BOITAOUTILS (BOÎTE À OUTILS POUR J) R.Coquidé (01/12/2022)

CONSTANTES

```
RC =: 10{a.
```

Fin de ligne avec retour chariot (passage à la ligne)

```
MD =: 10 58 10{a.
```

Séparation monad--dyad dans une programmation explicite

OUTILS GENERAUX

```
E =: 1 !:2&2
```

Affiche et retourne son unique argument de droite

Ex :
```
    55 + E (E 22* 56 67 78) - E 20 <. 12 56 7
1232 1474 1716
12 20 7
1220 1454 1709
1275 1509 1764
```

```
NAT =: 4!:0
```

Nature des objets.

Retourne : **_2** (invalide), **_1** (inutilisé), **0** (nom), **1** (adverbe), **2** (conjonction), **3** (verbe)

Ex :
```
    NAT  <'Chrono'
3
    NAT '*xx';'bonjour';'RC';'TABLE';'TA';'Feuler'
_2 _1 0 1 2 3
```

```
TYP =: 3!:0
```

Type de données

Retourne : **1** (booléen), **2** (caractère), **4** (entier), **8** (flottant), **16** (complexe), **32** (en boîte), **64** (entier étendu), **128** (rationnel), **1024** (sparse booléen), **2048** (sparse caractère), **4096** (sparse entier), **8192** (sparse flottant), **16384** (sparse complexe), **32768** (sparse en boîte), **65536** (symbol), **131072** (caractère2), **262144** (caractère4).

Ex :
```
    a =. 0 0 1 0 [ b =.'ZERT'[ c =. 45
    d =. 3.14159 [ e =.4j8 [ f =. 234567x [ g =. 44r35
    TYP f
64
    TYP <f
32
    TYP a
1
    TYP b
2
    TYP a;b;c
32
    TYP &.> a;b;c;d;e;f;g
┌─┬─┬─┬─┬──┬──┬───┐
│1│2│4│8│16│64│128│
└─┴─┴─┴─┴──┴──┴───┘
```

```
P =: 9!:11
```

Fixe le nombre de chiffres affichés

Ex :
```
        P  15
        1p1
    3.14159265358979
```

```
EFFACE =: 4!:55
```

Efface des objets en les détruisant (permet de libérer de la mémoire RAM)

Ex :
```
        EFFACE <'LULU'
    1
        EFFACE 'RIRI';'JOJO';'MIMI'
    1 1 1
```

```
Chrono =: 6!:2
```

Mesure du temps d'exécution d'une instruction (variable selon état de l'ordinateur)

Ex : `Chrono 'W=.*:&.>1x+i. 1000000x'` NB. en secondes
```
    1.22680
        Chrono 'W=.*:&.>1x+i. 1000000x'
    2.25732
        10 Chrono 'W=.*:&.>1x+i. 1000000x'
    1.69680
```
NB. Moyenne sur 10 exécutions de l'instruction

CHAINES DE CARACTERES ET SUITES DE NOMBRES

```
SUP =: (' '&~:#]) : (([~:])#])
```

Suppression des blancs dans une chaîne de car (utilisation monadique)

Ex1 :
```
        SUP 'Bonjour chez vous'
    Bonjourchezvous
```

Suppression d'un caractère précis dans une chaîne (utilisation dyadique)

Ex2 :
```
        'Z' SUP 'aZERtydZzZcvZfZZgZqsyb'
    aERtydzcvfgqsyb
```

Suppression d'un nombre dans un vecteur numérique (utilisation dyadique)

Ex3 :
```
        5 SUP 6 5 7 5 5 4 2 5 9 5 8
    6 7 4 2 9 8
```

```
Sauf =: -.
```

Eléments du vecteur x sauf les éléments du vecteur y

```
SAUF =: /:~@~.@:-.
```

Idem avec suppression des dupliques et rangement dans l'ordre croissant

Ex :
```
        X=.12 45 34 6 3 5 7 34 12 5 7 9 62 [ Y=. 45 12 3
        X Sauf Y
34 6 5 7 34 5 7 9 62

        [A=.2 3 $ 45 12 3 12 34 9
    45 12 3
    12 34 9
        X Sauf A
    6 5 7 5 7 62
```

```
    X SAUF Y
5 6 7 9 34 62
    X SAUF A
5 6 7 62
```

```
Minr =: ((({:<}.),1:)*.1:,}.<}:)#]
```

Minis relatifs d’une suite de nombres : R=. Minr Suite

Ex :

```
    V=.55 3 8 9 12 15 8 6 9 2 4 6 0
    Minr V
3 6 2 0
```

```
Maxr =: ((({:>}.),1:)*.1:,}.>}:)#]
```

Maxis relatifs d une suite de nombres : R=. Maxr Suite

Ex :

```
    V=.55 3 8 9 12 15 8 6 9 2 4 6 0
    Maxr V
55 15 9 6
```

```
Pairs   =: (0:=2:|])#]
Impairs =: (1:=2:|])#]
```

Extraction des nombres entiers pairs et impairs d’une liste de nombres

Ex :

```
    V=.7 _8 5 6 32 9 1 _3 2 12 21 43.5 0 57 44.008 15.06
    Pairs V
_8 6 32 2 12 0
    Impairs V
7 5 9 1 _3 21 43 57
```

```
Delta =: 2&(-~/\)
```

Suite des différences des éléments d'une suite numérique

Ex :

```
    [S=. (i.13)^4
0 1 16 81 256 625 1296 2401 4096 6561 10000 14641 20736
    Delta S
1 15 65 175 369 671 1105 1695 2465 3439 4641 6095
    Delta Delta S
14 50 110 194 302 434 590 770 974 1202 1454
    Delta^:3 S
36 60 84 108 132 156 180 204 228 252
    Delta^:4 S
24 24 24 24 24 24 24 24 24
```

```
INTER  =: /:~@~.@:([-.-.)
UNION  =: /:~@~.@([,])
```

Intersection (éléments communs) et union de 2 chaînes de caractères ou 2 vecteurs numériques, avec suppression des dupliques et ordre croissant.

Ex :

```
    'bhedvchadsfredsxcgqk' INTER 'zxmunoaesbmpdf'
abdefsx
    6 4 2 7 3 2 2 4 45 98 77 22 45 INTER 66 45 45 12 6 2 88
2 6 45
    'A CHEVAL' UNION 'LA VACHE'
ACEHLV
    2 6 4 123 0 8 UNION 8 5 1 8 7 11 2 0
0 1 2 4 5 6 7 8 11 123
```

```
INS =: {{(z{.y),x,(z}.y) [ z=.m (([:#])|[)y}}
```

Insertion de composantes dans une chaîne ou un vecteur numérique :

Ex :
```
    (1000 2000) 2 INS 1 2 3 4 5 6 7 8 9
1 2 1000 2000 3 4 5 6 7 8 9
    (1000 2000) _3 INS 1 2 3 4 5 6 7 8 9
1 2 3 4 5 6 1000 2000 7 8 9
    '*+-' 3 INS 'ABCDEFGHIJKLM'
ABC*+-DEFGHIJKLM
    '*+-' _4 INS 'ABCDEFGHIJKLM'
ABCDEFGHI*+-JKLM
```

```
Chg =: 4 : '($y)$(1{x)(((0{x)=,y)#i.$,y)},y'
```

Change l'élément 0{x en 1{x dans y

Ex1 :
```
     'c!' Chg 'abcdtfgsbcadbcbeacdg'
 ab!dtfgsb!adb!bea!dg
```

Ex2 :
```
     V=. 5 7 0 2 0 1 9 4 0 3
     0 _1 Chg V
 5 7 _1 2 _1 1 9 4 _1 3
```

Chg travaille aussi avec des matrices

Ex3 :
```
     M; 0 _ Chg M=.4 5$?20$7
┌─────────┬─────────┐
│5 5 0 4 5│5 5 _ 4 5│
│0 0 4 3 5│_ _ 4 3 5│
│1 6 5 2 1│1 6 5 2 1│
│1 0 1 5 3│1 _ 1 5 3│
└─────────┴─────────┘
```

Ex4 :
```
     M;R=.'*.' Chg M=.7 6$'az*ml*p*i**hg*'
┌──────┬──────┐
│az*ml*│az.ml.│
│p*i**h│p.i..h│
│g*az*m│g.az.m│
│l*p*i*│l.p.i.│
│*hg*az│.hg.az│
│*ml*p*│.ml.p.│
│i**hg*│i..hg.│
└──────┴──────┘
```

```
SELC =: {{(u n{y)n}y}}
```

Monad f appliqué à une sélection de composantes du nom y

Ex 1 :
```
     (+: SELC 1 4) 10  10 10 10 10 10 10
 10 20 10 10 20 10 10
```

Ex2 :
```
     M;( (100&*) SELC ((<1;2),(<2;3),(<3;0)) ) M =. i.4 6
┌─────────────────┬───────────────────────┐
│ 0  1  2  3  4  5│   0  1   2    3  4  5│
│ 6  7  8  9 10 11│   6  7 800    9 10 11│
│12 13 14 15 16 17│  12 13  14 1500 16 17│
│18 19 20 21 22 23│1800 19  20   21 22 23│
└─────────────────┴───────────────────────┘
```

MANIPULATION DE MATRICES

```
SymH =: |.
```

Symétrique d une matrice (axe horizontal)

Ex : `M;M1 =. SymH M=. ?3 5$9`

```
┌─────────┬─────────┐
│2 2 6 1 7│3 5 2 7 3│
│0 2 8 7 5│0 2 8 7 5│
│3 5 2 7 3│2 2 6 1 7│
└─────────┴─────────┘
```

```
SymV =: |.&.|:
```

Symétrique d une matrice (axe vertical)

Ex : `M;M1 =. SymV M`

```
┌─────────┬─────────┐
│2 2 6 1 7│7 1 6 2 2│
│0 2 8 7 5│5 7 8 2 0│
│3 5 2 7 3│3 7 2 5 3│
└─────────┴─────────┘
```

```
SymC =: SymH @ SymV
```

Symétrique d une matrice (centre)

Ex : `M;M1 =. SymC M`

```
┌─────────┬─────────┐
│2 2 6 1 7│3 7 2 5 3│
│0 2 8 7 5│5 7 8 2 0│
│3 5 2 7 3│7 1 6 2 2│
└─────────┴─────────┘
```

```
Tr1 =: |:
```

Transposition (1ere diagonale) ou transposition classique

Ex : `M;Tr1 M`

```
┌─────────┬─────┐
│2 2 6 1 7│2 0 3│
│0 2 8 7 5│2 2 5│
│3 5 2 7 3│6 8 2│
│         │1 7 7│
│         │7 5 3│
└─────────┴─────┘
```

```
Tr2 =: |:&.|.
```

Transposition (2e diagonale)

Ex : `M;Tr2 M`

```
┌─────────┬─────┐
│2 2 6 1 7│3 5 7│
│0 2 8 7 5│7 7 1│
│3 5 2 7 3│2 8 6│
│         │5 2 2│
│         │3 0 2│
└─────────┴─────┘
```

Remarque : Les 5 précédents pro-verbes sont leur propre inverse.

```
Rotlig =: >@([|.&.><"1@])
```

Rotation sélective dans les lignes d une matrice

Ex1 :
```
      M;1 _2 3 Rotlig M=.i.3 5
┌──────────────┬──────────────┐
│ 0  1  2  3  4│ 1  2  3  4  0│
│ 5  6  7  8  9│ 8  9  5  6  7│
│10 11 12 13 14│13 14 10 11 12│
└──────────────┴──────────────┘
```

Ex2 :
```
      M;2 Rotlig M
┌──────────────┬──────────────┐
│ 0  1  2  3  4│ 2  3  4  0  1│
│ 5  6  7  8  9│ 7  8  9  5  6│
│10 11 12 13 14│12 13 14 10 11│
└──────────────┴──────────────┘
```

```
Rotcol =: Rotlig&.|:
```

Rotation sélective dans les colonnes d une matrice

Ex1 :
```
      M ;1 _1 0 2 0 Rotcol M
┌──────────────┬──────────────┐
│ 0  1  2  3  4│ 5 11  2 13  4│
│ 5  6  7  8  9│10  1  7  3  9│
│10 11 12 13 14│ 0  6 12  8 14│
└──────────────┴──────────────┘
```

Ex2 :
```
      M ;_2 Rotcol M
┌──────────────┬──────────────┐
│ 0  1  2  3  4│ 5  6  7  8  9│
│ 5  6  7  8  9│10 11 12 13 14│
│10 11 12 13 14│ 0  1  2  3  4│
└──────────────┴──────────────┘
```

```
Diag =: (<0 1)&|: :(([:(>:*i.)[:#]){})
```

Lecture des éléments diagonaux d'une matrice

Ex :
```
      [M=.10 + ? 5 5$90
25 96 82 73 91
61 71 76 72 21
59 34 57 90 74
86 72 67 16 21
17 47 71 33 17
      Diag M
25 71 57 16 17
```

Ecriture dans la diagonale d'une matrice

Ex1 :
```
      W=.0 6 2 7 3
      [M1=.W Diag M
 0 96 82 73 91
61  6 76 72 21
59 34  2 90 74
86 72 67  7 21
17 47 71 33  3
```

Ex2 :
```
      [M2=.0 Diag M
 0 96 82 73 91
61  0 76 72 21
59 34  0 90 74
86 72 67  0 21
17 47 71 33  0
```

```
Pk=: [:,./^:2*/
```

Produit de Kronecker de 2 matrices

Ex : `M1=.2 2$0 1 2 3     [   M2=. 2 3$ 1 1 2 0 2 1`

```
  M1;  M2 ; M1 pk M2
┌───┬─────┬───────────┐
│0 1│1 1 2│0 0 0 1 1 2│
│2 3│0 2 1│0 0 0 0 2 1│
│   │     │2 2 4 3 3 6│
│   │     │0 4 2 0 6 3│
└───┴─────┴───────────┘
```

```
Pbk=: [:,./^:2*./
```

Produit booléen de Kronecker de 2 matrices booléennes

```
Ex:  M1=.?2 3$2 [ M2=.? 2 2$ 2
   M1 ; M2; M1 pbk M2
┌─────┬───┬───────────┐
│0 1 1│1 1│0 0 1 1 1 1│
│1 1 0│0 1│0 0 0 1 0 1│
│     │   │1 1 1 1 0 0│
│     │   │0 1 0 1 0 0│
└─────┴───┴───────────┘
```

MATRICES DE BLOCS

```
Colenboite =: 3 : ('(''''''''';1)<;.1 y',MD,'(''''''''';x)<;.1 y')
```

Range les colonnes d une matrice dans des boites

Ex1 : `Colenboite M=. 10+?3 7$90`   NB. Utilisation monadique

27 50 20	88 40 40	41 46 71	46 69 91	25 66 98	82 47 73	50 76 90

Ex2 : `1 0 1 1 0 0 1 Colenboite M`   NB. Utilisation dyadique

27 88 50 40 20 40	41 46 71	46 25 82 69 66 47 91 98 73	50 76 90

```
Ligenboite =: 3 : (',.(1;'''''''')<;.1 y',MD,',.x<;.1 y')
```

Range les lignes d une matrice dans des boîtes

Ex1 : `Ligenboite M=.10+?5 6$90`   NB. Utilisation monadique

79 42 44 95 91 66 77
45 21 10 77 83 81 64
44 97 29 68 76 26 22
25 42 48 90 94 84 44

Ex2 : `1 1 1 0 Ligenboite M` NB. Utilisation dyadique

```
+--------------------+
|79 42 44 95 91 66 77|
+--------------------+
|45 21 10 77 83 81 64|
+--------------------+
|44 97 29 68 76 26 22|
|25 42 48 90 94 84 44|
+--------------------+
```

```
MatBloc =: <;.1
```

Découpe une matrice en blocs

Ex1 : `MatBloc 10+?3 6$80` NB. Utilisation monadique

```
+-----------------+
|21 47 25 56 83 24|
|87 39 11 49 57 10|
|59 37 14 63 46 33|
+-----------------+
```

NB. Utilisation dyadique

Ex2 : `[MB=.(1 0 1 0 0;1 0 0 1 1 0 1 0) MatBloc 10+?5 8$90`

```
+--------+--+-----+-----+
|68 53 64|30|51 44|79 10|
|31 50 56|49|12 97|58 50|
+--------+--+-----+-----+
|11 67 33|71|53 75|75 12|
|52 54 15|51|48 77|47 28|
|43 32 56|14|24 80|72 59|
+--------+--+-----+-----+
```

```
MatDbloc =: [: > 3 : ',"(#$y)&.>/y'^:(#@$)
```

Suprime les blocs d une matrice de blocs

Ex : `MatDbloc MB`

```
68 53 64 30 51 44 79 10
31 50 56 49 12 97 58 50
11 67 33 71 53 75 75 12
52 54 15 51 48 77 47 28
43 32 56 14 24 80 72 59
```

TABLEAUX DIVERS

```
TA =: 2 : ('[:',MD,'(n;,.x),.({.;}.)":y,x u/y')
```

Résultat d'un verbe mis sous forme d'un tableau

Ex1 :
```
HYPOTHENUSE =: +/&.*:
3 5 7 HYPOTHENUSE TA 'HYP' 2 4 6 8
```

```
+---+-----------------------------------+
|HYP|2       4       6       8          |
+---+-----------------------------------+
|  3|3.60555 5       6.7082  8.544      |
|  5|5.38516 6.40312 7.81025 9.43398    |
|  7|7.28011 8.06226 9.21954 10.6301    |
+---+-----------------------------------+
```

Ex2:

```
2 12 88 +. TA 'PGCD' V=.2 3 4 6 12 30 36 44 88
```

PGCD	2	3	4	6	12	30	36	44	88
2	2	1	2	2	2	2	2	2	2
12	2	3	4	6	12	6	12	4	4
88	2	1	4	2	4	2	4	44	88

```
TAB =: 2 : ('[:',MD,'(x;(;:n)),((;:m),.<"0 y)')
```

Ex :

```
M =. 3 4 $9 6 5 7 2 0 5 8 9 3 1 8
'Ventes' 'Eu Paris Aix' TAB 'Vis Clous Ecrous Boulons' M
```

Ventes	Vis	Clous	Ecrous	Boulons
Eu	9	6	5	7
Paris	2	0	5	8
Aix	9	3	1	8

```
TABLH =: ([:;:[),:[:<"0 ]
```

Ex :

```
'A B C D' TABLH 4 15 308 7
```

A	B	C	D
4	15	308	7

```
TABLV =: ([:;:[),.[:<"0]
```

Ex :

```
'Vis Clous Boulons' TABLV 12.4 7.35 5.2
```

Vis	12.4
Clous	7.35
Boulons	5.2

```
TABLE =: 1 : (':'; '(((#~LF-.@e.])5!:5<''u'');,.y),.({.;_}.)":x,y u/x')~
```

Ex1 :

```
HYP =: +&.*:   NB. hypthénuse  ( avec P 4 )
3 5 9  HYP  TABLE  2 3 4 5 6
```

HYP	2	3	4	5	6
3	3.606	4.243	5	5.831	6.708
5	5.385	5.831	6.403	7.071	7.810
9	9.220	9.487	9.849	10.30	10.82

Ex2 : `0 1 *. TABLE 0 1` NB. Table de vérité du « et »

```
┌──┬───┐
│*.│0 1│
├──┼───┤
│0 │0 0│
│1 │0 1│
└──┴───┘
```

```
TABBLOC =: 2 : ('[:',MD,'((<'' ''),(;:n)),(;:m),.x <;.1 y')
```

Ex : `V=. 'Eu Paris Aix' [ W=. 'vis boulons ecrous clous'`
`(1 0 1 1 0;1 1 0 0 1 1 0) V TABBLOC W  (?5 7$99)`

	vis	boulons	ecrous	clous
Eu	46 2	9 37 23 81 52 59	68 6	89 78 0 93
Paris	1	25 88 73	19	0 84
Aix	13 9	94 11 37 75 14 53	20 92	61 75 17 73

```
Factor=:3 : '~./:~((#:i.@(*/))#~#q)<@(/:~)@(*//.)"1 q=.q: y'
```

Décompositions d'un nombre en produits de facteurs

Ex : `Factor 105`

3 5 7	3 35	5 21	7 15	105

Pro-conjonctions (divers produits de composition de 2 verbes ou pro-verbes)

Utilisation monadique

		Programmation	Résultat
`UVY =: 2 : 'u@v'`	NB.	**(u UVY v) y**	**u(v y)**
`VUY =: 2 : 'v@u'`	NB.	**(u VUY v) y**	**v(u y)**
`UY_VY =: 2 : '([:u])v]'`	NB.	**(u UY_VY v) y**	**(u y)v y**
`VY_UY =: 2 : '([:v])u]'`	NB.	**(u VY_UY v) y**	**(v y)u y**
`U_YVY =: 2 : 'u@(]v])'`	NB.	**(u U_YVY v) y**	**u(y v y)**
`V_YUY =: 2 : 'v@(]u])'`	NB.	**(u V_YUY v) y**	**v(y u y)**
`YU_VY =: 2 : ']u[:v]'`	NB.	**(u YU_VY v) y**	**y u (v y)**
`YV_UY =: 2 : ']v[:u]'`	NB.	**(u YV_UY v) y**	**y v (u y)**

Utilisation dyadique

`XUVY =: 2 : 'u v'`	NB.	**x (u XUVY v) y**	**x u (v y)**
`YUVX =: 2 : '(u v)~'`	NB.	**x (u YUVX v) y**	**y u (v x)**
`UY_VX =: 2 : 'v~u'`	NB.	**x (u UY_VX v) y**	**(u y) v x**
`UX_VY =: 2 : '(v~u)~'`	NB.	**x (u UX_VY v) y**	**(u x) v y**
`VY_UX =: 2 : 'u~v'`	NB.	**x (u VY_UX v) y**	**(v y) u x**
`VX_UY =: 2 : '(u~v)~'`	NB.	**x (u VX_UY v) y**	**(v x) u y**
`UXVY =: 2 : 'u@v'`	NB.	**x (u UXVY v) y**	**u (x v y)**
`VXUY =: 2 : 'v@u'`	NB.	**x (u VXUY v) y**	**v (x u y)**
`UYVX =: 2 : 'u@(v~)'`	NB.	**x (u UYVX v) y**	**u (y v x)**
`VYUX =: 2 : 'v@(u~)'`	NB.	**x (u VYUX v) y**	**v (y u x)**

Ex :
```
   X =. 6 [ Y =. 5
   (% UVY ^) Y
0.00673795
   (% YV_UY ^) Y
1.37973
   X (% UY_VX ^) Y
6.4e_5
   X (% VX_UY ^) Y
80.6858
```

ARRONDIS

```
Flot =: x:^:_1
```

Ecriture approximative en flottant de nombres entiers étendus ou rationnels (ici avec P 8)

Ex :
```
   Flot _73r119 1234567891213141517856x
_0.61344538 1.2345679e21
```

Approximations :

```
arpe=:(0 arpe]) :((10^[)*[:>.]%10^[)"0 0 0          NB. Par excès
arpd=:(0 arpd]) :((10^[)*[:<.]%10^[)"0 0 0          NB. Par défaut
arpp=:(0 arpp]) :((10^[)*[:<. 0.5+]%10^[)"0 0 0     NB. Au plus près
```

Ex1 : 2 chiffres après le point décimal
```
   _2 arpp 53.96543 _6.90743  8.7245     NB. Au plus près
53.97 _6.91 8.72
   _2 arpe 53.96543 _6.90743  8.7245     NB. Par excès
53.97 _6.9 8.73
   _2 arpd 53.96543 _6.90743  8.7245     NB. Par défaut
53.96 _6.91 8.72
```

Ex2 : 2 chiffres avant le point décimal (multiples de 100)
```
   2 arpp 4567.8 _65432.965 12345.89     NB. Au plus près
4600 _65400 12300
   2 arpe 4567.8 _65432.965 12345.89     NB. Par excès
4600 _65400 12400
   2 arpd 4567.8 _65432.965 12345.89     NB. Par défaut
4500 _65500 12300
```

Ex3 : par valeurs entières
```
   0 arpp 4567.8 _65432.965 12345.89     NB. Au plus près
4568 _65433 12346
   0 arpe 4567.8 _65432.965 12345.89     NB. Par excès
4568 _65432 12346
   0 arpd 4567.8 _65432.965 12345.89     NB. Par défaut
4567 _65433 12345
```

```
AR0 =: (([<|@])*])"0 0
```

Arrondit à 0 les éléments de y de valeur absolue < x   (x>0)

NB. ***R=. x AR0 y***

Ex :
```
   1e_7 AR0 14 0.024 _0.000000089 _0.007 1r123456789
14 0.024 0 _0.007 0
```

Egalité avec tolérance (m entier)

```
EGTA =: 1 : ('[:',MD,'0=<.(10^-m)*|x-y')
```

Tolérance absolue 10^m

```
EGTR =: 1 : ('[:',MD,'0=<.(10^-m)*(|x-y)%(|x)<.(|y)')
```

Tolérance relative 10^m

Ex:
```
   50 (3 EGTA) 5 0.000005 0.05 50 500 5000 5000000
1 1 1 1 1 0 0
```
Ex :
```
   50 (3 EGTR) 5 0.000005 0.05 50 500 5000 5000000
1 0 1 1 1 1 0
```

```
SURF =: 11&o.@-:@(+ +/ .* 1&|.) :([:SURF[j.])
```

Surface algébrique délimitée par un polygone plan défini par ses sommets

Ex1 :
```
   4 6 8 9 7 5 2 3 SURF 2 3 5 7 6 4 3 1
11
   SURF 4j2 6j3 8j5 9j7 7j6 5j4 2j3 3j1
11
```
Ex2 :
```
   4.56 7.89 9.54 4.12 2.34 SURF 1.2 7 5.3 8.04 5.76
16.9073
   SURF 4.56j1.2 7.89j7 9.54j5.3 4.12j8.04 2.34j5.76
16.9073
```

QUELQUES FONCTIONS CLASSIQUES

```
Feuler =: !@<: :(*&$:%$:@+)
```

Fonctions gamma (forme monadique) et beta (forme dyadique) d'EULER

$$Gamma(t) = \int_0^{\infty} e^{-z} z^{t-1} dz \;\; ; t > 0 \text{ (s'étend à t} \in E = \mathbb{C}\text{-\{0 \_1 \_2 \_3 ...\})}$$

$$Beta(p,q) = \int_0^{1} z^{p-1}(1-z)^{q-1} dz \;\; ; p > 0 \text{ et } q > 0 \text{ (s'étend à } p \in E \text{ et } q \in E)$$

Ex1 : Calcul de Gamma(5),Gamma(5.7531),Gamma(_3.2),Gamma(1j2)
```
   Feuler  5  5.7531  _3.2  1j2
24 79.191 0.689056 0.151904j0.0198049
```

Ex2 : Calcul de Beta(3,5),Beta(5.2,_3.7),Beta(1j3,5j2)
```
   3  5.2  1j3  Feuler  5  _3.7  5j2
0.00952381 9.25055 0.0096253j0.037409
```

```
Fgauss =: 1r2p1&%@{.@(((+/%#)&.:^.,+/%#)^:_)
```

La fonction de GAUSS

$$Fgauss(a,b) = \frac{\pi}{2M_{ag}(a,b)} \text{ où } a > 0 \text{ , } b > 0$$

$M_{ag}(a,b)$=Moyenne arithmético-géométrique de a et b est utile pour calculer certaines intégrales :

$$\int_0^\infty \frac{dt}{\sqrt{(t^2+a^2)(t^2+b^2)}} = 2\int_0^{\sqrt{ab}} \frac{dt}{\sqrt{(t^2+a^2)(t^2+b^2)}} = \frac{1}{2}\int_0^\infty \frac{dt}{\sqrt{t(t+a^2)(t+b^2)}} = Fgauss(a,b) \text{ où } a>0,\ b>0$$

$$\int_0^\infty \frac{dt}{\sqrt{t^4+s^2t^2+p^2)}} = 2\int_0^{\sqrt{p}} \frac{dt}{\sqrt{t^4+s^2t^2+p^2)}} = Fgauss(\frac{1}{2}\sqrt{2p+s},\sqrt{p}) \text{ où } |s|<2p$$

Ex : Calcul de Fgauss(a,b)

```
        Fgauss 3 5.8
    0.366588570138
        Fgauss 0.35 9.82
    0.480827
```

```
    BetaI=:{{)d
'p q'=.x
if. y<:1r2 do. ((y^p)%p)*(p,-.q)H.(p+1)y
else. (p Feuler q)-(|.x) BetaI -.y end.
}}"_ 0
```

Ft beta incomplète

R=. (p,q) BetaI z 0<:z<:1 et p,q réels ou complexes sauf _1,_2,_3,_4 **...**

Ex : **(2.3 1.56) BetaI 0.68**

```
0.123763
    (2j3 _1j4.5) BetaI 0.75
1.24936j0.320299
```

`POC =: ^!.1`	NB. Symbole de Pochhammer de 1e espèce
`Poc =: ^!._1`	NB. Symbole de Pochhammer de 2e espèce

a POC k <==> $(a)_k$ = a(a+1)(a+2)...(a+k-1) = Gamma(a+k) / Gamma(a) ; $(a)_0 = 1$

a Poc k <==> $[a]_k$= a(a-1)(a-2)...(a-k+1) = Gamma(a) / Gamma(a-k) ; $[a]_0 = 1$

avec $k \in \mathbb{N}$ et $a \in E_1 = \mathbb{C} - \{0,-1,-2,-3...\}$

Ex : **5 4.3 2.4j6.5 _3j5.1 POC 3**

```
210 143.577 _395.046j_55.705 150.06j_76.551
    5 4.3 2.4j6.5 _3j5.1 Poc 3
60 32.637 _176.106j_242.905 252.12j107.049
```

```
Fh   =: {{(>0{x)H.(>1{x)y}}"1 0
DFh  =: {{((*/x0)%*/x1)*(1+x0=.>0{x)H.(1+x1=.>1{x)y}}"1 0
D2Fh =: {{((*/x0*1+x0)% */x1*1+x1)*(2+x0=.>0{x)H.(2+x1=.>1{x)y}}"1 0
```

Fonctions Hypergéométriques, dérivées 1e et 2e

Ex : **(1 2 3;4 5 6) Fh 0.27 0.82 2.35**

```
1.01371 1.04302 1.13584
    (1 2 3;4 5 6) DFh 0.27 0.82 2.35
0.0515809 0.0550525 0.0668786
    (1 2 3;4 5 6) D2Fh 0.27 0.82 2.35
0.00599889 0.0066397 0.00897348
```

```
sint =: {.@]+-/@|.@]*i.@>:@[%[
```

Découpage de l intervalle y en x sous-intervalles de même longueur

Ex : **10 sint 0 2**

0 0.2 0.4 0.6 0.8 1 1.2 1.4 1.6 1.8 2

```
     MINu =: {{)a
I=.i.10 [ Y=.y
for_i. I do.
zm=.<./Z=.u"0 X=.20 sint Y
Y=.(_1 1 + Z i. zm){X end.
-:+/Y
}}
```

Calcul de l'Abscisse du mini d'une ft continue sans utilisation de dérivée

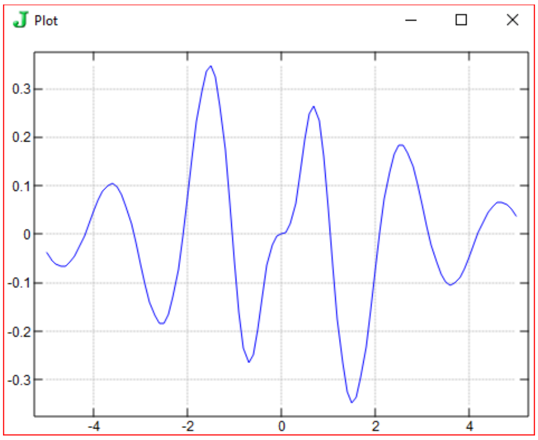
NB. ***R=. f MINu A,B*** NB. [A,B] intervalle de recherche de Y=f(X)

Ex : soit la fonction fi(x) = Arctg(x^2)*sin(3.x)/(1+x^2) qui se programme en J

```
    fi=: {{(1 o. 3*y)*(_3 o.*:y)%1+*:y}}
    P 12
    fi MINu _5 5
1.51542245525
```

Vérifions avec un « plot »

```
    plot(];fi) 100 sint _5 5
```



Pour le polynôme de LEGENDRE de degré 7 (objectif : intégration)

racines (LEGr7), coefficients de pondération (LEGp7) :

LEGr7=: _0.949107912342759 _0.741531185599395 _0.405845151377397 0

LEGr7=:LEGr7, 0.405845151377397 0.741531185599394 0.949107912342759

LEGp7=: 0.0647424830844263 0.139852695744671 0.190915025252494 0.208979591836819

LEGp7=:LEGp7,0.19091502525249 0.139852695744674 0.0647424830844254

```
     INTGLC =: {{)a
msi=.(>@(2&(<\))) x sint y
VR=.i.0 [ I=.i.x
for_i. I do. VR=.VR,+/LEGp7*u LEGr7 {{0.5*(+/y)-x*(-/y)}}"0 _ i{msi end.
(+/VR)*(--/y)%x
}}
```

Intégration numérique méthode de Gauss-Legendre composite utilisant le polynôme de LEGENDRE de degré 7. (a,b) : intervalle d'intégration, n : nb de subdivisions, f : fonction à intégrer

NB. ***R =. n f INTGLC a,b***

Ex : `100 ((1 _3 2&p.)*(1&o.)) INTGLC 0.1 1p1`

```
4.31038
    fi=: (1&o.)@*:
    400 fi INTGLC _1r2p1 1r2p1
1.65623
```

FONCTIONS À RÉPONSE BOOLÉENNE   B ϵ {0,1}

BOITE01 =: 32&=@(3!:0)	**ETENDU01 =: 64 e.~3!:0**
OUVERT01 =: 32&~:@(3!:0)	**RATION01 =: 128 e.~3!:0**
GEROND01 =: 0:`(2 32 e.~3!:0@>)@.(32=3!:0)"0	**FLOT01 =: 8"_=3!:0**
SCAL01 =: 0=#@$	**COMPLEXE01 =: 16"_=3!:0**
VECT01 =: 1=#@$	**NONDUP01 =: [:-.0 e.~:**
MATR01 =: 2=#@$	**XDANSY01 =: +./@:E."1**
HMATR01 =: 3<:#@$	**IDLIBRE01 =: _1=[:NAT]**
CARNUM01 =: ([:*./'0123456789'e.~(}.~e.&' +-_'@{.))"1	**NOM01 =: 0=[:NAT]**
CARAC01 =: 2&=@(3!:0)	**ADVERBE01 =: 1=[:NAT]**
UNICODE01 =: 131072"_=3!:0	**CONJ01 =: 2=[:NAT]**
BIN01 =: 1 e.~3!:0	**VERBE01 =: 3=[:NAT]**
ENTIER01 =: 1 4 e.~3!:0	

UTILISATION (réponse : 0 (non) ou 1 (oui))   ID : identifiant d'un objet

B =. BOITE01 ID	: est-ce une boîte ?
B =. OUVERT01 ID	: est-ce ouvert (non emboîté) ?
B =. GEROND01 ID	: est-ce un gérondif ?
B =. SCAL01 ID	: est-ce un scalaire ? (rang 0)
B =. VECT01 ID	: est-ce un vecteur (ou une chaîne) ? (rang 1)
B =. MATR01 ID	: est-ce une matrice ? (rang 2)
B =. HMATR01 ID	: est-ce une hyper-matrice ? (rang > 2)
B =. CARNUM01 ID	: est-ce du chiffre littéral (ASCII) (premier caractère éventuellement blanc + - _) ?
B =. CARAC01 ID	: est-ce du littéral (ASCII) ?
B =. UNICODE01 ID	: est-ce du littéral (unicode) ?
B =. BIN01 ID	: est-ce du binaire ?
B =. ENTIER01 ID	: est-ce du numérique entier ?
B =. ETENDU01 ID	: est-ce de l'entier étendu ?
B =. RATION01 ID	: est-ce du rationnel ?
B =. FLOT01 ID	: est-ce du numérique flottant ?
B =. COMPLEXE01 ID	: est-ce du numérique complexe ?
B =. NONDUP01 ID	: est-ce dépourvu de dupliques ?
B =. x XDANSY01 y	: la chaîne x est-elle dans la chaîne y ?
B =. IDLIBRE01 <'ID'	: est-ce un ID correct non encore utilisé ?
B =. NOM01 <'ID'	: est-ce un NOM ?
B =. ADVERBE01 <'ID'	: est-ce un ADVERBE ?
B =. CONJ01 <'ID'	: est-ce une CONJONCTION ?
B =. VERBE01 <'ID'	: est-ce un VERBE ?

Re =: 9&o.	Im =: 11&o.	Ro=: 10&o.	Ar=: 12&o.

Partie réelle (Re), imaginaire (Im), module (Ro) et argument (Ar) d'un nombre complexe complexe

Ex :
```
    Re 5 _3.2j6.1 89j4.9 12j0.002 0j8
5 _3.2 89 12 0
    Im 5 _3.2j6.1 89j4.9 12j0.002 0j8
0 6.1 4.9 0.002 8
```
NB.
```
    Ro 5 _3.2j6.1 89j4.9 12j0.002 0j8
5 6.88839604 89.1347856 12.0000002 8
    Ar 5 _3.2j6.1 89j4.9 12j0.002 0j8
0 2.05392199 0.0550006523 0.000166666665 1.57079633
```