CodeXlogo : Fractions

POUR PGCD :T1 :T2 5
POUR PPCM :T1 :T2 5
POUR MULTIPLE? :N :Q 5
POUR ETRANGERS? :A :B 5
POUR QUOTIENT2 :O :D 5
POUR RESTE2 :O :D 5
POUR GZÉROS :N 6
POUR GZ :X :R 6
POUR GNZQ :N 6
POUR GNZG :N 6
POUR GFTZ :O :N 6
POUR GFTE :O :N 6
POUR GTETE :O :X 6
POUR GTET2 :O :X 6
POUR GQUEUE :O :X 6
POUR GTRONQUED :O :X 7
POUR GENTIER? :N 7
POUR GNUL? :N 7
POUR GNÉGATIF? :N 7
POUR GOPPOSÉ :N 7
POUR GPLG? :N1 :N2 7
POUR GPLP? :N1 :N2 7
POUR GMAX :A :B 7
POUR GSOMME :T1 :T2 7
POUR GDIFFÉRENCE :T1 :T2 8
POUR GPRODUIT :T1 :T2 8
POUR GPROD1 :T1 :T2 9
POUR GPROD2 :T1 :T2 9
POUR GQUOTIENT :T1 :T2 9
POUR GQUOT2 :T1 :T2 9
POUR GRESTE :T1 :T2 10
POUR FRACTION :F 10
#LES FONCTIONS DE BASE 10
POUR FRACTION? :F 10
POUR RATIONNEL? :O 10
POUR SIMPLIFIABLE? :F 10
POUR SIMPLIFIE :F :Q 10
POUR IRRÉDUCTIBLE :F 10
POUR BNF :O 11
POUR BNF2 :N :D 11
POUR NUMÉRATEUR :O 11
POUR DÉNOMINATEUR :O 11
POUR NUMÉR :O 11
POUR DÉNOM :O 11
POUR DÉCIMAL :F 11
POUR FÉGAL? :F1 :F2 11
POUR FPLP? :F1 :F2 12
POUR FPLG? :F1 :F2 12
POUR FSOMME :F1 :F2 12
POUR FSOM2 :F1 :F2 12
POUR FPRODUIT :F1 :F2 12
POUR FPRODUIT2 :F1 :F2 12
POUR FOPPOSÉ :F 12
POUR FDIFF :F1 :F2 12
POUR FDIV :F1 :F2 12
POUR CONVERTNL :O 13
POUR RECHPOS :C :O 13

POUR RECHPO2 :C :O :N 13
POUR RECHJQ :C :O :R 13
POUR ÉVALUE :O 13
POUR ÉGAL? :O1 :O2 13
POUR RIEN 13
POUR LDC 13
POUR LANCERÉGLAGES 13
POUR RESTAURERÉGLAGES 13
POUR SAUVECONTEXTE 13
POUR RESTAURECONTEXTE 14
POUR FIXENEWCONTEXTE 14
POUR TAILLEMAXAFFICHE 14
POUR AFFICHECADRE :XG :XD :YH :YB 15
POUR AFFICHEMODE 15
POUR AFFMARQUE :C 15
POUR AFFICHENUM3 15
POUR AFFICHENUM2 15
POUR RECHPOLICE 15
POUR APPELRCHPOL :NMPL :DELTA :POL1 :NUMXPOL 16
POUR RCHPOL :NMPL :NMAX :POL1 :NUMXPOL 16
POUR INITCLAV 16
POUR LANCESAISIECLAV 16
POUR INPUTNUM2 16
POUR ANALTOUCHE :LATOUCHE 17
POUR ROGNEACCU2 17
POUR BARRE2 18
POUR POINTE2 18
POUR COMPLETE2 :TOUCHE 18
POUR FIXECARADMIS 18
POUR ITEREOP 18
POUR FRECOPIE :O1 :O2 19
POUR POINTOTÉ 19
POUR SLASHOTÉ 19
POUR RETIRE :O :C 19
POUR INJECTE :O :C 19
POUR MAJUS2 :CODE 19
POUR OPPOSE2 19
POUR INVERSE2 19
POUR REDUIT2 20
POUR DÉCIMAL2 20
POUR FRACTION2 20
POUR TRANSFERT1 20
POUR TRANSFERT2 21
POUR EXCHANGE2 21
POUR VIDECODEOP2 21
POUR MEMORISE2 21
POUR BACKUP2 21
POUR VIDANGEACCU2 22
POUR VIDANGETOUT2 22
POUR INVMODE2 22
POUR SIMPLIFIE2 22
POUR EXPANS2 22
POUR PREMIERDIVISEUR :N 23
POUR PQB :O 23
POUR VERIFCODEOP 23
POUR ERRORACCU2 23
POUR ALERTEERRORACCU2 23
POUR ENDERNIER? :C :O 23
POUR NUL? :O 23
POUR RETOURCHARIOT :PIX 23

POUR CONSIGNES **23**
POUR GLISSEX :O **24**
POUR POINTENDERNIER? :O **24**
POUR SLASHENDERNIER? :O **25**
POUR ALERTELONGACCU2 ..... **25**
POUR LANCEALERTEACCU2 :LACOM **25**
POUR TROPLONGACCU2 ..... **25**
POUR EFFACENUM1 ..... **25**
POUR EFFACENUM2 ..... **25**
POUR EFFACENUM3 ..... **25**
POUR ALERTETAMPONPLEIN **25**
POUR ALERTELONGACCU1 ..... **25**
POUR DIGIT? :O **25**
POUR INCRECOMPTEUR **26**
POUR RAZCOMPTEUR **26**
POUR AFFICHECOMPTEUR **26**
POUR EFFACECOMPTEUR **26**

P

pour affichecadre :xg :xd :yh :yb 15
pour affichecompteur 26
pour affichemode 15
pour affichenum2 15
pour affichenum3 15
pour affmarque :c 15
pour alerteerroraccu2 23
pour alertelongaccu1 25
pour alertelongaccu2 25
pour alertetamponplein 25
pour analtouche :latouche 17
pour appelrchpol :nmpl :delta :pol1 :numxpol 16
pour backup2 21
pour barre2 18
pour bnf :o 11
pour bnf2 :n :d 11
pour complete2 :touche 18
pour consignes 23
pour convertnl :o 13
pour décimal :f 11
pour décimal2 20
pour dénom :o 11
pour dénominateur :o 11
pour digit? :o 25
pour effacecompteur 26
pour effacenum1 25
pour effacenum2 25
pour effacenum3 25
pour égal? :o1 :o2 13
pour endernier? :c :o 23
pour erroraccu2 23
pour etrangers? :a :b 5
pour évalue :o 13
pour exchange2 21
pour expans2 22
pour fdiff :f1 :f2 12
pour fdiv :f1 :f2 12
pour égal? :f1 :f2 11
pour finverse :f 12
pour fixecaradmis 18
pour fixenewcontexte 14
pour fopposé :f 12
pour fplg? :f1 :f2 12
pour fplp? :f1 :f2 12
pour fproduit :f1 :f2 12
pour fproduit2 :f1 :f2 12
pour fraction :f 10
pour fraction? :f 10
pour fraction2 20
pour frecopie :o1 :o2 19
pour fsom2 :f1 :f2 12
pour fsomme :f1 :f2 12
pour gdifférence :t1 :t2 8
pour gentier? :n 7
pour gfte :o :n 6
pour gftz :o :n 6
pour glissex :o 24
pour gmax :a :b 7
pour gnégatif? :n 7
pour gnul? :n 7
pour gnzg :n 6
pour gnzq :n 6
pour gopposé :n 7
pour gplg? :n1 :n2 7
pour gplp? :n1 :n2 7
pour gprod1 :t1 :t2 9
pour gprod2 :t1 :t2 9
pour gproduit :t1 :t2 8
pour gqueue :o :x 6
pour gquot2 :t1 :t2 9

pour gquotient :t1 :t2 9
pour greste :t1 :t2 10
pour gsomme :t1 :t2 7
pour gtet2 :o :x 6
pour gtete :o :x 6
pour gtronqued :o :x 7
pour gz :x :r 6
pour gzéros :n 6
pour increcompteur 26
pour initclav 16
pour injecte :o :c 19
pour inputnum2 16
pour inverse2 19
pour invmode2 22
pour irréductible :f 10
pour itereop 18
pour lancealerteaccu2 :lacom 25
pour lanceréglages 13
pour lancesaisieclav 16
pour ldc 13
pour majus2 :code 19
pour memorise2 21
pour multiple? :n :q 5
pour nul? :o 23
pour numér :o 11
pour numérateur :o 11
pour oppose2 19
pour pgcd :t1 :t2 5
pour pointe2 18
pour pointendernier? :o 24
pour pointoté 19
pour ppcm :t1 :t2 5
pour pqb :o 23
pour premierdiviseur :n 23
pour quotient2 :o :d 5
pour rationnel? :o 10
pour razcompteur 26
pour rchpol :nmpl :nmax :pol1 :numxpol 16
pour rechjq :c :o :r 13
pour rechpolice 15
pour rechpos :c :o 13
pour reduit2 20
pour restaurecontexte 14
pour restaureréglages 13
pour reste2 :o :d 5
pour retire :o :c 19
pour retourchariot :pix 23
pour rien 13
pour rogneaccu2 17
pour sauvecontexte 13
pour simplifiable? :f 10
pour simplifie :f :q 10
pour simplifie2 22
pour slashendernier? :o 25
pour slashoté 19
pour taillemaxaffiche 14
pour transfert1 20
pour transfert2 21
pour troplongaccu2 25
pour verifcodeop 23
pour vidangeaccu2 22
pour vidangetout2 22
pour videcodeop2 21

```
#----------------------------------------
# attention ci-dessous, versions fonctionnelles pour mon ldc
# pour une utilisation plus générale utiliser les procédures gproduit au lieu de produit, greste au lieu de reste etc.

pour pgcd :t1 :t2
#t1 et t2 sont assumés positifs
soit "repons 0
soit "aux 0
soit "lereste 0
tantque [:repons = 0] [
si :t1 < :t2 [donne "aux :t1 donne "t1 :t2 donne "t2 :aux] #un petit swap
donne "lereste reste :t1 :t2
si egal? 0 :lereste [donne "repons :t2] [donne "aux :t2 donne "t2 :lereste donne "t1 :aux]
] # fin de la boucle
ret :repons
fin

pour ppcm :t1 :t2
si :t1 < :t2 [ret ppcm :t2 :t1]
# t1 est donc le plus grand
soit "p pgcd :t1 :t2
soit "q quotient :t1 :p
ret produit :q :t2
fin

# on gère des grands nombres considérés comme non entiers par XLogo
# Après coup, j'ai découvert les ressorts cachés de la primitive fdecimales …

pour multiple? :n :q
ret egal? 0 reste2 :n :q
fin

pour etrangers? :a :b
ret  egal? 1 pgcd2 :a :b
fin

pour quotient2 :o :d
# pas au sens des matheux : on travaille au signe près
si :o < 0 [ret moins quotient2 (moins :o) :d]
si :d < 0 [ret moins quotient2 :o (moins :d)]
# à ce point les deux arguments sont positifs
si :d > :o [ret 0]
si ou membre? ". :o membre? ". :d [ret quotient2 produit :o 10 produit :d 10] [ret quotient :o :d] # 1er terme de l'alternative par sécurité ...
fin

pour reste2 :o :d
# pas au sens des matheux : on travaille au signe près
si :o < 0 [ret moins reste2 (moins :o) :d]
si :d < 0 [ret moins reste2 :o (moins :d)]
# à ce point les deux arguments sont positifs
si  :d > :o [ret :o]
si gentier? :o [ret reste :o :d]
soit "r  :o moins :d * quotient2 :o :d # séquelle old version :: ne sert plus.
ret :r #temporaire
fin
#------------------------------
```

```
# on plonge ici l'arithmétique sur les très gros entiers
# mais en fait je n' ai pas besoin de tout grace à la primitive  fixedecimales
pour gzéros :n
# retourne :n zéros
si :n < 1 [ret "]
si :n < 11 [ret gz :n "] [ret mot "0000000000 gzéros (:n - 10)]
fin

pour gz :x :r
si egal? 0 :x [ret :r] [ret gz (:x - 1) mot :r 0]
fin


pour gnzq :n
# renvoit le nombre de zéros à droite de n, assuré nombre entier
si vide? :n [ret 0]
si non egal? 0 der :n [ret 0]
ret 1 + gnzq sd :n
fin

pour gnzg :n
#enlève les zéros excédentaires à gauche de l'écriture de :n
si ou (egal? 0 :n) (vide? :n) [ret 0]
si non egal? 0 prem :n [ret :n]
ret gnZg sp :n
fin

pour gftz :o :n
# formate :o sur :n caractères
# si la longueur de :o est inférieure à :n, ajoute des zéros à gauche
# si cette longueur est supérieure, tronque à droite
soit "longo compte :o
si :longo >= :n [ret gtronqued :o (:longo -:n)]
ret mot gzéros (:n - :longo) :o
fin

pour gfte :o :n
# formate :o sur au moins :n caractères
# si la longueur de :o est inférieure à :n, ajoute des zéros à gauche
# si cette longueur est supérieure, ne fait rien
soit "longo compte :o
si :longo >= :n [ret :o]
ret  mot gzéros (:n - :longo) :o
fin

pour gtete :o :x
#ret les :x caractères les plus à gauches de :o
soit "repons gtet2 (mot :o "@ ) :x # à cause du point possible dans l'écriture de :o
si endernier? "@  :repons [donne "repons sd :repons]
ret :repons
fin

pour gtet2 :o :x
si ou vide? :o egal? 0 :x [ret " ] [ret mot prem :o gtet2 sp :o (:x - 1)]
fin

pour gqueue :o :x
# ret les :x caractères les plus à droite de :o
```

```
si ou vide? :o egal? 0 :x [ret "] [ret mot gqueue sd :o (:x - 1) der :o]
fin

pour gtronqued :o :x
# ret :o privé de :x cars à droite
si egal? 0 :x [ret :o]
si vide? :o [ret :o]
ret gtronqued sd :o (:x - 1)
fin

pour gentier? :n
si gnégatif? :n [ret gentier? sp :n]
ret non ou ( membre? "/ :n) (membre? ". :n)
fin

pour gnul? :n
si vide? :n [ret vrai]
si gnégatif? :n [ret gnul? sp :n]
ret egal? 0 :n
fin

pour gnégatif? :n
ret egal? "- prem mot  :n  "@ # pas de test d'intégrité sur :n on ajoute @ en queu par sécurité
fin

pour gopposé :n
si gnégatif? :n [ret gnzg (sp :n)] [ret mot "- :n]
fin

pour gplg? :n1 :n2
si gnégatif? :n1 [si gnégatif? :n2 [ret gplg? gopposé :n2 gopposé :n1] [ret faux]]
si gnégatif? :n2 [ret vrai] [ret  :n1 >  :n2]
fin

pour gplp? :n1 :n2
si egal? :n1 :n2 [ret faux] [ret non gplg? :n1 :n2]
fin

pour gmax :a :b
si gplg? :a :b [ret :a] [ret :b]
fin
#---------------------------
# Les procédures ci-dessous ne sont pas utiles pour mon ldc du fait de la primitive fdecimales
pour gsomme :t1 :t2
si vide? :t1 [ret gsomme 0 :t2]
si vide? :t2 [ret gsomme :t1 0]
si et (compte :t1)  < 11 (compte :t2) <11 [ret somme :t1 :t2] # Pas de souci dans ce cas là
si egal? 0 :t1 [ret :t2]
si egal? 0 :t2 [ret :t1]
si gnégatif? :t1 [ret gdifférence :t2 gopposé :t1]
si gnégatif? :t2 [ret gdifférence :t1 gopposé :t2]
# à ce point les deux nombres sont grands et positifs
si :t2 > :t1 [soit "terme1 :t2 soit "terme2 :t1] [soit "terme1 :t1 soit "terme2 :t2]
# terme1 est donc le plus long des deux
soit "rtn 0 #retenue baladeuse
soit "pf1 "
soit "pf2 "
soit "pf3 "
```

```
soit "repons "
tantque [non vide? :terme2] [donne "pf1 gqueue :terme1 10
donne "terme1 gtronqued :terme1 10
si non vide? :rtn [donne "pf1 somme :pf1 :rtn]
donne "pf2 gqueue :terme2 10
donne "terme2 gtronqued :terme2 10
donne "pf3 gfte somme :pf1 :pf2 10
donne "rtn gtronqued :pf3 10 # le premier chiffre ou rien si moins de 10 chiffres
donne "repons mot gqueue :pf3 10 :repons
si vide? :terme2 [si vide? :rtn [donne "repons mot :terme1 :repons] [donne "terme2 :rtn donne "rtn "]]
] #ici la fin du tantque
ret gnzg :repons
fin
```

pour gdifférence :t1 :t2

```
si vide? :t1 [ret gdifférence 0 :t2]
si vide? :t2 [ret gdifférence :t1 0]
si et (compte :t1)  < 11 (compte :t2) <11 [ret difference :t1 :t2] # Pas de souci dans ce cas là
si egal? 0 :t1 [ret gopposé :t2]
si egal? 0 :t2 [ret :t1]
si gnégatif? :t2 [ret gsomme :t1 gopposé :t2]
si gnégatif? :t1 [ret gopposé gsomme gopposé :t1 :t2]
# A ce point les deux termes sont positifs
si :t2 > :t1 [ret gopposé gdifférence :t2 :t1]
# t1 et t2 sont positifs et t1 > t2
soit "longComp10 "10000000000
soit "rtn 0 #retenue baladeuxe
soit "pf1 "
soit "pf2 "
soit "pf3 "
soit "repons "
tantque [non vide? :t2] [donne "pf1 gqueue :t1 10
donne "t1 gtronqued :t1 10
donne "pf2 gqueue :t2 10
donne "t2 gtronqued :t2 10
donne "pf2 somme :pf2 :rtn #la retenue vient du coup d'avant
donne "pf3 difference :pf1 :pf2
#mais pf1 peut être un bloc trop petit => pf3 va être négatif et on prend la complémentation à 10^10
si :pf3 < 0 [donne "pf3 somme :pf3 :longComp10 donne "rtn 1] [donne "rtn 0]
donne "repons mot (gfte :pf3 10) :repons
si vide? :t2 [si gnul? :rtn [donne "repons mot :t1 :repons donne "t1 "] [donne "t2 :rtn donne "rtn 0]]
] #ici la fin du tantque
ret gnzg :repons
fin
```

pour gproduit :t1 :t2

```
si gnul? :t1 [ret 0]
si gnul? :t2 [ret 0]
si gnégatif? :t1 [ret gopposé gproduit gopposé :t1 :t2]
si gnégatif? :t2 [ret gopposé gproduit :t1 gopposé :t2]
# à ce point t1 et t2 sont strictement positifs
si et (compte :t1) < 8  (compte :t2) < 8 [ret :t1 * :t2]
soit "n1 gnzq :t1
donne "t1 gtronqued :t1 :n1 # on applique la règle des zéros
soit "n2 gnzq :t2
```

```
soit "n0 somme :n1 :n2 #on pourrait optimiser mais bon ...
donne "t2 gtronqued :t2 :n2 # re la règle des zéros
ret mot gprod1 :t1 :t2 gzéros :n0 #on délègue le calcul du produit
fin

pour gprod1 :t1 :t2
# en entrée t1 et t2 sont strictement positifs et ne contiennent aucun 0 à droite
soit "t2faible gqueue :t2 8
donne "t2 gnzg gtronqued :t2 8
si gnul? :t2 [ret gprod2 :t1 :t2faible]
soit "repons gsomme (mot gprod1 :t1 :t2 gzéros 8) gprod2 :t1 :t2faible
# ec (ph [repons dans gprod1 =] :repons)
ret :repons
fin

pour gprod2 :t1 :t2
# ec (ph [debug dans gprod2 t1 = ] :t1 [t2 =] :t2)
# t1 est de longueur quelconque mais t2 est de longueur 8 maxi
si (compte :t1) <8 [ret :t1 * :t2]
soit "t1faible gqueue :t1 8
donne "t1 gnzg gtronqued :t1 8
soit "retenue1 :t1faible * :t2
donne "t1faible gqueue :retenue1 8
donne "retenue1 gtronqued :retenue1 8
soit "repons gnzg mot (gsomme gprod2 :t1 :t2 :retenue1) :t1faible
# ec (ph [repons dans gprod2 =] :repons)
ret :repons
fin

pour gquotient :t1 :t2
si gnul? :t2 [message [Division par zéro impossible] stoptout]
si gnul? :t1 [ret 0]
si gnégatif? :t1 [si gnégatif? :t2 [ret gdifférence (gquot2 gopposé :t1 gopposé :t2) 1] [ret gopposé gsomme 1
gquot2 gopposé :t1 :t2]]
# à partir de maintenant t1 et t2 sont tous deux positifs
ret gquot2 :t1 :t2
fin

pour gquot2 :t1 :t2
si et (compte :t1) < 9 (compte :t2) < 9 [ret quotient :t1 :t2]
si :t1 < :t2 [ret 0]
soit "pfor1 gtete :t1 8
soit "pfor2 gtete :t2 7
soit "zd1 gmax 0 (compte :t1) - 8
soit "zd2 gmax 0 (compte :t2) - 7
soit "zd0 :zd1 - :zd2
soit "laqueue gzéros :zd0
soit "qfor quotient :pfor1 :pfor2 #candidat  pour le quotient
soit "soustot1 mot gprod1 :qfor :t2 :laqueue # le multiple de t2 qu'on veut oter
si (compte :soustot1) > (compte :t1) [donne "qfor sd :qfor donne "soustot1 mot gprod1 :qfor :t2 :laqueue]
# cas très particulier comme dans 98765432109 divisé par 12345678901 par exemple
si :soustot1 > :t1 [donne "qfor gdifférence :qfor 1 donne "soustot1 mot gprod1 :qfor :t2 :laqueue]
donne "t1 gdifférence :t1 :soustot1
donne "qfor mot :qfor :laqueue # le vrai quotient à garder
ret gsomme :qfor  gquot2 :t1 :t2 # encore un appel récursif
fin
```

pour greste :t1 :t2
si gnul? :t2 [message [Division par zéro impossible] stoptout]
si gnégatif? :t1 [si gnégatif? :t2 [ret gdifférence gopposé :t2 greste gopposé :t1 gopposé :t2] [ret gdifférence :t2 greste gopposé :t1 :t2]]
A ce point t1 et t2 sont positifs
si :t1 < :t2 [ret :t1]
si et (compte :t1) < 9 (compte :t2) < 9 [ret reste :t1 :t2]
ret gdifférence :t1 gproduit :t2 gquotient :t1 :t2 # algorithme bête, on doit pouvoir faire bien mieux
fin

----
les fractions sont codées comme des mots
exemple : "123/789 attention 123/789 serait interprétée comme 0.155893536121673
#pour une fraction négative : "-123/789
les écritures fractionnaires sont possibles comme dans "-1.2/25.3
#--------------------------
pour fraction :f
retourne sous forme fractionnaire l'entrée fournie 4.5 donne 45/10 mais 4 donne toujours 4
si fraction? :f [ret bnf :f][ret bnf2 :f 1]
fin

#les fonctions de base

pour fraction? :f
SI non rationnel? :f [EC PH :f [n'est ni une fraction ni un nombre décimal...] stoptout] # retour niveausup
ret membre? "/ :f
fin

pour rationnel? :o
retourne vrai si l'entrée est un nombre entier ou décimal ou une écriture de type fraction
si nombre? :o [ret "VRAI]
si non membre? "/ :o [ret "Faux]
soit "do dénominateur :o
soit "no numérateur :o
ret et nombre? :no nombre? :do
fin

pour simplifiable? :f
si gnégatif? :f [ret gopposé :f]
soit "f bnf :f
ret non etranger? numérateur :f dénominateur :f
fin

pour simplifie :f :q
simplifie num et denom d'une fraction par un même facteur
soit "f bnf :f
soit "a numérateur :f soit "b dénominateur :f
si et multiple? :a :q multiple? :b :q [ret bnf2 quotient2 :a :q quotient2 :b :q][ret :f]
fin

attention ci-dessous, versions fonctionnelles pour mon ldc
pour une utilisation plus générale utiliser les procédures gproduit au lieu de produit, greste au lieu de reste etc.

pour irréductible :f
soit "f bnf :f
si gentier? :f [ret :f] #rien à faire ici
si gnégatif? :f [ret gopposé irréductible gopposé :f]

```
# les deux termes de la fraction sont positifs maintenant
soit "a numér :f soit "b dénom :f
soit "d pgcd  :a :b
si egal? 1 :d [ret :f] [ret bnf2 quotient :a :d quotient :b :d]
fin

pour bnf :o
si endernier? ". :o [ret bnf sd :o]
si endernier? "/ :o [ret bnf sd :o]
si nombre? :o [soit "no :o soit  "do 1] [soit "no numér :o soit "do dénom :o ] # on ne contrôle pas plus que cela
SI :do <  0 [ret bnf2 moins :no  moins :do] [ret bnf2 :no :do]
fin

pour bnf2 :n :d
si egal? 0 :n [ret 0]
si ou membre? ". :n membre? ". :d [ret bnf2 produit :n 10 produit :d 10]
si egal? :d 1 [ret :n] [ret (mot :n "/ :d)]
fin

pour numérateur :o
# en entrée on tient quelquechose du genre "saaa/bbbb - l'existence de "/ a été vérifiée avant l'appel
# on retourne "saaa sans analyser le type de l'extrait (entier ou non)
ret numér bnf :o
fin

pour dénominateur :o
ret dénom bnf :o
fin

pour numér :o
# en entrée on tient quelquechose du genre "saaa/bbbb - l'existence de "/ a été vérifiée avant l'appel
# on retourne "saaa sans analyser le type de l'extrait (entier ou non)
soit "resultat rechjq "/ :o "
si nombre? :resultat [ret somme 0  :resultat] [ec (ph [Erreur :]  :o [n'est pas une écriture valide]) stoptout]
fin

pour dénom :o
# en entrée on tient quelquechose du genre "saaa/bbbb - l'existence de "/ a priori
# on retourne "bbbb quand c'est possible
soit "resultat membre "/ :o # on obtient faux ou /bbbb
si egal? :resultat "faux [ret 1] # il n'y a pas de dénominateur lisible, mais l'écriture n'est peut-etre pas celle d'un
nombre !
donne "resultat sp :resultat
si endernier? ".  :resultat [donne "resultat sd :resultat] #on élimine le point décimal s'il est en dernière position
si vide? :resultat [ ec (ph [Erreur :]  :o [n'est pas une écriture valide]) stoptout]
si nombre? :resultat [ret somme 0  :resultat] [ec (ph [Erreur :]  :o [n'est pas une écriture valide]) stoptout]
fin

pour décimal :f
# traitement brutal confié à Xlogo
si nombre? :f [ret :f]
sI non rationnel? :f [ec ph :f [n'est ni une fraction ni un nombre décimal...] stoppetout] # retour niveausup
ret div numérateur :f dénominateur :f
fin

pour fégal? :f1 :f2
soit "f1 bnf :f1 soit "f2 bnf :f2
soit "nf1numérateur :f1 soit "df1 dénominateur :f1
```

```
soit "nf2 numérateur :f2 soit "df2 dénominateur :f2
ret egal? produit :nf1 :df2 produit :nf2 :df1
fin

pour fplp? :f1 :f2
SI NOMBRE? :f1 [RET FPLP? FRACTION :f1 :f2]
SI NOMBRE? :f2 [RET FPLP? :f1 FRACTION :f2]
RET (produit numér :f1 dénom :f2) < (produit numér :f2 dénom :f1)
fin

pour fplg? :f1 :f2
ret fplp? :f2 :f1
fin

pour fsomme :f1 :f2
soit "ent1 gentier? :f1 soit "ent2 gentier? :f2
si et :ent1 :ent2 [ret somme :f1 :f2]
si :ent1 [donne "f1 mot :f1 "/1] [donne "f1 fraction :f1]
si :ent2 [donne "f2 mot :f2 "/1] [donne "f2 fraction :f2]
ret fsom2 :f1 :f2
fin

pour fsom2 :f1 :f2
soit "df1 dénom :f1 soit "df2 dénom :f2 soit "p ppcm  :df1  :df2
soit "d1 quotient :p :df1 soit "d2 quotient :p :df2
RET bnf2 somme produit numér :f1 :d1 produit numér :f2 :d2 :p
fin

pour fproduit :f1 :f2
soit "ent1 gentier? :f1 soit "ent2 gentier? :f2
si et :ent1 :ent2 [ret produit :f1 :f2]
si :ent1 [donne "f1 mot :f1 "/1] [donne "f1 fraction :f1]
si :ent2 [donne "f2 mot :f2 "/1] [donne "f2 fraction :f2]
ret fproduit2  :f1  :f2
fin

pour fproduit2 :f1 :f2
ret bnf2  produit  numér :f1 numér :f2 produit dénom :f1 dénom :f2
fin

pour finverse :f
si gnégatif? :f [ret gopposé finverse gopposé :f]
# maintenant f est de signe +
si gentier? :f [ret mot "1/  :f]
donne "f fraction :f
ret bnf2 dénom :f numér :f
fin

pour fopposé :f
si fraction? :f [soit "f bnf :f  ret bnf2 moins numér :f  dénom :f] [ret gopposé :f]
fin

pour fdiff :f1 :f2
soit "ent1 gentier? :f1 soit "ent2 gentier? :f2
si et :ent1 :ent2 [ret difference :f1 :f2]
RET fsomme :f1 fopposé :f2
fin

pour fdiv :f1 :f2
```

```
si gentier? :f1 [donne "f1 mot :f1 "/1] [donne "f1 fraction :f1]
si gentier? :f2 [donne "f2 mot :f2 "/1] [donne "f2 fraction :f2]
ret fproduit2 :f1 finverse :f2
fin

pour convertnl :o
# transforme une écriture du genre "snnnn/dddd en une liste [snnn dddd]
ret liste numérateur :o dénominateur :o
fin

pour rechpos :c :o
ret rechpo2 :c :o 1
fin

pour rechpo2 :c :o :n
# :o est un mot et rien d'autre ; permet de trouver le rang du car :c dans le mot :o
si vide? :o [ret 0]
si egal? :c prem :o [ret :n] [ret rechpo2 :c sp :o (:n + 1)]
fin

pour rechjq :c :o :r
si vide? :o [ret :r]
si egal? :c prem :o [ret :r] [ret rechjq :c sp :o mot :r prem :o]
fin

pour évalue :o
soit "lavaleur exec :o
ret :lavaleur
fin

pour égal? :o1 :o2
ret (:o1 = :o2)
fin

pour rien
retourne "
fin
#-------------------------
pour ldc
lanceréglages
initclav
lancesaisieclav
restaureréglages
fin

pour lanceréglages
sauvecontexte
fixenewcontexte
fin

pour restaureréglages
restaurecontexte
ve
ec "Terminé!
fin

pour sauvecontexte
dprop "oldcontexte "couleurcrayon cc
```

```
dprop "oldcontexte "couleurfond cf
dprop "oldcontexte "taillecrayon tc
dprop "oldcontexte "formecrayon fc
dprop "oldcontexte "tailledessin tailledessin
dprop "oldcontexte "taillepolice tp
dprop "oldcontexte "nompolice np
dprop "oldcontexte "separation sep
fin
```

pour restaurecontexte

```
fcc rprop "oldcontexte "couleurcrayon
fcfg rprop "oldcontexte "couleurfond
ftc rprop "oldcontexte "taillecrayon
ffc rprop "oldcontexte "formecrayon
ftd rprop "oldcontexte "tailledessin
ftp rprop "oldcontexte "taillepolice
fnp prem rprop "oldcontexte "nompolice
fsep rprop "oldcontexte "separation
fixedecimales -1
fin
```

pour fixenewcontexte

```
ftd [900 700] ve ct vt fsep 0.9 ftp 64  fnp rechpolice # "calibri attendue
soit  "rightCadre (quotient prem tailledessin 2) - 10 soit "leftCadre moins :rightCadre
soit "topCadre (quotient der tailledessin 2) - 10 soit  "hauteurCadre 4 * tp
soit "basCadre :topCadre - :hauteurCadre
ftc  tp # taille du crayon = taille de la police. Normalement on doit tout effacer comme il faut. (quotient tp 2) +
Donne "decalageV quotient tc 4 donne "demicrayon quotient tc 2
donne "zéroafficheur1 liste (:leftCadre + :demicrayon + 2) (:topcadre - :demicrayon - :decalageV - 2)  # l'afficheur
contient deux lignes - la plus élevée part d'ici
donne "zéroafficheur2 liste (:rightCadre - :demicrayon - 2 ) (:basCadre + 2 + :demicrayon - :decalageV)  # la
seconde ligne part d'ici -mais la gestion se fait à rebrousse-poil
donne "xdebut2 prem :zéroafficheur2
donne "zéroafficheur3 liste (prem :zéroafficheur2) der :zéroafficheur1
soit "x4 somme (prem :zéroafficheur1)  tc
soit "y4 quotient (somme der :zéroafficheur1  der :zéroafficheur2) 2
donne "zéroafficheur4 liste :x4 :y4
donne "accu1 0 donne "accu2 0 donne "accu3 rien donne "op rien
afficheCadre :leftCadre :rightCadre :topCadre :basCadre afficheNum1 afficheNum2 afficheNum3
donne "zéroconsigne liste prem :zéroafficheur1 difference :bascadre 24 consignes
donne "zéroafficheur5 liste prem :zéroafficheur2 der :zéroconsigne # pour afficher soit [Mode Calculette] soit
[Mode Evaluateur]
donne "ModeCalculette faux AfficheMode
donne "MaxCompteurAccu taillemaxaffiche # pas plus de :MaxCompteurAccu  dans un accu, signe -,  point (s)
ou barre de fraction compris. Calcul fait un peu à l'arrache ...
# donne "longafficheur1 difference prem :zéroafficheur3 prem :zéroafficheur1
# donne "longafficheur1 difference :longafficheur1 ( 2 * le "/)
donne "Compteuraccu2 1 # sert à vérifier la validité de la saisie
fixedecimales somme 5 produit  :MaxCompteurAccu 2
donne "CompteurIter 0 # va servir de à marquer le nombre de fois consécutives qu'on appuie sur la touche [=]
donne "zérocompteur liste prem :zéroafficheur2 quotient (somme der :zéroafficheur1 der :zéroafficheur4) 2 #
pour écrire cette valeur - à rebrousse poil
fin
```

pour taillemaxaffiche

```
soit "oldtp tp ftp 64
```

```
soit "temp "-9/ soit "numcar 3
soit "laf difference prem :zéroafficheur3 prem :zéroafficheur1
donne "laf difference :laf ( 2 * le "/)
tantque [ (le :temp) < :laf] [donne "temp mot :temp 9 donne "numcar somme :numcar 1]
# en sortie du tantque on tient une chaine qui dépasse juste un peu
ftp :oldtp
ret  :numcar - 1
fin
```

pour affichecadre :xg :xd :yh :yb

```
# un petit filet atour de l'éditeur en ligne
soit "oldtc taillecrayon ftc 2 soit "oldpos pos
lc fpos liste :xg :yh bc fixex :xd fixey :yb fixex :xg fixey :yh lc
fpos :oldpos ftc :oldtc
fin
```

pour affichemode

```
# affiche le mode actif de ma machine selon la valeur de la variable globale ModeCalculette
soit "oldcc cc soit "oldtp tp soit "oldtc tc soit "oldpos pos
ftp 12 fcc violet
si :modecalculette [soit "offset le [Mode Evaluateur]] [soit "offset le [Mode Calculette]]
lc fpos :zéroafficheur5 fixex difference prem pos :offset
ftc 24 gomme fixex prem :zéroafficheur5 de lc # on gomme l'ancienne mention
# maintenant on va écrire
si :modecalculette [ soit "lacom [Mode Calculette]] [soit "lacom [Mode Evaluateur]]
donne "offset le :lacom lc fpos :zéroafficheur5 fixex difference prem pos :offset etiquette :lacom
fpos :oldpos ftc :oldtc ftp :oldtp fcc :oldcc
fin
```

pour affmarque :c

```
# fait apparaitre le signe courant suite à l'appui sur une des touches + - * /
effacenum3 donne "accu3 :c
affichenum3
fin
```

pour affichenum3

```
soit "oldpos pos lc soit "longEcriture le :accu3 soit "oldcoul cc
fcc rougefonce fpos :zéroafficheur3 fixex (prem pos) - :longEcriture
etiquette :accu3
fpos :oldpos fcc :oldcoul
fin
```

pour affichenum2

```
soit "oldpos pos lc soit "longEcriture le :accu2 soit "oldcoul cc
fcc vertfonce fpos :zéroafficheur2 fixex (prem pos) - :longEcriture
etiquette :accu2
fpos :oldpos fcc :oldcoul
fin
pour affichenum1
soit "oldpos pos lc soit "oldcoul cc
fcc bleufonce fpos :zéroafficheur1
etiquette :accu1
fpos :oldpos fcc :oldcoul
fin
```

pour rechpolice

```
soit  "newnp appelrchpol 0 100 "Arial 0 # on est certain  que cette police existe
```

```
ret  appelrchpol 0 100 "Calibri :newnp # mais c'est celle-là que l'on veut
fin

pour appelrchpol :nmpl :delta :pol1 :numxpol
soit "resultat rchpol :nmpl (:nmpl + :delta) :pol1 :numxpol
si egal? -1 :resultat [ret :numxpol] # on a tout listé, sans succès
si egal? -20 :resultat [ret appelrchpol (:nmpl + :delta) :delta :pol1 :numxpol] # sur le tronçon listé, on a pas trouvé
ret :resultat
fin

pour rchpol :nmpl :nmax :pol1 :numxpol
si :nmpl > :nmax [ret -20] # on veut pas déborder la pile un peu faible de xlogo
soit "oldnp prem np
fnp :nmpl soit  "lapol np soit  "rangpol prem :lapol
fnp :oldnp
si non egal? :rangpol :nmpl [ret -1] # on a bouclé sans trouver la police :pol1 -- calibri en pratique
si egal? prem der :lapol :pol1 [ret :nmpl] # on a trouvé le nom correspondant à la police principale cherchée
ret  rchpol (1 + :nmpl) :nmax :pol1 :numxpol
fin

pour initclav
# on attend la première frappe de l'utilisateur. La variable caradmis est réglée en conséquence
donne "carAdmis "123456789. # au début de la saisie, il faut au moins taper un chiffre ou la virgule 0 étant affiché
# Ca va évoluer ensuite via les procédures de plus bas niveau.
# Ici on attend une touche dans [= + - * / n esc] ; n <=> +/- ; les  touches numériques sont gérées au niveau de la routine Inputnum2
donne "codefonctions "+-*oOiIrRsSeEdDfFxXyYtTuUmMbBcCaAkK
donne "codeOp "frecopie # de l'accu2 vers l'accu 1
donne "accu3 rien
donne "accuAux 0 # registre auxilliaire, permet de stocker depuis accu2 une donnée
fin

pour lancesaisieclav
soit "latouche inputnum2
si egal? :latouche "esc [stop]
si egal? :latouche "enter [itereOp increcompteur ] # on applique le dernier opérateur connu à "accu1 et "accu2. Le résultat va dans "accu1 qui est raffraichi.
# on analyse maintenant une touche parmi "oOiIrRsSdDfFxXyYtTuUbBcCaA
si egal? :latouche "W [alertelongaccu2] [analTouche :latouche] # registre de saisie trop long, on alerte et on bloque sinon on analyse
lancesaisieclav
fin

pour inputnum2
soit "codetouche liscar # on attend touche
si :codetouche < 1 [ret inputnum2] # appui sur une touche du clavier numérique alors qu'il n'est pas en mode numlock
si egal? :codetouche 27 [ret "esc] # on va vers l'arrêt de la ligne de commande
si ou egal? :codetouche 10 egal? :codetouche 61 [ret "enter] # touche [enter] ou touche [=]
# touche [<-] alias backspace enfoncée : on enlève le dernier caractère d'accu2 et on boucle de suite
si egal? :codetouche 8 [rogneAccu2 afficheNum2 ret inputnum2 ]
soit "touche car :codetouche
si membre? :touche :codefonctions [ret majus2 :codetouche] # a priori dans "+-*ooiIrRsSdDfF ;
# on fait gérer par le niveau supérieur après avoir transformé en majuscule
# on gère ici la saisie d'une touche numérique ou de la touche [.] ou [/] si autorisée
```

```
si non (membre? :touche :carAdmis) [  ret inputnum2] # on boucle en attente d'une saisie plus conforme à l'attente
si :Compteuraccu2 >= :MaxCompteuraccu  [si digit? :touche [ret inputnum2]] # on bloque toute saisie numérique car trop long
soit "suite " # rien à priori
effacenum2
si egal? 46 :codetouche [pointe2] # . enfoncé
si egal? 47 :codetouche [donne "suite barre2 si egal? "/ :suite [affichenum2 ret "/]] # barre enfoncée => sortie extraordinaire evnetuellement
si nombre? :touche [complete2 :touche]
affichenum2
si  :Compteuraccu2 >= :MaxCompteuraccu [ret "W ] # W pour wide = trop large on bloquera toute nouvelle saisie
ret inputnum2  #on boucle sauf si division demandée ... pas très propre
fin
```

pour analtouche :latouche

```
si et  :modecalculette  membre? :latouche "+-/* [itereop vidangeaccu2] #en mode calculette on déclenche éventuellement un calcul avant de changer le codeOp
# puis on modifie le codeop si telle était la commande
si egal? :latouche "+ [razcompteur donne "codeOp "fsomme affMarque "+  stop ]
si egal? :latouche "- [razcompteur donne "codeOp "fdiff  affMarque "- stop]
si egal? :latouche "/ [razcompteur donne "codeOp "fdiv  affMarque "/ stop]
si egal? :latouche "* [razcompteur donne "codeOp "fproduit  affMarque "* stop]
si egal? :latouche "O [razcompteur oppose2 stop] # on change de signe le contenu de l'accu2 / équivaut à la touche [+/-] de certaines calculettes.
si egal? :latouche "I [razcompteur inverse2 stop] # on calcule l'inverse / équivaut à la touche [1/x] de certaines calculettes
si egal? :latouche "R [reduit2 stop] # on transforme sous forme de fraction irréductible l'accu2
si egal? :latouche "S [simplifie2 stop] # on simplifie la fraction par le plus petit diviseur commun aux numérateur et dénominateur.
si egal? :latouche "E [expans2 stop] # on multiplie les deux termes de la fraction par un entier quand c'est possible
si egal? :latouche "D [razcompteur  décimal2 stop] # on remplace par une écriture décimale l'accu2
si egal? :latouche "F [razcompteur fraction2 stop] # on remplace l'accu2 par son écriture fractionnaire normalisée
si egal? :latouche "X [razcompteur exchange2 stop] # on intervertit les deux accus qui à réduire sur accu1
si egal? :latouche "Y [razcompteur transfert1 stop] # transfert du registre de saisie alias accu2 dans l'accumulateur alias accu1
si egal? :latouche "T [razcompteur transfert2 stop] # on rapatrie accu1 sur accu2 (dans l'autre sens Y, on peut aussi  tuer le codeop puis appuyer sur la touche [enter].
si egal? :latouche "U [videCodeOp2 stop] # on tue le code Op.
si egal? :latouche "M [memorise2 stop] # on mémorise accu2 dans un registre auxilliaire
si egal? :latouche "B [razcompteur backup2 stop] # on rapatrie dans accu2 le contenu du registre auxilliaire
si egal? :latouche "C [razcompteur vidangeAccu2 stop] # on remet à zéro accu2
si egal? :latouche "A [razcompteur vidangeTout2] # on vide les deux accus et la mémoire
si egal? :latouche "K [invmode2] # on switch de [Mode Calculette] à [Mode Evaluateur]
fin
```

pour rogneaccu2

```
si nul? :accu2 [ stop] #on peut rien faire # avant j'avais mis : verifCodeOp
soit "derniercode unicode der mot "@ :accu2 # ritournelle sur XLogo
effacenum2
soit "premiercar "
si gnégatif?  :accu2 [donne "premiercar "- donne "accu2 sp :accu2]  # signe enlevé temporairement
```

```
donne "accu2 sd :accu2
si nul? :accu2 [donne "accu2 "0 donne "Compteuraccu2 1 donne "carAdmis  "123456789.  stop]  # verifCodeOp
oté
# cas 1 : on vient d'enlever un point (alias virgule) : on autorise la frappe de / si celle-ci n'est pas déjà employée
si egal? 46 :derniercode [pointOté ] # Dans ce cas ci ...
# cas 2 : on vient d'enlever un slash (alias /) : on autorise la frappe de / et ou . selon les cas
si egal? 47 :derniercode [slashOté  ] #... et dans ce cas là,
# cas 3 : lissage dans le cas xxx. ou xx.xx/yy.
si pointendernier? :accu2 [donne "carAdmis retire :carAdmis "/]
# pour finir, on récupère le signe devant le nombre
si non nul? :accu2 [donne "accu2 mot :premiercar :accu2 donne "Compteuraccu2 difference :Compteuraccu2 1 ]
[donne "Compteuraccu2 1]
# verifCodeOp # ne pas laisser / dispo si accu2 est nul #annulé depuis
fin
```

pour barre2

```
si endernier? "/ :accu2 [donne "accu2 sd :accu2 donne "Compteuraccu2 difference :Compteuraccu2 1 ret "/] #
on a frappé 2 barres de suite => appel division
si membre? "/ :accu2 [ret "/] # c'est une deuxième frappe de la touche [/] : on va diviser
donne "accu2 mot pqb :accu2 "/  donne "Compteuraccu2 compte :accu2 # tout à fait exceptionnel
donne "carAdmis "123456789./
# donne "Compteuraccu2 0 # on remet le compteur à blanc #j'ai changé de stratégie depuis.
ret " # on rends rien => inputnum2 va se reproduire
fin
```

pour pointe2

```
# le point a été saisi
si endernier? "/  :accu2 [donne "accu2 mot :accu2 "0 donne "Compteuraccu2 somme :Compteuraccu2 1 ]
donne "accu2 mot :accu2 ". donne "Compteuraccu2 somme :Compteuraccu2 1
donne "carAdmis "0123456789/
fin
```

pour complete2 :touche

```
# n'est activée que parce qu'une touche numérique a été enfoncée
si nul? :accu2 [donne "accu2 :touche ] [donne "accu2 mot :accu2 :touche donne "Compteuraccu2 somme 1
:Compteuraccu2]
donne "carAdmis injecte :carAdmis "0 #parce que la première fois, on avait pas droit à zero.
donne "carAdmis injecte :carAdmis "/   # comme ci-dessus
fin
```

pour fixecaradmis

```
# uniquement quand on réassigne depuis accu1 le registre accu2. accu1 a toujours une bonne forme
si nul? :accu2 [donne "carAdmis "123456789. stop]
si nombre? :accu2 [donne "carAdmis "0123456789/ stop]
# dernier cas : une fraction
soit "d dénom :accu2
si membre? ". :d [donne "carAdmis "0123456789/] [donne "carAdmis "0123456789./]
fin
```

pour itereop

```
si erroraccu2 [alerteerroraccu2 stop]
soit "lacom (ph [donne "newaccu1 irréductible ] :codeOp [:accu1 :accu2])
execute :lacom
si (compte :newaccu1) > :MaxCompteuraccu [alertelongaccu1 stop]
effacenum1
donne "accu1 :newaccu1
affichenum1
```

```
fin

pour frecopie :o1 :o2
# par souci d'homogénéité ne fait rien que de renvoyer :o2
ret :o2
fin

pour pointoté
donne "carAdmis injecte :carAdmis ".  # si on l'enlève, c'est qu'on l'a mis et donc on peut le remettre
si nombre? :accu2 [donne "carAdmis injecte :carAdmis "/ stop] # ona pas affaire à une écriture bizarre genre "xxx/y
# mais on a peut-être affaire à qqchose du type xxx/0 => on élimine le 0 et on corrige carAdmis
soit "accu22 sd :accu2
si non vide? :accu22 [si endernier? "/  :accu22 [si endernier? "0 :accu2 [donne "accu2 :accu22 donne "carAdmis "123456789. donne "Compteuraccu2 difference :compteuraccu2 1  ]]]
fin

pour slashoté
donne "carAdmis "0123456789/  # si on l'enlève, c'est qu'on l'a mis et donc on peut le remettre
si non membre? ". :accu2 [donne "carAdmis injecte :carAdmis ".]
# donne "Compteuraccu2 NbrChiffres :accu2
fin

pour retire :o :c
soit "trouve membre :c :o
si egal? :trouve "faux [ret :o]
donne "trouve sp :trouve
ret mot rechjq :c mot :o "@ "   :trouve # j'injecte "@ en queue du mot :o pour éliminer un bug d'interprétation de Xlogo
fin

pour injecte :o :c
si membre? :c :o [ret :o] [ret mot :o car unicode :c]
fin

pour majus2 :code
si :code > 96 [soit "code :code - 32]
ret car :code
fin

pour oppose2
# on met ou ou enlève le signe -
si nul? :accu2 [stop]
si gnégatif? :accu2 [soit "delta 1 ] [soit "delta moins 1]
soit "longtampon somme :CompteurAccu2 :delta
si :longtampon > :MaxCompteurAccu  [alertetamponplein stop]
effacenum2
donne "accu2 gopposé  :accu2 donne "CompteurAccu2 :longtampon
affichenum2
fin

pour inverse2
# on veut inverser sans simplifier : 4 -> 1/4 2.3/5.6 -> 5.6/2.3 etc ... mais 1/5 -> 5
si nul? :accu2 [stop] # on ne génère pas de message, mais la touche est sans effet
si gnégatif? :accu2 [soit "tampon gopposé :accu2 soit "signe "- ] [soit "tampon :accu2 soit "signe " ]
si gentier? :tampon [donne "tampon mot "1/ :tampon] [donne "tampon  bnf :tampon donne "tampon bnf2 dénom :tampon numér :tampon]
donne "tampon mot :signe :tampon
```

```
soit "longtampon compte :tampon
si :longtampon > :MaxCompteurAccu [alertetamponplein stop]
effacenum2
donne "accu2 :tampon
affichenum2
donne "CompteurAccu2  :longtampon
fixecaradmis
fin
```

```
pour reduit2
soit "tampon irréductible :accu2
effacenum2
donne "accu2 :tampon
affichenum2
donne "CompteurAccu2  compte :accu2
fin
```

```
pour décimal2
si non fraction? :accu2 [stop] # inutile d'aller plus loin
soit "tampon bnf :accu2
si gnégatif? :tampon [donne "tampon gopposé :tampon soit  "signe "- ] [soit "signe " ]
soit "olddécimales decimales fixedecimales 2 * decimales # par sécurité
donne "tampon div numér :tampon dénom :tampon
donne "tampon mot :signe :tampon donne "tampon gtete :tampon :MaxCompteurAccu donne "tampon pqb
:tampon
fixedecimales :olddécimales
effacenum2
donne "accu2 :tampon
affichenum2
donne "CompteurAccu2 compte :tampon fixecaradmis
fin
```

```
pour fraction2
soit "tampon bnf :accu2
si egal? :accu2 :tampon [stop] # cas d'un entier
# soit "ln NbrChiffres numér :tampon
# soit "ld NbrChiffres dénom :tampon
# on essaye de convertir en fraction décimale. Mais si l'un des facteurs est trop grand, on essaye de passer à la
fraction irréductible
si (compte :tampon) > :MaxCompteurAccu [donne "tampon irréductible :accu2]
# Et même comme cela, il est possible que cela foire ...
soit "longtampon compte :tampon
si  :longtampon > :MaxCompteurAccu [alertetamponplein stop]
effacenum2
donne "accu2 :tampon
affichenum2
fixecaradmis donne "CompteurAccu2 :longtampon
fin
```

```
pour transfert1
# de accu2 vers accu1
si erroraccu2 [alerteerroraccu2 stop]
soit "tampon irréductible :accu2
soit "longtampon compte :tampon
si :longtampon > :MaxCompteurAccu [alertetamponplein stop]
effacenum1
```

```
donne "accu1 :tampon
affichenum1
fin

pour transfert2
# de accu1 vers accu2
soit "longaccu1 compte :accu1
si :longaccu1 > :MaxCompteurAccu [alertetamponplein stop]   # par sécurité ?
effacenum2
donne "accu2 :accu1
affichenum2
fixecaradmis donne "CompteurAccu2  :longaccu1
fin

pour exchange2
si erroraccu2 [alerteerroraccu2 stop]
soit "tampon irréductible :accu2
soit "longtampon compte :tampon # par sécurité uniquement
si  :longtampon > :MaxCompteurAccu [alertetamponplein stop]
# normalement, tout doit bien se passer depuis accu1 mais je préferre jouer la sécurité
soit "longaccu1 compte :accu1
si :longaccu1 > :MaxCompteurAccu [alertetamponplein stop]  # idem
effacenum2
donne "accu2 :accu1
affichenum2
effacenum1
donne "accu1 :tampon
affichenum1
fixecaradmis donne "CompteurAccu2  :longtampon
fin

pour videcodeop2
effacenum3
donne "codeOp "frecopie # de l'accu2 vers l'accu 1
donne "accu3 rien
fin

pour memorise2
#accu2 vers mémoire auxilliaire
donne "tampon bnf :accu2 # avant je prenais l'irréductible
si (compte :tampon) > :MaxCompteurAccu [alertetamponplein stop] # par sécurité ?
donne "accuAux :tampon
fin

pour backup2
#de la mémoire auxilliaire vers le tampon
si egal? :accuAux :accu2 [stop]
soit "longaccuaux compte :accuAux
si :longaccuaux  > :MaxCompteurAccu [alertetamponplein stop]  # par sécurité ?
effacenum2
donne "accu2 :accuAux
affichenum2
# verifcodeop
fixecaradmis
donne "CompteurAccu2  :longaccuaux
fin
```

pour vidangeaccu2
effacenum2
donne "accu2 0
donne "CompteurAccu2 1
affichenum2
videcodeop2 #dans une version initiale mais j'ai changé d'avis
donne "carAdmis "123456789.
fin

pour vidangetout2
effacenum1
donne "accu1 0
affichenum1
vidangeAccu2
donne "accuAux 0
videcodeop2
fin

pour invmode2
A l'appui sur la touche K on bascule de [Mode Calculette] à [Mode Evaluateur] et réciproquement
donne "modecalculette non :modecalculette
affichemode
fin

pour simplifie2
simplifie les fractions entier sur entier
si non fraction? :accu2 [stop]
si membre ". :accu2 [soit "lacom [Action impossible : les deux termes de la fraction doivent être des entiers!]
lancealerteaccu2 :lacom stop]
si gnégatif? :accu2 [soit "tampon gopposé :accu2 soit "signe "-] [soit "tampon :accu2 soit "signe "]
soit "nt numér :tampon soit "dt dénom :tampon soit "pgcdt pgcd :nt :dt
si egal? 1 :pgcdt [soit "lacom [Action impossible : la fraction est irréductible!] lancealerteaccu2 :lacom stop]
soit "divt premierdiviseur :pgcdt soit "lacom ph [La fraction va être simpliée par :] :divt lancealerteaccu2 :lacom
donne "nt quotient :nt :divt donne "dt quotient :dt :divt
effacenum2 donne "accu2 mot :signe bnf (mot :nt "/ :dt) affichenum2
fin

pour expans2
multiplie numérateur et dénominateur de la fraction par un même entier
si non fraction? :accu2 [stop]
si membre? ". :accu2 [soit "lacom [Action impossible : les deux termes de la fraction doivent être des entiers!]
lancealerteaccu2 :lacom stop]
si gnégatif? :accu2 [soit "tampon gopposé :accu2 soit "signe "-] [soit "tampon :accu2 soit "signe "]
soit "lacom [Par quel nombre entier, voulez-vous multiplier le numérateur et le dénominateur :]
temporaire pour essai
lis :lacom "repons vt
si gnégatif? :repons [donne "repons gopposé :repons] # on nie le signe -
si ou egal? 1 :repons egal? 0 :repons [stop]
si non rationnel? :repons [soit "lacom (ph [Donnée incompatible avec] :accu2 ".) lancealerteaccu2 :lacom stop]
donne "repons irréductible :repons
soit "nr numérateur :repons soit "dr dénominateur :repons
soit "nt numérateur :tampon soit "dt dénominateur :tampon
si non et (multiple? :nt :dr) (multiple? :dt :dr) [soit "lacom(ph [Donnée incompatible avec] :accu2 ".)
lancealerteaccu2 :lacom stop]
soit "nt produit :nr quotient :nt :dr soit "dt produit :nr quotient :dt :dr
soit "tampon mot :signe bnf (mot :nt "/ :dt)

```
si (compte :tampon) > :MaxCompteuraccu [alertetamponplein stop]
effacenum2 donne "accu2 :tampon affichenum2
fin

pour premierdiviseur :n
soit "arret 1 + tronque racine :n
si egal? 0 reste :n 2 [ret 2]
si egal? 0 reste :n 3 [ret 3]
soit "t 6 soit "repons :n # sera valide si :n est premier, sinon tué ci-dessous
tantque [:arret >= :t] [  # debut corps tantque
soit "t1 difference :t 1 si egal? 0 reste :n :t1 [donne "repons :t1 donne "t :arret + 1] [soit "t1 somme :t 1
si égal? 0 reste :n :t1 [donne "repons :t1 donne "t :arret + 1]]
donne "t somme :t 6
 ] #fin du tantque
ret :repons
fin

pour pqb :o
# prépare la queue pour poser la barre
si non membre? ". :o [ret :o] # on n'a rien à faire
si endernier?  ".  :o [ret sd :o]
si egal? "0  der :o  [ret pqb sd :o]
ret :o
fin

pour verifcodeop
si non egal? "/ :accu3 [stop]
si nul? :accu2 [videCodeop2]
fin

pour erroraccu2
si endernier? "/ :accu2 [ret vrai]
si et nul? :accu2 egal? "/ :accu3 [ret vrai]
si egal? 0 dénom :accu2 [ret vrai]
ret faux
fin

pour alerteerroraccu2
soit "lacom [Opération demandée incompatible avec l'état du registre de saisie . ]
lancealerteaccu2 :lacom
fin

pour endernier? :c :o
soit "corrigé mot "@ :o
ret egal? :c der :corrigé
fin

pour nul? :o
si vide? :o [ret vrai]
si egal? "- prem mot :o "@  [ret nul? sp :o] # les négatifs ...
si (compte :o) > 1 [ret faux] [ret (egal? :o 0)]
fin

pour retourchariot :pix
lc fpos liste prem pos difference der pos :pix #(-420)
fin

pour consignes
```

```
lc soit "oldpos pos  fpos :zéroconsigne
soit "oldcc cc soit "oldtp tp ftp 20
 fcc marron etiquette [Résumé des touches disponibles] retourchariot 30
soit  "lemessage  [Saisie des nombres :] soit "offset  8 +  le :lemessage
fcc marron etiquette :lemessage glisseX :offset
soit "lemessage  [ 0 1 2 3 4 5 6 7 8 9 . /]  fcc gris etiquette :lemessage soit "offset2 128 + le :lemessage
glisseX :offset2  fcc marron soit "lemessage [Changement de mode :] etiquette :lemessage
soit "offset3 8 + le :lemessage glisseX :offset3  fcc gris etiquette [touche K]
donne "offset (somme :offset :offset2 :offset3) glisseX moins :offset  retourchariot 30
soit  "lemessage  [Touches du calcul à venir :] soit "offset  8 +  le :lemessage
fcc marron etiquette :lemessage glisseX :offset  fcc gris etiquette [+ pour additionner |  - pour soustraire | *
pour multiplier]  glisseX moins :offset  retourchariot 20
etiquette [La touche / est comprise comme barre de fraction à la première frappe, appel de divison à la
seconde.]  retourchariot 20
etiquette [Touche = ou touche Enter := pour déclencher le calcul, quelque soit le mode courant.] retourchariot
30
fcc marron  etiquette [Gestion du registre de saisie] retourchariot 30
soit "lemessage  [O := changement de signe x <-> -x] soit "offset  20 + le :lemessage
fcc gris etiquette :lemessage glisseX :offset etiquette [ I := inversion x <->1/x ou n/d <-> d/n] glisseX moins
:offset retourchariot 20
soit  "lemessage  [D := remplacement par une écriture décimale] soit "offset  20 + le :lemessage
etiquette :lemessage glisseX :offset etiquette [ F := remplacement par une écriture fractionnaire] glisseX moins
:offset retourchariot 20
soit  "lemessage  [R := mise sous forme fractionnaire irréductible] soit "offset  20 + le :lemessage
etiquette :lemessage glisseX :offset etiquette [ S := simplification de la fraction] glisseX moins :offset
retourchariot 30
 fcc marron etiquette [Gestion des registres] retourchariot 20
fcc gris etiquette [Y := recopie du registre de saisie dans l'accumulateur] retourchariot 20
soit  "lemessage  [T := recopie de l'accumulateur dans le registre de saisie] soit "offset  20 + le :lemessage
etiquette :lemessage glisseX :offset etiquette [ X := échange des deux registres.] glisseX moins :offset
retourchariot 20
fcc [96 96 160] soit  "lemessage  [M := sauvegarde registre de saisie -> mémoire aux.] soit "offset  20 + le
:lemessage
etiquette :lemessage glisseX :offset etiquette [ B := restauration mém. aux. ->registre de saisie.] glisseX moins
:offset retourchariot 20
fcc gris etiquette [ A := remise à zéro de l'ensemble des registres de la ligne de commande] retourchariot 20
soit  "lemessage  [C := remise à zéro du registre de saisie] soit "offset  20 + le :lemessage
etiquette :lemessage glisseX :offset etiquette [ U := annulation du codeOp.] glisseX moins :offset retourchariot
30
ftp 16 fcc violet etiquette [Attention : malgré toute mon attention, une erreur de programmation peut avoir
subsisté. En cas de plantage, relancez le module]
retourchariot 16 etiquette [en tapant la commande ldc dans la ligne de commande tout en haut de la fenêtre de
travail.]
fpos :oldpos fcc :oldcc ftp :oldtp
fin

pour glissex :o
soit "x prem pos
fixex somme :o :x
fin

pour pointendernier? :o
soit "corrigé mot "@ :o
ret egal? car 46 der :corrigé
fin
```

```
pour slashendernier? :o
soit "corrigé mot "@ :o
ret egal? car 47 der :corrigé
fin

pour alertelongaccu2
soit "lacom [Attention ! Registre de saisie plein . ]
lancealerteaccu2 :lacom
fin

pour lancealerteaccu2 :lacom
soit "oldcc cc soit "oldtp tp soit "oldtc tc soit "oldpos pos
 fcc rouge ftp 12  lc fpos :zéroAfficheur4
donne "lacom ph :lacom [Touche Svp !]
etiquette :lacom soit "lasuite liscar
 ftc 24 soit "xend (somme prem pos le :lacom moins 12)
gomme fixex :xend de lc
fpos :oldpos ftc :oldtc  ftp :oldtp fcc :oldcc
fin

pour troplongaccu2
ret (compte :accu2) > :MaxCompteuraccu
fin

pour effacenum1
soit "oldpos pos lc fpos :zéroafficheur1
fixey (der pos) + :decalageV #on décale un peu la gomme verticalement
soit "xbout (prem :zéroafficheur3) - somme :demicrayon  le :accu3
gomme fixex :xbout de lc fpos :oldpos
fin

pour effacenum2
soit "oldpos pos lc fpos liste prem :zéroafficheur1 der :zéroafficheur2
fixey (der pos) + :decalageV #on décale un peu la gomme verticalement
soit "xbout (prem :zéroafficheur2) -  :demicrayon
gomme fixex :xbout de lc fpos :oldpos
fin

pour effacenum3
soit "oldpos pos lc soit "longEcriture le :accu3
fpos :zéroafficheur3 fixex (prem pos) - :longEcriture
fixey (der pos) + :decalageV #on décale un peu la gomme verticalement
soit "xbout (prem :zéroafficheur3) -  :demicrayon
gomme fixex :xbout de lc fpos :oldpos
fin

pour alertetamponplein
soit "lacom [Action impossible : le nombre manipulé déborde du registre.]
lancealerteaccu2 :lacom
fin

pour alertelongaccu1
soit "lacom [Attention ! Le résultat du calcul excède les capacités de l'accumulateur.]
lancealerteaccu2 :lacom
fin

pour digit? :o
si nombre? :o [ret vrai]
```

```
si egal? :o car 46 [ret vrai] # saisie d'un point
si egal? :o car 47 [ret non membre? :o :accu2] # on ne prend en compte que si la barre de fraction n'est pas encore saisie
ret faux
fin

pour increcompteur
# incrémente le compteur d'itération et affiche la nouvelle valeur si elle dépasse 2
# on commence par effacer l'ancienne valeur par sécurité
effacecompteur
donne "compteurIter somme 1 :compteurIter
si :compteurIter > 1 [affichecompteur]
fin

pour razcompteur
# chaque fois qu'une opération  ou une modification du registre de saisie est déclenchée
# on efface quand compteurIter est supérieur ou égal à 2
si :compteurIter > 1 [effacecompteur]
donne "compteurIter 0
fin

pour affichecompteur
soit "oldcc cc soit "oldtp tp soit "oldtc tc soit "oldpos pos ftp 24
lc fpos :zérocompteur glissex moins le :compteurIter
fcc gris etiquette :compteurIter
fpos :oldpos ftc :oldtc  ftp :oldtp fcc :oldcc
fin

pour effacecompteur
soit "oldcc cc soit "oldtp tp soit "oldtc tc soit "oldpos pos ftc 48
lc fpos :zérocompteur glissex moins le :compteurIter
gomme fixex prem  :zérocompteur de lc
fpos :oldpos ftc :oldtc  ftp :oldtp fcc :oldcc
fin
```