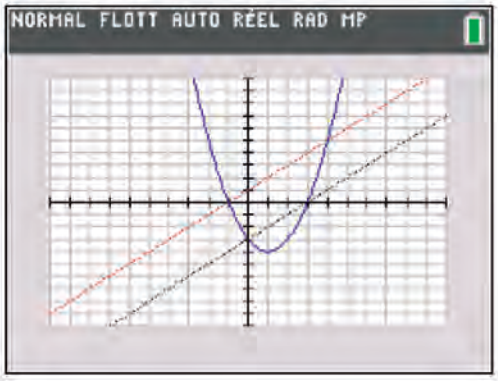
A présent que nous avons les racines et la forme factorisée de notre trinôme, nous allons pouvoir établir le tableau de signes détaillé de ce dernier, grâce aux signes de ses facteurs, à savoir $x+1$ et $x-3$.



Avant de passer à la construction du tableau proprement dit, il convient encore une fois de conjecturer les réponses, cette fois-ci à l'aide de la représentation graphique de notre fonction, mais également de ses facteurs.

Dans le menu [f(x)], nous saisissons donc **x+1** dans **Y2**, puis **x-3** dans **Y3**.

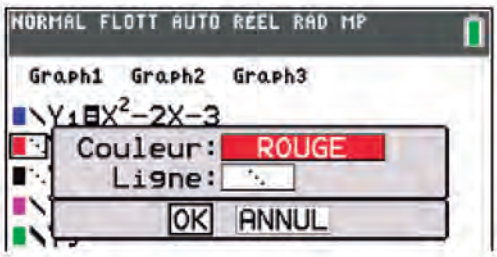
Dans le menu [zoom], nous sélectionnons la commande **6:ZStandard** et nous validons par [entrer].



Nous obtenons finalement la parabole représentative du trinôme f, accompagnée des 2 droites représentant ses 2 facteurs du 1er degré. Une lecture graphique nous permet ainsi de pouvoir vérifier tous les signes du tableau de signes ci-dessous.

Remarque :

Afin de différencier la fonction de ses facteurs, outre l'utilisation de la couleur, il peut être pratique d'utiliser des styles de tracé différents. Ces styles sont disponibles dans le menu [f(x)], en plaçant le curseur sur le rectangle de couleur en début de ligne et en appuyant sur [entrer]. Vous pouvez alors sélectionner une couleur différente et/ou un style différent.



x	-4		-1		3		4
$x+1$		$-$	0	$+$		$+$	
$x-3$		$-$		$-$	0	$+$	
$f(x)$		$+$	0	$-$	0	$+$	

Thème : Analyse

Fonction $x \mapsto ax^2 + b$

Résolution de système

TI-82 Advanced Edition Python

TI-83 Premium CE Edition Python

Enoncé

Soit f la fonction définie sur $\mathbb{R}$ par :

$$f(x) = ax^2 + b \ , \quad \text{où } a \text{ et } b \text{ sont des nombres réels.}$$

Déterminer les valeurs de a et de b sachant que $f(-1) = -2$ et $f(2) = 3$.

1. Mise en équation du problème

Nous commençons par transformer l'énoncé en système de 2 équations à 2 inconnues (a et b) :

$$\begin{cases} f(-1) = -2 \\ f(2) = 3 \end{cases} \Leftrightarrow \begin{cases} a \times (-1)^2 + b = -2 \\ a \times 2^2 + b = 3 \end{cases} \Leftrightarrow \begin{cases} a + b = -2 \\ 4a + b = 3 \end{cases}$$

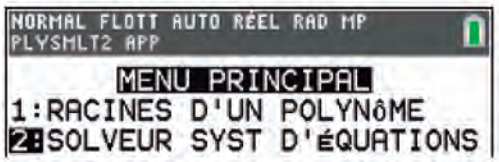
Pour tout système, il convient d'avoir les inconnues d'un côté de l'égalité, et les constantes de l'autre. Dans certains cas, il faudra donc faire davantage de manipulations, avant d'obtenir ce résultat.

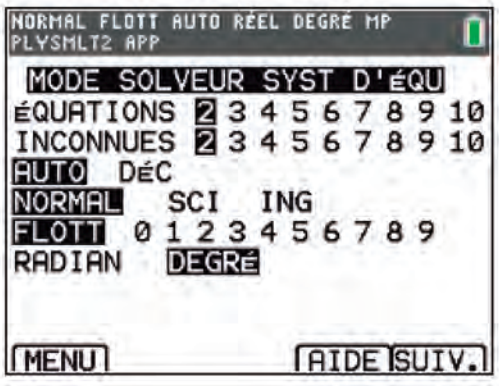
2. Résolution du système

Comme d'habitude en mathématiques, nous allons tenter de déterminer la réponse, avant de nous lancer dans la démarche de résolution de ce système.

La calculatrice possède un solveur de systèmes d'équations, qui va nous permettre de conjecturer la réponse de notre système.

- Dans le menu [résol] on sélectionne la commande **2:PlySmlt2**.
- Dans le menu principal nouvellement affiché de cette application, on sélectionne la commande **2:SOLVEUR SYST D'EQUATIONS**.
- Une fois dans le solveur proprement dit, on configure ce dernier. Pour notre exercice, aucune modification n'est nécessaire. On valide la configuration avec **SUIV.** ([table] [graphe]).



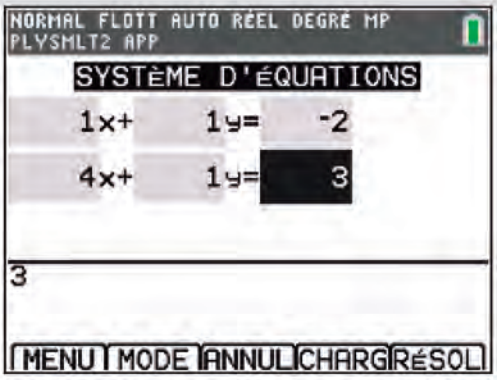


Fonction $x \mapsto ax^2 + b$

Résolution de système

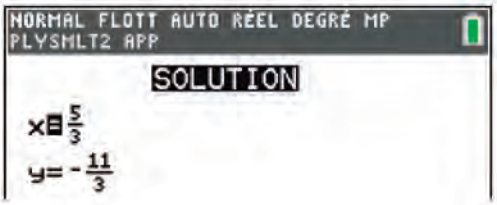
- A présent, nous allons saisir les coefficients de notre système. Les inconnues sont nommées x et y, par défaut. Elles joueront les rôles respectifs de nos inconnus a et b.
  On valide la saisie avec **RESOL** (table graphe).

  Attention à la gestion du symbole « - », qui peut signifier le signe d'un nombre, ou représenter la soustraction entre deux nombres.



- On lit le couple solution :

$$\left(\frac{5}{3} ; -\frac{11}{3}\right)$$



Maintenant que nous détenons la solution, résolvons algébriquement le système :

$$\begin{cases} a+b=-2 \quad (L_1) \\ 4a+b=3 \quad (L_2) \end{cases} \Leftrightarrow \begin{cases} a+b=-2 \quad (L_1) \\ 3a=5 \quad (L_2') = (L_2)-(L_1) \end{cases} \Leftrightarrow \begin{cases} a=\frac{5}{3} \quad (L_2') \\ \frac{5}{3}+b=-2 \quad (L_1) \end{cases} \Leftrightarrow \begin{cases} a=\frac{5}{3} \quad (L_2') \\ b=-\frac{11}{3} \quad (L_1) \end{cases}$$

3. Vérification graphique

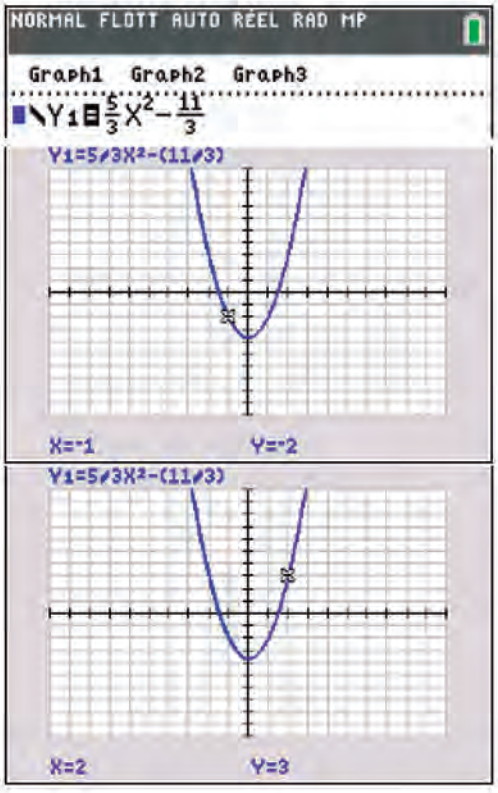
On peut effectuer une dernière vérification, graphique cette fois-ci.

- On entre l'expression $f(x) = \frac{5}{3}x^2 - \frac{11}{3}$ dans Y_1, du menu f(x).

- Dans le menu zoom on sélectionne la commande **6:ZStandard**.

- Dans le menu 2nde trace (calcule), on sélectionne la commande **1:image**.
  On entre alors la valeur -1 dans le bandeau inférieur, on valide par entrer et on vérifie bien que l'image de -1 est -2.

- On procède de même pour vérifier que $f(2) = 3$.



Thème : Analyse

Equations & inéquations du 2nd degré

TI-82 Advanced Edition Python

TI-83 Premium CE Edition Python

Enoncé

On considère les fonctions du 2nd degré f et g définies sur $\mathbb{R}$ par :

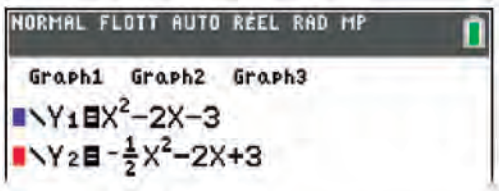
$$f(x) = x^2 - 2x - 3 \quad \text{et} \quad g(x) = -\frac{1}{2}x^2 - 2x + 3$$

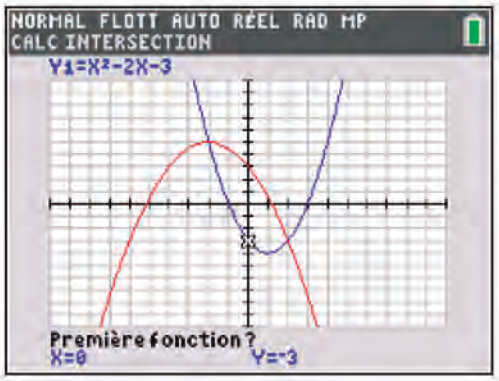
1. A l'aide de votre calculatrice graphique, conjecturer les coordonnées des points d'intersection des courbes représentatives des fonctions f et g.
2. Démontrer par le calcul les conjectures de la question précédente.
3. Résoudre graphiquement l'inéquation $f(x) < g(x)$.

1. Intersection

- Dans le menu [f(x)], on commence par saisir les expressions de nos fonctions dans Y1 et Y2.
- Dans le menu [zoom], on sélectionne la commande **6:ZStandard**, qui permet de cadrer notre graphique dans une fenêtre où les valeurs de x et y sont dans l'intervalle $[-10\,;+10]$.
- A l'aide des touches [2nde] [trace] (calcule f4), on obtient le menu où se trouve la commande 5:intersection. Cette dernière s'exécute en trois phases :
  - On sélectionne, à l'aide des flèches ▲ et ▼, la 1ère fonction (Y1 par défaut) et on valide par [entrer].
  - On effectue la même chose pour la 2nde fonction.
  - Enfin, on se place près du point d'intersection recherché, et on valide une dernière fois par [entrer].
- On procède ainsi pour les 2 points d'intersection observés et on obtient ainsi notre conjecture :

$$(-2\,;5) \quad \text{et} \quad (2\,;-3)$$





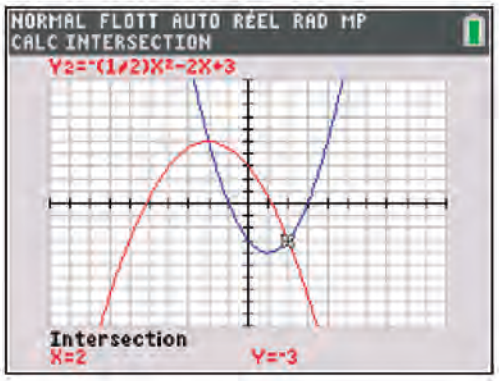
2. Résolution de l'équation

$$\begin{aligned} f(x) = g(x) &\Leftrightarrow x^2 - 2x - 3 = -\frac{1}{2}x^2 - 2x + 3 \\ &\Leftrightarrow 2x^2 - 4x - 6 = -x^2 - 4x + 6 \\ &\Leftrightarrow 3x^2 = 12 \\ &\Leftrightarrow x^2 = 4 \end{aligned}$$

D'où :

$$x = -2 \; ou \; 2$$

$$f(-2) = (-2)^2 - 2 \times (-2) - 3 = 4 + 4 - 3 = 5$$
$$f(2) = 2^2 - 2 \times 2 - 3 = 4 - 4 - 3 = -3$$

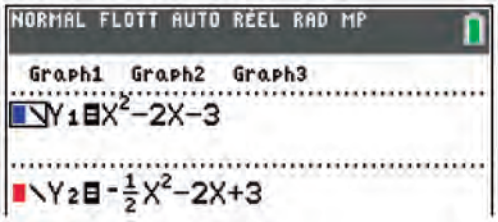


On retrouve ainsi les résultats graphiques précédents.

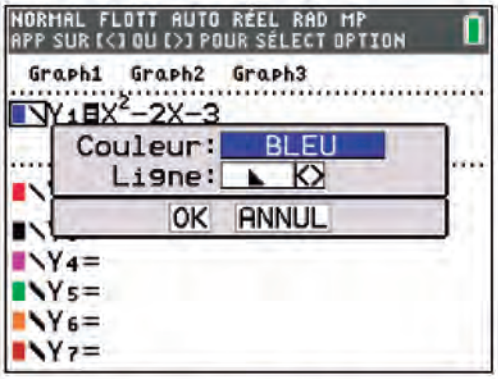
3. Résolution graphique de l'inéquation

La résolution graphique de cette inéquation peut bien entendu se faire à l'aide du graphique précédent, et c'est en général avec ce type de graphique que ces questions sont résolues.

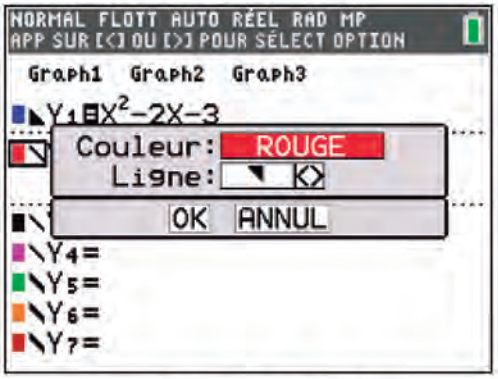
Pour autant, la calculatrice permet de visualiser encore mieux les solutions de tout type d'inéquation, et c'est ce qu'on se propose de vous faire découvrir ici.



- On commence par retourner dans le menu [f(x)], où sont définies nos 2 fonctions.

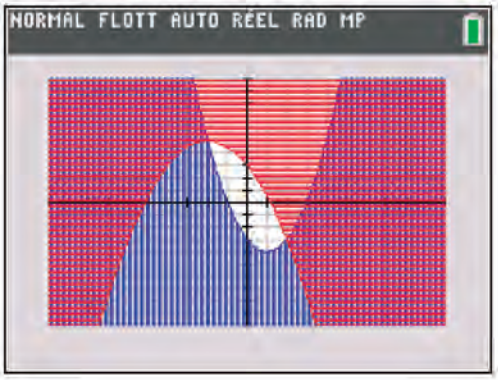


- Pour la fonction f saisie dans `Y1`, on vient placer le curseur au début de la ligne de saisie, sur le rectangle bleu, et on appuie sur [entrer].



- Dans le pop-up qui apparaît alors, on peut sélectionner la couleur de la courbe, mais surtout le type de ligne : on choisit alors le triangle noir inférieur gauche (voir écran ci-contre) et on valide par `Ok`. Lors du tracé, en plus de notre courbe représentative, la calculatrice va alors hachurer, de la même couleur, toute la région du plan qui va se trouver sous la courbe de f.

- On fait de même avec notre fonction g, saisie dans `Y2`, en sélectionnant cette fois-ci le triangle noir supérieur droit. Les hachures vont cette fois apparaître dans la région du plan située au-dessus de la courbe de g.



- On appuie alors sur [graphe] pour visualiser les courbes et les régions hachurées. Notre choix de hachures permet d'obtenir en blanc la région souhaitée par notre inéquation, ici la zone où la courbe de f se situe en-dessous de celle de g.

- Notre travail précédent sur les points d'intersection nous permet ainsi de conclure :

$$S =]-2\,;2[$$

Remarques :

- Nous vous conseillons ici de faire des tests de style pour vos courbes ; certains styles ne sont pas abordés dans cet ouvrage, mais peuvent vous servir à certaines occasions.

Dans le menu [f(x)], en plaçant votre curseur sur le symbole « = », vous pouvez désactiver l'affichage d'une fonction, sans l'effacer.

Equation réduite de droite

Enoncé

1. Déterminez l'équation réduite de la droite (GH) avec $G(2\,;3)$ et $H(6\,;5)$
2. Déterminez l'équation réduite de la droite (CD) avec $C(1\,;3)$ et $D(4\,;-1)$
3. Déterminez l'équation réduite de la droite (EF) avec $A(3\,;-3)$ et $B(5\,;-7)$

1. Equation réduite de (AB)

Bien que les automatismes ont vocation à se pratiquer sans calculatrice, il est intéressant de pouvoir vérifier ses résultats à l'aide de la machine.

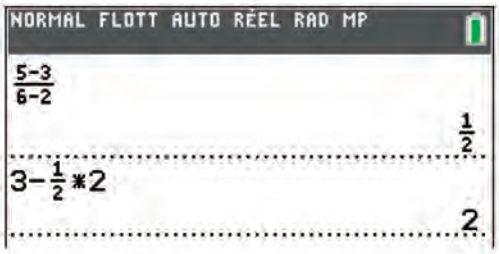
Commençons par nous servir de l'écran de calculs.

On connait la forme de l'équation réduite recherchée : $y = ax + b$.

On sait que a peut être déterminé à l'aide des coordonnées de G et H à l'aide de la formule $a = \frac{y_H - y_G}{x_H - x_G}$ puisque $x_G \neq x_H$. Ici $a = \frac{5-3}{6-2} = \frac{1}{2}$

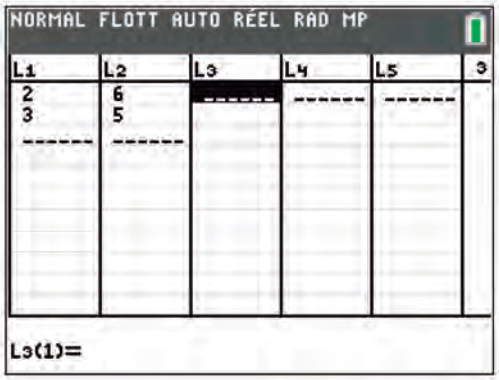
Une fois a trouvé, on peut calculer b à l'aide de la formule $b = y_G - a \times x_G$

Et donc $b = 3 - \frac{1}{2} \times 2 = 2$. L'équation réduite est donc $y = \frac{1}{2}x + 2$

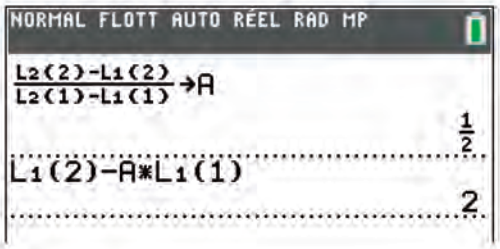


Mais dans le cadre d'une session d'entrainement, on peut supposer que l'on va être amené à reproduire ce calcul plusieurs fois, comme c'est le cas pour cet exercice.

Aussi il peut être plus judicieux d'utiliser les listes pour stocker les coordonnées. On place les coordonnées de G dans L1 et celles de H dans L2.



On saisit alors les formules $\frac{L_2(2)-L_1(2)}{L_2(1)-L_1(1)} \rightarrow$ A et L1(2)-A*L1(1)



Attention, dans ces formules, A est une variable de la calculatrice dans laquelle on stocke le coefficient directeur, pour le rappeler, sans avoir à le ressaisir dans la deuxième formule qui correspond au calcul de l'ordonnée à l'origine.

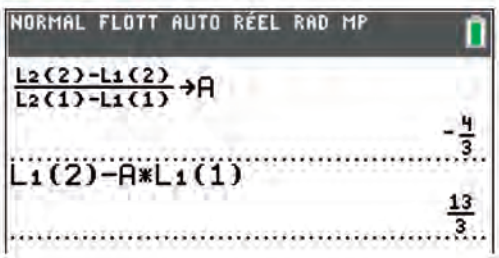
Il suffira ensuite de modifier les coordonnées des points dans l'éditeur de listes et de rappeler les formules saisies à l'aide de l'historique (flèche haute pour mettre en surbrillance la formule souhaitée puis [entrer] pour la copier coller et encore une fois [entrer] pour effectuer le calcul à nouveau).

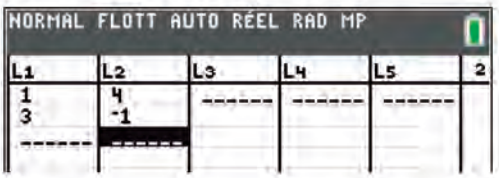
2. Equation réduite de (CD)

On a directement, par la méthode précédente que :

$a = -\frac{4}{3}$ et $b = \frac{13}{3}$

$y = -\frac{4}{3}x + \frac{13}{3}$





Equation réduite de droite

Mais nous allons utiliser une autre méthode mathématique.

La situation revient à résoudre le système suivant :

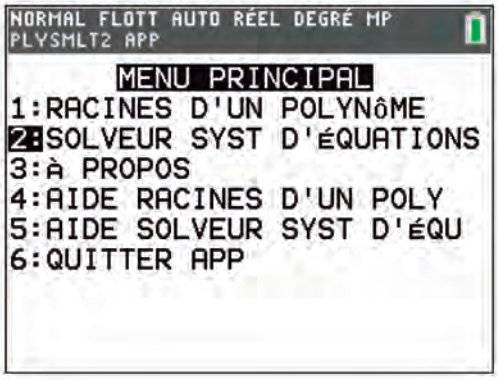
$\begin{cases} y_C = a \times x_C + b \\ y_D = a \times x_D + b \end{cases}$ où l'on cherche a et b.

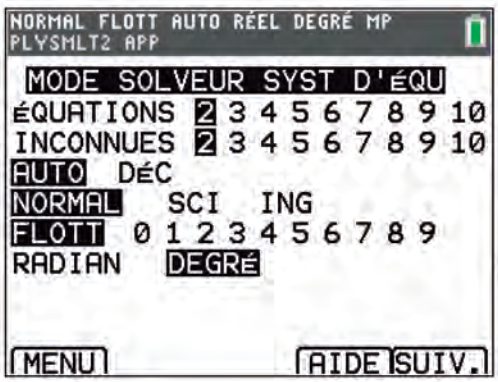
Si on remplace par nos valeurs numériques, on obtient le système suivant :

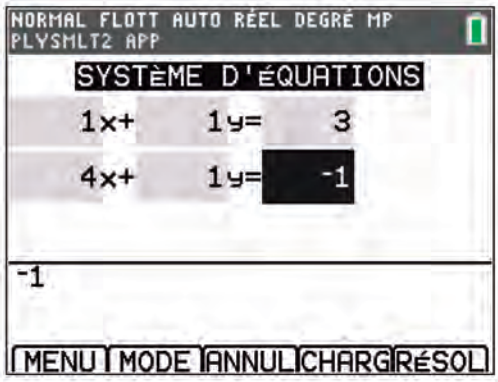
$\begin{cases} 3 = a \times 1 + b \\ -1 = a \times 4 + b \end{cases}$ ou bien encore $\begin{cases} a + b = 3 \\ 4a + b = -1 \end{cases}$

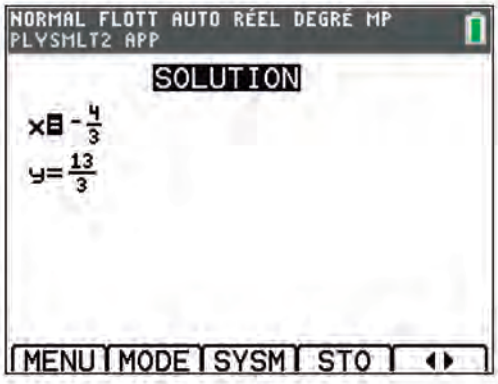
Notre calculatrice dispose d'un solveur de système dans l'application **PlySmlt2** accessible à l'aide de la touche [résol] .

Il s'agit ensuite de paramétrer notre système 2x2, de le saisir et à l'aide de l'onglet RÉSOL de demander sa résolution à la calculatrice.



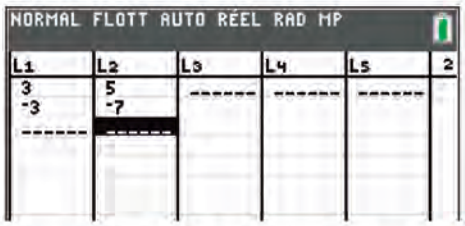


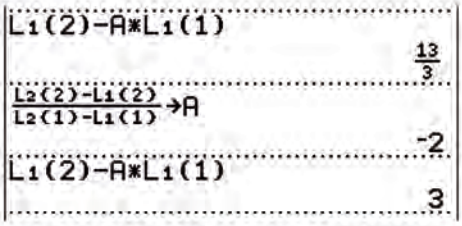


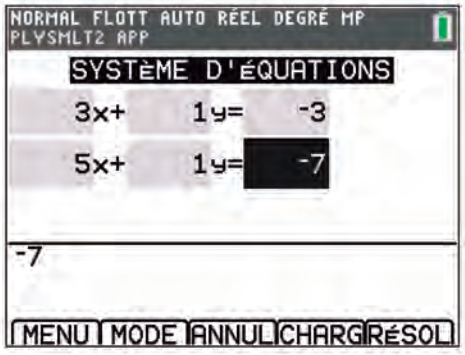


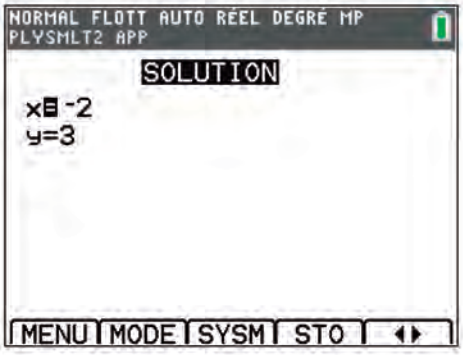
3. Equation réduite de (EF)

On peut donc au choix, utiliser la méthode par les listes pour trouver le coefficient directeur et l'ordonnée à l'origine ou bien la méthode par résolution du système.
On obtient $y = -2x + 3$









Thème : Analyse

Fonction du 2nd degré & Python

TI-82 Advanced
Edition Python

TI-83 Premium CE
Edition Python

Enoncé

Une entreprise fabrique des microscopes électroniques. Chaque mois, elle produit un nombre x de microscopes compris entre 1 000 et 3 000. Le coût de production, exprimé en euros, de x microscopes est donné par la fonction du 2nd degré :

$$f(x) = 0{,}448x^2 - 1\,452{,}7x + 1\,455\,391$$

Chaque microscope est vendu 249 € par l'entreprise. On suppose que l'entreprise parvient à vendre toute sa production.

1. L'entreprise réalise-t-elle un bénéfice lorsqu'elle fabrique et vend 1 200 microscopes ? 2 300 microscopes ?
2. On considère la fonction **Python** suivante :

```
def resultat(x):
    c=0.448*x**2-1452.7*x+1455391
    r=249*x
    return(r-c)
```

   Quel est le rôle de cette fonction ?

3. Tester cette fonction avec les valeurs de la première question.

1. Calcul d'image

On commence par saisir l'expression $f(x)$ dans Y_1 du menu [f(x)], mais également l'expression $249x$ dans Y_2.

Pour calculer les images de 1 200 et de 2 300 par f, nous allons utiliser le menu [var] de la calculatrice.

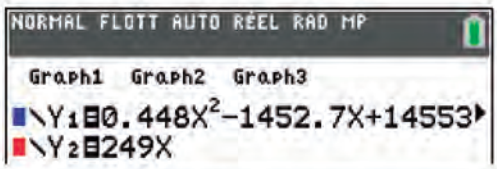
Dans ce menu, dans l'onglet **VAR Y**, nous sélectionnons la commande **1:Fonction**. Nous accédons alors aux noms des différentes fonctions saisies dans la calculatrice (les mêmes que dans le menu [f(x)] .

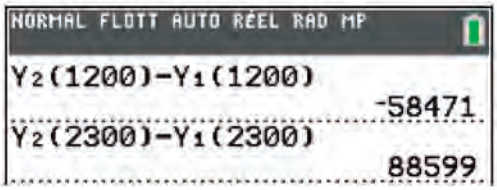
Nous saisissons alors le calcul **Y2(1200)-Y1(1200)**.

On obtient $-58\,471$ €. L'entreprise ne réalise donc pas de bénéfice.

On procède de même pour **2 300** et on obtient un bénéfice de 88 599 €.

Remarque : pour le 2nd calcul, on peut utiliser les touches [2nde] [entrer] pour récupérer le calcul précédent et modifier uniquement la valeur **1 200**.





2. Fonction Python

Le rôle de cette fonction est de calculer, en fonction de la valeur x entrée par l'utilisateur :

- Le coût de production c des x microscopes ; on reconnaît ici la fonction trinôme du 2nd degré f.
- La recette r des x microscopes, qui est une fonction linéaire.
- Le bénéfice réalisé en réalisant la différence des 2 fonctions précédentes.

Dans cette partie, nous allons utiliser le module **Python** disponible dans la calculatrice pour coder la fonction **resultat** dans un script, que nous exécuterons dans la console (**Shell**) afin de réaliser les tests de la 3ème question.

Fonction du 2nd degré & Python

Le module **Python** est accessible via la touche [prgm], en sélectionnant la commande **2:Python App**.

On parvient alors dans le « Gestionnaire de scripts », comme indiqué dans le bandeau supérieur. Nous allons donc créer un nouveau script à l'aide de l'onglet **Nouv**, accessible avec la touche **f3**.

A présent, on nomme notre script **MICRO** à l'aide de l'alphabet disponible sur les touches de la calculatrice. Notez que celle-ci est déjà en « mode alphabet », indiqué par le **A** en haut à droite de l'écran. On peut définir le style de notre script avec la touche **f3** ; le choix **2:Calculs Mathématiques** est une option qui permet de charger automatiquement la librairie mathématique de **Python**, qui donne accès aux fonctions usuelles (racine carrée, fonctions trigonométriques, etc.). On valide alors le nom du script par **Ok**.

Nous voici maintenant dans l'éditeur du script **MICRO**, comme indiqué dans le bandeau supérieur.

Dans l'onglet **Fns…** (touche **f1**), on accède immédiatement au menu **Fonc**, où se trouve les commandes nécessaires pour toute fonction.

On sélectionne ainsi **1:def fonction():** et, dans l'éditeur de script, on saisit immédiatement le nom de notre fonction, le curseur étant déjà correctement placé en mode insertion, à l'aide de la touche [alpha].

En appuyant sur la touche [▼], on se retrouve sur la ligne inférieure. Notez que l'indentation, si importante dans le langage **Python**, est déjà implémentée.

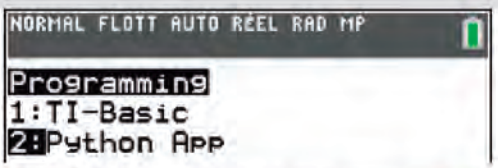
Des raccourcis ont été définis sur la calculatrice, pour faciliter le codage. Ainsi, le symbole = est accessible directement, via la touche [sto→]. La touche [x²] quant à elle, permet de coder le carré (**2 en **Python**).

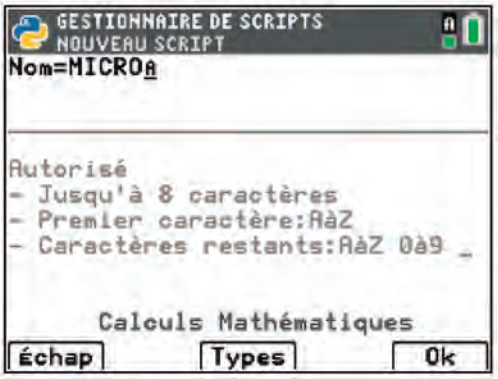
Pour la dernière ligne, rappelons que **return** est accessible via **Fns...**

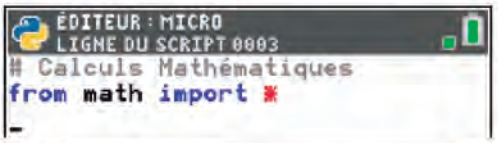
Une fois notre script terminé, on l'exécute **(Exéc**, via la touche **f4)**.

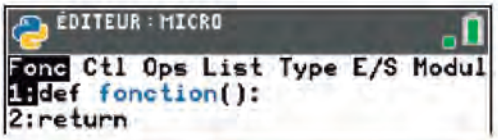
Remarques :

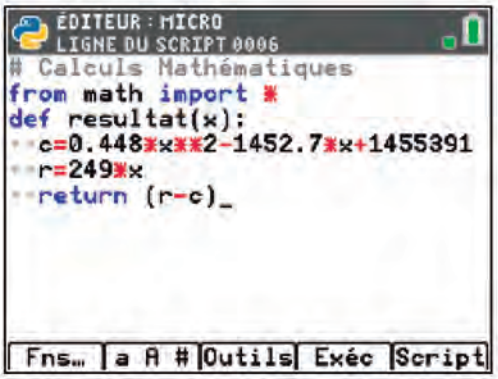
- Le mode « insertion » peut conduire à des erreurs de saisie par les débutants. Toutefois, la prise en main est rapide. Prenez votre temps lors de ce premier script !
- Attention à ne pas oublier les symboles de multiplication entre les nombres et la variable x. C'est une erreur très courante…











3. Tests

A présent, nous nous retrouvons dans la console Python : le Shell.

Notre fonction **resultat** est disponible à l'aide de la touche [var]

Une fois sélectionnée, nous pouvons la tester avec les valeurs 1 200 et 2 300 de l'énoncé.

Nous retrouvons bien les valeurs calculées précédemment.

Algorithme de seuil pour une suite

Enoncé

En 2019, le chiffre d'affaires d'un restaurant gastronomique était de 300 000 €.

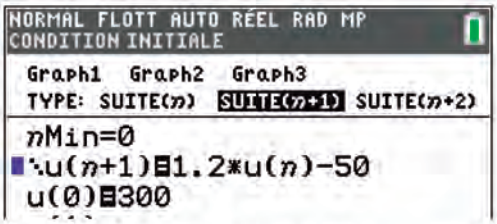
On modélise le chiffre d'affaires de ce restaurant (exprimé en milliers d'euros) pendant l'année $2019 + n$ par le n-ième terme, u_n , de la suite définie par : $u_0 = 300$ et $u_{n+1} = 1{,}2 \times u_n - 50$ pour $n \in \mathbb{N}$

1. A l'aide de la calculatrice et du mode **SUITE**, définir pour quelle valeur de n, le chiffre d'affaire du restaurant dépassera 500 000 euros. On utilisera un tableau de valeurs.
2. A l'aide du langage Python, définir la fonction **seuil1** qui renvoie n, la première valeur pour laquelle le chiffre d'affaire est supérieur à la valeur p passée en paramètre de la fonction. Vérifier pour $p = 500$.
3. A l'aide du langage Python et de la fonction **seuil1**, déterminer les années à partir desquelles le chiffre d'affaire dépasserait 5 millions d'euros puis 50 millions d'euros.

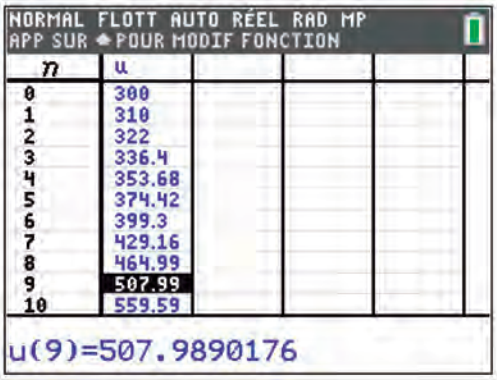
1. Déterminer n à l'aide du mode SUITE

Il s'agit de réutiliser les méthodes déjà vues dans ce livret.

On bascule la calculatrice en mode **SUITE** et on définit la suite (u_n).



On utilise la table des valeurs (2nde graphe) pour identifier la valeur de n recherchée.



On obtient alors que $n = 9$ puisque $U_8 \approx 465$ et $U_9 \approx 508$.

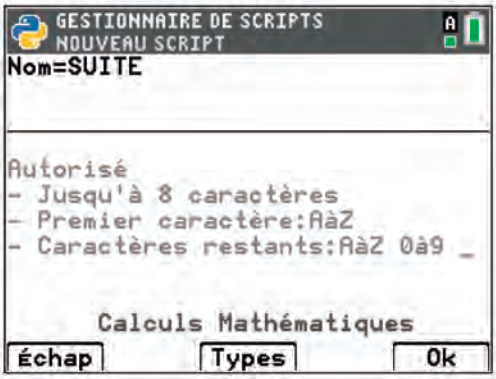
On cherche la première valeur de n qui permet de dépasser 500.

En effet, la suite est définie pour des valeurs exprimées en milliers d'euros.

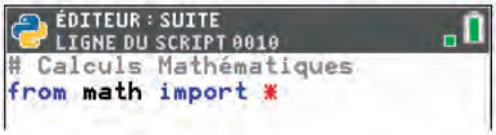
Il s'agit bien de $n = 9$.

2. Déterminer n en Python

On lance l'environnement Python (prgm) puis on crée un nouveau script (zoom pour l'onglet **Nouv**) que l'on nomme **SUITE** de type **Calculs Mathématiques** (zoom onglet **Types**).



La fenêtre de script s'ouvre et la librairie math est déjà importée.



La suite du travail va consister à compléter le script avec la définition de la fonction **seuil1**.

Algorithme de seuil pour une suite

On complète notre script avec les lignes de code ci-contre.

Il est important de faire appel aux différents menus de la calculatrice qui facilitent la saisie de notre programme. Ainsi en allant dans l'onglet [Fns…] à l'aide de la touche [f(x)], on accède au menu **Fonc** qui permet d'insérer directement les instructions de définition d'une fonction comme **def** ou **return** pour la valeur renvoyée par la fonction. L'environnement de la calculatrice complète automatiquement avec la bonne indentation marquée par des points gris. On peut également se servir du menu **Ctl** qui permet d'accéder aux structures de contrôle pour les importer directement indentée. Il n'y a alors plus qu'à compléter par la condition de sortie de la boucle dans le cas de l'instruction **while**. A noter qu'un certain nombre de caractères spéciaux sont accessibles dans l'onglet [fenêtre] à l'aide de la touche [a A #].

```
ÉDITEUR : SUITE
LIGNE DU SCRIPT 0010
# Calculs Mathématiques
from math import *

def seuil1(p):
··u = 300
··n = 0
··while u<p:
····u = 1.2*u-50
····n = n+1
··return n
Fns… | a A # | Outils | Exéc | Script
```

```
ÉDITEUR : SUITE
Fonc Ctl Ops List Type E/S Modul
1:def fonction():
2:return
Échap
```

```
ÉDITEUR : SUITE
Fonc Ctl Ops List Type E/S Modul
1:if ..
2:if .. else ..
3:if .. elif .. else
4:for i in range(taille):
5:for i in range(début,fin):
6:for i in range(début,fin,pas):
7:for i in liste:
8:while condition:
9:elif :
0:else:
Échap
```

```
ÉDITEUR : SUITE
LIGNE DU SCRIPT 0008
<_
  # " ' : , ; . ! ? _ \ ¯
  a b c d e f g h i j k l m
  n o p q r s t u v w x y z

ç à â è é ê ë î ï ô ù û ü ÿ æ
  () [] {} * ** % //
  = == != < <= > >=
  and or not True False
  << >> & | ^ ~
Échap | a◂▸A | Sélect | Coller
```

On exécute notre script à l'aide de l'onglet [Exéc], touche [trace]. Ceci nous permet d'accéder à la console Python liée à notre script.

A l'aide de la touche [var], on sélectionne la fonction **seuil1** et on complète avec le paramètre 500 (on aurait pu, bien sûr, saisir directement l'instruction **seuil1(500)**). La fonction renvoie 9. Ce qui est cohérent avec la question précédente.

```
PYTHON SHELL
VARS : SUITE
▸seuil1()
```

```
PYTHON SHELL
>>> # Shell Reinitialized
>>> # L'exécution de SUITE
>>> from SUITE import *
>>> seuil1(500)
9
>>>
```

3. Déterminer de nouvelles valeurs de n

On complète notre travail en console en saisissant les commandes **seuil1(5*10**3)** puis **seuil1(5*10**4)**. On obtient respectivement les valeurs 25 puis 38.

D'après ce modèle, le restaurant gastronomique dépassera le chiffre d'affaires de 5 millions d'euros en 2019+25 soit 2044 puis le chiffre d'affaires de 50 millions d'euros en 2019+38 soit 2057.

```
PYTHON SHELL
>>> # Shell Reinitialized
>>> # L'exécution de SUITE
>>> from SUITE import *
>>> seuil1(500)
9
>>> seuil1(5*10**3)
25
>>> seuil1(5*10**4)
38
>>>
Fns… | a A # | Outils | Éditer | Script
```

Taux de variation d'une fonction

Enoncé

On veut étudier des taux de variation du trinôme du 2nd degré, défini sur $\mathbb{R}$: $f(x) = -2x^2 + 6x - 1$

1. A l'aide du langage Python, définir **fonction1** qui renvoie l'image par f d'un réel x passé en paramètre. Et donner les images de 0, 1, 2 et 3 par la fonction f.
2. Dans le même script Python, définir la fonction **taux1**, qui renvoie le taux de variation de la fonction f entre les valeurs **a** et **b** distinctes passées en paramètres. Quel est le taux de variation de la fonction f entre 0 et 1 ? 1 et 2 ? 2 et 3 ?
3. En modifiant le script précédent, étudier les taux de variation de la fonction, définie sur $\mathbb{R}$: $f(x) = 2x^2 - 3x - 1$, entre 2 et 3 puis 3 et 2. Est-ce cohérent ?

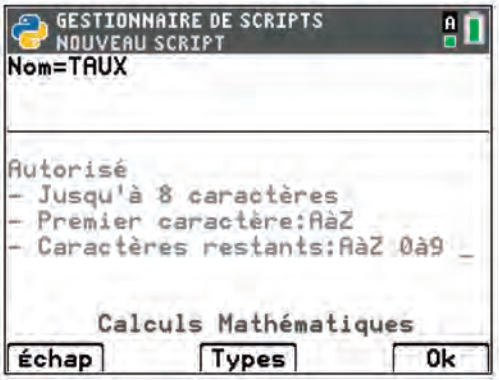
1. Définition de `fonction1`

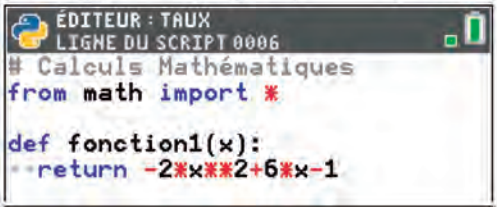
Nous commencons par créer un nouveau script Python appelé **Taux**, de type Calculs Mathématiques (la librairie **math** est ainsi déjà importée si nécessaire).

A l'aide de la touche [f(x)] nous sélectionnons dans l'onglet **fonc** l'instruction **def fonction() :** et complétons avec le nom imposé par l'énoncé **fonction1**.

Puis nous sélectionnons l'instruction **return** toujours dans le menu **fonc** et complétons avec l'instruction $-2 * x ** 2 + 6 * x - 1$

Nous obtenons le script ci-contre.



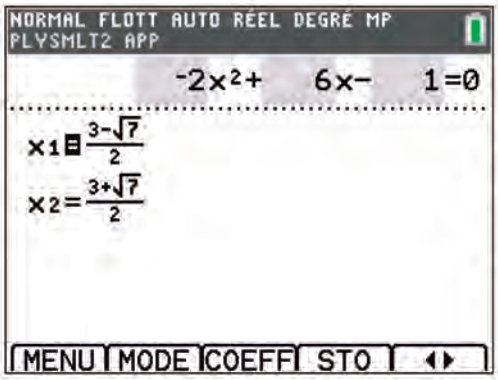


Nous l'exécutons pour le tester à l'aide des commandes **fonction1(0)**, **fonction1(1)**, **fonction1(2)** et **fonction1(3)**.

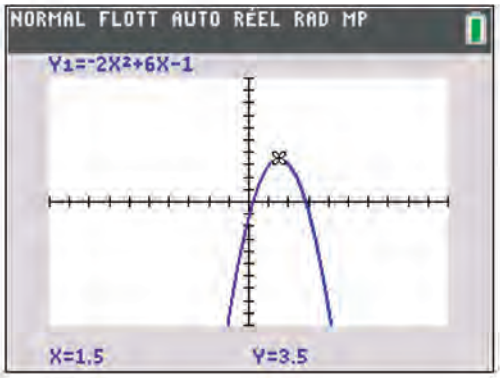
Nous obtenons bien que $f(0) = -1$, $f(1) = 3$, $f(2) = 3$ et $f(3) = -1$

D'après les propriétes de la fonction f, polynome de degré 2, on sait qu'elle possède un axe de symétrie.

Les manipulations réalisées permettent de se convaincre que celui-ci a pour équation $x = \frac{3}{2}$.



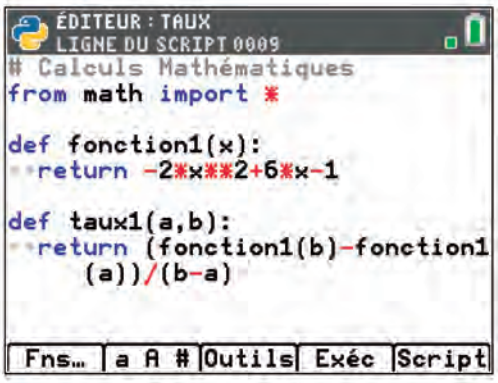
Pour le vérifier, vous pouvez réaliser la moyenne de ses deux racines obtenues à l'aide de l'application **PLYSMLT2** de la calculatrice ou bien représenter la fonction.



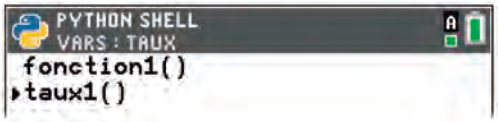
Taux de variation d'une fonction

2. Définition de `taux1`

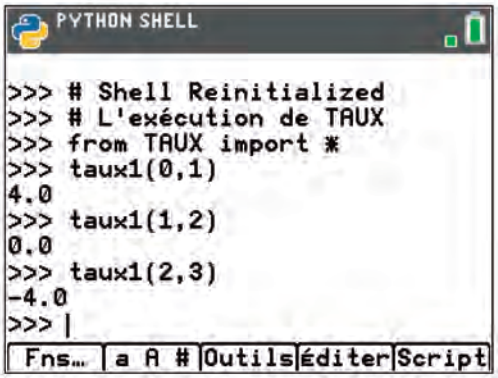
Nous éditons notre script Python précédent et le complétons avec la fonction **`taux1`** de paramètres **`a`** et **`b`** **`distincts`**, des nombres réels. Elle renvoie le quotient $\frac{fonction1(b)-fonction1(a)}{b-a}$. Ce qui donne le script ci-contre.



Nous testons ensuite la fonction **`taux1`** en lancant la console Python. En appuyant sur la touche [var], il est possible de sélectionner directement la fonction définie dans le script et que nous souhaitons utiliser.



On complète pour saisir les commandes **`taux1(0,1)`** puis **`taux1(1,2)`** et enfin **`taux1(2,3)`**



On obtient que :

- le taux de variation de f entre 0 et 1 vaut 4
- le taux de variation de f entre 1 et 2 vaut 0
- le taux de variation de f entre 2 et 3 vaut -4

Les résultats sont corrects puisque l'image de 0 par la fonction f vaut -1, l'image de 1 par la fonction f vaut 3 et $\frac{f(1)-f(0)}{1-0}=\frac{3-(-1)}{1-0}=4$.

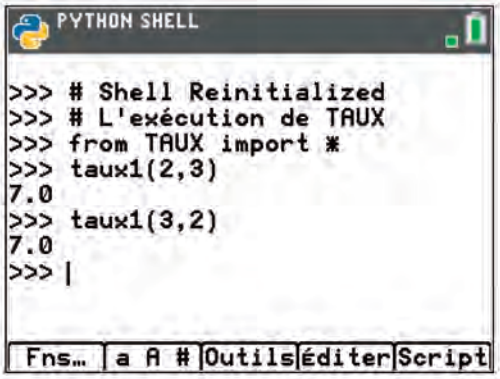
La fonction f admet un axe de symétrie d'équation $x=\frac{3}{2}$.

Donc $f(1)=f(2)$ et donc le taux de variation entre ces deux valeurs est nul.

Enfin, toujours par des considérations de symétrie, si le taux de variation vaut 4 entre 0 et 1, il est normal qu'il vaille -4 entre 2 et 3.

3. `taux1(a,b)` et `taux1(b,a)`

Nous éditons notre script avec la nouvelle définition de f dans **`fonction1`** et exécutons en console les commandes **`taux1(2,3)`** puis **`taux1(3,2)`**.



Nous trouvons la même valeur 7, ce qui est normal puisqu'en réalité si on conserve le même ordre au numérateur et au dénominateur pour les valeurs **a** et **b**, les deux fractions auront même valeur, les signes s'inversant simulatément au numérateur et au dénominateur. $\frac{f(b)-f(a)}{b-a}=\frac{f(a)-f(b)}{a-b}$

Algorithme de balayage pour la recherche d'un extremum

Enoncé

Soit f la fonction définie sur $\mathbb{R}$ par : $f(x) = -2x^2 + 6x + 8$. On souhaite rechercher le maximum de cette fonction. On va utiliser deux méthodes différentes.

1. Première méthode : retranscrire, en Python, la fonction **Maximum** ci-contre. On suppose que a < b et **f** est la fonction Python qui renvoie l'image d'un réel par f. A l'aide la fonction **Maximum**, utilisée avec les paramètres pertinents, déterminer le maximum de la fonction f.
2. Deuxième méthode : à l'aide de la calculatrice, déterminez les valeurs x_1 et x_2 pour lesquels la fonction f s'annule et en déduire la valeur du maximum, connaissant les propriétés de la fonction f.

```
Fonction Maximum(a,b,p)
    xmax ← a
    ymax ← f(a)
    Tant que x < b
        Si f(x) > f(xmax) alors
            xmax, ymax ← x, f(x)
        Fin Si
    x ← x + p
    Fin Tant que
    Renvoyer xmax, ymax
```

1. Méthode avec Python

On crée un script **BALAYAGE** dans lequel on commence par définir la fonction Python **f** qui renvoie l'image de x par f à l'aide de l'instruction :
`return -2*x**2 + 6*x + 8`

On définit ensuite la fonction **Maximum** de paramètres **a**,**b** et **p**.

Il existe plusieurs façons de réaliser un balayage. Ici, on a fixé le pas, c'est-à-dire la valeur avec laquelle on augmente **x** à chaque itération. On aurait pu opter pour passer en paramètre le nombre de valeurs de **x** testées.

On utilise donc une boucle **while**. Tant que **x**, notre variable intermédiaire, est inférieure à **b**, on teste si son image par la fonction **f** est plus grande que l'image maximale enregistrée. Si oui, on affecte les nouvelles valeurs. Dans tous les cas, on poursuit notre balayage jusqu'à dépasser **b**.

Il faut penser à initialiser les variables **xmax** et **ymax** qui seront renvoyées par la fonction.

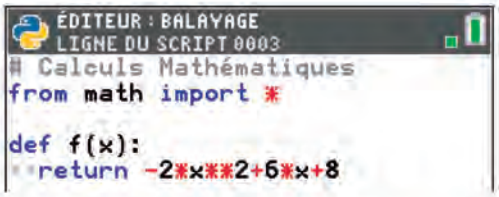
En mode console, on commence par évaluer quelques images de la fonction **f** pour avoir une idée où rechercher notre maximum.

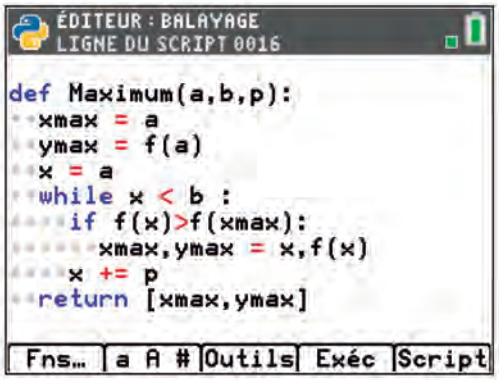
On a $f(-3) = -28$, $f(1) = 12$ et $f(5) = -12$. On teste notre fonction à l'aide de l'instruction **Maximum(-3,5,0.1)**.

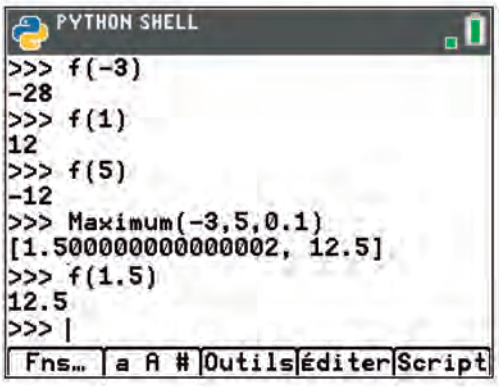
On obtient un maximum pour $x = 1{,}5$. On évalue notamment $f(1{,}5)$ pour comprendre qu'il s'agit bien d'un affichage, en quelque sorte, arrondi par Python.

Une question qui peut se poser est comment faire maintenant pour rechercher le maximum d'une autre fonction ? Par exemple, le maximum de la fonction h définie sur $\mathbb{R}$ par : $h(x) = -2(x+1)(x-6)$

On peut bien sûr redéfinir dans notre script la fonction **f**. Mais il est possible de rendre notre fonction **Maximum** plus « universelle » en lui ajoutant en paramètre le nom de la fonction dont on souhaite rechercher le maximum par balayage.





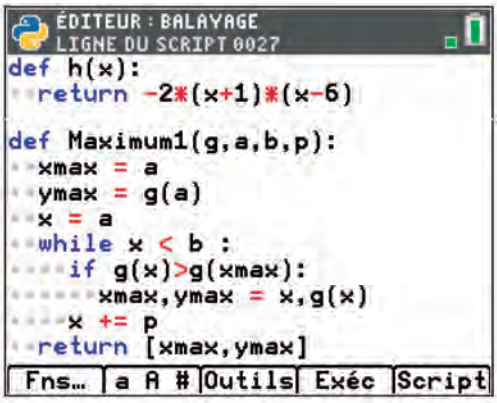


Algorithme de balayage pour la recherche d'un extremum

Créons une fonction **h** et une fonction **Maximum1** selon le script ci-contre.

Afin de faciliter la compréhension, on a appelé **g** la fonction passée en paramètre. **Maximum1** est proche du code de **Maximum** mais maintenant nous pouvons utiliser différentes fonctions dans notre script et ne plus dépendre de la seule fonction **f**.

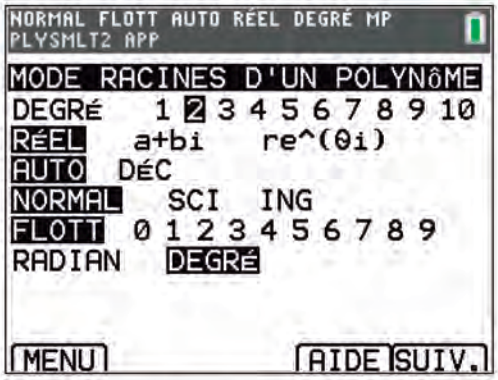
```
>>> Maximum1(f,-3,5,0.1)
[1.500000000000002, 12.5]
>>> Maximum1(h,-3,5,0.1)
[2.500000000000003, 24.5]
>>> |
```

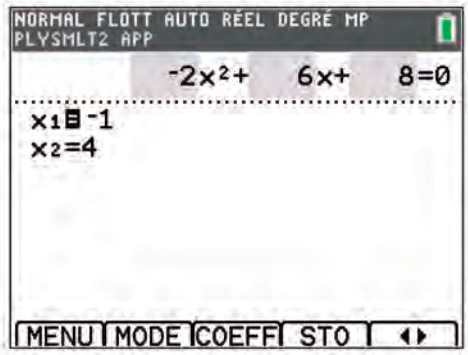


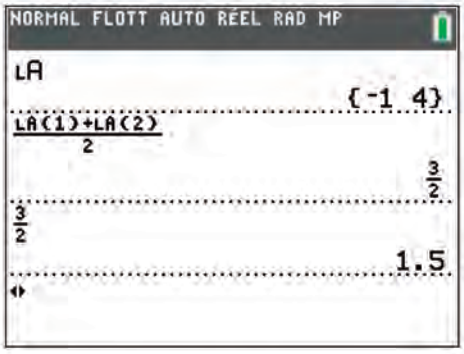
2. Méthode sans Python

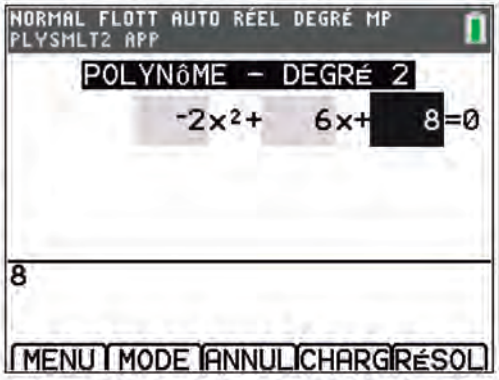
La calculatrice dispose de plusieurs outils pour ce genre d'exploration de fonctions. En particulier, elle dispose d'une application **PlySmlt2** capable de donner les racines d'un polynôme de degré 2. Nous allons nous en servir dans le cas de notre travail. D'autant que les solutions peuvent être exportées dans les listes pour être ensuite exploitées dans des calculs.

On configure l'application pour trouver les racines d'un polynôme de degré 2, que l'on saisit ensuite. L'onglet RÉSOL permet d'obtenir les racines, que l'on sauvegarde dans une liste LA à l'aide de l'onglet STO et rappelée dans l'écran de calcul à l'aide du menu **listes** (2nde + stats) puis **A.**



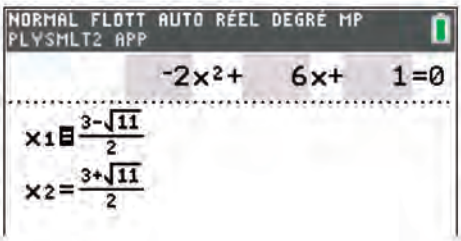


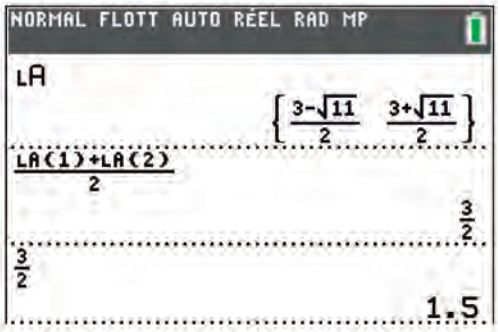




Dans le cas du polynôme f ou h, les racines sont suffisamment simples pour se passer de cette manipulation mais cela devient intéressant par exemple avec la fonction définie sur $\mathbb{R}$ par : $g(x) = -2x^2 + 6x + 1$.

En effet, de par les propriétés de symétrie, le maximum (ou minimum selon la nature du trinôme) est atteint pour la valeur moyenne des racines. Celles-ci peuvent parfois être longues à saisir.





Suite quelconque

Enoncé

En 2020, le chiffre d'affaires d'un restaurant gastronomique était de 300 000 €.

On modélise le chiffre d'affaires de ce restaurant (exprimé en milliers d'euros) pendant l'année $2020 + n$ par le n-ième terme u_n , de la suite (u_n) définie par :

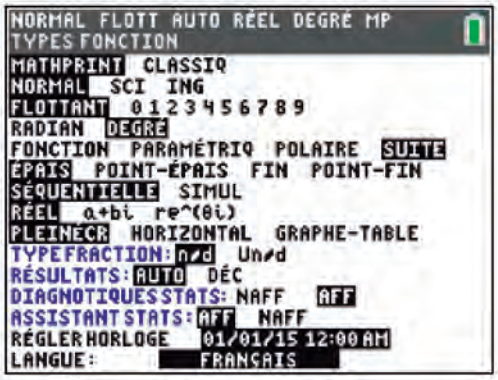
$$u_0 = 300 \quad \text{et} \quad u_{n+1} = 1{,}2 \times u_n - 50 \text{ , pour tout entier naturel } n.$$

1. Calculer u_1, puis établir, à l'aide de la calculatrice, le tableau de valeurs de (u_n), pour $n \in [0\,;10]$.
2. Tracer le nuage de points représentant (u_n), pour $n \in [0\,;10]$. Conjecturer le sens de variation de la suite (u_n).
3. La suite (u_n) est-elle arithmétique ? Donner un argument graphique, puis démontrer votre conjecture.

Définir une suite

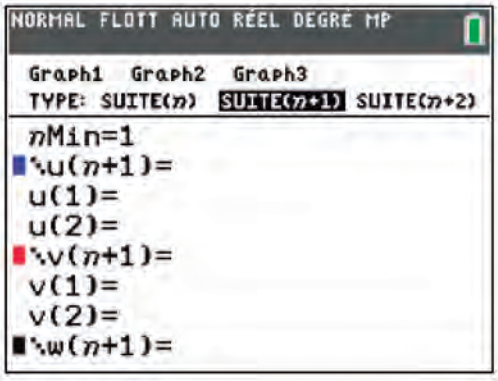
Afin de définir une suite à l'aide de la calculatrice, il faut tout d'abord la mettre en mode « suite ». La calculatrice **TI** permet de travailler dans quatre modes différents : fonction, paramétrique, polaire et suite. C'est évidemment ce dernier mode que nous allons activer pour cette activité.

On commence par utiliser le menu [mode] pour sélectionner le mode **suite** à la cinquième ligne, en validant la sélection par [entrer] .



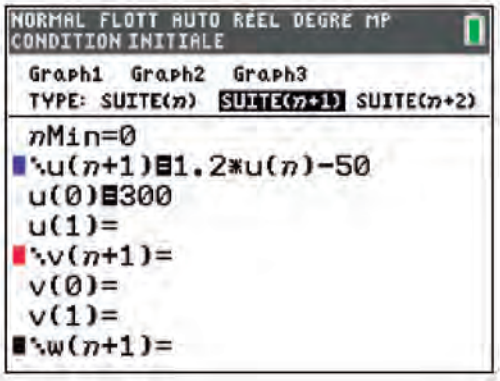
Une fois le mode **suite** activé, on entre dans le menu de saisie des suites, à l'aide de la touche [f(x)] . Ce menu, très différent du menu habituel des fonctions numériques, nous permet d'entrer des suites explicites ou récurrentes (sur un ou deux rangs).

Dans le bandeau supérieur, on sélectionne donc **SUITE(n+1)** en appuyant sur la touche [entrer] .



On renseigne ensuite les différentes informations : la valeur du 1er rang, la relation de récurrence et enfin la valeur initiale de la suite.

- **nMin** est initialisé à 0 ;
- la relation de récurrence est notée sous la forme fonctionnelle **u(n)** ; notez que **u** est obtenu à l'aide de [2nde] [7] et que **n** est obtenu à l'aide de la touche [X,T,θ,n] .
- **u(0)** est initialisé à 300.



Suite quelconque

1. Valeurs de (u_n)

On calcule u_1 à l'aide du calcul suivant :

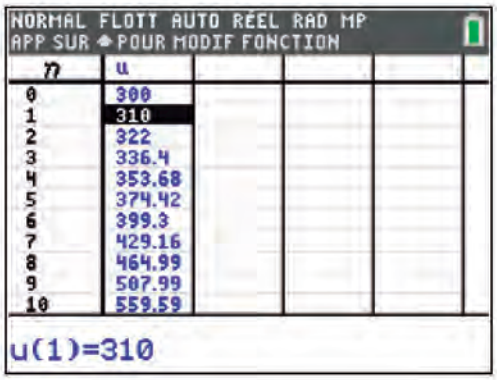
$u_1 = 1{,}2 \times u_0 - 50 = 1{,}2 \times 300 - 50 = 360 - 50 = 310$

Le chiffre d'affaires en 2021 est donc de 310 000 €.

A la calculatrice, à présent que notre suite est correctement définie, son tableau de valeurs est obtenu avec les touches [2nde] [graphe] .

La valeur 310 confirme donc bien notre calcul précédent.

Il est possible de configurer le tableau de valeurs (valeur initiale, pas, mode automatique ou à la demande), à l'aide de son menu de configuration disponible à l'aide de [2nde] [fenêtre] .



2. Nuage de points

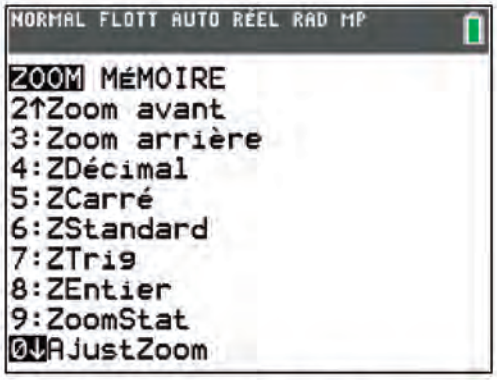
Pour obtenir le nuage de points représentant notre suite (u_n), nous pourrions configurer manuellement la [fenêtre] de notre graphique, à l'aide des valeurs du tableau précédent, comme nous le ferions habituellement pour une fonction classique.

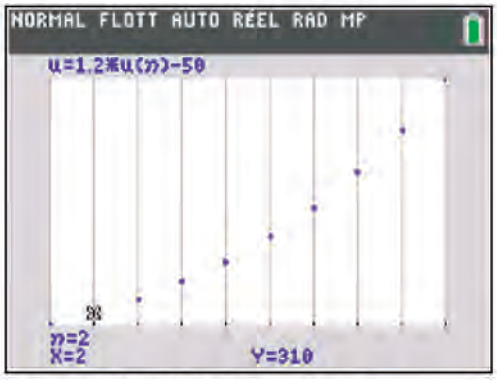
Toutefois, nous allons ici utiliser une fonctionnalité intéressante de la calculatrice.

Dans le menu [zoom] on sélectionne la commande **0:AjustZoom**, qui va cadrer pour nous notre nuage de points.

Nous obtenons ainsi directement le graphique ci-contre et nous pouvons conjecturer que la suite (u_n) semble strictement croissante.

A noter que l'on peut retrouver sur le nuage de points les valeurs du tableau, à l'aide de la commande [trace] .





3. Suite arithmétique ?

Les points de notre nuage de points n'étant pas alignés, la suite (u_n) n'est pas arithmétique. Par définition, il faut donc démontrer que l'on ne passe pas d'un terme au suivant en ajoutant un même nombre.

On commence par calculer u_2 :

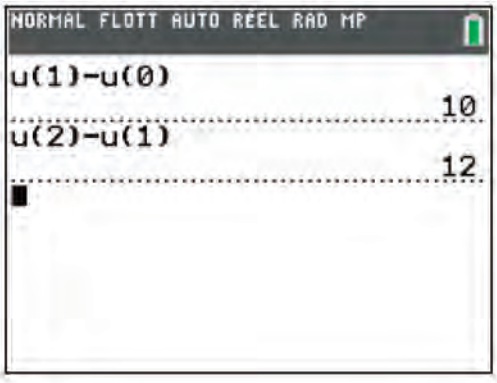
$u_2 = 1{,}2 \times u_1 - 50 = 1{,}2 \times 310 - 50 = 372 - 50 = 322$

Puis :

- $u_1 - u_0 = 310 - 300 = 10$
- $u_2 - u_1 = 322 - 310 = 12$

Puisque $10 \neq 12$, (u_n) n'est pas arithmétique.

On peut évidemment vérifier ces calculs à la calculatrice, comme proposé dans la capture d'écran ci-contre.



Suite arithmétique

Enoncé

En 2020, la population de la ville était de 23 600 habitants. On fait l'hypothèse que le nombre d'habitants augmente de 1 000 habitants par an. Pour tout entier naturel n, on note P_n le nombre d'habitants pour l'année $(2020 + n)$. La suite (P_n) est donc définie par récurrence par :

$$\begin{cases} P_0 = 23\,600 \\ P_{n+1} = P_n + 1\,000 \end{cases}, \quad \text{pour tout entier naturel } n$$

1. Déterminer la nature de la suite (P_n) en précisant ses caractéristiques. Calculer, en détaillant, la valeur de P_3. Interpréter le résultat dans le contexte de l'exercice.
2. On considère la fonction **ville** en langage Python ci-après. Quelle est la valeur affichée après l'exécution de cette fonction ? Interpréter le résultat dans le contexte de l'exercice.

```
def ville() :
    n=0
    p=23 600
    while p < 35 000 :
        p = p + 1 000
        n = n + 1
    return n
```

1. Définition de la suite (P_n)

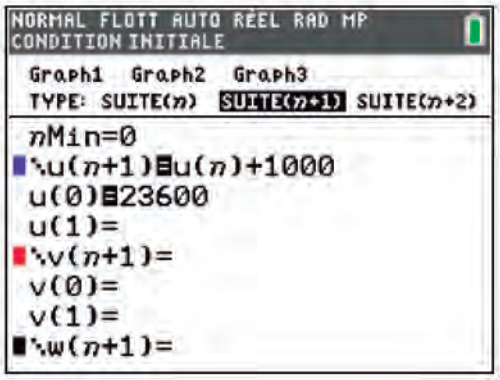
D'après la définition de la suite (P_n), un terme est obtenu en ajoutant 1 000 au terme précédent.

La suite (P_n) est donc une suite arithmétique de premier terme $P_0 = 23\,600$ et de raison $r = 1\,000$.

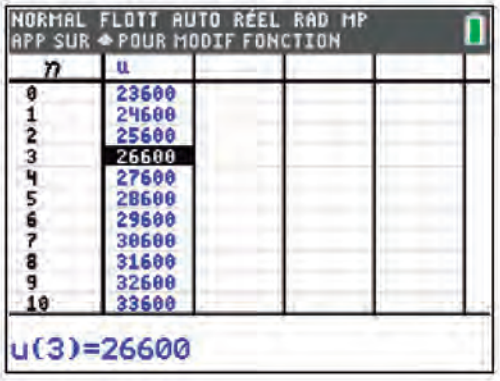
Le mode **suite** de la calculatrice peut être utilisé pour :

- définir la suite récurrente (P_n) ;
- établir le tableau de valeurs de la suite (P_n) ;
- tracer le nuage de points, représentant graphiquement (P_n).

Pour ce faire, consultez la fiche 04 - SUITE QUELCONQUE, et aidez-vous des captures d'écran ci-contre.



La moindre erreur de calcul dans une suite récurrente entraînant une succession de résultats erronés, il est primordial de conjecturer les termes de la suite, avant d'effectuer les calculs, à l'aide du tableau de valeurs.



On valide ensuite les conjectures obtenues à la calculatrice par les calculs détaillés suivants :

- $P_1 = P_0 + 1\,000 = 23\,600 + 1\,000 = 24\,600$
- $P_2 = P_1 + 1\,000 = 24\,600 + 1\,000 = 25\,600$
- $P_3 = P_2 + 1\,000 = 25\,600 + 1\,000 = 26\,600$

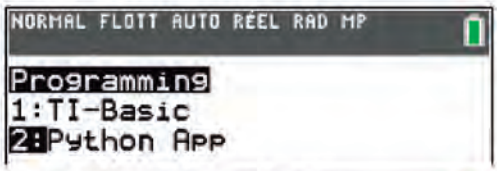
Ce dernier résultat permet d'affirmer que la population de la ville sera de 26 600 habitants en 2023.

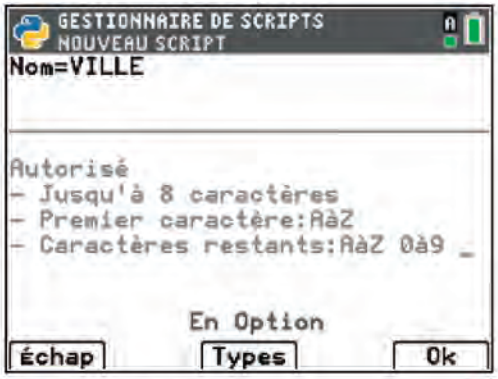
2. Boucle « tant que »

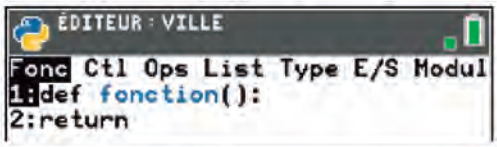
Cette question peut évidemment être résolue « à la main ». Dans cette activité, nous allons faire appel au module **Python** de la calculatrice et coder la fonction **ville**, puis l'exécuter pour obtenir le résultat.

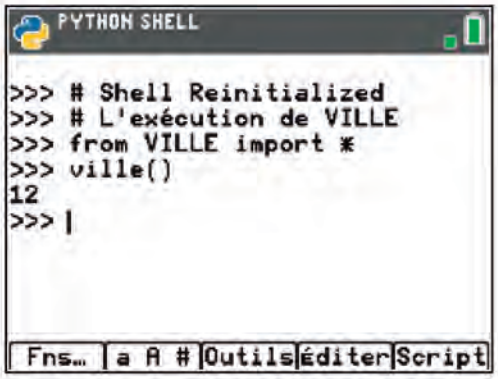
- Pour entrer dans le module Python, on utilise la touche [prgm] et on sélectionne la commande **2:PythonApp**.
- A l'aide de la touche **f3** ([zoom]) , on créé un nouveau script (**Nouv**), nommé **VILLE**, puis on valide par **Ok** ([graphe]).
- Nous sommes à présent dans l'éditeur du script **VILLE**, comme indiqué dans le bandeau supérieur de l'écran. Nous allons pouvoir coder la fonction **ville** à l'aide des renseignements suivants :
  - Dans le menu **Fns…**, accessible à l'aide de **f1** ([f(x)]), nous trouvons, dans l'onglet **Fonc**, la commande **1:def fonction():**, mais aussi la commande **2:return**. On notera qu'il suffit alors de compléter le nom de la fonction, le reste de la syntaxe étant automatiquement codé ; le curseur est déjà correctement placé, en mode insertion.
  - A l'aide des touches [2nde] [math] , on accède au menu des opérateurs logiques, où l'on trouve notamment le symbole = (on notera le raccourci de la touche [sto→] et aussi le symbole <.
  - Pour la structure de contrôle « tant que », on utilise le menu **Fns… (f1)**, puis l'onglet **Ctl** où se trouve la commande **8:while condition:**. De même que pour la définition de la fonction, le curseur est déjà correctement placé, en mode insertion, pour pouvoir écrire la condition d'arrêt de la boucle.
  - Pendant toute la saisie, il faut prendre garde à l'indentation du script, point très important du codage en langage **Python**.
- Une fois la fonction **ville** codée, on l'exécute à l'aide de **Exéc** (**f5**). On se retrouve alors dans la console (le **Shell**). A l'aide de la touche [var], on sélectionne notre fonction **ville** et on valide avec Ok (**f5**). On appuie alors sur [entrer] pour lancer notre fonction et on obtient le résultat 12, comme indiqué dans la capture d'écran ci-contre.

C'est donc en l'an 2032 que la population de la ville dépassera le nombre des 35 000 habitants. On peut évidemment vérifier ce résultat dans le tableau de valeurs obtenu dans la 1ère partie de l'activité.





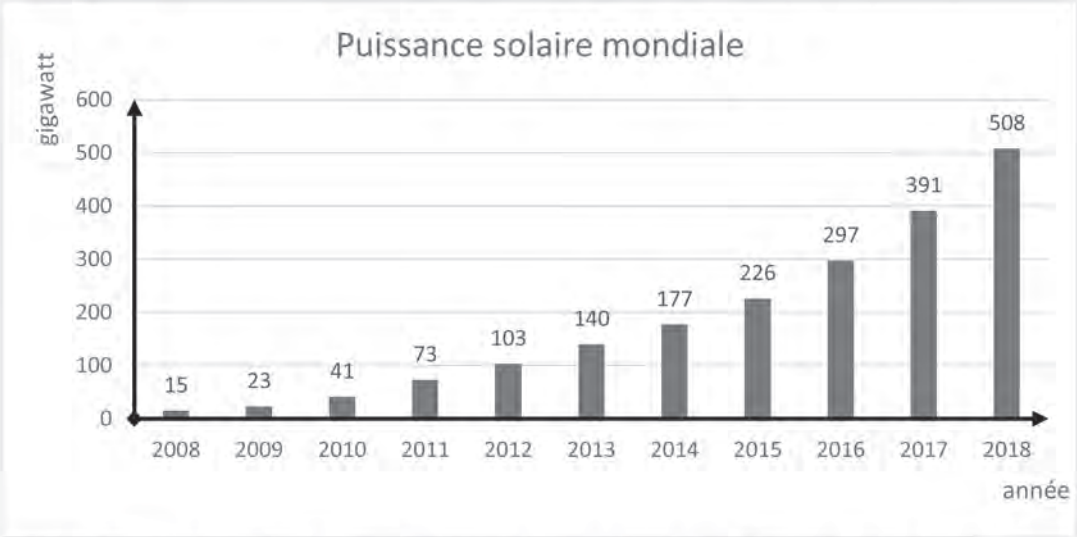




Suite géométrique

Enoncé

L'évolution de la puissance solaire photovoltaïque dans le monde entre fin 2008 et fin 2018 est résumée dans le graphique ci-dessous :



1. Montrer qu'entre fin 2008 et fin 2018, la puissance solaire photovoltaïque a augmenté d'environ 3287 %.

2. On se propose d'estimer la puissance solaire photovoltaïque dans le monde pour les années à venir en faisant l'hypothèse que le taux de croissance annuel restera constant et égal à 30%. On note P_n la puissance solaire photovoltaïque dans le monde, en gigawatt, à la fin de l'année 2018 + n. Ainsi, $P_0 = 508$.

   a. Justifier que, pour tout entier naturel n, $P_{n+1} = 1{,}3 \times P_n$. Quelle est la nature de la suite (P_n) ?

   b. Un chercheur affirme que si le taux de croissance se maintient à 30 %, la production dépassera les 2400 gigawatts avant fin 2024. A-t-il raison ? On justifiera la réponse par un calcul.

1. Taux d'évolution

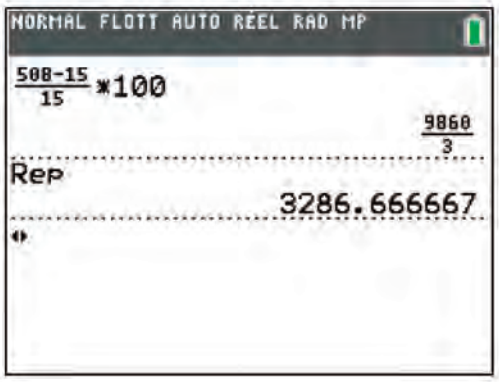
Pour cette question on se propose de calculer le taux d'évolution de la puissance solaire photovoltaïque entre 2008 et 2018.

Le calcul est donné par :

$$\frac{508 - 15}{15} \times 100$$

La calculatrice donne en priorité le résultat sous forme fractionnaire.

A l'aide de la touche « réponse » [2nde] [(−)] (rép 1), puis de la touche « toggle » [◆▶] on affiche également le résultat sous forme approchée sur le même écran.



Suite géométrique

2. a) Suite géométrique

Augmenter une valeur de 30% revient à la multiplier par 1,3.

La suite (P_n) est donc ainsi définie par récurrence par :

$$\begin{cases} P_0 = 508 \\ P_{n+1} = 1{,}3 \times P_n \end{cases}, \quad \text{pour tout entier naturel } n$$

Il s'agit d'une suite géométrique, de raison $q = 1{,}3 > 0$.

On peut dès lors utiliser la calculatrice pour :

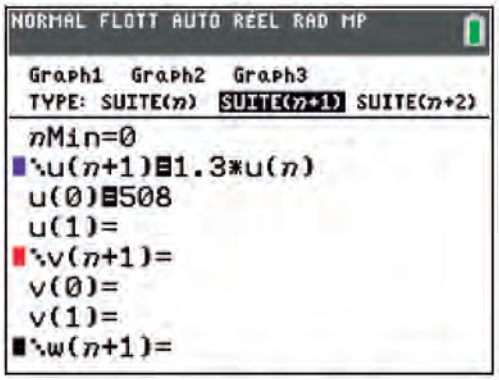
- Définir la suite (P_n) ;
- Visualiser les termes de la suite (P_n) dans son tableau de valeurs ;
- Représenter graphiquement le nuage de points de cette suite.

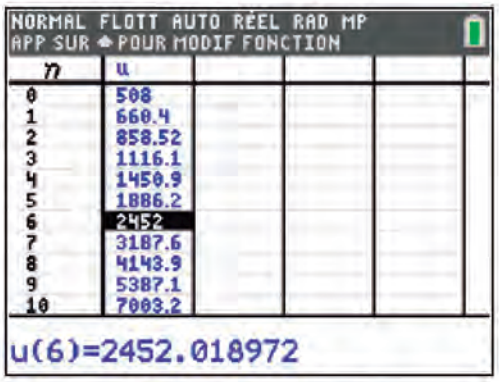
Vous pouvez vous référer à l'activité 04 - SUITE QUELCONQUE pour obtenir les détails de ces différentes étapes.

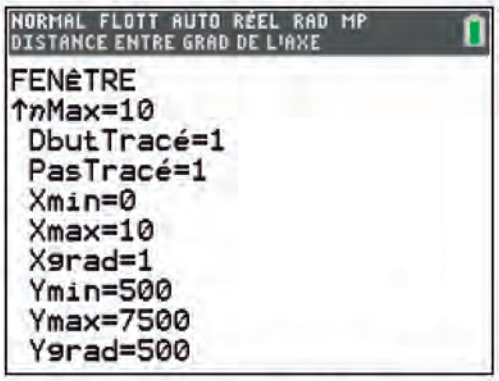
On notera notamment :

- L'utilisation du mode **SUITE(n+1)** lors de la définition de la suite ;
- Le mode **AUTO** du tableau de valeurs ;
- Les valeurs de la fenêtre graphique (**Xmin**, **Xmax**...etc.).

Vous trouverez ci-contre les captures d'écran nécessaires à la saisie des informations utiles.







2. b) Valeur de seuil

A l'aide du tableau de valeurs et/ou du nuage de points représentant la suite géométrique (P_n), il est aisé de confirmer les propos du chercheur.

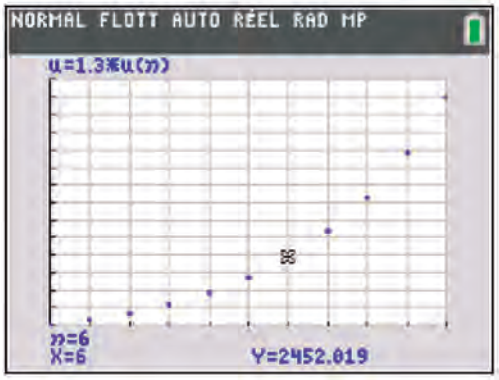
L'année 2024 correspond à l'année de rang $n = 6$. Il convient donc de déterminer le terme P_6 de notre suite géométrique.

On constate dans le tableau de valeurs que :

$$P_6 \approx 2452 \text{ gigawatts}$$

Cette valeur est bien supérieure aux 2400 gigawatts annoncés.

Il est également possible de vérifier cette information sur le nuage de points, à l'aide du mode [trace], comme sur la capture d'écran ci-contre.



Simuler le lancer de 3 dés

Enoncé

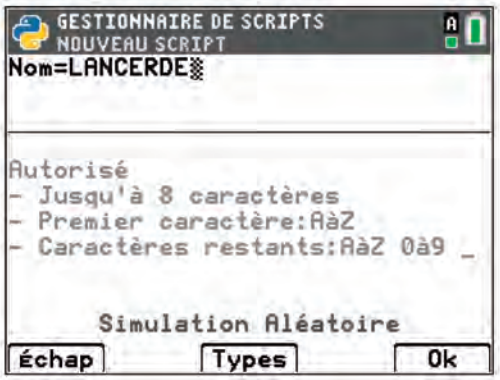
On lance 3 dés à 6 faces parfaitement équilibrés et on se demande quelle est la probabilité de réaliser au moins un 6.

1. A l'aide du langage Python, définir la fonction **Lancer3des** qui simule le lancer de 3 dés à 6 faces et renvoie 1 si au moins un des dés réalise un 6.
2. A l'aide du langage Python, définir la fonction **Frequence3des,** qui prend en paramètre le nombre n d'expériences réalisées et qui renvoie la fréquence f de réussite (obtenir au moins un 6). Donner la fréquence en pourcentage obtenue pour 1000 expériences.
3. Et en théorie, quel pourcentage devrait-on avoir ?

1. Définir la fonction Lancer3des

Il s'agit d'utiliser la notion de compteur en algorithmie pour différentes situations. Ici, si on obtient un 6 à l'aide de la fonction **randint(1,6)**, on augmentera notre variable **compteur** de 1.

On crée un script **LANCERDE** de type **Simulation Aléatoire**. (la librairie **random** sera automatiquement importée). On importe ensuite la librairie **math** à l'aide de [Fns…] , onglet **Modul** et enfin **math…**



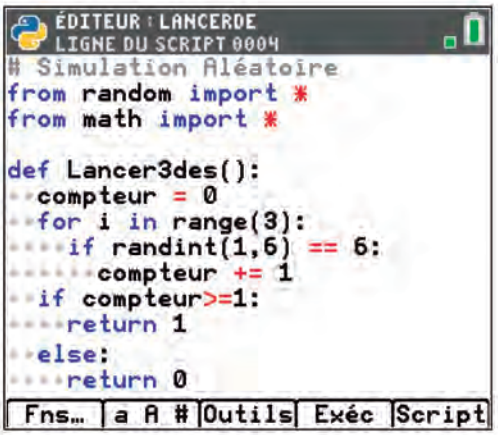
On définit ensuite la fonction **Lancer3des** selon le script ci-contre. On utilise la variable **compteur** (initialisée à 0) pour enregistrer le nombre de fois où l'on a obtenu un 6 à l'aide de la fonction **randint(1,6)**.

L'instruction **for i in range(3)** permet de réaliser nos 3 lancers de dés.

On a préféré l'instruction **compteur+=1** à **compteur = compteur + 1**. La première semble mieux convenir aux élèves pour percevoir son utilité, à savoir augmenter la variable compteur de 1. Dans la deuxième instruction la présence de compteur de part et d'autre du signe égal pose parfois quelques soucis de compréhension car la variable de droite contient l'ancien contenu et la variable de gauche le nouveau alors que lors de la résolution d'équations, les variables sont identiques dans leur contenu.

A la fin de la boucle **for**, si **compteur** est supérieur ou égal à 1, la fonction **Lancer3des** renvoie 1 sinon elle renvoie 0. On aurait pu choisir une typologie différente pour le retour de la fonction comme **True** ou **False** mais l'objectif étant par la suite de comptabiliser les réussites pour en établir les fréquences, il est plus facile de choisir dès maintenant un retour de type nombre entier.

On mettra à profit pour la saisie du script l'ensemble des menus de la calculatrice qui facilitent la saisie et l'indentation du code.



Simuler le lancer de 3 dés

2. Définir la fonction Frequence3des

Il s'agit de poursuivre l'utilisation de la notion de compteur pour évaluer une probabilité en répétant un grand nombre de fois l'expérience « lancer 3 dés » et observer les cas où l'on obtient au moins une fois le chiffre 6.

On va donc réutiliser la fonction **Lancer3des** précédente et compléter notre script avec la fonction **Frequence3des** qui prendra en paramètre n le nombre de répétitions souhaitées.

Lorsqu'on est en mode console et que l'on souhaite de nouveau modifier notre script, il suffit de sélectionner l'onglet **Editer** à l'aide de la touche [zoom] de la calculatrice.

On complète alors notre script avec la fonction **Frequence3des**. Vous remarquerez qu'en allant à la ligne, en repartant de la fonction précédente, l'indentation est conservée (marquée par les points gris). Il faut donc les supprimer à l'aide de la touche [suppr] pour faire comprendre à l'environnement Python que l'on souhaite démarrer une nouvelle fonction. Une fois l'opération réalisée, vous pouvez saisir comme d'habitude votre fonction.

L'idée consiste à comptabiliser les résultats (0 ou 1) de **Lancer3des** dans la variable **compteur** et ce n fois. On retourne la fréquence des réussites.

Dans la console, des appels successifs à la fonction **Frequence3des** pour notamment 1000 puis 100 000 lancers laissent penser que la probabilité de faire au moins un 6 en lancant 3 dés se situe aux alentours de 42%

```
PYTHON SHELL
>>> from LANCERDE import *
>>> Lancer3des()
1
>>>
>>> Lancer3des()
0
>>> Lancer3des()
1
>>> Lancer3des()
1
>>> |
Fns… | a A # | Outils | éditer | Script
```

```
ÉDITEUR : LANCERDE
LIGNE DU SCRIPT 0019
def Frequence3des(n):
··compteur = 0
··for i in range(n):
····compteur += Lancer3des()
··return compteur/n
Fns… | a A # | Outils | Exéc | Script
```

```
PYTHON SHELL
>>> # L'exécution de LANCERDE
>>> from LANCERDE import *
>>> Frequence3des(10)
0.4
>>> Frequence3des(100)
0.42
>>> Frequence3des(1000)
0.413
>>> Frequence3des(10000)
0.4246
>>> |
Fns… | a A # | Outils | éditer | Script
```

```
PYTHON SHELL
>>> Frequence3des(100000)
0.42165
>>> Frequence3des(100000)
0.42076
>>> Frequence3des(100000)
0.42166
>>> Frequence3des(100000)
0.42237
>>> Frequence3des(100000)
0.4218
>>> |
Fns… | a A # | Outils | éditer | Script
```

3. Et en théorie ?

Lors du lancer de 3 dés à 6 faces parfaitement équilibrés, on note A l'événement « obtenir au moins un 6 ».

Ainsi $\bar{A}$ est l'événement « ne pas obtenir de 6 ».

Les 3 lancers sont indépendants on a $p(\bar{A}) = \left(\frac{5}{6}\right)^3$.

On obtient donc $p(A) = 1 - p(\bar{A}) = 1 - \left(\frac{5}{6}\right)^3 = \frac{91}{216} \approx 42\%$.

```
NORMAL FLOTT AUTO RÉEL RAD MP
1-(5/6)^3
                          91/216
91/216
                    0.4212962963
```

Intérêts composés

Enoncé

On place une somme d'argent initiale sur un livret d'épargne qui rapporte 2,5 % par an.

1. On note (C_n) la suite, définie pour tout $n \in \mathbb{N}$, représentant le capital obtenu au bout de n années. On fixe $C_0 = 1500€$. Quelle est l'expression et la nature de cette suite ?
2. Au bout de 10 ans, quelle sera la valeur du capital si on ne retire ou n'ajoute aucune somme supplémentaire ? On souhaite réaliser un script en Python pour répondre à cette question qui consistera à définir la fonction **C** qui renverra le capital acquis au bout de n années où n est passé en paramètre.
3. On souhaite savoir au bout de combien de temps le capital acquis dépassera le montant de 2500 €. Définir la fonction **Annee** qui renvoie le nombre d'années nécessaire pour dépasser le capital c passé en paramètre.

1. Définir la suite (C_n)

Le capital va être augmenté chaque année de 2,5%. Exprimons d'abord le coefficient multiplicateur correspondant à cette augmentation.

$CM = 1 + \frac{2,5}{100} = 1,025$

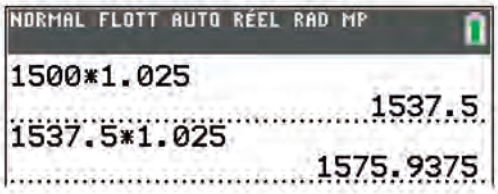
Ainsi à l'issue de la première année :

$C_1 = C_0 \times CM = C_0 \times 1,025 = 1537,50\,€.$

A l'issue de la deuxième année :

$C_2 = C_1 \times CM = C_1 \times 1,025 = 1575,9375\,€.$

Et ainsi de suite pour les années suivantes. Chacun des termes successifs de la suite est obtenu en multipliant le terme précédent par CM. Ainsi (C_n) est une suite géométrique de raison 1,025 et de premier terme $C_0 = 1500$.



2. Définir la suite (C_n) en Python

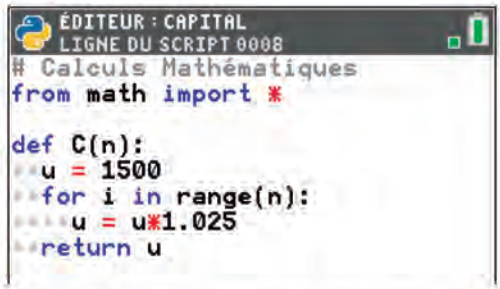
On crée le script **Capital** de type **Calculs Mathématiques**, à l'intérieur duquel on va définir la fonction **C** de paramètre **n** et qui renverra le terme C_n définie à la question précédente.

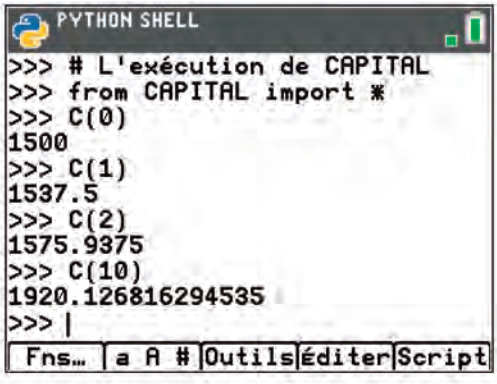
Pour cela on va utiliser, ce qu'on appelle en algorithmie, un accumulateur multiplicatif jouer par la variable **u**.

A la fin de l'exécution de la boucle, la fonction **C** renvoie la valeur accumulée dans **u**.

L'emploi de la boucle **for** est adapté à notre besoin puisque dans cette question, on connait le nombre d'itération que l'on souhaite réaliser. Ce nombre nous est donné par le paramètre **n**.

En console, on retrouve les résultats précédents et on obtient un capital d'environ 1920 € au bout de la 10 ième année.





Intérêts composés

2. Définir **Annee** en Python

On complète le script `Capital` avec la fonction `Annee` de paramètre `c` et qui renvoie le nombre d'année nécessaire pour que la suite dépasse `c`.

Cette fois-ci, on va utiliser une structure de boucle de type `while` puisqu'on ne connait pas à l'avance le nombre d'itération. On connait en revanche le seuil à dépasser puisqu'il nous est indiqué en paramètre.

Une fois la fonction saisie, on vérifie en console que le seuil de 1500€ est dépassé dès $n = 0$ puisqu'il s'agit de la valeur initiale de notre suite.

Il faut une année pour dépasser le seuil de 1530€.

Il faut dix années pour dépasser le seuil de 1920€.

Enfin, le seuil de 2500€ sera dépassé au bout de 21 années.

Un prolongement peut être proposé. En réalité, la fonction `c` peut être assez facilement généralisée pour le calcul de n'importe quel terme d'une suite géométrique.

Pour cela, il suffit de passer en paramètre, en plus du rang souhaité, le premier terme `u0` de la suite géométrique ainsi que sa raison `q`.

Appelons `G` notre nouvelle fonction de paramètre `u0`, `q` et `n`.

De la même façon, il suffit de modifier `Annee` en ajoutant les paramètres `u0` et `q` pour la transformer en une fonction de seuil « universelle » pour suite géométrique.

Appelons `Seuil` notre nouvelle fonction de paramètre `u0`, `q` et `c`.

Notre script est désormais enrichi des fonctions ci-contre. Testons les en console.

Ces nouvelles fonctions permettent de tester immédiatement de nouveaux paramètres au problème.

Ainsi `G(2000,1.03,10)` permet de voir qu'au bout de 10 ans, 2000€ placés à 3% l'an garantissent un capital acquis d'environ 2688€.

On retrouve bien qu'il faudra 21 années pour dépasser le seuil de 2500€ si l'on place 1500€ à 2,5% d'intérêts à l'aide de la commande `Seuil(1500,1.025,2500)`.

Tandis que la commande `Seuil(500,1.025,2500)` nous apprend qu'il faudra 66 années pour dépasser ce seuil de 2500€.

De même si on place 2000€ à 1% l'an, le capital sera de 2209€ au bout de 10 ans et si l'on place 1000€ à 2,5% pendant 10 ans également, le capital sera de 1280€.

ÉDITEUR : CAPITAL
LIGNE DU SCRIPT 0014

```
def C(n):
  u = 1500
  for i in range(n):
    u = u*1.025
  return u

def Annee(c):
  n = 0
  while C(n)<c:
    n += 1
  return n
```

Fns… | a A # | Outils | Exéc | Script

PYTHON SHELL

```
>>> # L'exécution de CAPITAL
>>> from CAPITAL import *
>>> Annee(1500)
0
>>> Annee(1530)
1
>>> Annee(1920)
10
>>> Annee(2500)
21
>>> |
```

Fns… | a A # | Outils | Éditer | Script

ÉDITEUR : CAPITAL
LIGNE DU SCRIPT 0026

```
def G(u0,q,n):
  u = u0
  for i in range(n):
    u = u*q
  return u

def Seuil(u0,q,c):
  n = 0
  while G(u0,q,n)<c:
    n += 1
  return n
```

Fns… | a A # | Outils | Exéc | Script

PYTHON SHELL

```
>>> G(2000,1.03,10)
2687.832758688244
>>> Seuil(1500,1.025,2500)
21
>>> Seuil(500,1.025,2500)
66
>>> G(2000,1.01,10)
2209.244250822409
>>> G(1000,1.025,10)
1280.084544196357
>>> |
```

Fns… | a A # | Outils | Éditer | Script

Générer des listes pour le 421

Enoncé

On lance 3 dés à 6 faces parfaitement équilibrés et on se pose la question de la probabilité de réaliser un 421.

1. A l'aide du langage Python, définir la fonction **Lancer3des** qui simule le lancer de 3 dés à 6 faces et renvoie sous la forme d'un entier à 3 chiffres le résultat du lancer sans classement. Définir ensuite la fonction **ListeLancer** qui renvoie la liste des résultats obtenus pour les n lancers où n est passé en paramètre.
2. A l'aide du langage Python, définir la fonction **Frequence421** qui renvoie la fréquence f de réussite de l'expérience (obtenir un 421 au premier lancer de 3 dés). Cette fonction prend en paramètre la liste des expériences réalisées. Donner la fréquence en pourcentage obtenue pour 1000 expériences.
3. Et en théorie, quel pourcentage devrait-on avoir ?

1. Définir la fonction **Lancer3des**

L'exercice peut sembler proche dans sa forme de celui proposé sur le lancer de 3 dés où l'on s'interroge sur la probabilité de faire au moins un 6. Ici, on souhaite conserver la mémoire des différents résultats obtenus à chaque lancer et on va voir comment les listes en Python vont nous y aider. Cela nous permettra d'aborder différentes modalités de construction des listes.

On crée un script **LNCER421** de type **Simulation Aléatoire**. On importe ensuite la librairie **math** à l'aide de l'onglet [Fns…] puis Menu **Modul** et enfin **math…**(la librairie **random** est déjà importée).

On définit ensuite la fonction **Lancer3des** selon le script ci-contre. On utilise la fonction **randint(1,6)** trois fois pour générer trois nombres aléatoires entre 1 et 6 qui, combinés par multiplication de multiple de dix, renvoie un nombre entre 111 et 666. Dans notre simulation, 123 est un résultat différent de 321. 123 signifie que le dé n°1 a renvoyé le chiffre 1 tandis que 321 signifie que le dé 1 a renvoyé le chiffre 3.

Intéressons nous à la définition de la fonction **ListeLancer**. Nous allons faire 3 propositions de rédaction pour un même résultat mais par des techniques de définition de listes différentes.

Commençons par saisir **ListeLancer1**. Il s'agit d'initialiser la liste par une liste de la taille du nombre **n** de lancers qui vont être réalisés et où chaque élément vaut 0. Ils seront ensuite remplacés par les **n** résultats obtenus à l'aide de la fonction **Lancer3des** précédente.

On teste alors la fonction **ListeLancer1** en console et il est intéressant de voir que nous sommes limités pour le nombre de lancers par la taille maximale d'éléments que peut contenir une liste. A mettre en regard avec un exercice précédent de lancers de dés où nous utilisions des compteurs et des accumulateurs pour notre travail et parvenions à réaliser 100 000 lancers.

Il existe d'autres façons de construire des listes en Python.

En particulier, on ne connait pas toujours à l'avance le nombre d'éléments qu'occupera notre liste à la fin de l'algorithme.

ÉDITEUR : LNCER421
LIGNE DU SCRIPT 0006

```
# Simulation Aléatoire
from random import *
from math import *

def Lancer3des():
  return randint(1,6)*100+randin
      t(1,6)*10+randint(1,6)
```

PYTHON SHELL

```
>>> # L'exécution de LNCER421
>>> from LNCER421 import *
>>> Lancer3des()
664
>>> Lancer3des()
134
>>> Lancer3des()
635
>>> Lancer3des()
255
>>>
```

Fns… | a A # | Outils | Éditer | Script

```
def ListeLancer1(n):
  l = [0]*n
  for i in range(n):
    l[i] = Lancer3des()
  return l
```

PYTHON SHELL

```
>>> # Shell Reinitialized
>>> # L'exécution de LNCER421
>>> from LNCER421 import *
>>> ListeLancer1(10)
[366, 442, 114, 121, 143, 625, 3
26, 553, 514, 653]
>>> ListeLancer1(10)
[641, 623, 665, 352, 333, 443, 2
25, 434, 415, 522]
>>>
```

Fns… | a A # | Outils | Éditer | Script

Générer des listes pour le 421

Aussi, saisissons maintenant `ListeLancer2`.

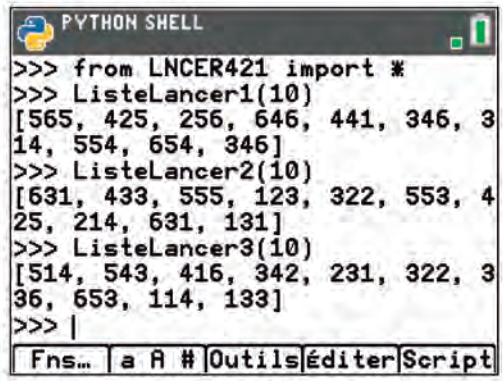
Il s'agit d'initialiser la liste avec la liste vide `[]`.

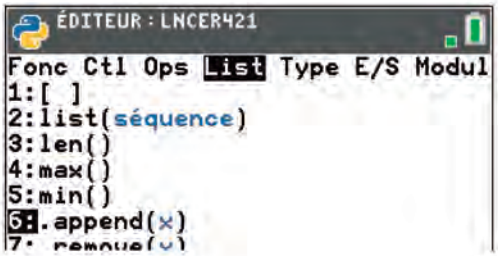
Cette fois-ci, chaque résultat obtenu avec la fonction `Lancer3des` est ajouté à la liste dont la taille grandit au fur et à mesure de la réalisation de l'algorithme.

Ces ajouts successifs d'éléments à la liste sont possibles à l'aide de la méthode **append** qui ajoute l'élément passé en paramètre à la liste depuis laquelle elle est appelée `l.append(Lancer3des())`.

Vous remarquerez dans le sous-menu `List` du menu [Fns…] le nombre de méthodes et fonctions disponibles pour travailler avec les listes. On en reparlera par la suite.

```
def ListeLancer2(n):
    l = []
    for i in range(n):
        l.append(Lancer3des())
    return l
```





Enfin, on souhaite présenter une dernière façon de générer une liste. Observer le code de la fonction `ListeLancer3` ci-contre.

Elle permet d'obtenir le même résultat que les fonctions précédentes en une seule ligne de code. On définit directement à l'interieur de la liste la description des éléments souhaités et leur nombre.

```
def ListeLancer3(n):
    return [Lancer3des() for i in
        range(n)]
```

2. Définir la fonction `Frequence421`

Il faut maintenant trier les résultats.

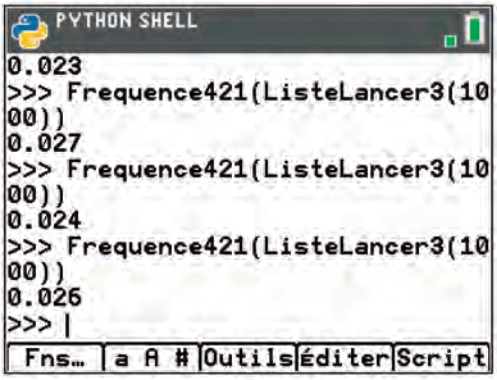
A l'aide de la méthode `count`, nous allons comptabiliser les occurrences de 124,142,214,241,412 et 421 dans la liste `l` passée en paramètre.

Nous utilisons un compteur pour sommer les différentes apparitions favorables et retournons la fréquence en divisant la valeur de `compteur` par le nombre d'éléments de la liste à l'aide de la fonction `len`.

Nous testons plusieurs fois en console notre fonction à l'aide de la commande `Frequence421(ListeLancer3(1000))`.

La fréquence obtenue fluctue et semble se situer aux alentours des 2,5 % dans notre courte série de tests.

```
def Frequence421(l):
    resultat = [124,142,214,241,41
        2,421]
    compteur = 0
    for i in resultat:
        compteur += l.count(i)
    return compteur/len(l)
```



3. Et en théorie ?

A l'aide du dénombrement, on peut établir qu'il y a 6 possibilités qui nous sont favorables sur un total de $6 \times 6 \times 6 = 216$ possibilités. Ainsi, si on appelle P_{421} la probabilité de faire un 421 au premier lancer, on a :

$$P_{421} = \frac{6}{216} \approx 2{,}8\%.$$

Ce qui est proche des résultats de nos simulations précédentes.

Simulation de la somme de 2 dés

Enoncé

On lance 2 dés à 6 faces parfaitement équilibrés, on observe la somme des deux faces obtenues et on se pose la question de parier sur une somme valant 6 ou plutôt sur une somme valant 7.

1. Réaliser la simulation de cette expérience, un très grand nombre de fois (999 fois, nombre d'éléments maximal d'une liste pour la calculatrice) en utilisant les générateurs de listes et l'éditeur de listes.
2. Réaliser la simulation de cette expérience, un très grand nombre de fois en utilisant l'application **ProbSim** et l'éditeur de listes.

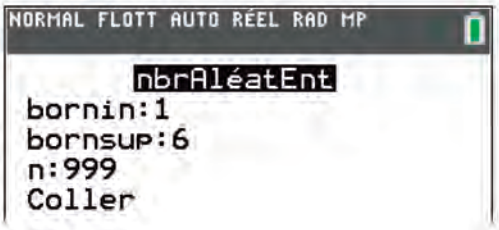
1. Simulation à l'aide des listes

Sans avoir recours à Python, il est possible de se servir de l'environnement de la calculatrice pour procéder à quelques simulations.

Pour étudier ce grand classique, nous allons utiliser la fonction **nbrAléatEnt** en appuyant sur la touche [math] puis dans le menu **PROB**. On accède alors à une fenêtre demandant de compléter les paramètres pour la génération.

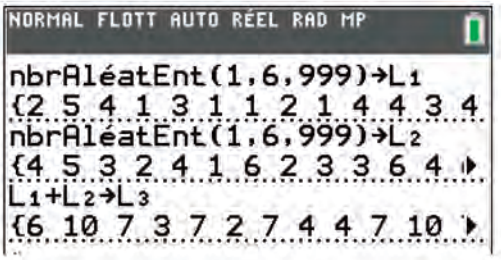
On va réaliser 999 tirages d'un dé à 6 faces et stocker l'ensemble des résultats dans la liste **L1**.

Pour cela, on complète de la manière ci-contre la fenêtre de saisie, puis on colle. Dans l'écran de calcul, avant de valider **nbrAléatEnt(1,6,999)**, on précise que l'on souhaite enregistrer ([sto→]) le résultat dans **L1**.



Attention, **L1** s'obtient par la combinaison de touche [2nde] [1] et non en saisissant L puis 1.

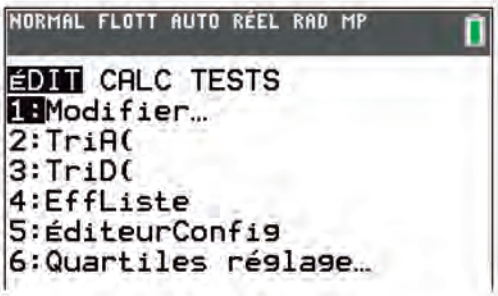
Une fois l'opération réalisée, on la répète en stockant cette fois-ci le résultat dans la liste **L2** ([2nde] [2]).

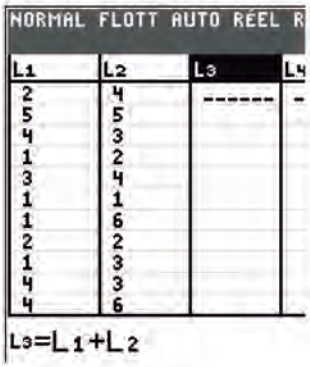


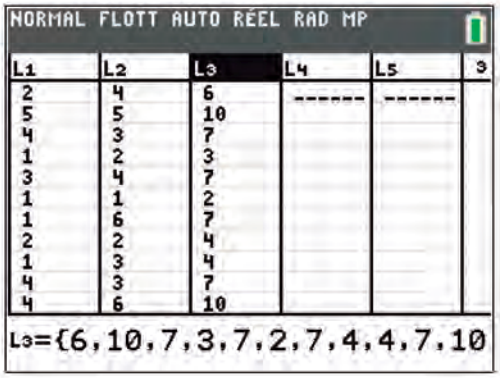
Il s'agit enfin de réaliser la somme des deux dés en la stockant dans la liste **L3**. Pour cela, plusieurs possibilités s'offrent à nous.

A partir de l'écran de calcul, en saisissant directement **L1 + L2 → L3**

Mais il est également possible de réaliser cette opération dans l'éditeur de liste de la calculatrice. On appuie sur la touche [stats] puis **Modifier…** On se place dans l'entête de la colonne **L3**, on saisit **L1+L2** puis on valide.







Simulation de la somme de 2 dés

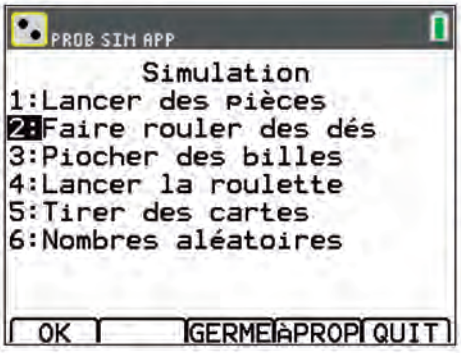
2. Simulation à l'aide de ProbSim

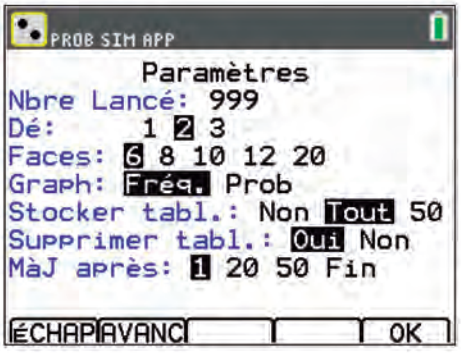
Dans ce cas particulier d'exercice, il est possible d'avoir recours à une application embarquée dans la calculatrice qui se charge, une fois configurée, de réaliser cette expérience.

On y accède en appuyant sur [2nde] [résol] et on sélectionne l'application **Prob Sim** (son rang dans le menu peut différer de la capture d'écran selon le modèle de votre calculatrice).

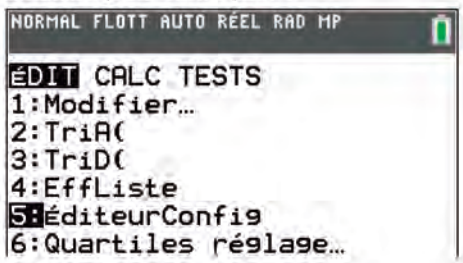
On choisit ensuite de faire rouler des dés. On configure notre simulateur dans l'onglet **Param**.

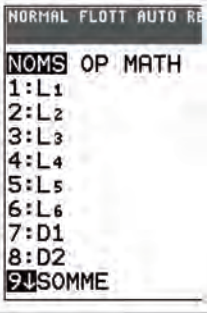
On peut choisir de réaliser 100 lancers plusieurs fois. Nous avons choisi de réaliser les 999 lancers en une seule fois avec une actualisation de la représentation graphique après chaque lancers de deux dés. L'application réalise automatiquement la somme des deux dés. Après avoir confirmé la modification des paramètres, on lance la simulation avec l'onglet **ROULE** et on peut observer la réalisation du diagramme en bâton représentant les différentes sommes obtenues avec leurs effectifs respectifs.

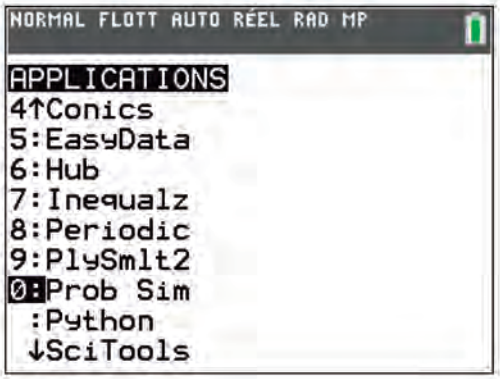


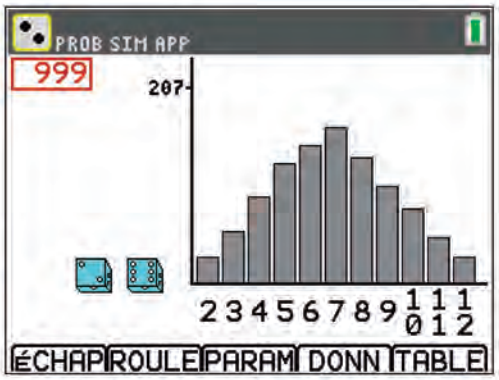


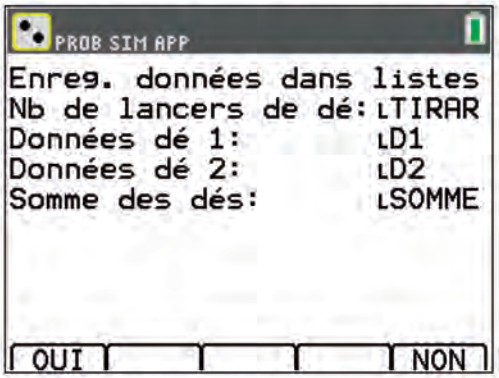
L'application offre la possibilité de sauvegarder dans des listes les lancers réalisés. Nous les avons limités à 999 car c'est le nombre maximum d'éléments que peut contenir une liste dans la calculatrice. Pour réaliser cette opération, on va dans l'onglet **DONN** puis **OUI**. Les données sont désormais disponibles pour être exploitées dans l'éditeur de listes de la calculatrice dans les listes **LD1**, **LD2** et **LSOMME**. Pour cela nous allons configurer notre éditeur qui, par défaut travaille avec **L1**, **L2** etc. Après avoir quitté **ProbSim**, nous utilisons la commande **EditeurConfig** accessible via la touche [stats] pour saisir **EditeurConfig LSOMME**. (Rappel : [2nde] [stats] pour accéder aux listes disponibles). Désormais notre éditeur ne contient plus que la liste **LSOMME**. Serions-nous capables de réaliser le diagramme en bâton de l'application **ProbSim** pour prendre une décision pour notre pari ?

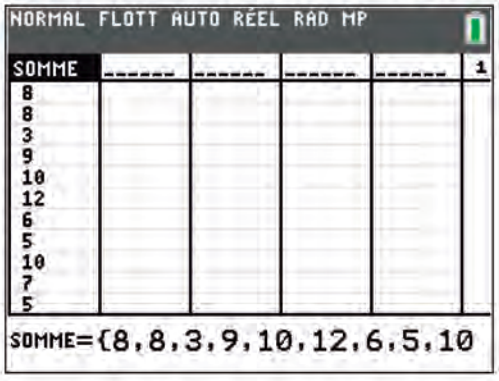












Représentation de la somme de 2 dés

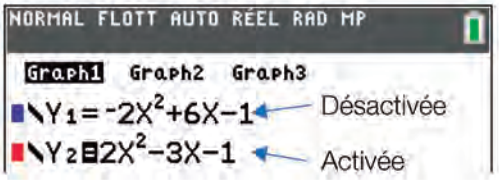
Enoncé

On lance 2 dés à 6 faces parfaitement équilibrés, on observe la somme des deux faces obtenues et on se pose la question de parier sur une somme valant 6 ou plutôt sur une somme valant 7. La simulation a été réalisée et nous voulons maintenant prendre une décision.

1. La représentation sous forme de nuage de points des différentes sommes permet-elle de prendre une décision ?
2. La représentation sous forme de boite à moustache permet-elle de prendre une décision ?
3. La représentation sous forme de diagramme en bâtons permet-elle de prendre une décision ?

0. Preambule

Il s'agit dans un premier temps de désactiver l'affichage de la représentation graphique des fonctions. On appuie sur la touche [f(x)] et on vérifie que le signe = n'est pas sélectionné pour chaque fonction définie. S'il l'est, on déplace le curseur dessus et on appuie sur la touche [entrer] . La fonction existe toujours mais elle n'est plus représentée lorsqu'on se rend dans l'écran de représentation graphique à l'aide de la touche [graphe] .

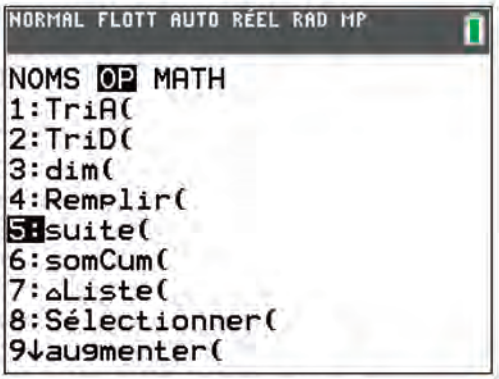


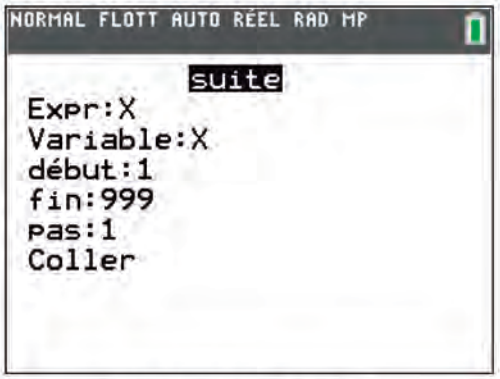
Maintenant, on va pouvoir paramétrer nos différentes représentations graphiques de nos données statistiques.

1. Nuage de points

On suppose que nos sommes successives sont stockées dans la liste L2. Si vos données sont contenues dans la liste LSOMME, issue de l'application ProbSim, vous pouvez recopier vos données, si nécessaire, à l'aide de la commande LSOMME → L2.

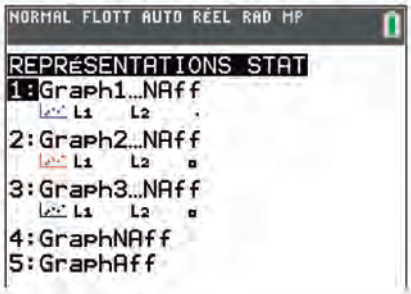
Pour réaliser notre représentation sous la forme d'un nuage de points, nous avons besoin d'une liste qui va contenir les numéros de lancers respectifs, c'est-à-dire la liste des entiers de 1 à 999 puisque nous avions réalisé 999 tirages. Nous choisissons d'utiliser la fonction SUITE accessible à l'aide des touches [2nde] [stats] onglet OP.

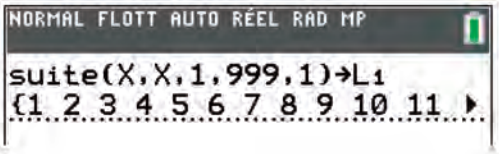




On complète la fenêtre de la manière indiquée dans la capture, on colle et stocke le résultat dans L1.
On se rend ensuite dans l'éditeur de représentation statistiques à l'aide des touches [2nde] [f(x)] .

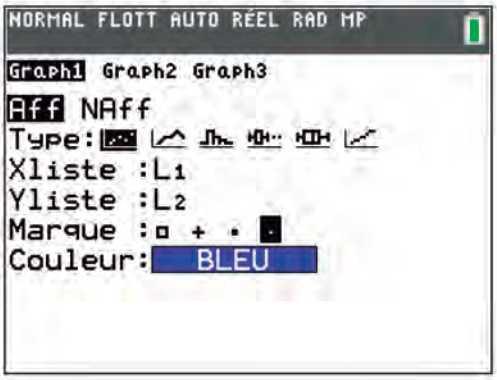
Nous allons travailler avec Graph1.

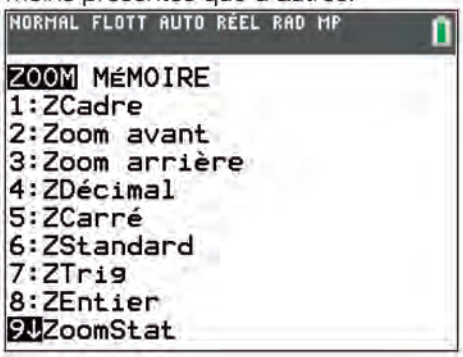


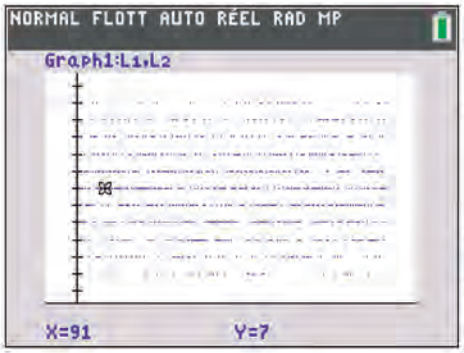


Représentation de la somme de 2 dés

Par défaut, l'éditeur est configuré pour représenter le nuage de points où les abscisses sont contenues dans **L1** et les ordonnées respectives sont dans **L2**. Nous n'avons rien à changer et pouvons activer l'option **Aff** pour afficher le nuage de points. Préalablement on utilise la fonction **ZoomStat** accessible via la touche [zoom] pour fenêtrer correctement notre affichage. Cette représentation ne semble, cependant, pas la plus indiquée pour nous convaincre dans notre décision , même si l'on constate visuellement que certaines valeurs semblent moins présentes que d'autres.

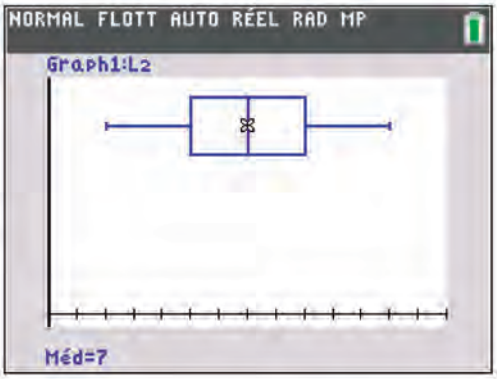






2. Boite à moustache

On retourne dans le menu de configuration de notre représentation statistique **Graph1** et sélectionnons cette fois-ci la boite à moustache n°2, qui est la représentation habituelle (non modifiée). On fera attention à bien paramétrer la liste **L2** qui comprend nos données à représenter. Par défaut, c'est **L1** qui est proposée. En utilisant la fonction **trace** (touche [trace]), on peut remarquer que Q1, la médiane et Q3 sont assez proches, valant respectivement 5, 7 et 9. C'est une piste de réflexion mais cette représentation ne semble pas, malgré tout, la plus indiquée pour nous convaincre dans notre décision entre parier sur 6 ou parier sur 7. Poursuivons.

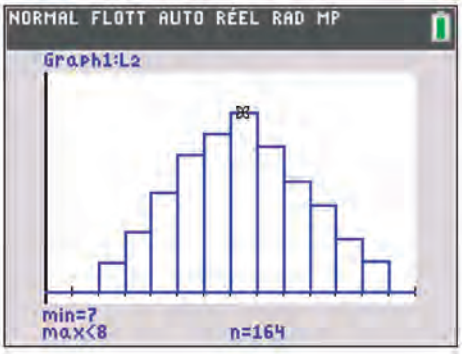


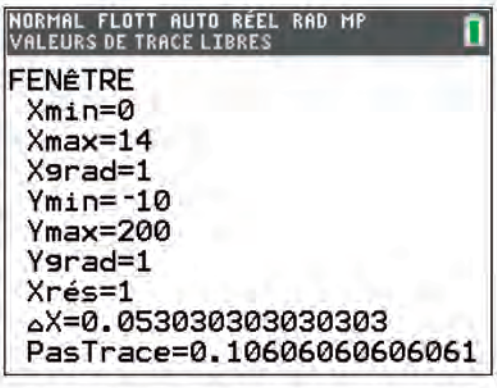
3. Diagramme en bâtons

On retourne une dernière fois dans le menu de configuration de **Graph1** et sélectionnons la représentation sous forme de diagramme en bâtons.

Connaissant notre plage de données, il peut être plus intéressant de fenêtrer directement notre représentation graphique avant d'afficher la représentation.

La touche [trace] , nous permet de nous convaincre, quitte à recommencer la simulation plusieurs fois des 999 lancers, que nous aurions davantage intérêt à miser sur le 7.





Comparaison de deux populations

Enoncé

On effectue un sondage dans deux populations différentes où chacun est interrogé sur son poids.

On obtient les résultats suivants :

	65 kg	70 kg	75 kg	80 kg	85 kg	90 kg	95 kg	100 kg
Population 1	18	80	90	95	84	63	15	10
Population 2	10	85	105	116	75	51	10	3

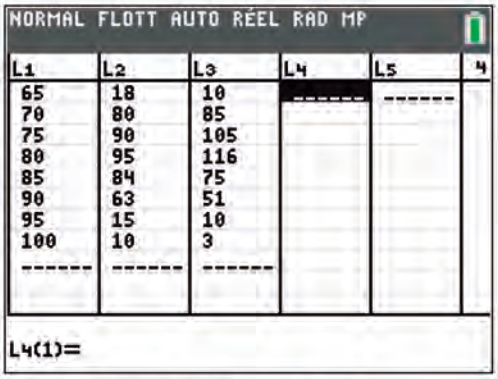
1. Etablir pour les deux populations, l'ensemble des indicateurs statistiques (Moyenne, Quartiles 1 et 3, Médiane, Ecart Type).
2. Etablir pour les deux populations, la représentation sous forme de boite à moustaches.

   Que penser de la répartition des poids dans les deux populations?

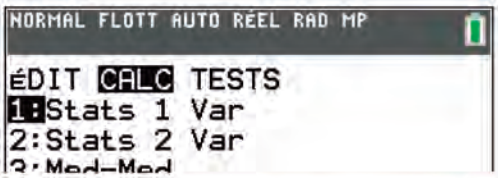
1. Indicateurs statistiques

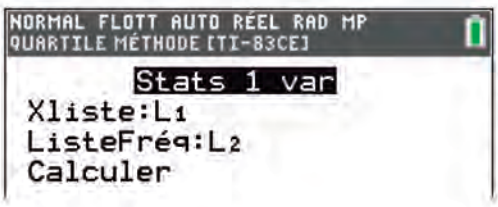
On commence par saisir dans l'éditeur de listes (stats puis **Modifier…**), l'ensemble des données de l'exercice.

On saisira dans **L1** les différentes valeurs observées, dans **L2** les effectifs de la population 1 et dans **L3**, les effectifs de la population 2.

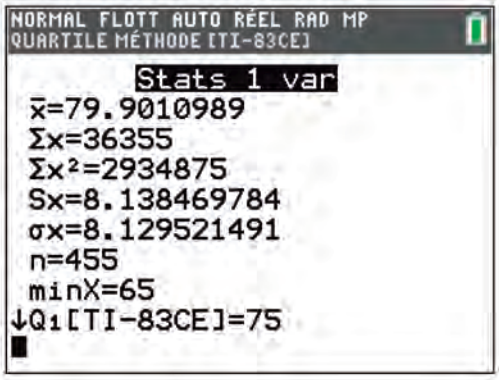


Pour établir l'ensemble des indicateurs statistiques, on va utiliser la fonctionnalité **Stats 1 var** de la calculatrice. Elle est accessible en appuyant sur la touche stats puis en allant dans le menu **CALC.**





On configure alors en spécifiant que les valeurs se trouvent dans la liste **L1** et les effectifs respectifs dans la liste **L2**.

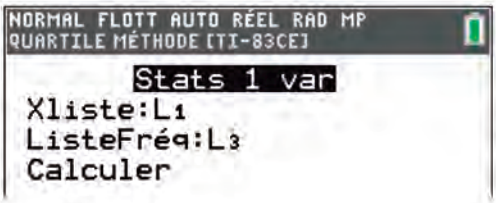


En lançant le calcul, on obtient les valeurs suivantes :

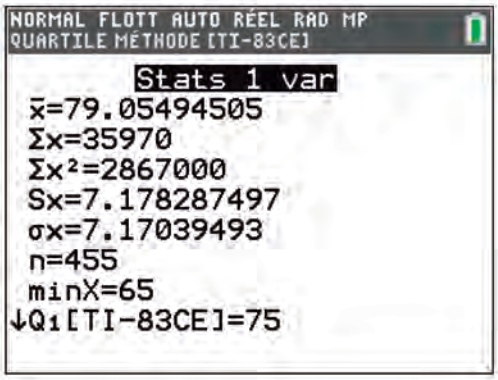
la moyenne $\bar{x}$ vaut environ 79,9 kg, l'écart type σx vaut environ 8,1 kg, Q_1 vaut 75 kg, la médiane 80 kg et enfin Q_3 vaut 85 kg.

On utilisera les flèches ⌄ ou ⌃ de la calculatrice pour faire défiler l'ensemble des indicateurs.

Comparaison de deux populations



On recommence l'opération, en configurant, cette fois-ci, les effectifs avec la liste L3 pour établir les indicateurs statistiques de la population 2.



En lançant le calcul, on obtient les valeurs suivantes :

la moyenne $\bar{x}$ vaut environ 79,1 kg, l'écart type σx vaut environ 7,1 kg, Q_1 vaut 75 kg, la médiane 80 kg et enfin Q_3 vaut 85 kg.

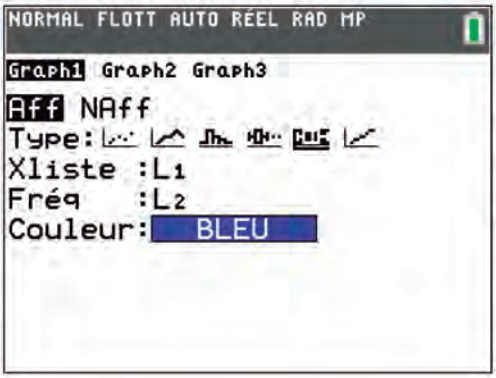
Les deux populations ont un effectif n de 455 individus.

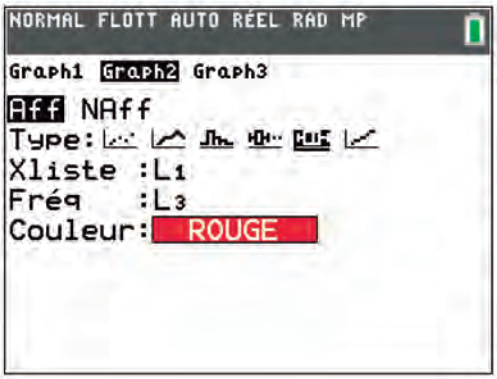
2. Représentation graphique

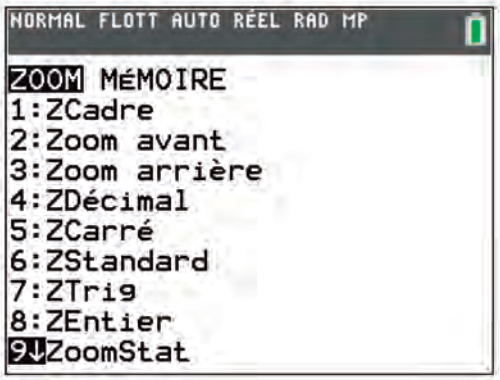
On configure nos deux représentations graphiques à l'aide de la combinaison de touches [2nde] [f(x)] .

La population 1 sera représentée en bleu et la population 2 sera représentée en rouge.

Une fois nos graphiques configurés, on fait appel à la fonction **ZoomStat** pour fenêtrer automatiquement la représentation de nos boites à moustaches.



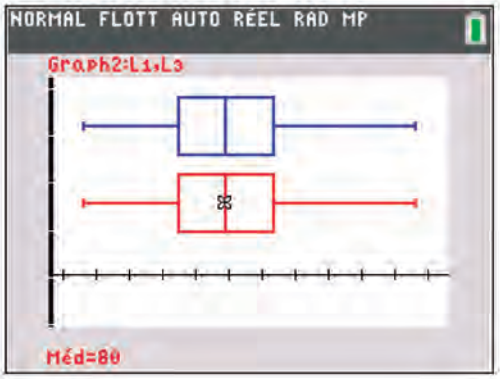




On obtient alors la représentation simultanée de nos deux boites à moustaches qui sont finalement identiques.

On retrouve bien, visuellement, les données obtenues précédemment concernant la répartition des poids de ces deux populations, à savoir que Q_1, Q_3 et la médiane sont identiques.

Pour un même effectif de population, on constate une légère différence au niveau de la moyenne, inférieure pour la population 1 avec un écart type également légèrement inférieur, ce qui laisse penser à une moindre dispersion autour de la moyenne.



Polynôme de degré 3

Enoncé

On considère le polynôme de degré 3 défini sur $\mathbb{R}$ suivant : $P(x) = (x+1)(x-2)(x-5)$

1. Résoudre l'équation $P(x) = 0$.
2. Compléter le tableau de signes suivant :

x	$-\infty$	-1	2	5	$+\infty$
$x+1$					
$x-5$					
$x-2$					
$P(x)$					

3. Compléter le tableau de variations suivant (arrondir à 10^{-2} si nécessaire) :

x	-2	$\cdots$	$\cdots$	6
$P(x)$	$\cdots$	$\cdots$	$\cdots$	$\cdots$

1. Détermination des racines

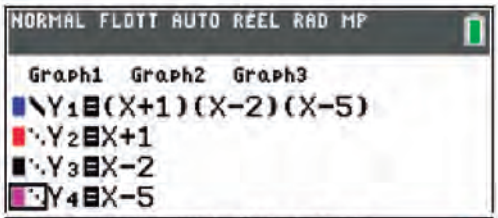
Un produit de facteurs est nul si et seulement si un de ses facteurs est nul ($A \times B = 0 \Leftrightarrow A = 0$ ou $B = 0$). D'où :

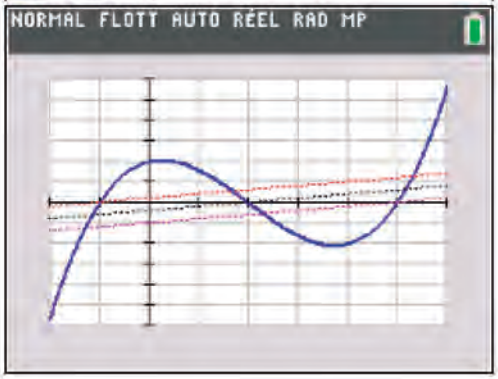
$$P(x) = 0 \Leftrightarrow x+1 = 0 \text{ ou } x-2 = 0 \text{ ou } x-5 = 0$$
$$\Leftrightarrow x = -1 \text{ ou } x = 2 \text{ ou } x = 5$$

2. Tableau de signes

Afin de nous aider à compléter ce tableau de signes, nous allons tracer la représentation graphique du polynôme P. De plus, pour chacun de ses facteurs, nous allons également tracer la représentation graphique des fonctions affines associées.

- Dans le menu [f(x)], on saisit l'expression de $P(X)$ dans Y1, puis celles de ses facteurs dans Y2, Y3 et Y4. En plaçant le curseur sur le rectangle de couleur et en appuyant sur [entrer], il est possible de modifier la couleur, ainsi que le style du tracé.
- Avant de tracer ces fonctions, on utilise le tableau de valeurs de ces dernières pour configurer la [fenêtre]. Pour une aide à ce sujet, consulter la fiche 04-Tracer et cadrer une fonction.





La lecture de ce graphique nous permet alors de compléter le tableau de signes avec précision :

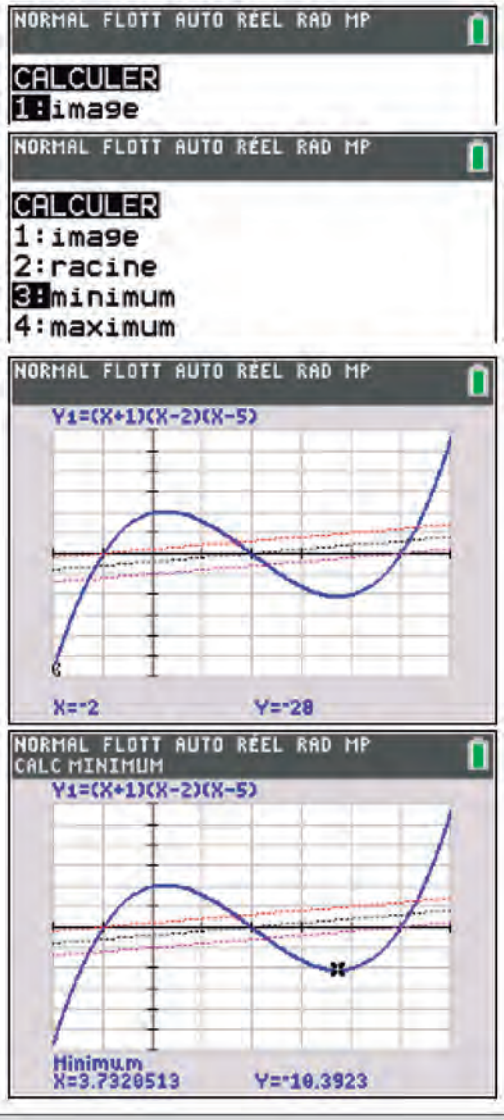
x	$-\infty$		-1		2		5		$+\infty$
$x+1$		$-$	0	$+$		$+$		$+$	
$x-5$		$-$		$-$		$-$	0	$+$	
$x-2$		$-$		$-$	0	$+$		$+$	
$P(x)$		$-$	0	$+$	0	$-$	0	$+$	

En cas de doute sur les racines de ces différentes fonctions, on peut toujours utiliser la commande **2:racine** du menu 2nde calculs f4 trace (vous pouvez consulter la ficher 05- Racines d'un trinôme du 2nd degré pour davantage de précisions).

3. Tableau de variations

Afin de compléter ce tableau de variations, nous allons procéder en 2 étapes distinctes :

- Tout d'abord, nous allons déterminer les images de -2 et de 6 par le polynôme P. A cette fin, dans le menu 2nde calculs f4 trace, on utilise la commande **1:image**. Une fois dans la fenêtre graphique, à l'aide du bandeau inférieur, on entre la valeur de notre antécédent, en commençant par -2. On obtient le résultat $P(-2) = -28$. On procède de même pour l'image de 6 par P pour obtenir $P(6) = 28$.
- Pour obtenir les coordonnées manquantes du tableau, dans le même menu que précédemment, on utilise les commandes **3:minimum** et **4:maximum**. A chaque fois, une fois dans la fenêtre graphique, on procède en 3 étapes :
  - On se place à gauche de notre extremum local, et on valide par entrer.
  - On se place ensuite à droite de notre extremum local, et on valide par entrer.
  - Enfin, on valide par entrer.



On obtient alors le tableau complet suivant :

x	-2		≈ 0.27		≈ 3.73		6
$P(x)$	-28	↗	≈ 10.39	↘	≈ -10.39	↗	28

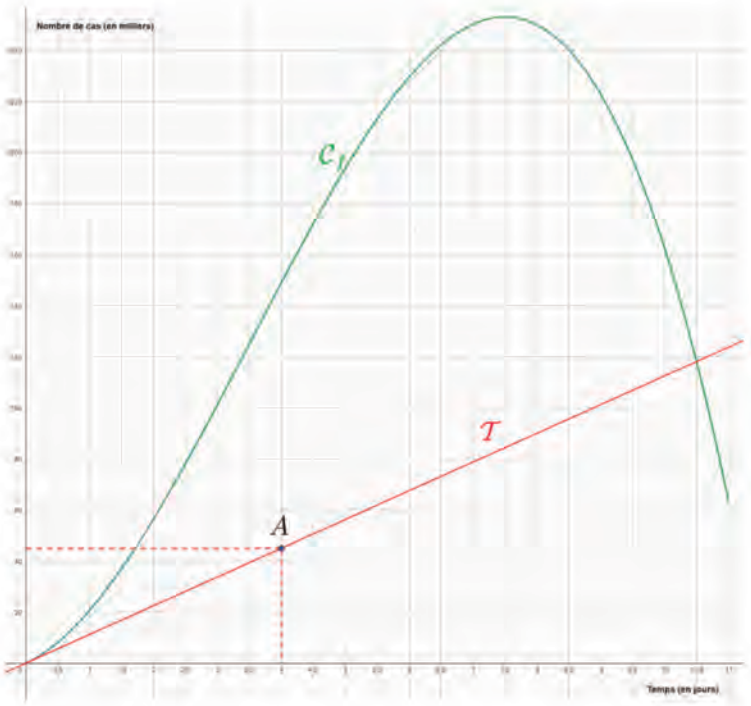
Nombre dérivé et tangente

Enoncé

Lors d'une épidémie observée sur une période de onze jours, un institut de veille sanitaire a étudié l'évolution du nombre de personnes malades. La durée, écoulée à partir du début de la période, est exprimée en jours. Elle est notée t. On modélise le nombre de cas grâce à la fonction f suivante, où $f(t)$ représente le nombre personnes malades, en milliers, à l'instant $t \in [0\,;11]$:

$$f(t) = -t^3 + \frac{21}{2}t^2 + \frac{45}{4}t$$

On donne ci-contre la courbe représentative $\mathcal{C}_f$ de la fonction f. La droite $\mathcal{T}$ est la tangente à la courbe $\mathcal{C}_f$ au point d'abscisse 0 et passe par le point A de coordonnées $(4\,;45)$.

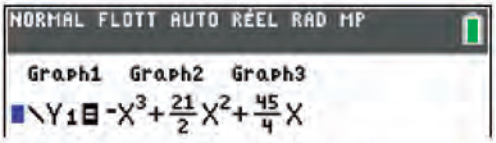


1. Déterminer graphiquement $f(0)$ et $f'(0)$.
2. Calculer $f'(t)$ pour tout t dans l'intervalle $[0\,;11]$.
3. En déduire l'équation réduite de la tangente $\mathcal{T}$, en justifiant tous les calculs.

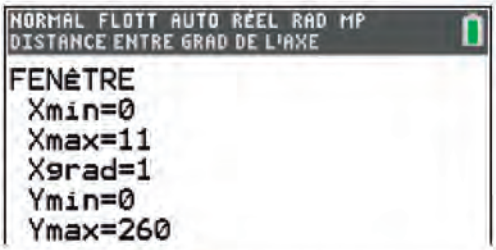
1. Image et nombre dérivé

Nous allons commencer par utiliser la calculatrice pour obtenir la courbe représentative $\mathcal{C}_f$.

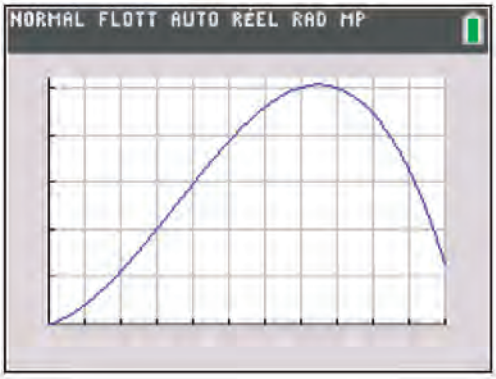
- Dans le menu [f(x)] , on saisit l'expression $f(x)$ dans `Y1`.



- Avant de passer au graphique, on peut utiliser celui de l'énoncé pour configurer aisément notre fenêtre graphique, en appuyant sur la touche [fenêtre]. La capture d'écran ci-contre vous permet de voir cette utilisation astucieuse de l'énoncé, pour gagner un temps précieux sur le cadrage de la courbe.



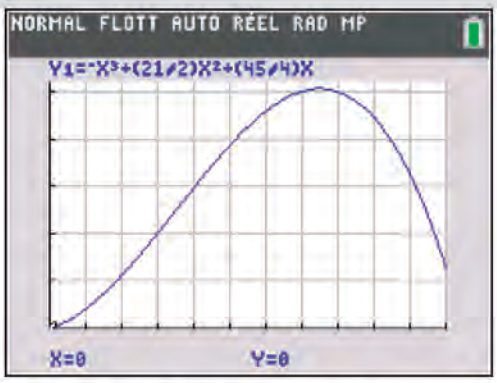
- Enfin, on trace la courbe $\mathcal{C}_f$ en appuyant sur la touche [graphe] . L'avantage de représenter cette courbe sur la calculatrice est que nous allons pouvoir vérifier les conjectures graphiques demandées dans cette 1ère question.



Nombre dérivé et tangente

- Commençons par $f(0)$. A l'aide des touches [2nde] + [trace] (calculs f4), on sélectionne la commande **1:image**. Puis on entre la valeur 0 dans le bandeau inférieur de la fenêtre graphique alors affichée. On valide par [entrer] et on lit l'image de 0 par la fonction f :

$$f(0) = 0$$

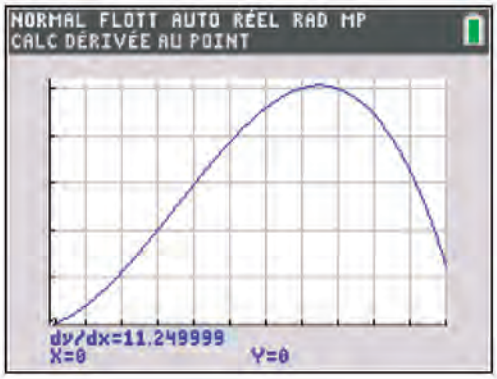


- Nous allons suivre le même procédé pour obtenir le nombre dérivé de f en $x = 0$.

  A l'aide des touches [2nde] [trace] (calculs f4), on sélectionne la commande **6:dy/dx**. Contrairement à la commande précédente, il n'y a pas de bandeau qui apparaît automatiquement en bas de la fenêtre graphique. Toutefois, en appuyant sur la touche 0, le bandeau s'affiche. On valide par [entrer] et on lit le nombre dérivé de f en $x = 0$.

  Attention : la valeur affichée par la calculatrice sera fréquemment une valeur approchée. Ici, la valeur est très certainement de **11.25**, c'est-à-dire $\frac{45}{4}$, ce qui doit nous rappeler les coordonnées du point A et donc nous aider à réaliser la conjecture demandée.

$$f'(0) = \frac{45}{4}$$



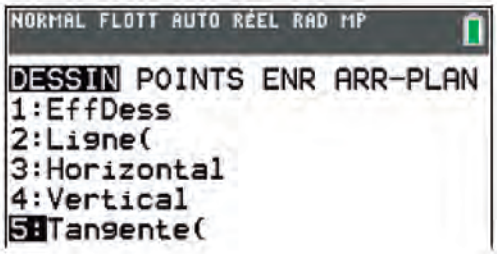
2. Fonction dérivée f'

Pour tout t dans l'intervalle $[0\,;11]$, $f'(t) = -3t^2 + \frac{21}{2} \times 2t + \frac{45}{4} = -3t^2 + 21t + \frac{45}{4}$

3. Equation de la tangente $\mathcal{T}$

Afin d'obtenir l'équation de la tangente à l'aide de la calculatrice, on utilise les touches [2nde] [prgm] (dessin C) pour entrer dans l'onglet **dessin**.

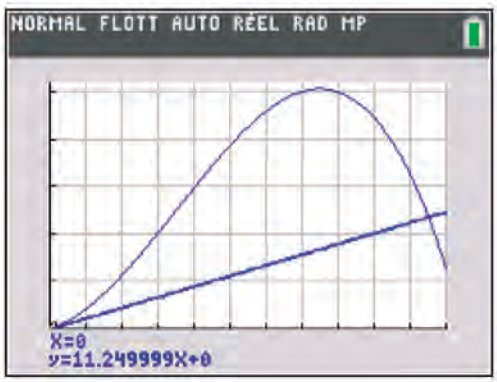
On sélectionne alors la commande **5:Tangente** (puis on valide avec [entrer]).



On se retrouve alors dans la fenêtre graphique. Il suffit alors d'entrer la valeur 0 (le bandeau **X=** s'affiche alors, comme pour le nombre dérivé) et de valider une dernière fois par [entrer]. La tangente $\mathcal{T}$ est alors tracée et son équation réduite apparaît dans la partie inférieure de l'écran.

Reste à valider la conjecture par les calculs :

- $f(0) = -0^3 + \frac{21}{2} \times 0^2 + \frac{45}{4} \times 0 = 0$
- $f'(0) = -3 \times 0^2 + 21 \times 0 + \frac{45}{4} = \frac{45}{4}$



Ainsi, l'équation réduite de $\mathcal{T}$ est :

$$\mathcal{T} : y = f'(0) \times (t - 0) + f(0)$$

C'est-à-dire :

$$\mathcal{T} : y = \frac{45}{4}t$$

Combien de 6 ?

Enoncé

On lance deux fois un dé à 6 faces, parfaitement équilibré.

Le jeu est le suivant :

- Si on obtient deux 6, on gagne 5 €
- Si on obtient un 6, on ne perd ou ne gagne rien.
- Si on obtient aucun 6, on perd 2 €

On définit la variable aléatoire X qui, à chaque issue du jeu, relève la somme gagnée ou perdue

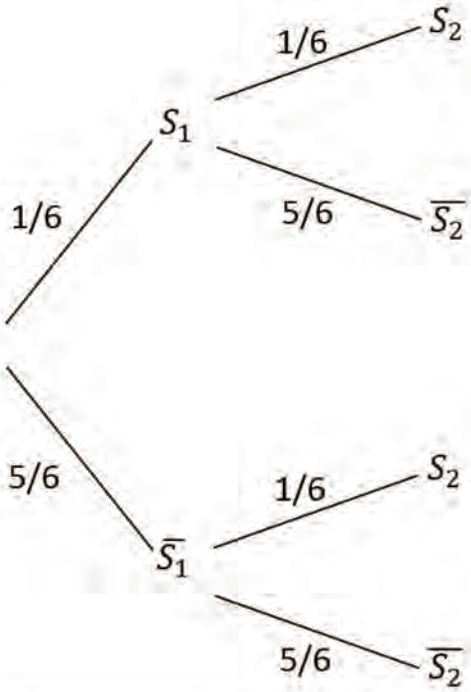
1. Après avoir déterminé les différentes issues possibles puis les différentes valeurs possibles, déterminer la loi de probabilité de la variable aléatoire X.
2. Calculer l'espérance de la variable aléatoire X. Que penser de nos chances de gagner de l'argent à ce jeu ?
3. Combien faudrait-il que le joueur perde, lorsqu'il n'obtient aucun 6, pour que l'espérance du jeu soit nulle ? Ou bien combien faudrait-il que le joueur gagne, lorsqu'il obtient deux 6, pour que l'espérance du jeu soit nulle ?

1. Loi de probabilité

Pour répondre à l'ensemble des questions, il est intéressant de dresser l'arbre de probabilité de l'expérience aléatoire.

On désigne par S_1 et S_2 les évènements obtenir un 6 respectivement au premier et deuxième tirage, puis $\overline{S_1}$ et $\overline{S_2}$ les évènements respectifs contraire.

On a $p(S_1) = p(S_2) = \frac{1}{6}$ et $p(\overline{S_1}) = p(\overline{S_2}) = \frac{5}{6}$.



Nous pouvons maintenant dresser la loi de probabilité de la variable aléatoire X.

A l'issue de l'expérience aléatoire, X peut prendre les valeurs 5, 0 ou -2

$$P(X = 5) = \frac{1}{6} \times \frac{1}{6} = \frac{1}{36}$$

$$P(X = 0) = \frac{1}{6} \times \frac{5}{6} + \frac{5}{6} \times \frac{1}{6} = \frac{5}{36} + \frac{5}{36} = \frac{10}{36} = \frac{5}{18}$$

$$P(X = -2) = \frac{5}{6} \times \frac{5}{6} = \frac{25}{36}$$

x_i	-2	0	5
$P(X = x_i)$	$\frac{25}{36}$	$\frac{5}{18}$	$\frac{1}{36}$

Combien de 6 ?

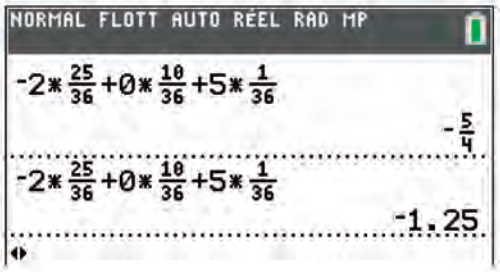
2. Espérance de la variable aléatoire

$$E(X) = -2 \times P(X = -2) + 0 \times P(X = 0) + 5 \times P(X = 5)$$

$$E(X) = -2 \times \frac{25}{36} + 0 \times \frac{5}{18} + 5 \times \frac{1}{36} = -\frac{5}{4}$$

On peut interpréter l'espérance de la manière suivante : si on joue un grand nombre de fois, en moyenne, on perdra 1,25€.

Le jeu nous est défavorable.



3. Espérance nulle

On appelle a la valeur prise par la variable aléatoire lorsque le joueur perd.

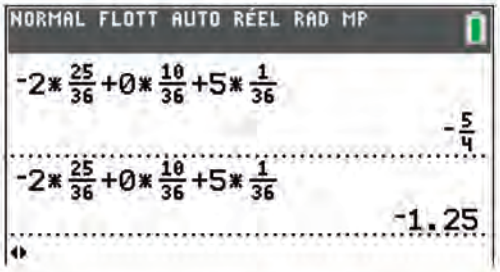
Il s'agit donc de trouver la valeur de a qui annule l'espérance.

$$E(X) = a \times P(X = -2) + 0 \times P(X = 0) + 5 \times P(X = 5)$$

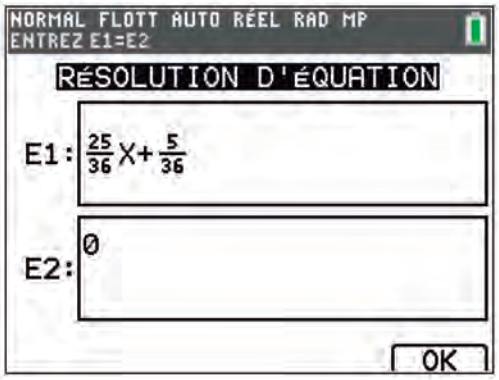
$$E(X) = \frac{25}{36}a + \frac{5}{36}$$

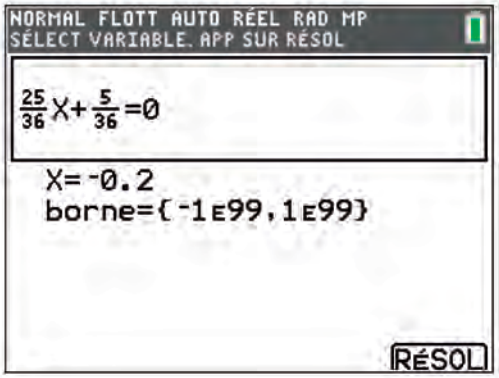
On doit donc résoudre l'équation $\frac{25}{36}a + \frac{5}{36} = 0$

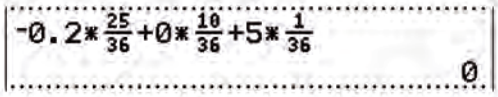
Pour cela, on utilise l'application `Résoudre` accessible via la touche [résol]



On trouve la valeur $a = -0{,}2$ c'est-à-dire que le joueur doit perdre 20 centimes pour que l'espérance du jeu soit nulle.





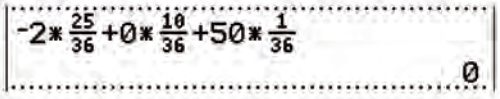


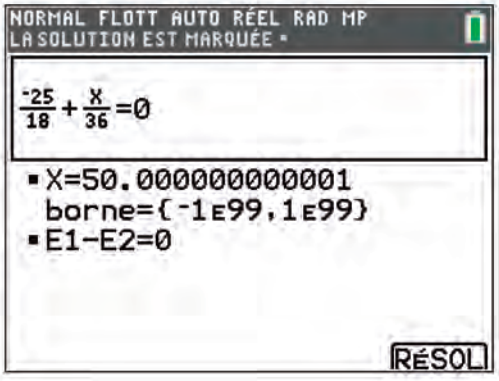
On reproduit la même démarche pour trouver, cette fois-ci, la valeur prise par la variable aléatoire lorsque le joueur gagne. Dans ce cas, on a :

$$E(X) = \frac{-25}{18} + \frac{a}{36}$$

Et on doit résoudre l'équation $\frac{-25}{18} + \frac{a}{36} = 0$

Dans ce cas, le joueur doit gagner 50€ pour que l'espérance du jeu soit nulle.





Répétition d'expériences de Bernoulli

Enoncé

Une entreprise fabrique des petites voitures et les vend par lot de 100. La probabilité qu'un jouet produit présente un léger défaut de peinture a été établi à 28%.

L'entreprise examine un lot et trouve 35 % de jouets présentant un défaut. Elle se demande si sa chaîne de production est dégradée.

```
Fonction JouetDefaut(p,t)
  compteur ← 0
  Pour i allant de 0 à t-1
      Si Nombre Aleatoire < p alors
            compteur ← compteur + 1
      Fin Si
      Fin Pour
      Renvoyer compteur/t
```

1. Pourquoi la situation présentée peut se modéliser par une répétition d'expériences de Bernoulli ?
2. Réaliser la simulation en Python de la situation. Pour cela retranscrire la fonction **JouetDefaut** de paramètres **p** et **t** où **p** représente la probabilité de présenter un défaut et où **t** représente la taille du lot. La fonction renvoie la fréquence de pièces défectueuses.
3. On souhaite répéter plusieurs fois la situation précédente en stockant l'ensemble des fréquences obtenues pour pouvoir les étudier et prendre une décision. Définir en Python la fonction **Echantillon** de paramètre **p, t** et **n** où **p** représente la probabilité de présenter un défaut, **t** représente la taille du lot et **n** représente le nombre de lots testés. La fonction renverra la liste des fréquences obtenues. On pourra représenter le nuage de points des fréquences obtenues.

1. Répétition d'expériences de Bernoulli

La situation consiste à examiner chacune des pièces d'un lot.

Pour chaque pièce, deux issues possibles : défectueuse ou pas défectueuse, avec une probabilité déclarée de 28%. On a une épreuve de Bernoulli.

La répétition consiste à examiner chacune des petites voitures du lot pour déterminer si elle est ou non défectueuse.

On a donc bien la répétition d'expériences de Bernoulli.

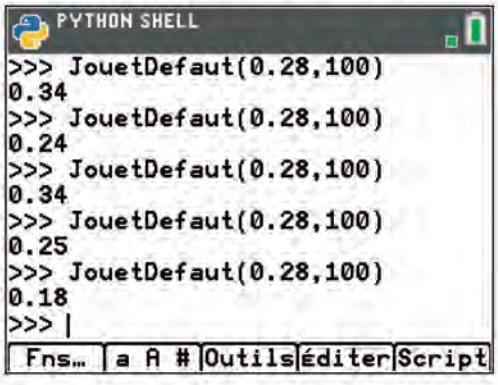
2. Définition de JouetDefaut en Python

On crée un script **BERNOULL** dans lequel on importe la librairie **math**, la librairie **random** et la librairie **ti_plotlib** dont on raccourcit le nom en **plt**. Il s'agit d'une librairie de représentation graphique dont on se servira en fin d'exercice.

Dans le code de notre fonction **JouetDefaut**, nous réutilisons des principes déjà rencontrés dans les fiches précédentes, à savoir la notion de compteur pour cumuler les réussites (ici des pièces défectueuses). Ce nombre est divisé à la fin par le nombre d'expériences réalisées pour retourner la fréquence des succès. L'instruction **random()<p** permet de simuler la probabilité passée en paramètre et ce, **t** fois pour la taille du lot. On teste notre fonction, en console (Exéc), à l'aide de la commande **JouetDefaut(0.28,100)**.

On constate la variation des fréquences d'un lot à l'autre.

On souhaite maintenant conserver une trace des différentes fréquences obtenues.



Répétition d'expériences de Bernoulli

3. Définition de **Echantillon** en Python

On complète notre script avec la fonction **Echantillon**. Elle consiste à stocker dans une liste les résultats renvoyés par notre fonction **JouetDefaut** précédente, avec les paramètres **p** et **t**, ceci **n** fois, le paramètre **n** correspondant au nombre de lots que nous souhaitons prélever.

Là encore, nos travaux précédents nous ont conduit à voir plusieurs possibilités de codage de cette situation. C'est pour cela que nous vous présentons deux solutions. La première consiste à produire un code très court dans lequel nous définissons la liste des fréquences par « compréhension » à l'aide de l'instruction **[JouetDefaut(p,t) for i in range(n)]**. Chaque élément est le résultat de la fonction **JouetDefaut** appelée **n** fois.

L'autre solution, plus classique, consiste à construire au fur et à mesure la liste par ajout successif du résultat de la fonction **JouetDefaut**. La boucle n'est plus à l'intérieur de la définition de la liste mais dans une structure, plus habituelle, à l'extérieur.

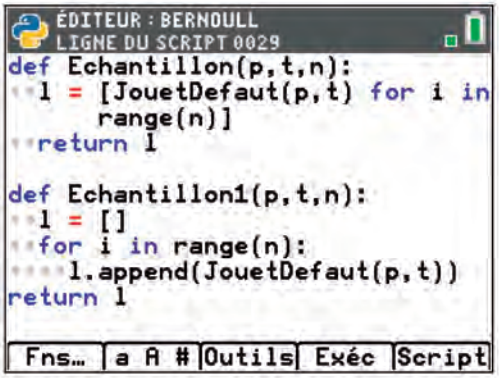
Pour la suite, nous modifions **JouetDefaut** pour qu'elle renvoie la fréquence en pourcentage, arrondi à l'unité, à l'aide de l'instruction **return int(compteur/t*100).** Elle s'appelera **JouetDefaut1.**

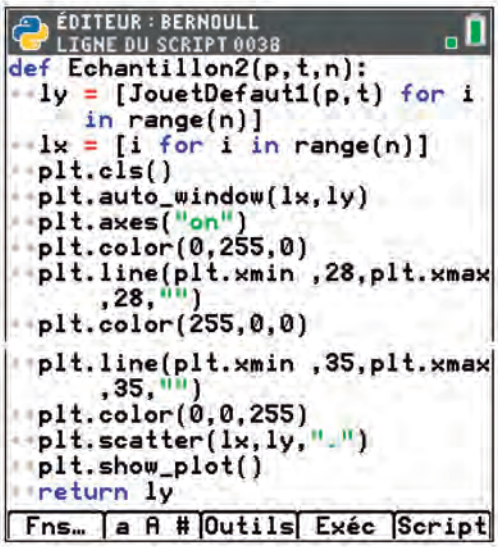
Nous créons la fonction **Echantillon2**, à partir de la fonction **Echantillon**. La liste **l** s'appelle désormais **ly** et nous créons une liste des **n** premiers entiers stockés dans **lx** à l'aide de l'instruction **[i for i in range(n)]**.

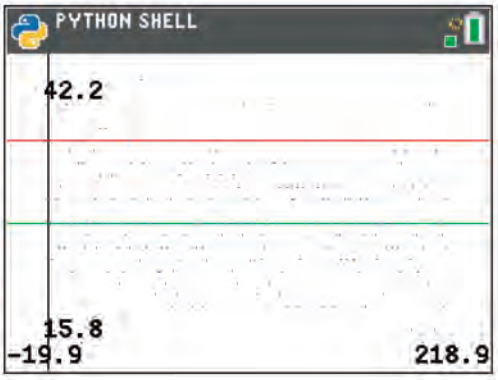
Nous pouvons maintenant préparer l'environnement graphique pour représenter le nuage de points (x;y) construit à partir des éléments respectifs de **lx** et **ly**. Vous trouverez dans le module **ti_plotlib** toutes les instructions nécessaires pour calibrer automatiquement la fenêtre graphique en prévision des points à représenter, afficher les axes de la représentation, dessiner en rouge la droite d'équation y = 35, dessiner en vert la droite d'équation y = 28 et enfin le nuage de points, en bleu, à l'aide de l'instruction **plt.scatter(lx,ly,"")**.

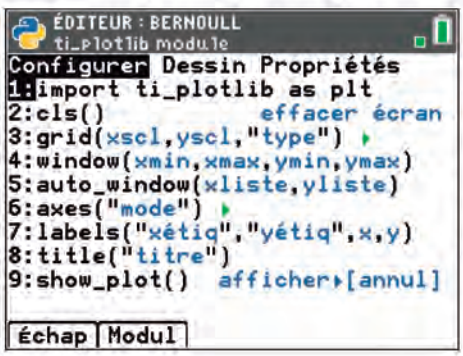
On affiche enfin le graphique via l'instruction **plt.showplot**. Lors de l'exécution, il faut appuyer ensuite sur la touche [annul] pour quitter la représentation graphique et afficher la liste des fréquences en pourcentage.

On parvient à se convaincre que 35% de jouets défectueux, bien que dans une limite haute, peut se produire plusieurs fois.









Algorithme de calcul de moyenne et d'écart type

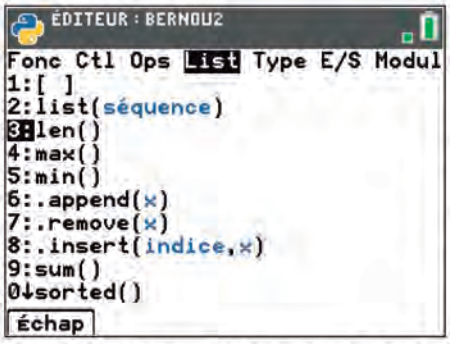
Enoncé

Une entreprise fabrique des petites voitures et les vend par lot de 100. La probabilité qu'un jouet produit présente un léger défaut de peinture a été établi à 28%.

L'entreprise examine un lot et trouve 35 % de jouets présentant un défaut. Elle se demande si sa chaîne de production est dégradée.

1. A l'aide de l'environnement Python, définissez la fonction **EcartType** qui prend en paramètre la liste **l** et renvoie son écart type. Les fonctions intermédiaires **Moyenne** et **Variance** sont fournies.
2. Définissez la fonction **Proportion** qui prend en paramètres **l**, **a** et **b** où **l** est la liste à examiner, **a** et **b** les bornes distinctes d'un Intervalle. La fonction renvoie la proportion, en pourcentage, d'éléments de la liste dans l'intervalle [a;b].
3. A l'aide des fonctions **Echantillon**, **EcartType** et **Proportion**, observez le pourcentage de fréquence appartenant aux intervalles [p-s;p+s], [p-2s ;p+2s] et enfin [p-3s;p+3s] où s est la valeur de l'écart type et p la probabilité qu'un jouet soit défectueux.

1. Définition d'**EcartType** en Python



Nous repartons du script précédent que nous dupliquons à l'aide de l'onglet **Gérer** dans la fenêtre de gestionnaire de script. Il s'appelle désormais **BERNOU2**.

Nous travaillerons en pourcentage arrondi à l'unité, que ce soit pour les fonctions **JouetDefaut** ou **Proportion** de la question suivante.

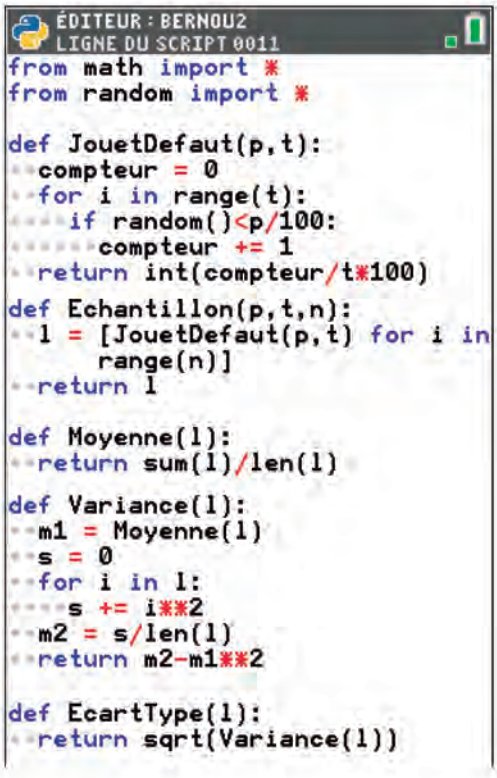
Les fonctions **sum** et **len** renvoient respectivement les sommes des éléments de la liste passée en paramètre et sa longueur, c'est-à-dire le nombre de ses éléments. Ces fonctions sont accessibles dans le sous menu List du menu [Fns…].

Nous avons choisi une formule de calcul de la variance qui nous permet de nous passer de l'utilisation d'une liste intermédiaire et fonctionne par accumulateur.

L'import de la bibliothèque **math**, bien que réalisé systématiquement par réflexe, est ici indispensable pour faire appel à la fonction **sqrt** qui renvoie la racine carrée de la variance.

Nous sommes maintenant presqu'outillés en Python pour pouvoir passer à quelques manipulations et études de nos fréquences.

Il nous reste à définir la fonction qui va permettre de déterminer le pourcentage des valeurs dans un intervalle donné.




```
from math import *
from random import *

def JouetDefaut(p,t):
    compteur = 0
    for i in range(t):
        if random()<p/100:
            compteur += 1
    return int(compteur/t*100)
def Echantillon(p,t,n):
    l = [JouetDefaut(p,t) for i in
        range(n)]
    return l

def Moyenne(l):
    return sum(l)/len(l)

def Variance(l):
    m1 = Moyenne(l)
    s = 0
    for i in l:
        s += i**2
    m2 = s/len(l)
    return m2-m1**2

def EcartType(l):
    return sqrt(Variance(l))
```

Algorithme de calcul de moyenne et d'écart type

2. Définition de **Proportion** en Python

Nous complétons notre script avec la fonction **Proportion**, qui comptabilise le nombre d'éléments de la liste **l** passée en paramètre dans un intervalle de valeurs données par les paramètres **a** et **b** distincts.

Finalement la fonction renvoie la proportion sous forme de pourcentage arrondi à l'unité en divisant la variable p par la longueur de la liste. C'est la fonction **int** qui arrondit à l'unité.

```
def Proportion(l,a,b):
 p = 0
 for i in l:
  if a<=i  and i<=b:
   p += 1
 return int(p/len(l)*100)
```

Fns… | a A # | Outils | Exéc | Script

3. Manipulation

La suite de l'exercice consiste donc à manipuler nos fonctions en console et observer les résultats obtenus. On commence par définir la variable **p** à 28 (notre probabilité de référence), **t** (la taille d'un échantillon) à 100 et **n** à 100.

On définit **l** une première liste de fréquences issues de la fonction Echantillon.

On stocke dans s l'écart type de notre liste.

On utilise la touche [var] pour appeler plus rapidement chacune de nos fonctions et réaliser les manipulations demandées.

PYTHON SHELL

```
>>> # Shell Reinitialized
>>> # L'exécution de BERNOU2
>>> from BERNOU2 import *
>>> p = 28
>>> t = 100
>>> n = 100
>>> l = Echantillon(p,t,n)
>>> s = EcartType(l)
>>> |
```

Fns… | a A # | Outils | Éditer | Script

Lors de notre première manipulation, nous avons obtenu que 93% des fréquences issues de nos 100 échantillons de taille 100 sont dans l'intervalle [p-2;p+2s]. Il semble donc que les 35% de jouets défectueux, obtenus lors du test d'un lot ne soit pas aberrant.

PYTHON SHELL

```
>>> Proportion(l,p-s,p+s)
75
>>> Proportion(l,p-2*s,p+2*s)
93
>>> Proportion(l,p-3*s,p+3*s)
100
>>> p+2*s
36.87988738667334
>>> p+3*s
41.31983108001001
>>> |
```

Fns… | a A # | Outils | Éditer | Script

On relance plusieurs fois ces manipulations avec les mêmes paramètres de taille d'échantillons et nombre de lots. Les résultats se confirment à quelques variations près.

Il est possible de relancer ensuite les manipulations en augmentant le nombre d'échantillons jusqu'à 500.

Là encore les résultats nous amèneront à la même conclusion.

PYTHON SHELL

```
>>> l=Echantillon(p,t,n)
>>> s=EcartType(l)
>>> Proportion(l,p-s,p+s)
66
>>> Proportion(l,p-2*s,p+2*s)
96
>>> Proportion(l,p-3*s,p+3*s)
99
>>> p+2*s
37.05427103636731
>>> |
```

Fns… | a A # | Outils | Éditer | Script

PYTHON SHELL

```
>>> l=Echantillon(p,t,n)
>>> s=EcartType(l)
>>> Proportion(l,p-s,p+s)
68
>>> Proportion(l,p-2*s,p+2*s)
95
>>> Proportion(l,p-3*s,p+3*s)
100
>>> p+2*s
37.10514140472295
>>> |
```

Fns… | a A # | Outils | Éditer | Script

Thème : Analyse

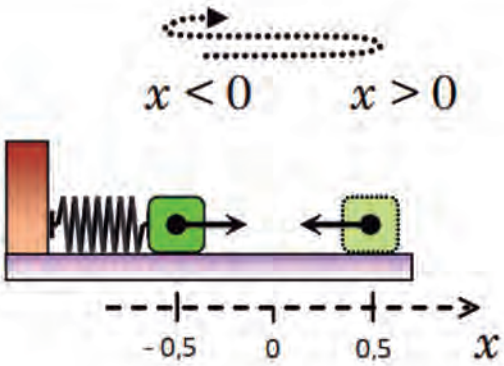
Fonction trigonométrique

TI-82 Advanced
Edition Python

TI-83 Premium CE
Edition Python

Enoncé

On s'intéresse aux oscillations horizontales d'un objet autour de sa position d'origine, en fonction du temps. Ces oscillations sont créées par un ressort attaché à l'objet, et fixé à un support, ainsi qu'à une modification de la position d'origine. Les frottements sont négligés.



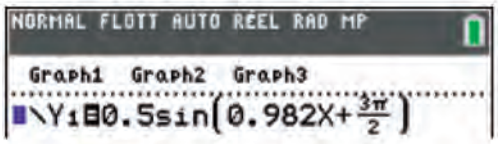
La position du centre de gravité de l'objet, selon l'axe x, en mètres, est donné par la fonction trigonométrique :

$$x(t) = 0{,}5 \times \sin\left(0{,}982t + \frac{3\pi}{2}\right)$$, où t est exprimée en secondes (s).

1. Déterminer $x(0)$ et interpréter ce résultat dans le cadre de l'expérience.
2. Déterminer le temps t_0 que met l'objet à retrouver, pour la première fois, sa position d'équilibre, c.à.d. tel que $x(t_0) = 0$.
3. A l'aide de votre calculatrice, conjecturer le nombre de fois où l'objet repasse par sa position d'équilibre pendant les 39 premières secondes de l'expérience. Démontrer cette conjecture.

1. Position initiale

Pour conjecturer la réponse, on peut effectuer un simple calcul sur la calculatrice, mais la lecture complète de l'énoncé nous suggère d'utiliser la fonction trigonométrique $x(t)$. Dans le menu [f(x)], on saisit ainsi l'expression de la fonction dans Y_1.



On appuie sur [2nde] [mode] (quitter) pour revenir à l'écran de calcul. Ensuite, on appuie sur [var] puis, dans le menu **VAR Y**, on sélectionne **1:Fonction…** et enfin **1:Y1**. On complète notre saisie pour calculer **Y1(0)** et on valide par [entrer].

On démontre ensuite notre conjecture :

$$x(0) = 0{,}5 \times \sin\left(\frac{3\pi}{2}\right) = 0{,}5 \times (-1) = -0{,}5$$

La position initiale de l'objet, au moment $t = 0$ où l'on lâche l'objet, est située à $-0{,}5$ m sur l'axe x.

2. Détermination de t_0

La représentation graphique de la fonction x est sûrement le meilleur moyen de visualiser la réponse à cette question.

Pour nous aider à représenter cette fonction, il est utile de rappeler ici le vocabulaire du cours :

- $A = 0{,}5$ est l'amplitude de cette fonction, qui va donc osciller entre $-0{,}5$ m et $+0{,}5$ m ;
- $\omega = 0{,}982$ est la pulsation, exprimée en $rad.s^{-1}$;
- $T = \frac{2\pi}{\omega} \approx 6{,}4\ secondes$ est la période de la fonction ;
- $0{,}982t + \frac{3\pi}{2}$ est la phase instantanée, en radians ; $\frac{3\pi}{2}$ est la phase à l'origine, en radians également.

Fonction trigonométrique

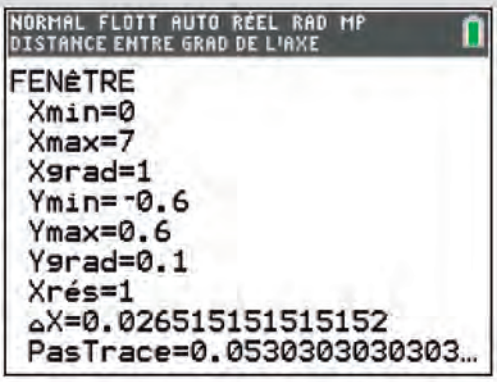
Dans le menu [fenêtre] , on saisit ainsi les valeurs minimales et maximales de nos axes, ainsi que les graduations.

- Pour l'axe des abscisses et notre variable t en secondes, on prend

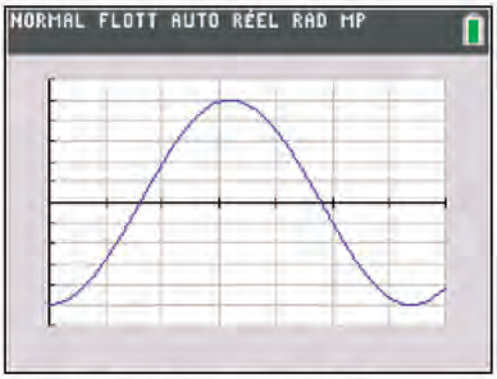
  `Xmin=0,  Xmax=7,  Xgrad=1`

- Pour l'axe des ordonnées et notre variable x en mètres, on prend

  `Ymin=-0.6,  Ymax=0.6,  Ygrad=0.1`

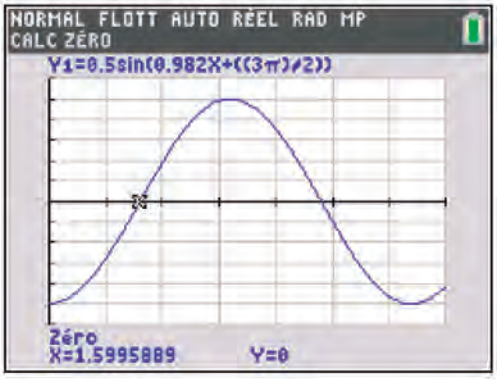


On appuie alors sur la touche [graphe] et on visualise notre fonction trigonométrique.



Pour conjecturer la 1ère racine de $x(t)$, dans le menu [2nde] [trace] (calcule 14) , on sélectionne la commande **`2:racine`**. On effectue alors la procédure :

- On place le curseur à gauche de la racine recherchée et on valide par [entrer].
- On fait de même en plaçant le curseur à droite de la racine.
- On valide une dernière fois par [entrer] .



On obtient ainsi la valeur approchée $t_0 \approx 1{,}6$.

Puisqu'on commence à $t = 0$, le sinus s'annule une 1ère fois pour :

$$0{,}982t_0 + \frac{3\pi}{2} = 2\pi \Leftrightarrow 0{,}982t_0 = \frac{\pi}{2} \Leftrightarrow t_0 = \frac{\pi}{2 \times 0{,}982} \approx 1{,}6\ s$$

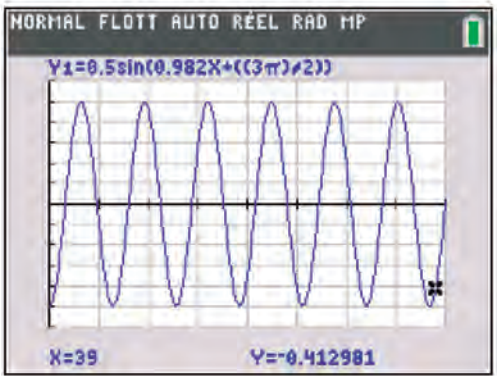
3. Nombre de passages

Dans cette question, on commence par changer notre fenêtre graphique, dans le menu [fenêtre] : `Xmax=40,  Xgrad=5`

On appuie sur [graphe] et on peut conjecturer le résultat : l'objet repasse ainsi 12 par sa position d'équilibre lors des 39 premières secondes.

Vérification :

- $\frac{39}{T} \approx 6{,}1$: Lors des 39 secondes, l'objet effectue un peu plus de 6 périodes.
- Lors d'une période, l'objet passe 2 fois par sa position d'équilibre.
- $6 \times 2 = 12$ et on retrouve bien le résultat conjecturé graphiquement.



Thème: Analyse

Produit scalaire & Python

TI-82 Advanced
Edition Python

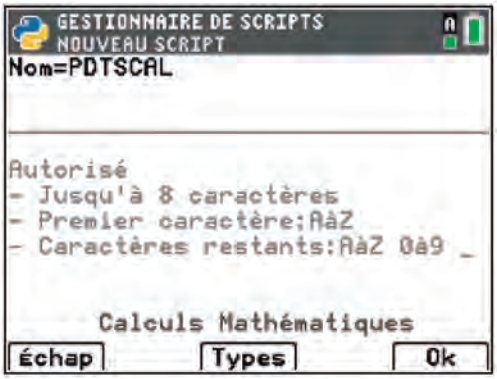
TI-83 Premium CE
Edition Python

Enoncé

1. Créer un script **Python**, de type **Calculs Mathématiques**, nommé **PDTSCAL**. Les fonctions des questions suivantes seront codées dans ce script.
2. Connaissant les coordonnées de 2 vecteurs $\vec{u}\begin{pmatrix}x_{\vec{u}}\\y_{\vec{u}}\end{pmatrix}$ et $\vec{v}\begin{pmatrix}x_{\vec{v}}\\y_{\vec{v}}\end{pmatrix}$, écrire une fonction **Python**, nommée **psvec**, prenant pour arguments les valeurs de **xu**, **yu**, **xv** et **yv**, et retournant en sortie la valeur du produit scalaire des vecteurs $\vec{u}$ et $\vec{v}$. Tester cette fonction.
3. Connaissant 2 points $A(x_A\,;y_A)$ et $B(x_B\,;y_B)$, écrire une fonction **Python**, nommée **vec**, prenant pour arguments les valeurs **xa**, **ya**, **xb** et **yb**, et retournant en sortie les coordonnées du vecteur $\overrightarrow{AB}$. Tester cette fonction.
4. Connaissant 4 points $A(x_A\,;y_A)$, $B(x_B\,;y_B)$, $C(x_C\,;y_C)$ et $D(x_D\,;y_D)$ écrire une fonction **Python**, nommée **pspts**, prenant pour arguments les valeurs **xa**, **ya**, **xb**, **yb**, **xc**, **yc**, **xd** et **yd**, et retournant en sortie la valeur du produit scalaire des vecteurs $\vec{u}$ et $\vec{v}$. Cette fonction **pspts** fera appel aux 2 fonctions précédentes. Tester cette fonction.

1. Création du script

- On commence par aller dans le module **Python** de la calculatrice, en appuyant sur [prgm] et en sélectionnant **2:Python App**.
- Une fois dans le module Python, on créé un nouveau script, à l'aide **Nouv** (touche **f3**). On saisit alors le nom du script, **PDTSCAL**.
- A l'aide de la touche **f3**, on sélectionne le type **2:Calculs Mathématiques** et on valide par **Ok** (touche **f5**).



2. Fonction psvec

On rappelle l'expression analytique du produit scalaire :

$$\vec{u}\cdot\vec{v} = x_{\vec{u}} \times x_{\vec{v}} + y_{\vec{u}} \times y_{\vec{v}}$$

- Les commandes nécessaires à toute fonction **Python** se trouvent dans le menu **Fns…** (touche **f1**), dans le 1er onglet nommé **Fonc**.
- Vous trouverez ci-contre le script de cette fonction. Une fois la saisie terminée, on exécute le script via l'onglet **Exéc** (touche **f4**).
- On n'oublie évidemment pas de tester cette fonction, avec des valeurs simples qui permettent de vérifier aisément l'exactitude des résultats obtenus. Notre fonction est disponible via la touche [var] de la calculatrice.

Produit scalaire & Python

3. Fonction **vec**

On rappelle l'expression des coordonnées du vecteur $\overrightarrow{AB}$:

$$\overrightarrow{AB}\begin{pmatrix} x_B - x_A \\ y_B - y_A \end{pmatrix}$$

On procède de la même manière que pour la fonction précédente et on effectue différents tests. Attention toutefois à ne pas coder cette 2nde fonction à l'intérieur de la 1ère ; c'est l'indentation qui vous l'indique.

A noter que cette fonction retourne un couple de coordonnées. Si l'on veut accéder à une de ces 2 coordonnées (ce qui sera utile pour la question suivante), il faudra saisir (si on nomme **coord** le résultat renvoyé par notre fonction) :

- `coord[0]` pour la 1ère coordonnée ;
- `coord[1]` pour la 2nde.

```
ÉDITEUR : PDTSCAL
LIGNE DU SCRIPT 0007
# Calculs Mathématiques
from math import *
def psvec(xu,yu,xv,yv):
  return xu*xv+yu*yv

def vec(xa,ya,xb,yb):
  return xb-xa,yb-ya
```

```
PYTHON SHELL
>>> # Shell Reinitialized
>>> # L'exécution de PDTSCAL
>>> from PDTSCAL import *
>>> coord=vec(1,3,5,-2)
>>> coord
(4, -5)
>>> coord[0]
4
>>> coord[1]
-5
>>> |
Fns… | a A # | Outils | éditer | Script
```

4. Fonction **pspts**

Pour cette dernière fonction, il convient de récapituler les étapes, avant de coder proprement dit :

a) Les arguments de notre fonction sont les 8 coordonnées de nos 4 points.

b) A l'aide de ces 8 coordonnées, et de la fonction **vec**, on fabrique nos 2 vecteurs, que l'on peut nommer **vec1** et **vec2**, par exemple.

c) Une fois nos 2 vecteurs créés à partir des coordonnées des points, on peut utiliser notre fonction **psvec**, en faisant attention de bien utiliser les valeurs, et non les couples de coordonnées.

d) On retourne le produit scalaire en sortie.

Vous trouverez ci-contre le script complet de l'exercice.

Comme d'habitude, il est nécessaire de tester cette dernière fonction avec différentes valeurs et de vérifier que les résultats retournés sont cohérents.

Cette façon de travailler, en divisant le travail à l'aide de « petites » fonctions très ciblées, se retrouve fréquemment en mathématiques et en informatique.

« Diviser pour régner » n'a rien de néfaste dans notre matière, bien au contraire !

```
ÉDITEUR : PDTSCAL
LIGNE DU SCRIPT 0001
# Calculs Mathématiques
from math import *
def psvec(xu,yu,xv,yv):
  return xu*xv+yu*yv

def vec(xa,ya,xb,yb):
  return xb-xa,yb-ya

def pspts(xa,ya,xb,yb,xc,yc,xd,y
    d):
  vec1=vec(xa,ya,xb,yb)
  vec2=vec(xc,yc,xd,yd)
  ps=psvec(vec1[0],vec1[1],vec2[
    0],vec2[1])
  return ps
Fns… | a A # | Outils | Exéc | Script
```

```
PYTHON SHELL
>>> # Shell Reinitialized
>>> # L'exécution de PDTSCAL
>>> from PDTSCAL import *
>>> pspts(0,0,1,0,-2,3,-1,4)
1
>>> pspts(0,0,1,3,5,1,2,3)
3
>>> pspts(1,0,5,0,0,2,-2,4)
-8
>>> |
Fns… | a A # | Outils | éditer | Script
```

Plus d'informations sur le site internet de T³ France :

- Des ressources pédagogiques pour votre classe
- Un programme de formations gratuites sur site et en ligne
- Des vidéos d'aide à la prise en main de la technologie

t3france.fr

Un service après-vente est également accessible depuis le site **education.ti.com/fr/csc**