


## Contenu

Entrées-Sorties - tests - répétitions .....	2
Listes .....	3
Graphiques .....	3
Nombres aléatoires - Lois de probabilités discrètes .....	4
Lois de probabilités continues .....	5
Constantes - fonctions .....	6
Matrices .....	7

Illustrations sur le site planète maths : [fiche thématique Algorithmique](#) et [logiciels](#)

# Entrées-Sorties - tests - répétitions

		Langage algorithmique				
		<b>Scilab</b> Certaines fonctions nécessitent l'installation du module « lycée » <b>Le point virgule permet d'écrire plusieurs instructions sur la même ligne, Il supprime aussi l'affichage</b>	<b>Python 2.6</b> <b>3.x seulement</b> <b>Attention : en v 2.6, 3/2 vaut 1, et non 1,5</b>	<b>TI</b> <b>82-84</b>	<b>Casio</b> <b>35+</b> (non USB)	<b>XCAS 0.9.4</b> <b>Le point-virgule sépare les instructions</b> <b>Pour plusieurs instructions le français et l'anglais sont acceptés.</b>
Insérer un commentaire		<i>// mon commentaire</i>	<i># mon commentaire</i>			<i>// mon commentaire</i>
Saisir a		a=input ( "donner a ")	a=input("donner a ") a=float(input("réel a ?")) a=int(input("entier a ?"))	:Prompt A :Input "X1=", X	"A=": ? → A dans shift PRGM	input("a= ",a)  saisir ("a= ",a)
Afficher a <i>(Xcas : Unquoted : sans guillemets)</i>		a ou afficher( " a= "+string(a)) ou disp( " a= "+string(a))	V 2.6 : print «'a= ', a V 3.x : print ( 'a= ', a)	:Disp "A=" , A	"A=" : A ▴ dans shift PRGM	print ("a=",a) print ("a=",a)  afficher ("a=",a) afficher ("a=",a)
affectation: a → b b prend la valeur a,		b = a	b = a	A  B	A → B touche directe	b := a
Tests, logique =, ≠, ≤, ≥ et, ou, non, ou exclusif,		== , <> , <= , >= & ,   (AltGr+6) , ~ (AltGr+2), voir	==, !=, <=, >= and, or, not, xor	Menu 2nd TEST =, ≠, ≤, ≥ and, or, not, xor	=, ≠, ≤, ≥ Dans shift PRGM REL and, or, not dans OPTN LOGIC	=, !=, <=, >= and, or, not, xor
Bloc d'instructions		Les blocs sont définis par la structure	Les instructions d'un bloc ont la même marge à gauche (indentation)	Le bloc est terminé par End	Le bloc est terminé par End, ifEnd, whileEnd,...	Le bloc est encadré par des accolades: {instructions} , sauf s'il est encadré par la structure, voir ci-dessous. Pour ne pas se tromper utiliser le menu <b>Add</b>
Si condition Alors Instruction1 Facultatif [ Sinon Instruction2] Fin du si	Condition2 est la négation de condition1	if condition then Instructions1 else Instructions2 end <small>(if et then doivent être sur la même ligne)</small>	if condition: Instruction1 [ else: Instruction2]	:If condition :Then Instructions1 [:Else Instructions2] :End	If condition Then Instructions1 [Else Instructions2] IfEnd dans PRGM COM	if ( condition) then {Instructions1} [ else {Instructions2}] fsi  si (condition) alors Instructions1 [ sinon Instructions2 ] fsi
Répéter Instruction(s) Jusqu'à condition1		while %T then Instruction(s) if condition then break end	while True : Instruction(s) if condition : break end	:Repeat condtion1 Instruction(s) :End	Do Instruction(s) LpWhile condition2 dans PRGM COM	repeat instruction(s) until (condition1)  repetar instruction(s) jusqu_a (condition1)
Tant condition Instruction(s) Fin TantQue		while condition then Instruction(s) end	while condition : Instruction(s)	:While condtion Instruction(s) :End	While condition Instruction(s) Endwhile dans PRGM COM	tantque (condition) faire instruction(s) ftantque;
Pour i variant de 1 à n Faire Instruction(s) Fin du pour		for i=1:n Instruction(s) end	for i in range(1,n+1): Instruction(s)	:For (l,1,N) Instruction(s) :End	For 1->A To 10 Instruction(s) Next dans PRGM COM	for j from 1 to n do instruction(s) end_for (ne pas utiliser « i » )  pour j de 1 jusque n faire instruction(s) fpour (éviter la lettre i)

# Listes

	Scilab	Python 2.6 3.x <i>seulement</i>	TI 82-84	Casio 35+ (non USB)	XCAS 0.9.4
Créer une liste	<i>En Scilab, les listes sont aussi appelées vecteurs, indice minimum : 1</i> $l=[5,8,9]$ $l(1)$ vaut 5, $l(2)$ vaut 8...	$l=[5,8,9]$ l'indice commence à <u>zéro</u> , $l[0]$ vaut 5, $l[1]$ vaut 8,...	Les listes $L_1, L_2$ , existent dans le mode Statistique	Les listes <b>List 1, List 2</b> existent dans le menu STAT	$l:=[5,8,9]$ l'indice commence à <u>zéro</u> , $l[0]$ vaut 5, $l[1]$ vaut 8,...
Vider une liste $l$ Créer une liste vide (Scilab, python, Xcas)	$l=[]$	$l=[]$	<b>ClrList</b>	Menu <i>Stat</i> puis <b>DEL-A</b> ou $\{0\} \rightarrow$ List 1	$l := []$
Créer et remplir une liste de six 0, de $n+1$ 0 Créer $l=[5^2,7^2,9^2,11^2,13^2]$	$l = \text{zeros}(1,6)$ <i>Avec une boucle, ou...</i>	<i>Avec une boucle et la fonction : append</i>	<b>6 STO dim(L<sub>1</sub>)</b> Seq(X^2,X,5,13,2) <b>STO</b> L <sub>1</sub>	6 $\rightarrow$ Dim List 1 Dans OPTN LIST Seq(X^2,X,5,13,2) $\rightarrow$ List 1	$l := [0\$6], l := [0\$(n+1)]$ ou $l := [0\$(k=1..6)]$ $l := \text{seq}(k^2,k,5,13,2);$
Ajouter un élément $a$ à la fin de la liste $l$	<i>Si <math>l</math> comporte déjà <math>n</math> éléments : <math>l(n+1)=a</math></i>	$l.append(a)$	$a$ <b>STO</b> L <sub>1</sub> ( $l$ ) $l$ étant le premier indice non encore utilisé	Dans le menu <i>List</i> , entrer directement l'élément $a$ à la fin de la liste ! (inutilisable dans un programme)	$l := \text{append}(l, a)$
Accès à l'élément numéro $k$	$l(k)$	$l[k]$	L <sub>1</sub> ( $k$ )	List1[ $k$ ]	$l[k]$
Longueur d'une liste	$\text{taille}(l)$	$\text{len}(l)$	<b>dim</b> (L <sub>1</sub> )	Dim List 1 Dans OPTN LIST	$\text{length}(l)$

# Graphiques

Quelques instructions pour les graphiques					
	Scilab	Python 2.6 ou 3.x <i>from graphsecondev2_3</i> <i>import* télécharger</i> le module	TI 82-84	Casio 35+	XCAS
Passer en mode graphique / mode calcul	<b>automatique</b>	<i>fenetre(xmin,xmax,ymin,ymax)</i> <i>affiche à la fin du pgm</i>	<b>DispGraph</b>	<b>DrawGraph</b> Dans shift PRGM DISP Grph	<b>DispG</b> <b>DispHome</b>
Effacer l'écran Graphique	<b>Clf</b> <b>Attention : clear efface les fonctions !</b>		<b>ClrDraw</b> ou <b>EffDessin</b>	<b>ClrGraph</b> Dans shift PRGM CLR	<b>ClrGraph</b> ou <b>efface</b>
Placer un point M(x,y) Choisir la taille	<b>plot</b> ( $x,y, " . "$ ) $\text{plot}([1],[1],',','MarkerSize',1)$	<b>point</b> ( $x,y[,couleur]$ )	<b>Pt-On</b> ( $x,y[,marquee]$ )	<b>Plot</b> $x,y$ Dans shift Sketch(F4)	<b>point</b> ( $x,y$ )
Tracer le segment [AB] avec A(x <sub>A</sub> ; y <sub>A</sub> ) et B(x <sub>B</sub> ; y <sub>B</sub> )	<b>plot</b> ( [ $x_A, x_B$ ] , [ $y_A, y_B$ ] ) <b>Attention à l'ordre !</b>	<b>segment</b> (( $x_A, y_A, x_B, y_B$ [,couleur])	<b>Line</b> ( $x_A, y_A, x_B, y_B$ ) ou <b>Ligne</b> ( $x_A, y_A, x_B, y_B$ )	<b>F-line</b> $x_A, y_A, x_B, y_B$ Dans shift Sketch(F4)	$A := \text{point}(x_A, y_A);$ $B := \text{point}(x_B, y_B);$ <b>segment</b> (A,B);
Tracer un cercle	$t = \text{linspace}(0, 2 * \pi, 100);$ $x = x_0 + r * \cos(t); y = y_0 + r * \sin(t);$ $\text{plot}(x, y)$	<b>cercle_cr</b> ( $x,y,r[,couleur]$ ) <b>cercle_cp</b> ( $x,y,s,t[,couleur]$ )	<b>Circle</b> ( $x,y,r$ ) $x$ et $y$ coordonnées du centre et $r$ le rayon	<b>Circle</b> $x,y,r$ $x$ et $y$ coordonnées du centre, et $r$ le rayon Dans shift Sketch(F4)	<b>circle</b> ( $\text{point}(x,y),r$ ) voir autres possibilités dans l'index

# Nombres aléatoires - Lois de probabilités discrètes

	Scilab	Python 2.6 3.x	TI	Casio	XCAS
Nombres aléatoires	Module Lycée	Avec from random import*	Touche MATH/ PRB	Run / touche OPTN / PROB	Le point virgule sépare les instructions
Initialisation	rand("seed")	seed()	-	-	srand
Nombre réel aléatoire dans [0;1[	tirage_reel (1,0,1) (liste de 1 seul réel aléatoire)	random()	rand	Rand#	rand(0,1) ou alea(0,1)
Nombre réel aléatoire dans [a;b[	tirage_reel(p,a,b) (liste de p réels aléatoires)	a + (b-a) x random() uniform(a,b) , dans [a,b]	a + (b-a) x rand	a + (b-a) x Rand#	rand(a,b) ou alea (a,b)
Entier aléatoire dans {a;a+1;...;b} avec a et b entiers	tirage_entier(p,a,b) (liste de p nombres entiers aléatoires)	randint(a,b)	rand(a,b)	a + Intg((b-a+1) x Rand#)	a+rand(b-a+1) ou a+alea(b-a+1)
Exemple : entier aléatoire dans : {0;1} puis dans {1;2;3;4;5;6}	L= tirage_entier (1,0,1) (liste de 1 seul entier aléatoire) L= tirage_entier (1,1,6)	randint(0,1) randint(1,6)	rand(0,1) randInt(1,6)	Intg(2*Rand#) 1+ Intg(6*Rand#)	rand(2) ou alea(2) 1+rand(6) ou 1+alea(6)
Remplir une liste l avec dix nombres Aléatoires pris dans {1;2;3;4;5;6}	l=tirage_entier (10,1,6)	Avec une boucle et append	Seq(randInt(1,6),X,1,10,1)	Seq(1+ Intg(6*Rand#),X,1,10,1) Seq : Dans OPTN LIST	l := [(1+rand(6))\$(k=1..10)] ou seq(1+rand(6),k,1,10)
Coefficients binomiaux, loi binomiale					
module	-	From loi_discrete import* Télécharger Ou avec scipy <a href="#">stats.binom</a>	-	-	-
$\binom{n}{p}$	combinaison(n,p)	nCr(n,p)	n Combinaison p ou n nCr p dans MATH PRB	n nCr p dans OPTN / PROB	nCr(n,p)
Loi binomiale B(n,p) P(X=k)	loi_binomiale(n,p,k)	binomial(n,p,k)	binomFdp(n,p,k) ou binompdf(n,p,k) dans 2 <sup>nd</sup> DISTR DISTRIB	BinomialPD(k,n,p) Dans OPTN / STAT / DIST / BINM / Bpd	binomial(n,p,k)
Loi binomiale B(n,p) cumulée P(X≤k)	1°/ c= cumsum(binomial(p,n)) 2°/ afficher c(k)	binomial_cdf(n,p,k)	binomFRép(n,p,k) ou binomcdf(n,p,k)	BinomialCD(k,n,p) Dans OPTN / STAT / DIST / BINM / Bcd	binomial_cdf(n,p,k)
Autres lois discrètes					
Loi de Poisson de paramètre μ P(X≤x)		Consulter scipy <a href="#">stats.poisson</a>	poissonFRép(μ,x) ou poissoncdf(μ,x) dans 2 <sup>nd</sup> DISTR DISTRIB	PoissonCD(x, μ) Dans OPTN / STAT / DIST / POISN / Pcd	poisson_cdf( μ ,x)

# Lois de probabilités continues



	Scilab	Python 2.6 3.x	TI	Casio	XCAS
Nombres aléatoires	Module Lycée	Avec from random import* from <a href="#">scypi.stats</a> import*	Touche MATH/ PRB	Run / touche OPTN / PROB	Le point virgule sépare les instructions
Initialisation du calcul aléatoire Pour les simulations	rand("seed")	seed()			srand
Lois normales					
Fontions de densité $N(0,1)$ , Et $N(\mu, \sigma^2)$ $\frac{1}{\sqrt{2\rho}} e^{-\frac{1}{2}x^2}$ et $\frac{1}{s\sqrt{2\rho}} e^{-\frac{1}{2}\frac{x-m_0^2}{s^2}}$		<a href="#">site scypi.stats.norm</a> norm.pdf(x) norm.pdf(x, $\mu, \sigma$ )	dans 2 <sup>nd</sup> DISTR DISTRIB normalpdf(x) et normalpdf(x, $m, S$ )	OPTN, puis STAT, puis DIST NormPD(x) NormPD(x, $\mu, \sigma$ )	normald(x) normald(mu,sigma,x)
Loi normale $N(\mu, \sigma^2)$ $P(a \leq X \leq b)$ ou $P(X \leq x)$	loi_normale(t, $\mu, \sigma$ ) retourne la probabilité $p(X \leq t)$	norm.cdf(x, $\mu, \sigma$ )	normalFRép(a,b, $\mu, \sigma$ ) normalcdf(a, b, $m, S$ )	NormCD(a,b, [ $\mu, \sigma$ ])  Dans OPTN / STAT / DIST / NORM / Ncd	normald_cdf( $\mu, \sigma, x$ )
Loi normale centrée réduite $P(a \leq X \leq b)$ ou $P(X \leq x)$	loi_normale(b,1,0) - loi_normale(a,1,0)	norm.cdf(b)- norm.cdf(a) norm.cdf(x)	normalFRép(a,b) ou normalcdf(a,b)	NormCD(a,b)	normald_cdf(mu,sigma,a,b) normald_cdf(mu,sigma,x)
Valeur de x telle que $P(X \leq x)=t$	<b>cdfnor("X",6,2,0.9,0.1)</b> donne le réel x tel que $P(N(6, 2^2) \leq x)=0,9$ .	norm.ppf(t) ou norm.ppf(t, $\mu, \sigma$ )	FracNormal( t, [ $\mu, \sigma$ ]) invNorm(t[ $\mu, \sigma$ ])	InvNormCD(t, [ $\mu, \sigma$ ]) Dans OPTN / STAT / DIST / NORM / invN	normald_icdf(mu,sigma,t)
Génération de 1000 nombres aléatoires suivant une loi normale $N(\mu, \sigma^2)$	<b>grand(2,3,"nor",5,4)</b> : matrice 2 x 3 dont les coef suivent $N(5,4^2)$	from <a href="#">scypi.stats</a> import* norm.rvs( $\mu, \sigma, size = 1000$ )	normAleat( $\mu, \sigma, 1000$ ) randNorm( $\mu, \sigma, 1000$ )	avec seq	seq(randnorm( $\mu, \sigma$ ),j,1,1000)
Autres lois continues					
Lois exponentielle s de paramètre $\ell$	-	Dans scipy : from <a href="#">Stats.expon</a> import*	-	-	-
Fonction de densité	-	from <a href="#">scypi.stats</a> import* expon.pdf(x,0,1/ $\ell$ ) ou bien expon.pdf(x,scale=1/ $\ell$ )	-	-	calcul direct
$P(X \leq x)$	<b>loi_exp( <math>\ell, x</math> )</b> retourne la probabilité $p(X \leq x)$ lorsque X suit la loi exp de param $\ell$	from <a href="#">scypi.stats</a> import* expon.cdf(x,0,1/ $\ell$ ) ou bien expon.cdf(x,scale=1/ $\ell$ )	-	-	calcul direct
Génération de 1000 nombres aléatoires suivant une loi exp. De paramètre $\ell$	<b>grand(2,3,"exp", <math>\ell</math> )</b> donne une matrice 2 x 3 dont les coef suivent la loi exp de param $\ell$	from <a href="#">scypi.stats</a> import* expon.rvs(0,1/ $\ell$ ,size=1000) ou bien expon.rvs(scale=1/ $\ell$ ,size=1000)	-	-	seq(randexp( $\ell$ ),j,1,1000)

## Constantes - fonctions

Quelques fonctions courantes - Constantes - Définir une fonction						
	Scilab	Python 2.6 <i>3.x seule.</i>	TI	Casio	XCAS	
Racine carrée	sqrt	sqrt (from math import*)	touche directe	touche directe	sqrt	
Puissance $a^b$	$a^b$	$a^{**}b$ ou pow(a,b) (from math import*)	$a^b$	$a^b$	$a^b$	
Valeur absolue	abs	fabs (from math import*)	abs	Abs dans OPTN NUM	abs	
ln, exp	log, exp	log exp (from math import*)	ln exp	ln exp	ln exp	
Partie entière(*) , plus grand entier relatif inférieur ou égal au nombre considéré	floor	floor (from math import*)	Int ou PartEnt dans MATH NUM	Intg dans OPTN / NUM	floor	
Troncature(*), c'est-à-dire nombre sans sa partie décimale éventuelle	int	trunc (from math import*)	iPart dans MATH NUM	Int dans OPTN / NUM	iPart	
Quotient division euclidienne	quotient (a,b)	V 2.6 a/b v 3.x a//b (avec a et b entiers)	Int(A/B) ou PartEnt(A/B)	Intg(A/B)	iquo(a,b)	
Reste division euclidienne	reste(a,b)	$a\%b$ , ou fmod(a,b) (from math import*)	$A-B * \text{Int}(A/B)$	$A-B * \text{Intg}(A/B)$	irem(a,b)	
pi, e, $i$	$\%pi$ , $\%e$ , $\%i$	pi, e, 1j (from math import*)	touche directe	touche directe	pi, e, $i$	
Définir une fonction Par exemple $f(x)=4x^3+2x+1$	function y=f(x) y=4*x^3+2*x+1 endfunction	def f(x): return 4*x**3+2*x+1	Touche Y= Y1=4*X^3+2*X+1	Menu GRAPH Y1= 4xX^3+2xX+1	$f(x) := 4*x^3+2*x+1$ ; ou pour des cas élaborés $f(x) := \{$ return 4*x^3+2*x+1 $\}$ ;	
Saisir une fonction Exemple de réponse	rep=input( " f(x) vaut :"+string" deff ('y=f(x)', "y="+rep 4*x^3+2*x+1		:Prompt Y <sub>1</sub>  "4*X^3+2*X+1		input(f) ou saisir(f)  x -> 4*x^3+2*x+1	
puis, obtenir une image $f(2)$ , $f(a)$	$f(2)$ , $f(a)$	$f(2)$ , $f(a)$	Y <sub>1</sub> (2), Y <sub>1</sub> (A), ATTENTION : Y <sub>1</sub> se trouve dans VARS /Y-VARS / Function	2→X : Y1 A→X : Y1 ATTENTION : Pour obtenir Y : VARS/GRPH/F1[Y]	Y1(2) Y1(A)  Sur 35+ USB	$f(2)$ , $f(a)$

(\*) Les fonctions Partie entière et Troncature diffèrent sur les nombres négatifs : si  $x=-2,31$  : partie entière : -3, troncature -2, partie décimale : 0,31, partie « fractionnaire » : 0,69

# Matrices

	Matrices : saisie, opérations				
	Scilab	Python 2.6 3.x seult.	TI	Casio	XCAS
Définir une matrice M carrée à n lignes et n colonnes, remplie de zéros		from numpy import*	touche MATRX / menu Edit	► MAT ou Menu / Mat - Sélectionner une matrice - Taper le nombre de lignes, par exemple 3, il apparaît 3x0 - Remplacer le 0 par le nombre de colonnes	M :=matrix [n,n] ; <i>Attention : les indices varient de 0 à n-1</i>
Saisie en ligne des coefficients de $\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$	M=[1 2 3 ;4 5 6 ;7 8 9] <i>On sépare les coefficients par des espaces ou des virgules Le point-virgule indique un passage à la ligne suivante</i>	M=array([(1,2,3),(4,5,6),(7,8,9)])	[[1,2,3][4,5,6][7,8,9]]  [M]	Compléter la matrice affichée	M := [[1,2,3],[4,5,6], [7,8,9]]
Utiliser un tableur pour saisir les coefficients	-	-	Directement dans le menu Matrix/EDIT	Directement dans ► MAT ou Menu / Mat	Ouvrir le tableur avec Alt+t Variable : M, nom de la matrice. On peut alors utiliser directement M dans les calculs et programmes de la session.
Créer la matrice identité d'ordre n					
Utiliser M dans une instruction	M	M	Menu Matrix/NOMS Affichage : [M]	Mat M dans OPTN / MAT	M
Accéder au coefficient de la ligne l et de la colonne c de la matrice	M(l,c)	M[l,c]	[M] (l,c)	Mat [l,c]	M[l,c]
Afficher une matrice dans un programme	disp(M) ou afficher (M)	print M	Matrix/NOMS M : [M] Enter	Mat M 	afficher( M) ;
Dimensions de M (liste des dimensions)	size (M) ou taille(M)	M.shape		DIM dans ► MAT	dim(M)
Additionner, multiplier deux matrices A et B	A+B A*B	A+B dot(A,B)	Affichage : [A]+[B] [A]*[B]	Mat A+Mat B Mat A*Mat B	A+B A*B
A^n (puissance entière de A) avec n fixé ( par exemple A^5 )	A^n	-	A^n	Mat A^n	A^n
A^n avec le paramètre n, A étant diagonalisable	-	-	-	-	assume n>0 ; B := matpow(A,n) ;
Multiplier M par un réel k	k*M ou M*k	k*M	Affichage : k*[M]	k*M	k*M
Matrice inverse de M	1/M	linalg.inv(M)	Touche x^-1 : [M]^-1	Touche x^-1 : Mat M^-1	inverse(M)
Résoudre A*X=B, X vaut :	A\B ou (1/A)*B	linalg.inv(A)*B	[A]^-1 * [B]	Mat A^-1 * Mat B	inverse(A)*B
Multiplication élément par élément	A.*B	A*B		-	A.*B
Remplir automatiquement une matrice à l'aide de formules, par exemple $m_{ij} = \frac{1}{i+j}$		Voir la ressource n°300	remplir	Fill ? Dans OPTN / MAT	