Chapitre 8 Programmation

1. Étapes élémentaires de la programmation

Les commandes et les calculs sont exécutés dans l'ordre, tout comme les instructions multiples d'un calcul manuel.

1. Depuis le menu principal, accédez au mode **PRGM**. A ce moment, une liste de programmes apparaît.

Zone de programme sélectionnée (utilisez ▲ et ▼ pour changer de zone)

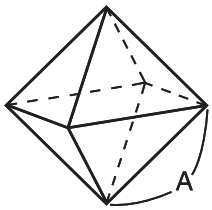
Les fichiers sont classés dans l'ordre alphabétique de leurs noms.

2. Enregistrez un nom de fichier.
3. Ecrivez le programme.
4. Lancez le programme.

- Les valeurs à droite dans la liste de programmes indiquent le nombre d'octets utilisés par chaque programme.
- Un nom de fichier peut contenir jusqu'à huit caractères.
- Vous pouvez utiliser les caractères suivants pour les noms de fichier : A à Z, r, θ, espaces, [,], {, }, ', ", ~, 0 à 9, ., +, −, ×, ÷
- L'enregistrement d'un nom de fichier utilise 32 octets de mémoire.

Exemple **Calculer l'aire (cm²) et le volume (cm³) de trois octaèdres réguliers dont les côtés mesurent 7, 10 et 15 cm**

Stockez la formule sous le nom de fichier OCTA.



Les formules utilisées pour le calcul de l'aire S et du volume V d'un octaèdre régulier dont la longueur d'un côté A est connue sont les suivants.

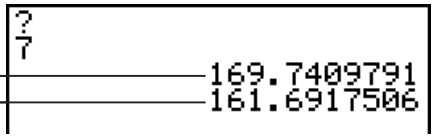
$$S = 2\sqrt{3}\,A^2, \quad V = \frac{\sqrt{2}}{3}A^3$$

① MENU PRGM

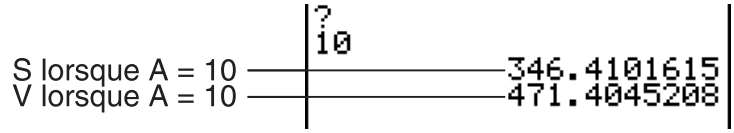
② F3(NEW) 9(O) ln(C) ÷(T) X,θ,T(A) EXE

③ SHIFT VARS(PRGM) F4(?) → ALPHA X,θ,T(A) F6(▷) F5(:)
2 × SHIFT x²(√) 3 × ALPHA X,θ,T(A) x² F6(▷) F6(▷) F5(◢)
SHIFT x²(√) 2 ÷ 3 × ALPHA X,θ,T(A) ^ 3
EXIT EXIT

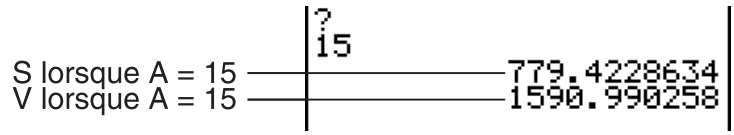
④ F1(EXE) ou EXE
7 EXE(Valeur de A)
EXE



[EXE] [EXE]
[1] [0] [EXE]
[EXE]



[EXE] [EXE]
[1] [5] [EXE]
[EXE] *1



*1 L'appui sur [EXE] lorsque le résultat final du programme est affiché, provoque la sortie du programme.

- Vous pouvez lancer un programme dans le mode **RUN•MAT** en entrant : Prog "<nom de fichier>" [EXE].
- Lorsque le résultat final d'un programme exécuté au moyen de cette méthode est affiché, une pression sur [EXE] réexécute le programme.
- Une erreur se produit si le programme spécifié par Prog "<nom de fichier>" ne peut pas être trouvé.

2. Touches de fonction du mode PRGM

- {**NEW**} ... {nouveau programme}

• Lorsque vous enregistrez un nom de fichier

- {**RUN**}/{**BASE**} ... entrée de programme {calcul général}/{base numérique}
- {**π0**} ... {enregistrement d'un mot de passe}
- {**SYBL**} ... {menu de symboles}

• Lorsque vous écrivez un programme —— [F1](RUN) … défaut

- {**TOP**}/{**BTM**} ... {début}/{fin} du programme
- {**SRC**} ... {recherche}
- {**MENU**} ... {liste des menus}
  - {**STAT**}/{**MAT**}/{**LIST**}/{**GRPH**}/{**DYNA**}/{**TABL**}/{**RECR**}
    ... menu {statistiques}/{matrice}/{liste}/{graphe}/{graphe dynamique}/{table}/{récurrence}
- {**A↔a**} ... {fait basculer entre majuscules et minuscules}
- {**CHAR**} ... {affiche un écran pour la sélection de symboles mathématiques, de symboles spéciaux et de caractères accentués}

- Lorsque vous appuyez sur [SHIFT] [VARS] (PRGM) le menu de programmation (PRGM) suivant apparaît.
  - {**COM**} ... {menu de commandes de programmation}
  - {**CTL**} ... {menu de commandes de contrôle de programmation}
  - {**JUMP**} ... {menu de commande de saut}
  - {**?**}/{◢} ... commande {de saisie}/{d'affichage}

- {**CLR**}/{**DISP**} ... menu de commande de {suppression}/{affichage}
- {**REL**} ... {menú de operadores relacionales de salto condicional}
- {**I/O**} ... {menu de commande de contrôle/transfert d'entrée/sortie}
- {**:**} ... {commande d'instructions multiples}
- {**STR**} ... {commande de chaîne}

Voir « Guide des commandes » à la page 8-7 pour tous les détails sur ces commandes.

- Lorsque vous appuyez sur SHIFT MENU (SET UP), le menu de commandes de mode ci-dessous apparaît.
  - {**ANGL**}/{**COOR**}/{**GRID**}/{**AXES**}/{**LABL**}/{**DISP**}/{**S/L**}/{**DRAW**}/{**DERV**}/{**BACK**}/{**FUNC**}/{**SIML**}/{**S-WIN**}/{**LIST**}/{**LOCS**}/{**T-VAR**}/{**ΣDSP**}/{**RESID**}/{**CPLX**}/{**FRAC**}/{**Y•SPD**}/{**DATE**}/{**PMT**}/{**PRD**}/{**INEQ**}/{**SIMP**}/{**Q1Q3**}

Voir « Menus de touches de fonction sur l'écran de configuration » à la page 1-27 pour les détails au sujet de chaque commande.

• Lorsque vous écrivez un programme —— F2 (BASE)*1

- {**TOP**}/{**BTM**}/{**SRC**}
- {**MENU**}
  - {**d~o**} ... saisie de valeurs {décimales}/{hexadécimales}/{binaires}/{octales}
  - {**LOG**} ... {opérateur des bits}
  - {**DISP**} ... conversion de la valeur affichée en valeur {décimale}/{hexadécimale}/{binaire}/{octale}
- {**A↔a**}/{**SYBL**}

- Lorsque vous appuyez sur SHIFT EXIT (PRGM), le menu de programmation (PRGM) suivant apparaît.
  - {**Prog**} ... {rappel de programme}
  - {**JUMP**}/{**?**}/{◢}
  - {**REL**} ... {menu d'opérateurs relationnels avec saut conditionnel}
  - {**:**} ... {commande d'instructions multiples}

- Lorsque vous appuyez sur SHIFT MENU (SET UP), le menu de commandes de mode ci-dessous apparaît.
  - {**Dec**}/{**Hex**}/{**Bin**}/{**Oct**}

*1 Les programmes écrits après une pression sur F2 (BASE) sont indiqués par B à la droite du nom de fichier.

- {**EXE**}/{**EDIT**} ... programme {execute}/{edit}
- {**NEW**} ... {nouveau programme}
- {**DEL**}/{**DEL•A**} ... suppression de {programme particulier}/{tous les programmes}
- {**SRC**}/{**REN**} ... nom fichier {recherche}/{modifier}

3. Édition du contenu d'un programme

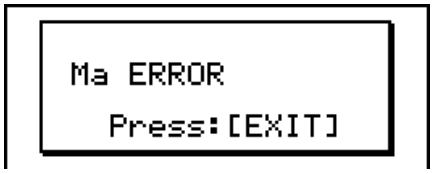
■ Mise au point d'un programme (débogage)

Un problème apparaissant dans un programme et l'empêchant de se dérouler normalement est appelé un « bogue » et l'élimination de ce problème est appelé « débogage ». Les symptômes suivants indiquent que votre programme contient une erreur (un bogue) et qu'une mise au point est nécessaire.

- Messages d'erreur apparaissant quand le programme est en route
- Résultats qui ne correspondent pas aux prévisions

• Pour éliminer une erreur à l'origine d'un message

Un message d'erreur comparable au message suivant apparaît quand un problème se présente pendant l'exécution d'un programme.



Quand ce type de message apparaît, appuyez sur [EXIT] pour afficher le point du programme où l'erreur s'est produite. Le curseur clignote à l'endroit où se trouve le problème. Contrôlez le « Tableau des messages d'erreur » (page α-1) pour savoir quelles dispositions prendre pour corriger le problème.

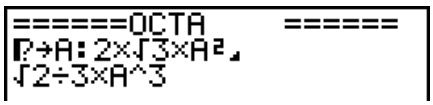
- Notez que la position de l'erreur ne sera pas indiquée lorsque vous appuyez sur [EXIT] si le programme est protégé par un mot de passe.

• Pour éliminer les erreurs à l'origine de mauvais résultats

Si le programme aboutit à un résultat qui ne correspond pas à vos attentes, vérifiez le contenu du programme et effectuez les modifications nécessaires.

[F1](TOP)... Positionne le curseur en début de programme

[F2](BTM)... Positionne le curseur en fin de programme



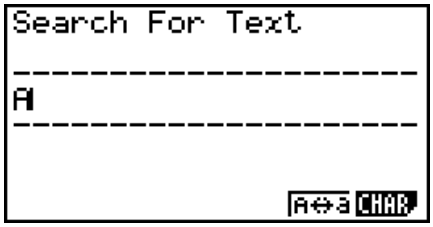
■ Recherche de données à l'intérieur d'un programme

Exemple **Rechercher la lettre « A » dans le programme nommé OCTA**

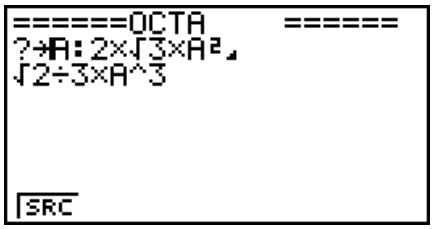
1. Rappeler le programme.
2. Appuyez sur [F3](SRC) et saisissez les données que vous recherchz.

[F3](SRC)

[ALPHA] [X,θ,T](A)



3. Appuyez sur [EXE] pour commencer la recherche.
   Le contenu du programme apparaît à l'écran avec le curseur sur la première occurrence de la donnée spécifiée.*1



4. Chaque pression de [EXE] ou [F1](SRC) provoque le passage du curseur sur le cas suivant des données spécifiées.*2

*1 Le message « Not Found » apparaîtra si la donnée recherchée ne pourra pas être localisée dans le programme.

*2 La recherche s'arrête lorsque les données recherchées ont toutes été localisées.

- Vous ne pouvez pas spécifier le retour à la ligne (↵) ni la commande d'affichage (◢) pour la donnée recherchée.
- Lorsque le contenu du programme est affiché, vous pouvez utiliser les touches du pavé directionnel pour placer le curseur à un autre endroit avant de chercher la prochaine occurrence de la donnée. La recherche ne s'effectuera que sur la partie du programme débutant à la position du curseur lorsque vous appuierez sur [EXE].
- Lorsque la donnée recherchée est localisée, la recherche s'arrête si vous saisissez des données ou déplacez le curseur.
- Si vous faites une erreur lors de la saisie de caractères, appuyez sur [AC] pour annuler la saisie et recommencez depuis le début.

4. Gestion de fichiers

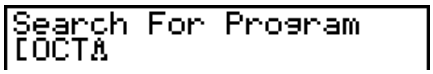
■ Recherche d'un fichier

• Pour localiser un fichier par ses initiales

Exemple **Faire une recherche par initiales pour rappeler le programme nommé OCTA**

1. Quand la liste de programmes est à l'écran, appuyez sur [F6](▷)[F1](SRC) et saisissez les premiers caractères du fichier souhaité.

   [F6](▷)[F1](SRC)
   [9](O)[ln](C)[÷](T)



2. Appuyez sur [EXE] pour commencer la recherche.
   - Le nom commençant par les caractères que vous avez saisis est mis en surbrillance.

- Si aucun programme ne commence par les caractères que vous avez saisis, le message « Not Found » apparaitra à l'écran. Dans ce cas, appuyez sur [EXIT] pour annuler le message d'erreur.

■ Édition d'un nom de fichier

1. Quand la liste de programmes est à l'écran, utilisez ▲ et ▼ pour amener la surbrillance sur le fichier dont vous voulez changer le nom, puis appuyez sur [F6](▷)[F2](REN).
2. Effectuez les changements souhaités.
3. Appuyez sur [EXE] pour enregistrer le nouveau nom et revenir à la liste de programmes.

La liste de programmes est retriée selon les changements effectués dans le nom de fichier.

- Si, après modification, le nouveau nom de fichier est identique à un nom de programme stocké en mémoire, le message « Already Exists » apparaitra. Dans ce cas, vous pouvez effectuer une des deux opérations suivantes pour corriger le problème.
  - Presser [EXIT] pour effacer l'erreur et revenir à l'écran d'édition du nom de fichier.
  - Presser [AC] pour nettoyer le nom de fichier entré et en introduire un nouveau.

■ Effacement d'un programme

• Pour supprimer un programme précis

1. Quand la liste de programmes est à l'écran, utilisez ▲ et ▼ pour amener la surbrillance sur le nom du programme que vous voulez supprimer.
2. Appuyez sur [F4](DEL).
3. Appuyez sur [F1](YES) pour supprimer le programme sélectionné ou sur [F6](NO) pour abandonner l'opération sans rien supprimer.

• Pour supprimer tous les programmes

1. Quand la liste de programmes est à l'écran, appuyez sur [F5](DEL•A).
2. Appuyez sur [F1](YES) pour supprimer tous les programmes ou sur [F6](NO) pour abandonner l'opération sans rien supprimer.

- Vous pouvez aussi effacer tous les programmes en accédant au mode **MEMORY**. Voir « Chapitre 11 Gestionnaire de la mémoire » pour les détails.

■ Enregistrement d'un mot de passe

Lorsque vous écrivez un programme, vous pouvez le protéger par un mot de passe sans lequel il ne sera pas possible d'accéder au contenu de ce programme.

- Il n'est pas nécessaire d'indiquer le mot de passe pour lancer un programme.
- La saisie d'un mot de passe est identique à la saisie d'un nom de fichier.

1. Quand la liste de programmes est à l'écran, appuyez sur [F3](NEW) pour enregistrer le nom de fichier du nouveau programme.
2. Appuyez sur [F5](π0) puis saisissez le mot de passe.
3. Appuyez sur [EXE] pour enregistrer le nom de fichier et le mot de passe. Vous pouvez maintenant introduire le contenu du programme.
4. Une fois que vous avez introduit le programme, appuyez sur [SHIFT][EXIT](QUIT) pour sortir du fichier et revenir à la liste de programmes. Les fichiers qui sont protégés par un mot de passe sont indiqués par un astérisque à la droite du nom de fichier.

■ Rappel d’un programme protégé par un mot de passe

1. Dans la liste de programmes, utilisez ▲ et ▼ pour amener la surbrillance sur le nom du programme que vous voulez rappeler.
2. Appuyez sur F2 (EDIT).
3. Entrez le mot de passe et appuyez sur EXE pour rappeler le programme.

• Le message d’erreur « Mismatch » apparaîtra si vous indiquez le mauvais mot de passe lors du rappel d’un programme protégé par un mot de passe.

5. Guide des commandes

■ Index des commandes

Les conventions utilisées dans cette section pour la description des différentes commandes sont les suivantes.

Texte en caractères gras .. Les commandes et autres paramètres qui doivent toujours être saisis sont en caractères gras.

{Accolades} ..... Les accolades sont utilisées pour indiquer un certain nombre de paramètres dont un doit être sélectionné lorsqu'une commande est utilisée. N'insérez pas d'accolades quand vous introduisez une commande.

[Crochets] ..... Les crochets doivent être utilisés pour indiquer des paramètres qui sont optionnels. N'insérez pas de crochets quand vous introduisez une commande.

Expressions numériques ..... Les expressions numériques, telles que 10, 10 + 20, A, indiquent des constantes, des calculs, des constantes numériques, ou autres.

Caractères alphabétiques .... Les caractères alphabétiques indiquent des chaînes, telles AB.

■ Commandes d'opérations élémentaires

? (Commande de saisie)

Fonction : Demande de saisir une valeur devant être affectée à une variable pendant la programmation.

Syntaxe : ? → <nom de la variable>, "<message à afficher>" ? → <nom de variable>

Exemple : ? → A ↵

Description :

- Cette commande interrompt momentanément l'exécution du programme et demande de saisir une valeur ou une expression à affecter à une variable. Si vous ne spécifiez pas de message à afficher, l'exécution de cette commande fera apparaître « ? » pour indiquer que la calculatrice attend que vous saisissiez une valeur. Si vous spécifiez le message à afficher, « <message à afficher> ? » apparaîtra pour demander de saisir une valeur. Le texte du message à afficher est limité à 255 octets.
- La réponse à cette commande doit être une valeur ou une expression, mais l'expression ne peut pas être un nom de la variable.
- Vous pouvez spécifier un nom de liste, un nom de matrice, un nom de chaîne, une mémoire de fonction (fn), un graphe (Yn), etc. comme nom de variable.

◢ (Commande d'affichage)

Fonction : Affiche un résultat intermédiaire pendant l'exécution d'un programme.

Description :

- Cette commande interrompt momentanément l'exécution d'un programme et affiche un texte en caractères alphabétiques ou le résultat du calcul précédant immédiatement cette commande.
- La commande d'affichage doit être utilisée aux endroits où vous appuieriez normalement sur la touche [EXE] pendant un calcul manuel.

: (Commande d'instructions multiples)

Fonction : Relie deux instructions pour qu'elles soient exécutées dans l'ordre sans interruption.

Description :

- Contrairement à la commande d'affichage (◢), les instructions reliées par cette commande sont exécutées sans interruption.
- La commande d'instructions multiples peut être utilisée pour mettre en relation deux expressions d'un calcul ou deux commandes.
- Vous pouvez utiliser un retour indiqué par ↵ au lieu de la commande d'instructions multiples.

↵ (Retour)

Fonction : Relie deux instructions pour qu'elles soient exécutées dans l'ordre sans interruption.

Description :

- Le retour fonctionne de la même façon que la commande d'instructions multiples.
- Vous pouvez créer une ligne vide dans un programme en tapant un retour à la ligne. L'utilisation du retour à la place de la commande d'instructions multiples facilite la lecture du programme affiché.

' (Délimiteur de commentaire)

Fonction : Indique un commentaire inséré à l'intérieur d'un programme.

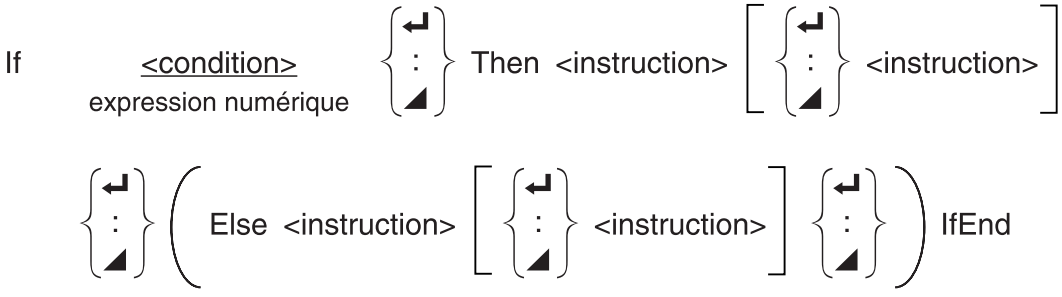
Description : Tout ce qui suit une apostrophe est traité comme commentaire et n'est pas exécutable.

■ Commandes de programmation (COM)

If~Then~(Else~)IfEnd

Fonction : L'instruction Then est exécutée seulement quand la condition If est vraie (pas zéro). L'instruction Else est exécutée quand la condition If est fausse (0). L'instruction IfEnd est toujours exécutée après l'instruction Then ou l'instruction Else.

Syntaxe :



Paramètres : condition, expression numérique

Description :

(1) If ~ Then ~ IfEnd

- Lorsque la condition est vraie, l'exécution passe à l'instruction Then puis continue par l'instruction suivant IfEnd.
- Lorsque la condition est fausse, l'exécution passe à l'instruction suivant IfEnd.

(2) If ~ Then ~ Else ~ IfEnd

- Lorsque la condition est vraie, l'exécution passe à l'instruction Then puis saute à l'instruction suivant IfEnd.

• Lorsque la condition est fausse, l'exécution saute à l'instruction Else et continue par l'instruction suivant IfEnd.

For~To~(Step~)Next

Fonction : Cette commande répète tout ce qui se trouve entre l'instruction For et l'instruction Next. La valeur initiale est affectée à la variable de référence à la première exécution, puis la valeur de la variable de référence change en fonction de la valeur de l'incrément à chaque exécution. L'exécution continue jusqu'à ce que la valeur de la variable de référence dépasse la valeur finale.

Syntaxe : For <valeur initiale> → <nom de la variable de référence> To <valeur finale>

(Step <valeur de l'incrément>) { ↵ / : / ◢ } Next

Paramètres :

• nom de la variable de référence : A à Z

• valeur initiale : valeur ou expression qui produit une valeur (i.e. sin x, A, etc.)

• valeur finale : valeur ou expression qui produit une valeur (i.e. sin x, A, etc.)

• valeur de l'incrément : valeur numérique (défaut : 1)

Description :

• La valeur par défaut de l'incrément est 1.

• La définition d'une valeur initiale inférieure à la valeur finale et d'un incrément positif incrémente la variable de référence à chaque exécution. La définition d'une valeur initiale supérieure à la valeur finale et d'un incrément négatif décrémente la valeur de la variable de référence à chaque exécution.

Do~LpWhile

Fonction : Cette commande répète des commandes particulières tant que sa condition est vraie (pas zéro).

Syntaxe :

Do { ↵ / : / ◢ } <instruction> { ↵ / : / ◢ } LpWhile <condition>
expression numérique

Paramètres : expression

Description :

• Cette commande répète les commandes contenues dans la boucle tant que sa condition est vraie (pas zéro). Quand la condition devient fausse (0), l'exécution continue à partir de l'instruction suivant l'instruction LpWhile.

• Comme la condition vient après l'instruction LpWhile, la condition est testée (vérifiée) après que toutes les commandes à l'intérieur de la boucle ont été exécutées.

While~WhileEnd

Fonction : Cette commande répète des commandes particulières tant que sa condition est vraie (pas zéro).

Syntaxe :

While <condition> ↵ / : / ◢ <instruction> ↵ / : / ◢ WhileEnd

(<condition> : expression numérique)

Paramètres : expression

Description :

- Cette commande répète les commandes contenues dans la boucle tant que sa condition est vraie (pas zéro). Quand la condition devient fausse (0), l'exécution se poursuit à partir de l'instruction suivant l'instruction WhileEnd.
- Comme la condition vient après l'instruction While, elle est testée (vérifiée) avant que les commandes à l'intérieur de la boucle soient exécutées.

■ Commandes de contrôle de la programmation (CTL)

Break

Fonction : Cette commande interrompt l'exécution d'une boucle et continue à partir de la commande suivante après la boucle.

Syntaxe : Break ↵

Description :

- Cette commande interrompt l'exécution d'une boucle et continue à partir de la commande suivante, après la boucle.
- Cette commande peut être utilisée pour interrompre l'exécution des instructions For, Do et While.

Prog

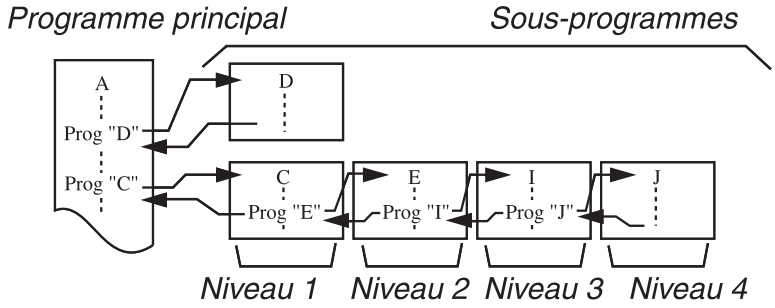
Fonction : Cette commande définit l'exécution d'un autre programme en tant que sous-programme. Dans le mode **RUN•MAT**, cette commande exécute un nouveau programme.

Syntaxe : Prog "nom de fichier" ↵

Exemple: Prog "ABC" ↵

Description :

- Même quand cette commande se trouve à l'intérieur d'une boucle, elle interrompt immédiatement la boucle et démarre le sous-programme.
- Cette commande peut être utilisée autant de fois que nécessaire à l'intérieur d'un programme principal pour faire appel à des sous-programmes qui exécutent des tâches particulières.
- Un sous-programme peut être utilisé à plusieurs endroits à l'intérieur d'un même programme principal, ou il peut être appelé par un certain nombre de programmes principaux.



- L'appel d'un sous-programme exécute celui-ci à partir du début. Quand l'exécution du sous-programme est terminée, on revient au programme principal et continue à partir de l'instruction suivant la commande Prog.
- Une commande Goto~Lbl à l'intérieur d'un sous-programme est valide à l'intérieur de ce sous-programme seulement. Elle ne peut pas être utilisée pour sauter à un label hors du sous-programme.
- Si le sous-programme correspondant au nom de fichier défini par la commande Prog n'existe pas, une erreur se produira.
- Dans le mode **RUN•MAT**, la saisie de la commande Prog et sa validation par [EXE] mettent en route le programme spécifié par la commande.

Return

Fonction : Cette commande fait revenir d'un sous-programme au programme d'origine.

Syntaxe : Return ↵

Description : L'exécution de la commande de retour à l'intérieur du programme principal interrompt l'exécution du programme. L'exécution de la commande de retour à l'intérieur d'un sous-programme interrompt le sous-programme et fait revenir au programme principal, à l'endroit où le sous-programme a commencé.

Stop

Fonction : Cette commande termine l'exécution d'un programme.

Syntaxe : Stop ↵

Description :

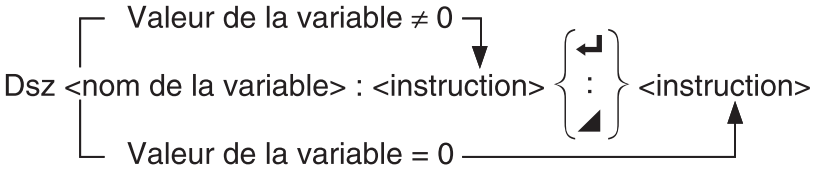
- Cette commande termine l'exécution du programme.
- L'exécution de cette commande à l'intérieur d'une boucle achève l'exécution du programme sans qu'aucune erreur ne se produise.

■ Commandes de saut (JUMP)

Dsz

Fonction : Cette commande est un saut avec compteur qui décrémente la valeur d'une variable de référence d'une unité, puis passe à l'instruction suivant la commande d'instruction multiple quand la valeur de la variable est égale à zéro.

Syntaxe :



Paramètres : nom de la variable : A à Z, r, θ

[Exemple] Dsz B : Décrémente la valeur affectée à la variable B d'une unité.

Description : Cette commande décrémente la valeur d'une variable de référence d'une unité, puis la teste (vérifie). Si la valeur actuelle n'est pas égale à zéro, l'exécution continue avec l'instruction suivante. Si la valeur est égale à zéro, l'exécution passe à l'instruction suivant la commande d'instructions multiples (:), la commande d'affichage de résultat (◢), ou la commande de retour (↵).

Goto~Lbl

Fonction : Cette commande effectue un saut inconditionnel à un endroit défini.

Syntaxe : Goto <nom de label> ~ Lbl <nom de label>

Paramètres : nom de label : valeur (0 à 9) variable (A à Z, r, θ)

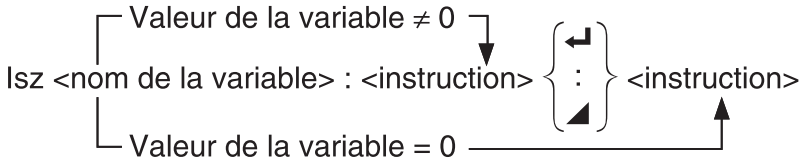
Description :

- Cette commande comprend deux parties : Goto n (n étant un paramètre comme décrit plus haut) et Lbl n (n étant un paramètre référencée par Goto n). Cette commande fait passer l'exécution du programme à l'instruction Lbl dont le paramètre n correspond à celui qui a été spécifié par l'instruction Goto.
- Cette commande peut être utilisée pour revenir au début d'un programme ou pour sauter à un endroit quelconque du programme.
- Cette commande peut être combinée aux sauts conditionnels et aux sauts avec compteurs.
- S'il n'y a aucune instruction Lbl dont la valeur correspond à celle définie par l'instruction Goto, une erreur se produira.

Isz

Fonction : Cette commande est un saut avec compteur qui incrémente la valeur de la variable de référence d'une unité, puis passe à l'instruction suivant la commande d'instruction multiple quand la valeur de la variable est égale à zéro.

Syntaxe :



Paramètres : nom de la variable : A à Z, r, θ

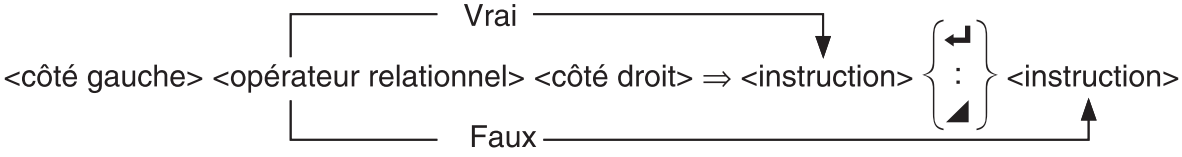
[Exemple] Isz A : Incrémente la valeur affectée à la variable A d'une unité.

Description : Cette commande incrémente la valeur d'une variable de référence d'une unité, puis la teste (vérifie). Si la valeur actuelle n'est pas égale à zéro, l'exécution continue avec l'instruction suivante. Si la valeur est égale à zéro, l'exécution passe à l'instruction suivant la commande d'instructions multiples (:), la commande d'affichage de résultat (◢) ou la commande de retour (↵).

⇒ (Code de saut)

Fonction : Ce code est utilisé pour poser les conditions d'un saut conditionnel. Le saut est exécuté quand les conditions sont fausses.

Syntaxe :



Paramètres :

- côté gauche/côté droit : variable (A à Z, r, θ), constante numérique, expression avec variable (comme : A × 2)
- opérateur relationnel : =, ≠, >, <, ≥, ≤ (page 8-19)

Description :

- Le saut conditionnel compare le contenu de deux variables ou les résultats de deux expressions, et le saut est exécuté ou non selon les résultats de la comparaison.
- Si le résultat de la comparaison est vrai, l'exécution se poursuit à partir de l'instruction qui suit la commande ⇒. Si le résultat de la comparaison est faux, l'exécution passe aux instructions suivant la commande d'instructions multiples (:), la commande d'affichage de résultat (◢) ou la commande de retour (↵).

Menu

Fonction : Crée un menu de branchement dans un programme.

Syntaxe : Menu "<chaîne (nom du menu)>", "<chaîne (nom de la branche) 1>" ,<valeur ou variable 1>, "<chaîne (nom de la branche) 2>" ,<valeur ou variable 2>, ... , "<chaîne (nom de la branche) n>" ,<valeur ou variable n>

Paramètres : valeur (0 à 9), variable (A à Z, r, θ)

Description :

- Chaque partie "<chaîne (nom de la branche) n>",<valeur ou variable n> correspond à un ensemble de branche et l'ensemble doit être inclus en entier.
- On peut inclure de deux à neuf ensembles de branche. Une erreur se produit si le nombre d'ensembles de branche est 1 ou supérieur à 9.
- Lors de la sélection d'une branche dans le menu pendant le déroulement d'un programme provoque un saut ver le même type d'étiquette (Lbl n) que celle utilisée avec la commande Goto. En spécifiant « "OK", 3 » pour la partie « "<chaîne (nom de la branche) n>",<valeur ou variable n> » spécifie un saut vers Lbl 3.

Exemple : Lbl 2↵

Menu "IS IT DONE?", "OK", 1, "EXIT", 2↵

Lbl 1↵

"IT'S DONE !"

■ Commandes d'effacement (CLR)

ClrGraph

Fonction : Cette commande efface l'écran graphique.

Syntaxe : ClrGraph↵

Description : Cette commande efface l'écran graphique pendant l'exécution du programme.

ClrList

Fonction : Cette commande supprime les données d'une liste.

Syntaxe : ClrList <nom de liste>

ClrList

Paramètres : nom de liste : 1 à 26, Ans

Description : Cette commande supprime les données de la liste désignée par « nom de liste ». Toutes les données de la liste sont supprimées si rien n'est spécifié pour le « nom de liste ».

ClrMat

Fonction : Cette commande supprime les données de matrice.

Syntaxe : ClrMat <nom de matrice>

ClrMat

Paramètres : nom de matrice : A à Z, Ans

Description : Cette commande supprime les données de la matrice désignée par « nom de matrice ». Toutes les données de toutes les matrices sont supprimées si aucun « nom de matrice » n'est désigné.

ClrText

Fonction : Cette commande efface l'écran de texte.

Syntaxe : ClrText↵

Description : Cette commande efface le texte de l'écran pendant l'exécution du programme.

■ Commandes d'affichage (DISP)

DispF-Tbl, DispR-Tbl **Aucun paramètre**

Fonction : Ces commandes affichent des tables numériques.

Description :

- Ces commandes créent des tables numériques pendant l'exécution d'un programme selon les paramètres définis dans le programme.
- DispF-Tbl crée une table de fonctions, tandis que DispR-Tbl crée une table de récurrence.

DrawDyna **Aucun paramètre**

Fonction : Cette commande exécute un tracé de graphe dynamique.

Description : Cette commande trace un graphe dynamique au cours de l'exécution du programme selon les paramètres de traçage définis dans le programme.

DrawFTG-Con, DrawFTG-Plt **Aucun paramètre**

Fonction : Cette commande utilise les valeurs d'une table pour représenter graphiquement une fonction.

Description :

- Cette commande trace un graphe en fonction selon les paramètres définis dans le programme.
- DrawFTG-Con produit un graphe à points connectés, tandis que DrawFTG-Plt produit un graphe à points séparés.

DrawGraph **Aucun paramètre**

Fonction : Cette commande trace un graphe.

Description : Cette commande trace un graphe selon les paramètres de traçage définis dans le programme.

DrawR-Con, DrawR-Plt **Aucun paramètre**

Fonction : Ces commandes tracent des expressions de récurrence, avec a_n (b_n ou c_n) comme axe vertical et n comme axe horizontal.

Description :

- Ces commandes tracent des expressions de récurrence selon les paramètres définis dans le programme, avec an a_n (b_n ou c_n) comme axe vertical et n comme axe horizontal.
- DrawR-Con produit un graphe à points connectés, tandis que DrawR-Plt produit un graphe à points séparés.

DrawRΣ-Con, DrawRΣ-Plt **Aucun paramètre**

Fonction : Ces commandes tracent des expressions de récurrence, avec Σa_n (Σb_n ou Σc_n) comme axe vertical et n comme axe horizontal.

Description :

- Ces commandes tracent des expressions de récurrence selon les paramètres définis dans le programme, avec Σa_n (Σb_n ou Σc_n) comme axe vertical et n comme axe horizontal.
- DrawRΣ-Con produit un graphe à points connectés tandis que DrawRΣ-Plt produit un graphe à points séparés.

DrawStat

Fonction : Trace un graphe statistique.

Syntaxe : Voir « Utilisation de calculs et de graphes statistiques dans un programme » à la page 8-26.

Description : Cette commande trace un graphe statistique selon les paramètres de traçage définis dans le programme.

DrawWeb

Fonction : Cette commande représente graphiquement la convergence/divergence d'une expression de récurrence (graphe WEB).

Syntaxe : DrawWeb <type de récurrence>[, <nombre de lignes>]↵

Exemple: DrawWeb a_{n+1} (b_{n+1} ou c_{n+1}), 5↵

Description :

- Cette commande représente graphiquement la convergence/divergence d'une expression de récurrence (graphe WEB).
- L'omission de la définition du nombre de lignes impose automatiquement 30, la valeur par défaut.

PlotPhase

Fonction : Trace le graphe d'un diagramme cartésien (courbe reportée en coordonnées cartésiennes) de séquences numériques dont les éléments correspondent aux abscisses (axe x) et aux ordonnées (axe y).

Syntaxe : PlotPhase <nom de la séquence numérique de l'axe x>, <nom de la séquence numérique de l'axe y>

Description :

- Seules les commandes suivantes peuvent être entrées pour chaque argument afin de spécifier le tableau récursif :

a_n, b_n, c_n, a_{n+1}, b_{n+1}, c_{n+1}, a_{n+2}, b_{n+2}, c_{n+2}, Σa_n, Σb_n, Σc_n, Σa_{n+1}, Σb_{n+1}, Σc_{n+1}, Σa_{n+2}, Σb_{n+2}, Σc_{n+2}

- Une erreur « Memory ERROR » se produit lorsque vous spécifiez le nom d'une séquence numérique qui ne possède pas de valeurs stockées dans le tableau récursif.

Exemple : PlotPhase Σb_{n+1}, Σa_{n+1}

Trace le graphe d'un diagramme cartésien en utilisant Σb_{n+1} pour les abscisses et Σa_{n+1} pour les ordonnées.

■ Commandes d'entrée/sortie (I/O)

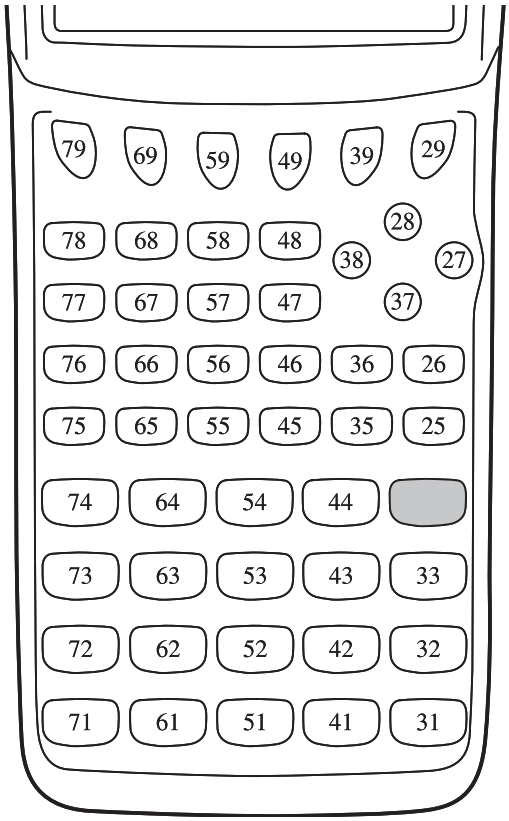
Getkey

Fonction : Cette commande retourne le code correspondant à la dernière touche appuyée.

Syntaxe : Getkey ↵

Description :

- Cette commande retourne le code correspondant à la dernière touche appuyée.



- On revient à la valeur zéro si aucune touche n'a été activée avant l'exécution de cette commande.
- Cette commande peut être utilisée à l'intérieur d'une boucle.

Locate

Fonction : Cette commande affiche des caractères alphanumériques à une position précise de l'écran de texte.

Syntaxe : Locate <numéro de colonne>, <numéro de ligne>, <valeur>

Locate <numéro de colonne>, <numéro de ligne>, <expression numérique>

Locate <numéro de colonne>, <numéro de ligne>, "<chaîne>"

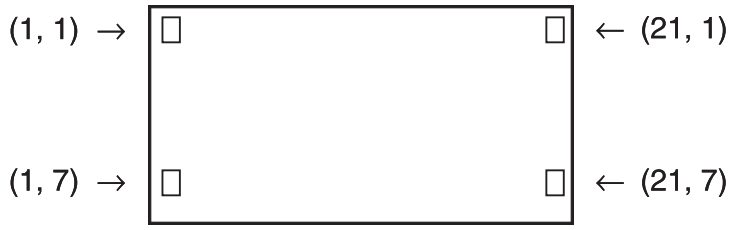
[Exemple] Locate 1, 1, "AB" ↵

Paramètres :

- numéro de ligne : numéro de 1 à 7
- numéro de colonne : numéro de 1 à 21
- valeur et expression numérique
- chaîne : chaîne de caractères

Description :

- Cette commande affiche des valeurs (y compris le contenu des variables) ou du texte à une position précise de l'écran de texte. Si un calcul est introduit, le résultat de ce calcul sera affiché.
- La ligne est désignée par une valeur de 1 à 7 et la colonne est désignée par une valeur de 1 à 21.



Exemple : Cls ↵
Locate 7, 1, "CASIO FX"

Ce programme affiche le texte « CASIO FX » au centre de la première ligne de l'écran.

- Dans certains cas, la commande ClrText doit être exécutée avant de mettre le programme précédent en route.

Receive(/ Send(

Fonction : Cette commande reçoit les données d'un appareil externe et envoie des données à un appareil externe.

Syntaxe : Receive (<données>) / Send (<données>)

Description :

- Cette commande reçoit des données et envoie des données à un appareil externe.
- Les types de données suivantes peuvent être reçues (envoyées) par cette commande.
  - Valeurs individuelles affectées aux variables
  - Données de matrices (toutes les valeurs - des valeurs individuelles ne peuvent pas être spécifiées)
  - Données de listes (toutes les valeurs - des valeurs individuelles ne peuvent pas être spécifiées)

OpenComport38k / CloseComport38k

Fonction : Ouvre et ferme le port COM à 3 broches (série).

Description : Voir la commande Receive38k/Send38k ci-dessous.

Receive38k / Send38k

Fonction : Exécute l'envoi et la réception de données à un débit de 38 kbps.

Syntaxe : Send38k <expression>

Receive38k {<nom de variable> / <nom de liste>}

Description :

- La commande OpenComport38k doit être exécutée avant l'exécution de la commande Receive38k/Send38k.
- La commande CloseComport38k doit être exécutée après l'exécution de la commande Receive38k/Send38k.
- Si le câble de communication est débranché lorsque vous exécutez cette commande, l'exécution du programme continuera sans générer d'erreur.

■ Opérateurs relationnels avec saut conditionnel (REL)

=, ≠, >, <, ≥, ≤

Fonction : Ces opérateurs relationnels sont utilisés communément avec la commande de saut conditionnel.

Syntaxe : <côté gauche> <opérateur relationnel> <côté droit>

Paramètres :

- côté gauche/côté droit : variable (A à Z, r, θ), constante numérique, expression avec variable (comme : A × 2)
- opérateur relationnel : =, ≠, >, <, ≥, ≤

■ Chaînes

Une chaîne est une séquence de caractères enfermés entre doubles guillemets. Dans un programme, les chaînes sont utilisées pour spécifier l'affichage de texte. Une chaîne composée de caractères numériques (telle que "123") ou une expression (telle que "x–1") ne peut être traitée dans un calcul.

Pour afficher une chaîne à un endroit spécifique de l'écran, utilisez la commande Locate (page 8-17).

- Pour inclure un double guillemet (") ou une barre oblique inverse (\) (back slash) dans une chaîne, mettez une barre oblique inverse (\) devant le double guillemet (") ou devant la barre oblique inverse (\).

  Exemple 1 : Pour inclure la séquence de caractères « Japan:"Tokyo" » dans une chaîne
  "Japan:\"Tokyo\""

  Exemple 2 : Pour inclure « main\abc » dans une chaîne
  "main\\abc"

Vous pouvez entrer une barre oblique inverse à partir du menu qui s'affiche en appuyant sur [F6](CHAR) [F2](SYBL) dans le mode **PRGM**, ou bien, à partir de la catégorie String (chaîne) du catalogue qui s'affiche quand vous appuyez sur [SHIFT] [4](CATALOG).

- Vous pouvez affecter des chaînes à la mémoire de stokage de chaîne (de Str 1 jusqu'à Str 20). Pour plus de détails sur les opérations avec des chaînes, voir « Mémoire de stockage des chaînes » (page 2-7).
- Vous pouvez utiliser la commande « + » (page 8-22) pour concaténer des chaînes dans un argument.

• Une fonction ou une commande à l'intérieur d'une fonction de chaîne (Exp(, StrCmp(, etc.) est traitée comme un caractère unique. Par exemple, la fonction « sin » est traitée comme un caractère unique.

Exp(

Fonction : Effectue la conversion d'une chaîne en une expression et exécute l'expression résultante.

Syntaxe : Exp("<chaîne>"[)]

Exp►Str(

Fonction : Effectue la conversion d'une expression de graphe en une chaîne et l'affecte à une variable spécifiée.

Syntaxe : Exp►Str(<formule>, <nom de variable de chaîne>[)]

Description : Pour le premier argument (<formule>) on peut utiliser une expression de graphe (Y_n, r, X_t, Y_t, X), une formule récursive (a_n, a_{n+1}, a_{n+2}, b_n, b_{n+1}, b_{n+2}, c_n, c_{n+1}, c_{n+2}) ou une mémoire de fonction (f_n).

StrCmp(

Fonction : Compare les chaînes « <chaîne 1> » et « <chaîne 2> » (comparaison basée sur les codes des caractères).

Syntaxe : StrCmp("<chaîne 1>", "<chaîne 2>"[)]

Description : Compare deux chaînes et retourne une des valeurs suivantes :

Retourne 0 lorsque « <chaîne 1> » = « <chaîne 2> ».

Retourne 1 lorsque « <chaîne 1> » > « <chaîne 2> ».

Retourne –1 lorsque « <chaîne 1> » < « <chaîne 2> ».

StrInv(

Fonction : Inverse la séquence de caractères d'une chaîne.

Syntaxe : StrInv("<chaîne>"[)]

StrJoin(

Fonction : Concatène « <chaîne 1> » et « <chaîne 2> ».

Syntaxe : StrJoin("<chaîne 1>", "<chaîne 2>"[)]

Note : On peut obtenir le même résultat en utilisant la commande « + » (page 8-22).

StrLeft(

Fonction : Copie une chaîne jusqu'au $n^{ième}$ caractère à partir de la gauche.

Syntaxe : StrLeft("<chaîne>", n[)] (0 ≦ n ≦ 9999, n est un nombre naturel)

StrLen(

Fonction : Retourne la longueur d'une chaîne (le nombre de caractères qui la composent).

Syntaxe : StrLen("<chaîne>"[)]

StrLwr(

Fonction : Effectue la conversion de tous les caractères d'une chaîne en minuscules (casse inférieure).

Syntaxe : StrLwr("<chaîne>"[)]

StrMid(

Fonction : Effectue l'extraction du $n^{ième}$ jusqu'au $m^{ième}$ caractère d'une chaîne.

Syntaxe : StrMid("<chaîne>", n [,m)] (0 ≦ n ≦ 9999, n est un nombre naturel)

Description : L'omission de « m » effectue l'extraction à partir du $n^{ième}$ jusqu'à la fin de la chaîne.

StrRight(

Fonction : Copie une chaîne jusqu'au $n^{ième}$ caractère à partir de la droite.

Syntaxe : StrRight("<chaîne>", n[)] (0 ≦ n ≦ 9999, n est un nombre naturel)

StrRotate(

Fonction : Déplace circulairement les caractères d'une chaîne situés à gauche et à droite du $n^{ième}$ caractère.

Syntaxe : StrRotate("<chaîne>", [,n)] (–9999 ≦ n ≦ 9999, n est un nombre entier)

Description : Le déplacement circulaire s'effectue vers la gauche si « n » est positif et vers la droite si « n » est négatif. Si « n » est omis la fonction utilise +1 comme valeur par défaut.

Exemple : StrRotate("abcde", 2)..... Retourne la chaîne « cdeab ».

StrShift(

Fonction : Déplace les caractères d'une chaîne à gauche ou à droite sur n caractères.

Syntaxe : StrShift("<chaîne>", [,n)] (–9999 ≦ n ≦ 9999, n est un nombre entier)

Description : Le déplacement s'effectue vers la gauche si « n » est positif et vers la droite si « n » est négatif. Si « n » est omis la fonction utilise +1 comme valeur par défaut.

Exemple : StrShift("abcde", 2) ..... Retourne la chaîne « cde ».

StrSrc(

Fonction : Effectue une recherche dans « <chaîne 1> », à partir du caractère spécifié ($n^{ième}$ caractère à partir du début de la chaîne), pour déterminer si elle contienne les données spécifiées par « <chaîne 2> ». Si les données sont trouvées, cette commande retourne la position du premier caractère de « <chaîne 2> », à partir du début de « <chaîne 1> ».

Syntaxe : StrSrc("<chaîne 1>", "<chaîne 2>"[,n)] (0 ≦ n ≦ 9999, n est un nombre naturel)

Description : L'omission du point de départ provoque une recherché à partir du début de « <chaîne 1> ».

StrUpr(

Fonction : Effectue la conversion de tous les caractères d'une chaîne en majuscules (casse supérieure).

Syntaxe : StrUpr("<chaîne>"[)]

+

Fonction : Concatène « <chaîne 1> » et « <chaîne 2> ».

Syntaxe : "<chaîne 1>"+"<chaîne 2>"

Exemple : "abc"+"de"→Str 1 ..... Affecte « abcde » à Str 1.

■ Divers

RclCapt

Fonction : Affiche le contenu spécifié par le numéro de capture d'écrans.

Syntaxe : RclCapt <numéro de capture d'écrans> (numéro de capture d'écrans : 1 à 20)

6. Utilisation des fonctions de la calculatrice dans un programme

■ Affichage de texte

Il suffit de mettre un texte entre guillemets pour l'inclure dans un programme. Ce texte sera affiché pendant l'exécution du programme, ce qui signifie que vous pouvez ajouter des labels pour entrer des messages et résultats.

Programme	Affichage
"CASIO"	CASIO
? → X	?
"X =" ? → X	X = ?

- Si le texte est suivi d'une formule de calcul, n'oubliez pas d'insérer une commande d'affichage (◢) entre le texte et le calcul.
- La saisie de plus de 21 caractères fait passer à la ligne suivante. L'écran défile automatiquement si le texte a plus de 21 caractères.
- Le texte d'un commentaire est limité à 255 octets.

■ Utilisation d'opérations sur les lignes d'une matrice dans un programme

Ces commandes vous permettent de travailler sur les lignes d'une matrice dans un programme.

- Pour ce programme, accédez au mode **RUN•MAT** et utilisez l'éditeur de matrices pour indiquer la matrice, puis accédez au mode **PRGM** pour écrire le programme.

• Pour échanger le contenu de deux lignes (Swap)

Exemple 1 **Echanger les valeurs de la ligne 2 et de la ligne 3 dans la matrice suivante :**

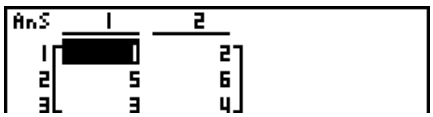
Matrice A = $\begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{bmatrix}$

La syntaxe utilisée pour ce programme est la suivante.

Swap A, 2, 3↵
- 2, 3 : *Lignes à échanger*
- A : *Nom de la matrice*

Mat A

L'exécution de ce programme produit le résultat suivant.



• Pour calculer un produit scalaire (∗ Row)

Exemple 2 **Calculer le produit scalaire de la ligne 2 de la matrice dans l'exemple 1, par 4**

La syntaxe utilisée pour ce programme est la suivante.

∗Row 4, A, 2↵
- 2 : *Ligne*
- A : *Nom de la matrice*
- 4 : *Multiplicateur*

Mat A

• Pour calculer le produit scalaire et ajouter le résultat à une autre ligne (∗Row+)

Exemple 3 **Calculer le produit scalaire de la ligne 2 de la matrice citée dans l'exemple 1, par 4, et ajouter le résultat aux valeurs de la ligne 3**

La syntaxe utilisée pour ce programme est la suivante.

∗Row+ 4, A, 2, 3↵
- 3 : *Lignes à ajouter*
- 2 : *Ligne pour laquelle le produit doit être calculé*
- A : *Nom de la matrice*
- 4 : *Multiplicateur*

Mat A

• Pour additionner deux lignes (Row+)

Exemple 4 **Additionner les valeurs de la ligne 2 et de la ligne 3 de la matrice citée dans l'exemple 1**

La syntaxe utilisée pour ce programme est la suivante.

Row+ A, 2, 3↵
- 3 : *Numéro de ligne à ajouter à*
- 2 : *Numéro de ligne à ajouter*
- A : *Nom de la matrice*

Mat A

■ Utilisation de fonctions graphiques dans un programme

Vous pouvez intégrer des fonctions graphiques dans un programme pour tracer des graphes, puis superposer plusieurs graphes. Les différentes syntaxes nécessaires pour la programmation de fonctions graphiques sont les suivantes.

- Fenêtre d'affichage — View Window –5, 5, 1, –5, 5, 1 ↵
- Saisie de la fonction graphique — Y = Type ↵ .....Définit le type de graphe.
  "$X^2 - 3$" → Y1*1 ↵
- Tracé de graphe — DrawGraph ↵

*1 Entrez ce Y1 avec [VARS] [F4](GRPH)[F1](Y)[1] (affiché en tant que Y1). Une erreur « Syntax ERROR » se produit lorsque vous entrez « Y » avec les touches de la calculatrice.

• Syntaxe d'autres fonctions de représentation graphique

- V-Window
  View Window <Xmin>, <Xmax>, <Xscale>, <Ymin>, <Ymax>, <Yscale>, <Tθ min>, <Tθ max>, <Tθ pitch>
  StoV-Win <zone de V-Win> ..... zone : 1 à 6
  RclV-Win <zone de V-Win> ..... zone : 1 à 6
- Zoom
  Factor <facteur X>, <facteur Y>
  ZoomAuto..... Aucun paramètre
- Pict
  StoPict <zone de l'image> ..... zone : 1 à 6
  expression numérique
  StoPict <zone de l'image> ..... zone : 1 à 6
  expression numérique
- Sketch
  PlotOn <abscisse X>, <ordonnée Y>
  PlotOff <abscisse X>, <ordonnée Y>
  PlotChg <abscisse X>, <ordonnée Y>
  PxlOn <numéro de ligne>, <numéro de colonne>
  PxlOff <numéro de ligne>, <numéro de colonne>
  PxlChg <numéro de ligne>, <numéro de colonne>
  PxlTest <numéro de ligne>, <numéro de colonne>
  Text <numéro de ligne>, <numéro de colonne>, "<texte>"
  Text <numéro de ligne>, <numéro de colonne>, <expression>
  SketchThick <dessin ou instruction de graphe>
  SketchBroken <dessin ou instruction de graphe>
  SketchDot <dessin ou instruction de graphe>
  SketchNormal <dessin ou instruction de graphe>
  Tangent <fonction>, <abscisse X>
  Normal <fonction>, <abscisse X>
  Inverse <fonction>
  Line
  F-Line <abscisse X 1>,<ordonnée Y 1>,<abscisse X 2>,<ordonnée Y 2>
  Circle <abscisse X du point central>,<ordonnée Y du point central>, <valeur R du rayon>

Vertical <abscisse X>

Horizontal <ordonnée Y>

■ Utilisation des fonctions de graphe dynamique dans un programme

L'utilisation des fonctions de graphe dynamique dans un programme permet de répéter les tracés d'un graphe dynamique. La définition de la plage du graphe dynamique à l'intérieur d'un programme s'effectue de la façon suivante.

- Plage du graphe dynamique

  1 → D Start ↵

  5 → D End ↵

  1 → D pitch ↵

■ Utilisation des fonctions de table et graphe dans un programme

L'utilisation des fonctions de table et graphe dans un programme permet de créer des tables numériques et d'effectuer des opérations graphiques. Les différentes syntaxes nécessaires lors de la programmation de fonctions avec table et graphe sont les suivantes.

- Définition échelle table

  1 → F Start ↵

  5 → F End ↵

  1 → F pitch ↵

- Génération d'une table numérique

  DispF-Tbl ↵

- Tracé de graphe

  Gràphe à points connectés : DrawFTG-Con ↵

  Graphe à points séparés : DrawFTG-Plt ↵

■ Utilisation des fonctions de récurrence de table et graphe dans un programme

L'intégration de fonctions de récurrence de table et graphe dans un programme permet de créer des tables numériques et d'effectuer des opérations graphiques. Les différentes syntaxes nécessaires lors de la programmation de fonctions de récurrence avec table et graphe sont les suivantes.

- Saisie de la formule de récurrence

  a_{n+1} Type ↵ .... Spécifier le type de récurrence.

  "3a_n + 2" → a_{n+1} ↵

  "4b_n + 6" → b_{n+1} ↵

- Définition de la plage de la table

  1 → R Start ↵

  5 → R End ↵

  1 → a_0 ↵

  2 → b_0 ↵

  1 → a_n Start ↵

  3 → b_n Start ↵

- Génération d'une table numérique

  DispR-Tbl ↵

- Opération de traçage d'un graphe

  Graphe à points connectés : DrawR-Con ↵,

  DrawRΣ-Con ↵

  Graphe à points séparés : DrawR-Plt ↵,

  DrawRΣ-Plt ↵

- Graph statistique convergence/divergence (graphe WEB)

  DrawWeb a_{n+1}, 10 ↵

■ Utilisation des fonctions de tri de listes dans un programme

Cette commande vous permet de trier les données de listes dans un ordre ascendant ou descendant.

• Ordre ascendant

①　②
SortA (List 1, List 2, List 3)

Listes à trier (six listes au maximum)

① F4 F3 F1　② OPTN F1 F1

• Ordre descendant

③
SortD (List 1, List 2, List 3)

Listes à trier (six listes au maximum)

③ F4 F3 F2

■ Utilisation de calculs et de graphes statistiques dans un programme

L'insertion de calculs et de graphes statistiques dans un programme vous permet de calculer et de représenter graphiquement des données statistiques.

• Pour définir les conditions et tracer un graphe statistique

Après une commande StatGraph (« S-Gph1 », « S-Gph2 » ou « S-Gph3 »), vous devez définir les conditions suivantes :

• État avec tracé ou sans tracé de graphe (DrawOn/DrawOff)
• Type de graphe
• Emplacement des données sur l'axe x (nom de liste)
• Emplacement des données sur l'axe y (nom de liste)
• Emplacement des valeurs d'effectifs de données (nom de liste)
• Type de point
• Paramètre d'affichage des graphes de type camembert (% ou Data)
• Spécification de la liste de données pour un graphe de type camembert à pourcentages (None ou nom de la liste)
• Données pour la première barre d'un histogramme (nom de liste)
• Données pour la deuxième et pour la troisième barre d'un histogramme (nom de liste)
• Orientation d'un histogramme à barres (Length ou Horizontal)

Les conditions de tracé du graphe dépendent du type de graphe. Voir « Changement des paramètres d'un graphe » (page 6-1).

• La définition typique d'un diagramme de corrélation ou d'un graphe linéaire xy est la suivante.

S-Gph1 DrawOn, Scatter, List 1, List 2, 1, Square ↵

Dans le cas d'un graphe linéaire xy, remplacez « Scatter » dans la définition précédente par « xyLine ».

• La définition typique d'un traçage de probabilité normale est la suivante.

S-Gph1 DrawOn, NPPlot, List 1, Square ↵

• La définition typique d'un graphe à variable unique est la suivante.

S-Gph1 DrawOn, Hist, List 1, List 2 ↵

Le même format peut être utilisé pour les types de graphes suivants en remplaçant simplement « Hist » de la définition précédente par le type de graphe applicable.

Histogramme	Hist	Loi de probabilité normale ...	N-Dist
Boîte-médiane	MedBox*1	Ligne brisée	Broken

*1 Outliers:On

S-Gph1 DrawOn, MedBox, List 1, 1, 1

Outliers:Off

S-Gph1 DrawOn, MedBox, List 1, 1, 0

• La définition typique d'un graphe de régression est la suivante.

S-Gph1 DrawOn, Linear, List 1, List 2, List 3 ↵

Le même format peut être utilisé pour les types de graphes suivants en remplaçant simplement « Linear » de la définition précédente par le type de graphe applicable.

Régression linéaire.....	Linear	Régression logarithmique	Log
Med-Med.....	Med-Med	Régression exponentielle.....	ExpReg(a·e^bx)
Régression quadratique..	Quad		ExpReg(a·b^x)
Régression cubique	Cubic	Régression de puissance	Power
Régression quartique.....	Quart		

• La définition typique d'un graphe de régression sinusoïdale est la suivante.

S-Gph1 DrawOn, Sinusoidal, List 1, List 2 ↵

• La définition typique d'un graphe de régression logistique est la suivante.

S-Gph1 DrawOn, Logistic, List 1, List 2 ↵

• La définition typique d'un graphe de type camembert est la suivante.

S-Gph1 DrawOn, Pie, List 1, %, None ↵

• La définition typique d'un graphe de type histogramme est la suivante.

S-Gph1 DrawOn, Bar, List 1, None, None, StickLength ↵

• Pour tracer un graphe statistique, insérez la commande « DrawStat » en suivant la ligne de spécification de la condition du graphe.

ClrGraph

S-Wind Auto

{1, 2, 3} → List 1

{1, 2, 3} → List 2

S-Gph1 DrawOn, Scatter, List 1, List 2, 1, Square ↵

DrawStat

■ Utilisation de graphes de distributions dans un programme

Pour tracer des graphes de distributions dans un programme on utilise des commandes spéciales.

• Pour tracer le graphe d'une distribution normale cumulative

① DrawDistNorm <Lower>, <Upper> [,σ, μ]

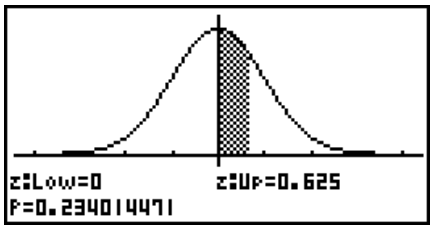
- μ : *Moyenne de la population*[*1]
- σ : *Écart-type de la population*[*1]
- Upper : *Limite supérieure des données*
- Lower : *Limite inférieure des données*

① F4 F1 F5 F1

[*1] Ce terme peut être omis. En cas d'omission le calcul se fait avec $\sigma = 1$ et $\mu = 0$.

$$p = \frac{1}{\sqrt{2\pi}\sigma}\int_{Lower}^{Upper} e^{-\frac{(x-\mu)^2}{2\sigma^2}}dx \qquad \text{ZLow} = \frac{Lower-\mu}{\sigma} \qquad \text{ZUp} = \frac{Upper-\mu}{\sigma}$$

- L'exécution de DrawDistNorm effectue le calcul ci-dessus selon les conditions spécifiées et trace le graphe. La région ZLow ≦ x ≦ ZUp du graphe est remplie à la même occasion.



- Simultanément, les valeurs des résultats du calcul de p, ZLow et ZUp sont affectées aux variables p, ZLow et ZUp respectivement et p est affectée à Ans.

• Pour tracer le graphe d'une distribution t de Student cumulative

① DrawDistT <Lower>, <Upper>, <df>

- df : *Degrés de liberté*
- Upper : *Limite supérieure des données*
- Lower : *Limite inférieure des données*

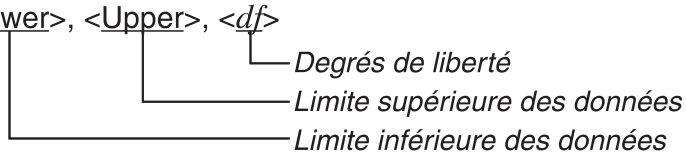
① F4 F1 F5 F2

$$p = \int_{Lower}^{Upper} \frac{\Gamma\left(\frac{df+1}{2}\right)}{\Gamma\left(\frac{df}{2}\right)} \times \frac{\left(1+\frac{x^2}{df}\right)^{-\frac{df+1}{2}}}{\sqrt{\pi \times df}}dx \qquad \text{tLow} = Lower \qquad \text{tUp} = Upper$$

- L'exécution de DrawDistT effectue le calcul ci-dessus selon les conditions spécifiées et trace le graphe. La région Lower ≦ x ≦ Upper du graphe est remplie à la même occasion.
- Simultanément, la valeur du résultat du calcul de p et les valeurs d'entrée inférieure et supérieure sont affectées aux variables p, tLow et tUp respectivement et p est affectée à Ans.

• Pour tracer le graphe d'une distribution χ^2 cumulative

① DrawDistChi <Lower>, <Upper>, <*df*>



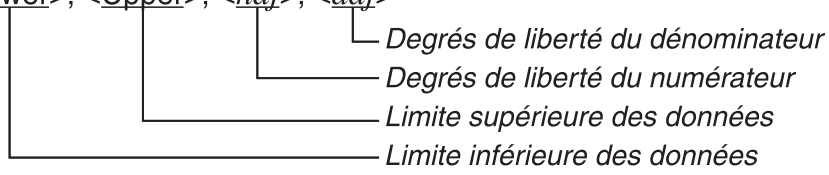
① F4 F1 F5 F3

$$p = \int_{Lower}^{Upper} \frac{1}{\Gamma\left(\frac{df}{2}\right)} \times \left(\frac{1}{2}\right)^{\frac{df}{2}} \times x^{\left(\frac{df}{2}-1\right)} \times e^{-\frac{x}{2}}\, dx$$

- L'exécution de DrawDistChi effectue le calcul ci-dessus selon les conditions spécifiées et trace le graphe. La région Lower ≦ x ≦ Upper du graphe est remplie à la même occasion.
- Simultanément, la valeur du résultat du calcul de p est affectée aux variables p et Ans.

• Pour tracer le graphe d'une distribution F cumulative

① DrawDistF <Lower>, <Upper>, <*ndf*>, <*ddf*>



① F4 F1 F5 F4

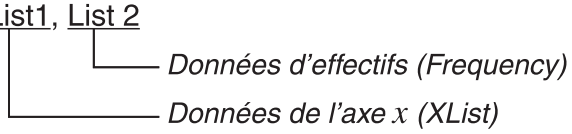
$$p = \int_{Lower}^{Upper} \frac{\Gamma\left(\frac{ndf+ddf}{2}\right)}{\Gamma\left(\frac{ndf}{2}\right) \times \Gamma\left(\frac{ddf}{2}\right)} \times \left(\frac{ndf}{ddf}\right)^{\frac{ndf}{2}} \times x^{\left(\frac{ndf}{2}-1\right)} \times \left(1+\frac{ndf \times x}{ddf}\right)^{-\frac{ndf+ddf}{2}} dx$$

- L'exécution de DrawDistF effectue le calcul ci-dessus selon les conditions spécifiées et trace le graphe. La région Lower ≦ x ≦ Upper du graphe est remplie à la même occasion.
- Simultanément, la valeur du résultat du calcul de p est affectée aux variables p et Ans.

■ Exécution de calculs statistiques dans un programme

- Calcul statistique à variable unique

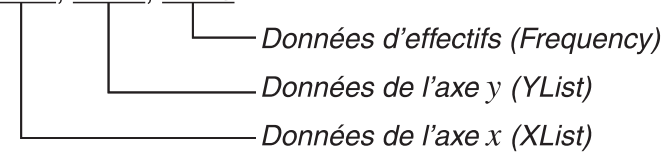
① 1-Variable List1, List 2



① F4 F1 F6 F1

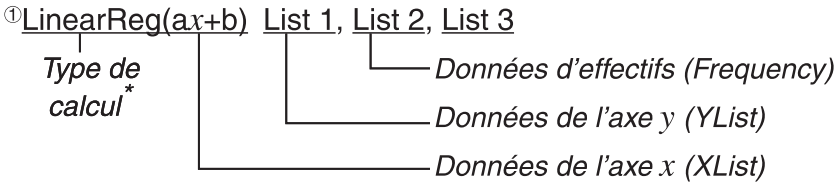
- Calcul statistique à variable double

① 2-Variable List 1, List 2, List 3



① F4 F1 F6 F2

• Calcul statistique de régression

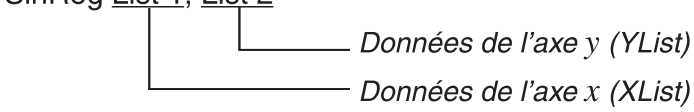


① F4 F1 F6 F6 F1 F1

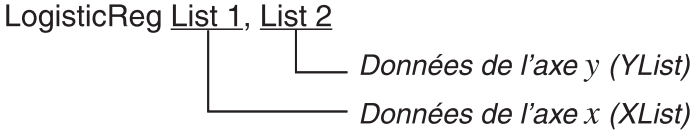
* Vous pouvez définir comme type de calcul les paramètres suivants.

LinearReg(ax+b).....régression linéaire (type $ax+b$)
LinearReg(a+bx).....régression linéaire (type $a+bx$)
Med-MedLine .....calcul Med-Med
QuadReg .....régression quadratique
CubicReg.....régression cubique
QuartReg .....régression quartique
LogReg .....régression logarithmique
ExpReg(a·e^bx).....régression exponentielle (type $a \cdot e^{bx}$)
ExpReg(a·b^x).....régression exponentielle (type $a \cdot b^x$)
PowerReg .....régression de puissance

• Calcul statistique de régression sinusoïdale



• Calcul statistique de régression logistique



■ Réalisation de calculs de distributions dans un programme

• Les valeurs suivantes sont substituées chaque fois que toute valeur délimitée par des crochets ([ ]) est omise.

σ=1, μ=0, tail=L (gauche)

• Pour obtenir de l'information sur la formule de calcul de chaque fonction de densité de probabilité, voir « Formule statistique » (page 6-55).

• Distribution normale

NormPD(: Retourne la densité de probabilité normale (valeur p) pour les données spécifiées.

Syntaxe : NormPD(x[, σ, μ)]

• L'argument x peut être spécifié par une valeur seule ou par une liste. Le résultat p du calcul est affecté aux variables p et Ans (ou à ListAns lorsque x correspond à une liste).

NormCD(: Retourne la distribution normale (valeur p) cumulative pour les données spécifiées.

Syntaxe : NormCD(Lower, Upper[, σ, μ)]

- Les arguments Lower et Upper peuvent être spécifiés par des valeurs seules ou par des listes. Les résultats du calcul de p, ZLow et ZUp sont affectés aux variables p, ZLow et ZUp respectivement. Le résultat p du calcul est affecté également à la variable Ans (ou à ListAns lorsque Lower et Upper sont des listes).

InvNormCD(: Retourne la distribution normale cumulative inverse (valeur(s) inférieur(es) et/ou supérieur(es)) pour la valeur p spécifiée.

Syntaxe : InvNormCD(["L(ou –1) ou R(ou 1) ou C(ou 0)",]p[,σ, μ])
extrémité (Left, Right, Central)

- L'argument p peut être spécifié par une valeur seule ou par une liste. Les résultats du calcul sont sortis selon le paramètre de queue spécifié, comme décrit ci-dessous :

  extrémité = Left

  La valeur Upper est affectée aux variables x1InvN et Ans (ou ListAns lorsque p correspond à une liste).

  extrémité = Right

  La valeur Lower est affectée aux variables x1InvN et Ans (ou ListAns lorsque p correspond à une liste).

  extrémité = Central

  Les valeurs Lower et Upper sont affectées aux variables x1InvN et x2InvN respectivement. Seulement Inférieur est affectée à Ans (ou à ListAns lorsque p correspond à une liste).

• Distribution *t* de Student

tPD(: Retourne la densité de probabilité t de Student (valeur p) pour les données spécifiées.

Syntaxe : tPD(x, df [])]

- L'argument x peut être spécifié par une valeur seule ou par une liste. Le résultat p du calcul est affecté aux variables p et Ans (ou à ListAns lorsque x correspond à une liste).

tCD(: Retourne la distribution t de Student (valeur-p) cumulative pour les données spécifiées.

Syntaxe : tCD(Lower,Upper,df [])]

- Les arguments Lower et Upper peuvent être spécifiés par des valeurs seules ou par des listes. Les résultats du calcul de p, tLow et tUp sont affectés aux variables p, tLow et tUp respectivement. Le résultat p du calcul est affecté également à la variable Ans (ou à ListAns lorsque Lower et Upper sont des listes).

InvTCD(: Retourne la distribution t de Student cumulative inverse (valeur Lower) pour le valeur p spécifiées.

Syntaxe : InvTCD(p,df [])]

- L'argument p peut être spécifié par une valeur seule ou par une liste. La valeur de Lower est affecté aux variables xInv et Ans (ou à ListAns lorsque p correspond à une liste).

• Distribution χ^2

ChiPD(: Retourne la densité de probabilité χ^2 (valeur p) pour les données spécifiées.

Syntaxe : ChiPD(x,df [])]

- L'argument x peut être spécifié par une valeur seule ou par une liste. Le résultat p du calcul est affecté aux variables p et Ans (ou à ListAns lorsque x correspond à une liste).

ChiCD(: Retourne la distribution χ^2 (valeur p) cumulative pour les données spécifiées.

Syntaxe : ChiCD(Lower,Upper,*df* [])]

- Les arguments Lower et Upper peuvent être spécifiés par des valeurs seules ou par des listes. Le résultat p du calcul est affecté aux variable p et Ans (ou à ListAns lorsque Lower et Upper sont des listes).

InvChiCD(: Retourne la distribution χ^2 cumulative inverse (valeur Lower) pour la valeur p spécifiées.

Syntaxe : InvChiCD(p,*df* [])]

- L'argument p peut être spécifié par une valeur seule ou par une liste. La valeur Lower est affecté aux variables xInv et Ans (ou à ListAns lorsque p correspond à une liste).

• Distribution F

FPD(: Retourne la densité de probabilité F (valeur p) pour les données spécifiées.

Syntaxe : FPD(x,*ndf*,*ddf* [])]

- L'argument x peut être spécifié par une valeur seule ou par une liste. Le résultat p du calcul est affecté aux variables p et Ans (ou à ListAns lorsque x correspond à une liste).

FCD(: Retourne la distribution F cumulative (valeur p) pour les données spécifiées.

Syntaxe : FCD(Lower,Upper,*ndf*,*ddf* [])]

- Les arguments Lower et Upper peuvent être spécifiés par des valeurs seules ou par des listes. Le résultat p du calcul est affecté aux variables p et Ans (ou à ListAns lorsque Lower et Upper sont des listes).

InvFCD(: Retourne la distribution F cumulative inverse (valeur Lower) pour les données spécifiées.

Syntaxe : InvFCD(p,*ndf*,*ddf* [])]

- L'argument p peut être spécifié par une valeur seule ou par une liste. La valeur Lower est affecté aux variables xInv et Ans (ou à ListAns lorsque p correspond à une liste).

• Distribution binomiale

BinomialPD(: Retourne la probabilité binomiale (valeur p) pour les données spécifiées.

Syntaxe : BinomialPD([x,]n,P[])]

- L'argument x peut être spécifié par une valeur seule ou par une liste. Le résultat p du calcul est affecté aux variables p et Ans (ou à ListAns lorsque x correspond à une liste).

BinomialCD(: Retourne la distribution binomiale cumulative (valeur p) pour les données spécifiées.

Syntaxe : BinomialCD([X,]n,P[])]

- Chaque argument X peut être spécifié par une valeur seule ou par une liste. Le résultat p du calcul est affecté aux variables p et Ans (ou à ListAns lorsque X est omis ou correspond à une liste).

InvBinomialCD(: Retourne la distribution binomiale cumulative inverse pour les données spécifiées.

Syntaxe : InvBinomialCD(p,n,P[])

- L'argument p peut être spécifié par une valeur seule ou par une liste. La valeur X du résultat du calcul est affectée aux variables xInv et Ans (ou à ListAns lorsque p correspond à une liste).

• Distribution de Poisson

PoissonPD(: Retourne la probabilité de Poisson (valeur p) pour les données spécifiées.

Syntaxe : PoissonPD(x, μ[])

- L'argument x peut être spécifié par une valeur seule ou par une liste. Le résultat p du calcul est affecté aux variables p et Ans (ou à ListAns lorsque x correspond à une liste).

PoissonCD(: Retourne la distribution de Poisson cumulative (valeur p) pour les données spécifiées.

Syntaxe : PoissonCD(X,μ[])

- Chaque argument X peut être spécifié par une valeur seule ou par une liste. Le résultat p du calcul est affecté aux variables p et Ans (ou à ListAns lorsque X correspond à une liste).

InvPoissonCD(: Retourne la distribution de Poisson cumulative inverse pour les données spécifiées.

Syntaxe : InvPoissonCD(p,μ[])

- L'argument p peut être spécifié par une valeur seule ou par une liste. La valeur X du résultat du calcul est affectée aux variables xInv et Ans (ou à ListAns lorsque p correspond à une liste).

• Distribution géométrique

GeoPD(: Retourne la probabilité géométrique (valeur p) pour les données spécifiées.

Syntaxe : GeoPD(x, P[])

- L'argument x peut être spécifié par une valeur seule ou par une liste. Le résultat p du calcul est affecté aux variables p et Ans (ou à ListAns lorsque x correspond à une liste).

GeoCD(: Retourne la distribution géométrique cumulative (valeur p) pour les données spécifiées.

Syntaxe : GeoCD(X,P[])

- Chaque argument X peut être spécifié par une valeur seule ou par une liste. Le résultat p du calcul est affecté aux variables p et Ans (ou à ListAns lorsque X correspond à une liste).

InvGeoCD(: Retourne la distribution géométrique cumulative inverse pour les données spécifiées.

Syntaxe : InvGeoCD(p,P[])

- L'argument p peut être spécifié par une valeur seule ou par une liste. La valeur X du résultat du calcul est affectée aux variables xInv et Ans (ou à ListAns lorsque p correspond à une liste).

• Distribution hypergéométrique

HypergeoPD(: Retourne la probabilité hypergéométrique (valeur p) pour les données spécifiées.

Syntaxe : HypergeoPD(x, n, M, N[])]

- L'argument x peut être spécifié par une valeur seule ou par une liste. Le résultat p du calcul est affecté aux variables p et Ans (ou à ListAns lorsque x correspond à une liste).

HypergeoCD(: Retourne la distribution hypergéométrique cumulative (valeur p) pour les données spécifiées.

Syntaxe : HypergeoCD(X, n, M, N[])]

- Chaque argument X peut être spécifié par une valeur seule ou par une liste. Le résultat p du calcul est affecté aux variables p et Ans (ou à ListAns lorsque X correspond à une liste).

InvHypergeoCD(: Retourne la distribution hypergéométrique cumulative inverse pour les données spécifiées.

Syntaxe : InvHypergeoCD(p, n, M, N[])]

- L'argument p peut être spécifié par une valeur seule ou par une liste. La valeur X du résultat du calcul est affectée aux variables xInv et Ans (ou à ListAns lorsque p correspond à une liste).

■ Utilisation de la commande TEST pour exécuter une commande dans un programme

- Les plages de spécification de l'argument « condition μ » de la commande sont les suivantes :

  « < » ou –1 lorsque $\mu < \mu_0$

  « ≠ » ou 0 lorsque $\mu \neq \mu_0$

  « > » ou 1 lorsque $\mu > \mu_0$

  Ce qui précède s'applique aussi aux méthodes de spécification de « condition ρ » et « condition β&ρ ».
- Pour obtenir des explications sur les arguments qui ne sont pas traités ici en détails, voir « Tests » (page 6-23) et « Termes des tests d'entrée et sortie, intervalle de confiance et loi de probabilité » (page 6-52).
- Pour obtenir de l'information sur la formule de calcul de chaque commande, voir « Formule statistique » (page 6-55).

• Test Z

OneSample$Z$Test : Exécute le calcul de test Z à 1 échantillon.

Syntaxe : OneSample$Z$Test "condition μ", μ_0, σ, $\bar{x}$, n

Valeurs en sortie : Z, p, $\bar{x}$, n sont affectés aux variables z, p, $\bar{x}$, n respectivement et aux éléments 1 à 4 de la liste ListAns.

Syntaxe : OneSample$Z$Test "condition μ_1", μ_0, σ, List[, Freq]

Valeurs en sortie : Z, p, $\bar{x}$, s_x, n sont affectés aux variables z, p, $\bar{x}$, s_x, n respectivement et aux éléments 1 à 5 de la liste ListAns.

TwoSample$Z$Test : Exécute le calcul de test Z à 2 échantillons.

Syntaxe : TwoSample$Z$Test "condition μ_1", σ_1, σ_2, $\bar{x}_1$, n_1, $\bar{x}_2$, n_2

Valeurs en sortie : Z, p, $\bar{x}_1$, $\bar{x}_2$, n_1, n_2 sont affectés aux variables z, p, $\bar{x}_1$, $\bar{x}_2$, n_1, n_2 respectivement et aux éléments 1 à 6 de la liste ListAns.

Syntaxe : TwoSample$Z$Test "condition μ_1", σ_1, σ_2, List1, List2[, Freq1 [, Freq2]]

Valeurs en sortie : Z, p, $\bar{x}_1$, $\bar{x}_2$, s_{x1}, s_{x2}, n_1, n_2 sont affectés aux variables z, p, $\bar{x}_1$, $\bar{x}_2$, s_{x1}, s_{x2}, n_1, n_2 respectivement et aux éléments 1 à 8 de la liste ListAns.

OneProp$Z$Test : Exécute le calcul de test Z à 1 proportion.

Syntaxe : OneProp$Z$Test "condition p", p_0, x, n

Valeurs en sortie : Z, p, $\hat{p}$, n sont affectés aux variables z, p, $\hat{p}$, n respectivement et aux éléments 1 à 4 de la liste ListAns.

TwoProp$Z$Test : Exécute le calcul de test Z à 2 proportions.

Syntaxe : TwoProp$Z$Test "condition p_1 ", x_1, n_1, x_2, n_2

Valeurs en sortie : Z, p, $\hat{p}_1$, $\hat{p}_2$, $\hat{p}$, n_1, n_2 sont affectés aux variables z, p, $\hat{p}_1$, $\hat{p}_2$, $\hat{p}$, n_1, n_2 respectivement et aux éléments 1 à 7 de la liste ListAns.

• Test t

OneSampleTTest : Exécute le calcul de test t à 1 échantillon.

Syntaxe : OneSampleTTest "condition μ", μ_0, $\bar{x}$, s_x, n
OneSampleTTest "condition μ", μ_0, List[, Freq]

Valeurs en sortie : t, p, $\bar{x}$, s_x, n sont affectés aux variables du même nom et aux éléments 1 à 5 de la liste ListAns.

TwoSampleTTest : Exécute le calcul de test t à 2 échantillons.

Syntaxe : TwoSampleTTest "condition μ_1", $\bar{x}_1$, s_{x1}, n_1, $\bar{x}_2$, s_{x2}, n_2[,condition Pooled]
TwoSampleTTest "condition μ_1", List1, List2, [, Freq1[, Freq2[, condition Pooled]]]

Valeurs en sortie : Quand condition Pooled = 0, alors t, p, df, $\bar{x}_1$ $\bar{x}_2$, s_{x1}, s_{x2}, n_1, n_2 sont affectés aux variables du même nom respectivement et au éléments 1 à 9 de la liste ListAns.
Quand condition Pooled = 1, alors t, p, df, $\bar{x}_1$, $\bar{x}_2$, s_{x1}, s_{x2}, s_p, n_1, n_2 sont affectés aux variables du même nom respectivement et au éléments 1 à 10 de la liste ListAns.

Note : Si vous voulez désactiver la condition d'analyse groupée (condition Pooled) spécifiez la valeur 0. Spécifiez 1 pour activer cette condition. L'omission de cette entrée est traitée comme condition Pooled désactivée.

LinRegTTest : Exécute le calcul de test t à régression linéaire.

Syntaxe : LinRegTTest "condition β&ρ", XList, YList[, Freq]

Valeurs en sortie : t, p, df, a, b, s, r, r^2 sont affectés aux variables du même nom et aux éléments 1 à 8 de la liste ListAns.

• Test χ^2

ChiGOFTest : Exécute un test chi carré de précision de l'ajustement (GOF « goodness-of-fit »).

Syntaxe : ChiGOFTest List1, List2, df, List3
(List 1 est la liste Observed, List 2 est la liste Expected et List 3 est la liste CNTRB.)

Valeurs en sortie : χ^2, p, df sont affectés aux variables du même nom et aux éléments 1 à 3 de la liste ListAns. La liste CNTRB est stockée dans List 3.

ChiTest : Exécute un test chi carré.

Syntaxe : ChiTest MatA, MatB
(MatA est la matrice Observed et MatB est la matrice Expected.)

Valeurs en sortie : χ^2, p, df sont affectés aux variables du même nom et aux éléments 1 à 3 de la liste ListAns. La matrice Expected est affecté à MatB.

• Test F

TwoSample$F$Test : Exécute le calcul de test F à 2 échantillons.

Syntaxe : TwoSample$F$Test "condition σ_1", s_{x1}, n_1, s_{x2}, n_2

Valeurs en sortie : F, p, s_{x1}, s_{x2}, n_1, n_2 sont affectés aux variables du même nom et aux éléments 1 à 6 de la liste ListAns.

Syntaxe : TwoSample$F$Test "condition σ_1", List1, List2, [, Freq1 [, Freq2]]

Valeurs en sortie : F, p, $\bar{x}_1$, $\bar{x}_2$, s_{x1}, s_{x2}, n_1, n_2 sont affectés aux variables du même nom et aux éléments 1 à 8 de la liste ListAns.

• ANOVA

OneWayANOVA : Exécute l'analyse de variance ANOVA à un seul facteur.

Syntaxe : OneWayANOVA List1, List2
(List1 est la liste Factor (A) et List2 est la liste Dependent.)

Valeurs en sortie : Adf, Ass, Ams, AF, Ap, ERRdf, ERRss, ERRms sont affectés aux variables Adf, SSa, MSa, Fa, pa, Edf, SSe, MSe respectivement.

Ces valeurs de sortie sont affectées aussi aux éléments de la matrice MatAns, comme indiqué ci-dessous :

$$\text{MatAns} = \begin{bmatrix} Adf & Ass & Ams & AF & Ap \\ ERRdf & ERRss & ERRms & 0 & 0 \end{bmatrix}$$

TwoWayANOVA : Exécute l'analyse de variance ANOVA à deux facteurs.

Syntaxe : TwoWayANOVA List1, List2, List3
(List1 est la liste de Factor (A), List2 est la liste Factor (B) et List3 est la liste Dependent.)

Valeurs en sortie : Adf, Ass, Ams, AF, Ap, Bdf, Bss, Bms, BF, Bp, ABdf, ABss, ABms, ABF, ABp, ERRdf, ERRss, ERRms sont affectés aux variables Adf, SSa, MSa, Fa, pa, Bdf, SSb, MSb, Fb, pb, ABdf, SSab, MSab, Fab, pab, Edf, SSe, MSe respectivement.

Ces valeurs de sortie sont affectées aussi aux éléments de la matrice MatAns, comme indiqué ci-dessous :

$$\text{MatAns} = \begin{bmatrix} Adf & Ass & Ams & AF & Ap \\ Bdf & Bss & Bms & BF & Bp \\ ABdf & ABss & ABms & ABF & ABp \\ ERRdf & ERRss & ERRms & 0 & 0 \end{bmatrix}$$

■ Réalisation de calculs financiers dans un programme

• Commandes de configuration

• Configuration du paramètre « Date Mode » pour les calculs financiers

DateMode365..... 365 jours

DateMode360..... 360 jours

• Configuration du paramètre de la période d'échéance

PmtBgn..... Début de la période

PmtEnd..... Fin de la période

• Période d'échéance des calculs d'obligations

PeriodsAnnual..... Annuel

PeriodsSemi..... Semestriel

• Commandes des calculs financiers

Pour obtenir de l'information sur la signification de chaque commande, voir « Chapitre 7 Calculs financiers (TVM) ».

• Intérêt simple

Smpl_SI : Retourne l'intérêt à partir d'un calcul d'intérêt simple.

Syntaxe : Smpl_SI(n, I%, PV)

Smpl_SFV : Retourne le total du capital et de l'intérêt à partir d'un calcul d'intérêt simple.

Syntaxe : Smpl_SFV(n, I%, PV)

• Intérêt composé

Note :

• P/Y et C/Y peuvent être omis pour tout calcul d'intérêt composé.
Dans ce cas, les calculs sont effectués par défaut avec P/Y=12 et C/Y=12.

• Si vous effectuez un calcul qu'utilise une fonction d'intérêt composé (Cmpd_n(, Cmpd_I%(, Cmpd_PV(, Cmpd_PMT(, Cmpd_FV(), les arguments saisis et les résultats du calcul seront sauvegardés dans les variables pertinentes (n, I%, PV, etc.). Si vous effectuez un calcul qu'utilise d'autres types de fonctions de calcul financier, l'argument et les résultats du calcul ne sont pas affectés à des variables.

Cmpd_n : Retourne le nombre d'échéances à intérêt composé.

Syntaxe : Cmpd_n(I%, PV, PMT, FV, P/Y, C/Y)

Cmpd_I% : Retourne le taux d'intérêt annuel.

Syntaxe : Cmpd_I%(n, PV, PMT, FV, P/Y, C/Y)

Cmpd_PV : Retourne la valeur actuelle (montant dû d'un prêt à la consommation, capital d'un plan d'épargne).

Syntaxe : Cmpd_PV(n, I%, PMT, FV, P/Y, C/Y)

Cmpd_PMT : Retourne des valeurs d'entrée/sortie égales (montant des échéances d'un prêt à la consommation, montant des dépôts d'une épargne) pour une période fixe.

Syntaxe : Cmpd_PMT(n, I%, PV, FV, P/Y, C/Y)

Cmpd_FV : Retourne le montant d'entrée/sortie final ou le capital total et l'intérêt total.

Syntaxe : Cmpd_FV(n, I%, PV, PMT, P/Y, C/Y)

• Flux de trésorerie (évaluation des investissements)

Cash_NPV : Retourne la valeur actualisée nette.

Syntaxe : Cash_NPV(I%, Csh)

Cash_IRR : Retourne le taux de rendement interne.

Syntaxe : Cash_IRR(Csh)

Cash_PBP : Retourne le délai de récupération.

Syntaxe : Cash_PBP(I%, Csh)

Cash_NFV : Retourne la valeur capitalisée nette.

Syntaxe : Cash_NFV(I%, Csh)

• Amortissement

Amt_BAL : Retourne le montant principal restant suite à l'échéance PM2.

Syntaxe : Amt_BAL(PM1, PM2, I%, PV, PMT, P/Y, C/Y)

Amt_INT : Retourne l'intérêt payé pour l'échéance PM1.

Syntaxe : Amt_INT(PM1, PM2, I%, PV, PMT, P/Y, C/Y)

Amt_PRN : Retourne le principal et l'intérêt payés pour l'échéance PM1.

Syntaxe : Amt_PRN(PM1, PM2, I%, PV, PMT, P/Y, C/Y)

Amt_ΣINT : Retourne le principal total et l'intérêt total payés entre les échéances PM1 et PM2.

Syntaxe : Amt_ΣINT(PM1, PM2, I%, PV, PMT, P/Y, C/Y)

Amt_ΣPRN : Retourne le principal total payé entre les échéances PM1 et PM2.

Syntaxe : Amt_ΣPRN(PM1, PM2, I%, PV, PMT, P/Y, C/Y)

• Conversion de taux d'intérêt

Cnvt_EFF : Retourne le taux d'intérêt converti à partir du taux d'intérêt nominal vers le taux d'intérêt effectif.

Syntaxe : Cnvt_EFF(n, I%)

Cnvt_APR : Retourne le taux d'intérêt converti à partir du taux d'intérêt effectif vers le taux d'intérêt nominal.

Syntaxe : Cnvt_APR(n, I%)

• Calculs de coût, prix de vente, marge bénéficiaire

Cost : Retourne le coût à partir d'un prix de vente et d'une marge bénéficiaire spécifiés.

Syntaxe : Cost(Sell, Margin)

Sell : Retourne le prix de vente à partir d'un coût et d'une marge bénéficiaire spécifiés.

Syntaxe : Sell(Cost, Margin)

Marge : Retourne la marge bénéficiaire à partir d'un coût et d'un prix de vente spécifiés.

Syntaxe : Margin(Cost, Sell)

• Calculs de jours/date

Days_Prd : Retourne le nombre de jours entre deux dates d1 et d2 spécifiées.

Syntaxe : Days_Prd(MM1, DD1, YYYY1, MM2, DD2, YYYY2)

• Calculs d'obligations

Bond_PRC : Retourne les prix des obligations sous des conditions spécifiées en forme de liste.

Syntaxe : Bond_PRC(MM1, DD1, YYYY1, MM2, DD2, YYYY2, RDV, CPN, YLD) = {PRC, INT, CST}

Bond_YLD : Retourne le rendement sous des conditions spécifiées.

Syntaxe : Bond_YLD(MM1, DD1, YYYY1, MM2, DD2, YYYY2, RDV, CPN, PRC)

7. Liste des commandes du mode PRGM

Les commandes listées ci-dessous ne sont pas toutes disponibles sur tous les modèles couverts par ce manuel.

Programme RUN

Touche F4 (MENU)			
Niveau 1	Niveau 2	Niveau 3	Commande
STAT	DRAW	On	DrawOn
		Off	DrawOff
	GRPH	GPH1	S-Gph1_
		GPH2	S-Gph2_
		GPH3	S-Gph3_
		Scat	Scatter
		xy	xyLine
		Hist	Hist
		Box	MedBox
		Bar	Bar
		N-Dis	N-Dist
		Brkn	Broken
		X	Linear
		Med	Med-Med
		X^2	Quad
		X^3	Cubic
		X^4	Quart
		Log	Log
		*1	
		Pwr	Power
		Sin	Sinusoidal
		NPP	NPPlot
		Lgst	Logistic
		Pie	Pie
	List		List_
	TYPE	*2	
	DIST	DrwN	DrawDistNorm_
		Drwt	DrawDistT_
		DrwC	DrawDistChi_
		DrwF	DrawDistF_
	CALC	1VAR	1-Variable_
		2VAR	2-Variable_
		*3	
		Med	Med-MedLine_
		X^2	QuadReg_
		X^3	CubicReg_
		X^4	QuartReg_
		Log	LogReg_
		*4	
		Pwr	PowerReg_
		Sin	SinReg_
		Lgst	LogisticReg_
MAT	Swap		Swap_
	×Rw		∗Row_
	×Rw+		∗Row+_
	Rw+		Row+_
LIST	Srt-A		SortA(
	Srt-D		SortD(
GRPH	SEL	On	G_SelOn_
		Off	G_SelOff_
	TYPE	Y=	Y=Type
		r=	r=Type
		Parm	ParamType
		X=	X=Type
		Y>	Y>Type
		Y<	Y<Type
		Y≥	Y≥Type
		Y≤	Y≤Type
		X>	X>Type
		X<	X<Type
		X≥	X≥Type
		X≤	X≤Type
	STYL	—	NormalG_
		—	ThickG_
		·····	BrokenThickG_
		·····	DotG_
	GMEM	Sto	StoGMEM_
		Rcl	RclGMEM_
DYNA	On		D_SelOn_
	Off		D_SelOff_
	Var		D_Var_
	TYPE	Y=	Y=Type
		r=	r=Type
		Parm	ParamType
TABL	On		T_SelOn_
	Off		T_SelOff_
	TYPE	Y=	Y=Type
		r=	r=Type
		Parm	ParamType
	STYL	—	NormalG_
		—	ThickG_
		·····	BrokenThickG_
		·····	DotG_
RECR	SEL+S	On	R_SelOn_
		Off	R_SelOff_
		—	NormalG_
		—	ThickG_
		·····	BrokenThickG_
		·····	DotG_
	TYPE	a_n	a_nType
		a_{n+1}	a_{n+1}Type
		a_{n+2}	a_{n+2}Type
	$n.a_{n\cdot\cdot}$	n	n
		a_n	a_n

		a_{n+1}	**a_{n+1}**
		a_{n+2}	**a_{n+2}**
		b_n	**b_n**
		b_{n+1}	**b_{n+1}**
		b_{n+2}	**b_{n+2}**
		c_n	**c_n**
		c_{n+1}	**c_{n+1}**
		c_{n+2}	**c_{n+2}**
		Σa_n	**Σa_n**
		Σa_{n+1}	**Σa_{n+1}**
		Σa_{n+2}	**Σa_{n+2}**
		Σb_n	**Σb_n**
		Σb_{n+1}	**Σb_{n+1}**
		Σb_{n+2}	**Σb_{n+2}**
		Σc_n	**Σc_n**
		Σc_{n+1}	**Σc_{n+1}**
		Σc_{n+2}	**Σc_{n+2}**
	RANG	a_0	**Sel_a_0**
		a_1	**Sel_a_1**

Touche OPTN

Niveau 1	Niveau 2	Niveau 3	Commande
LIST	List		**List_**
	L→M		**List→Mat(**
	Dim		**Dim_**
	Fill		**Fill(**
	Seq		**Seq(**
	Min		**Min(**
	Max		**Max(**
	Mean		**Mean(**
	Med		**Median(**
	Aug		**Augment(**
	Sum		**Sum_**
	Prod		**Prod_**
	Cuml		**Cuml_**
	%		**Percent_**
	⊿		**⊿List_**
MAT	Mat		**Mat_**
	M→L		**Mat→List(**
	Det		**Det_**
	Trn		**Trn_**
	Aug		**Augment(**
	Iden		**Identity_**
	Dim		**Dim_**
	Fill		**Fill(**
	Ref		**Ref_**
	Rref		**Rref_**
CPLX	i		**i**
	Abs		**Abs_**
	Arg		**Arg_**
	Conj		**Conjg_**
	ReP		**ReP_**
	ImP		**ImP_**
	▶r∠θ		**▶r∠θ**
	▶a+bi		**▶a+bi**
CALC	Solve		**Solve(**
	d/dx		**d/dx(**
	d^2/dx^2		**d^2/dx^2(**
	$\int dx$		**$\int$(**
	SolveN		**SolveN(**
	FMin		**FMin(**
	FMax		**FMax(**
	Σ(		**Σ(**
	$\log_a b$		**$\log_a b$(**
	Int÷		**Int÷**
	Rmdr		**Rmdr**
	Simp		**▶Simp**
STAT	$\hat{x}$		**$\hat{x}$**
	$\hat{y}$		**$\hat{y}$**
	DIST	*5	
	S·Dev		**StdDev(**
	Var		**Variance(**
	TEST	*6	
CONV	▶		**▶**
	LENG	fm	**[fm]**
		Å	**[Å]**
		μm	**[μm]**
		mm	**[mm]**
		cm	**[cm]**
		m	**[m]**
		km	**[km]**
		AU	**[AU]**
		l.y.	**[l.y.]**
		pc	**[pc]**
		Mil	**[Mil]**
		in	**[in]**
		ft	**[ft]**
		yd	**[yd]**
		fath	**[fath]**
		rd	**[rd]**
		mile	**[mile]**
		n mile	**[n mile]**
	AREA	cm^2	**[cm^2]**
		m^2	**[m^2]**
		ha	**[ha]**
		km^2	**[km^2]**
		in^2	**[in^2]**
		ft^2	**[ft^2]**
		yd^2	**[yd^2]**
		acre	**[acre]**
		$mile^2$	**[$mile^2$]**
	VLUM	cm^3	**[cm^3]**
		mL	**[mL]**
		L	**[L]**
		m^3	**[m^3]**
		in^3	**[in^3]**
		ft^3	**[ft^3]**
		fl_oz(UK)	**[fl_oz(UK)]**
		fl_oz(US)	**[fl_oz(US)]**
		gal(US)	**[gal(US)]**
		gal(UK)	**[gal(UK)]**
		pt	**[pt]**
		qt	**[qt]**
		tsp	**[tsp]**
		tbsp	**[tbsp]**
		cup	**[cup]**
	TIME	ns	**[ns]**
		μs	**[μs]**
		ms	**[ms]**
		s	**[s]**
		min	**[min]**
		h	**[h]**
		day	**[day]**
		week	**[week]**
		yr	**[yr]**
		s-yr	**[s-yr]**
		t-yr	**[t-yr]**
	TMPR	°C	**[°C]**
		K	**[K]**
		°F	**[°F]**
		°R	**[°R]**
	VELO	m/s	**[m/s]**
		km/h	**[km/h]**
		knot	**[knot]**
		ft/s	**[ft/s]**
		mile/h	**[mile/h]**
	MASS	u	**[u]**
		mg	**[mg]**
		g	**[g]**
		kg	**[kg]**
		mton	**[mton]**
		oz	**[oz]**
		lb	**[lb]**
		slug	**[slug]**
		ton(short)	**[ton(short)]**
		ton(long)	**[ton(long)]**
	RORC	N	**[N]**
		lbf	**[lbf]**
		tonf	**[tonf]**
		dyne	**[dyne]**
		kgf	**[kgf]**
	PRES	Pa	**[Pa]**
		kPa	**[kPa]**
		mmH_2O	**[mmH_2O]**
		mmHg	**[mmHg]**
		atm	**[atm]**
		inH_2O	**[inH_2O]**
		inHg	**[inHg]**
		lbf/in^2	**[lbf/in^2]**
		bar	**[bar]**
		kgf/cm^2	**[kgf/cm^2]**
	ENGY	eV	**[eV]**
		J	**[J]**
		cal_{th}	**[cal_{th}]**
		cal_{15}	**[cal_{15}]**
		cal_{IT}	**[cal_{IT}]**
		$kcal_{th}$	**[$kcal_{th}$]**
		$kcal_{15}$	**[$kcal_{15}$]**
		$kcal_{IT}$	**[$kcal_{IT}$]**
		l-atm	**[l-atm]**
		kW•h	**[kW•h]**
		ft•lbf	**[ft•lbf]**
		Btu	**[Btu]**
		erg	**[erg]**
		kgf•m	**[kgf•m]**
	PWR	W	**[W]**
		cal_{th}/s	**[cal_{th}/s]**
		hp	**[hp]**
		ft•lbf/s	**[ft•lbf/s]**
		Btu/min	**[Btu/min]**

HYP	sinh		**sinh_**
	cosh		**cosh_**
	tanh		**tanh_**
	$\sinh^{-1}$		**$\sinh^{-1}$_**
	$\cosh^{-1}$		**$\cosh^{-1}$_**
	$\tanh^{-1}$		**$\tanh^{-1}$_**
PROB	X!		**!**
	nPr		**P**
	nCr		**C**
	RAND	Ran#	**Ran#_**
		Int	**RanInt#(**
		Norm	**RanNorm#(**
		Bin	**RanBin#(**
		List	**RanList#(**
	P(		**P(**
	Q(		**Q(**
	R(		**R(**
	t(		**t(**
NUM	Abs		**Abs_**
	Int		**Int_**
	Frac		**Frac_**
	Rnd		**Rnd**
	Intg		**Intg_**
	RndFi		**RndFix(**
	GCD		**GCD(**
	LCM		**LCM(**
	MOD		**MOD(**
	MOD•E		**MOD_Exp(**
ANGL	°		**°**
	r		**r**
	g		**g**
	° ’ ”		**¤**
	Pol(		**Pol(**
	Rec(		**Rec(**
	▶DMS		**▶DMS**
ESYM	m		**m**
	μ		**μ**
	n		**n**
	p		**p**
	f		**f**
	k		**k**
	M		**M**
	G		**G**
	T		**T**
	P		**P**
	E		**E**
PICT	Sto		**StoPict_**
	Rcl		**RclPict_**
FMEM	fn		**fn**
LOGIC	And		**_And_**
	Or		**_Or_**
	Not		**Not_**
	Xor		**Xor_**
CAPT	Rcl		**RclCapt_**
TVM	SMPL	SI	**Smpl_SI(**
		SFV	**Smpl_SFV(**
	CMPD	n	**Cmpd_n(**
		I%	**Cmpd_I%(**
		PV	**Cmpd_PV(**
		PMT	**Cmpd_PMT(**
		FV	**Cmpd_FV(**
	CASH	NPV	**Cash_NPV(**
		IRR	**Cash_IRR(**
		PBP	**Cash_PBP(**
		NFV	**Cash_NFV(**
	AMT	BAL	**Amt_BAL(**
		INT	**Amt_INT(**
		PRN	**Amt_PRN(**
		ΣINT	**Amt_ΣINT(**
		ΣPRN	**Amt_ΣPRN(**
	CNVT	EFF	**Cnvt_EFF(**
		APR	**Cnvt_APR(**
	COST	Cost	**Cost(**
		Sell	**Sell(**
		Mrg	**Margin(**
	DAYS	PRD	**Days_Prd(**
	BOND	PRC	**Bond_PRC(**
		YLD	**Bond_YLD(**

Touche VARS

Niveau 1	Niveau 2	Niveau 3	Commande
V-WIN	X	min	**Xmin**
		max	**Xmax**
		scal	**Xscl**
		dot	**Xdot**
	Y	min	**Ymin**
		max	**Ymax**
		scal	**Yscl**
	T,θ	min	**Tθmin**
		max	**Tθmax**
		ptch	**Tθptch**
	R-X	min	**RightXmin**
		max	**RightXmax**
		scal	**RightXscl**
		dot	**RightXdot**
	R-Y	min	**RightYmin**
		max	**RightYmax**
		scal	**RightYscl**
	R-T, θ	min	**RightTθmin**
		max	**RightTθmax**
		ptch	**RightTθptch**
FACT	Xfct		**Xfct**
	Yfct		**Yfct**
STAT	X	n	**n**
		$\bar{x}$	**$\bar{x}$**
		Σx	**Σx**
		Σx^2	**Σx^2**
		σx	**σx**
		sx	**sx**
		minX	**minX**
		maxX	**maxX**
	Y	$\bar{y}$	**$\bar{y}$**
		Σy	**Σy**
		Σy^2	**Σy^2**
		Σxy	**Σxy**
		σy	**σy**
		sy	**sy**
		minY	**minY**
		maxY	**maxY**
	GRPH	a	**a**
		b	**b**
		c	**c**
		d	**d**
		e	**e**
		r	**r**
		r^2	**r^2**
		MSe	**MSe**
		Q_1	**Q_1**
		Med	**Med**
		Q_3	**Q_3**
		Mod	**Mod**
		Strt	**H_Start**
		Pitch	**H_pitch**
	PTS	x_1	**x_1**
		y_1	**y_1**
		x_2	**x_2**
		y_2	**y_2**
		x_3	**x_3**
		y_3	**y_3**
	INPT	n	**n**
		$\bar{x}$	**$\bar{x}$**
		sx	**sx**
		n1	**n1**
		n2	**n2**
		$\bar{x}_1$	**$\bar{x}_1$**
		$\bar{x}_2$	**$\bar{x}_2$**
		sx_1	**sx_1**
		sx_2	**sx_2**
		sp	**sp**
	RESLT	*7	
GRPH	Y		**Y**
	r		**r**
	Xt		**Xt**
	Yt		**Yt**
	X		**X**
DYNA	Strt		**D_Start**
	End		**D_End**
	Pitch		**D_pitch**
TABL	Strt		**F_Start**
	End		**F_End**
	Pitch		**F_pitch**
	Reslt		**F_Result**
RECR	FORM	a_n	**a_n**
		a_{n+1}	**a_{n+1}**
		a_{n+2}	**a_{n+2}**
		b_n	**b_n**
		b_{n+1}	**b_{n+1}**
		b_{n+2}	**b_{n+2}**
		c_n	**c_n**
		c_{n+1}	**c_{n+1}**
		c_{n+2}	**c_{n+2}**
	RANG	Strt	**R_Start**
		End	**R_End**
		a_0	**a_0**
		a_1	**a_1**
		a_2	**a_2**
		b_0	**b_0**
		b_1	**b_1**
		b_2	**b_2**
		c_0	**c_0**

		c_1	**c1**
		c_2	**c2**
		a_nSt	**anStart**
		b_nSt	**bnStart**
		c_nSt	**cnStart**
	Reslt		**R_Result**
EQUA	S-Rlt		**Sim_Result**
	S-Cof		**Sim_Coef**
	P-Rlt		**Ply_Result**
	P-Cof		**Ply_Coef**
TVM	n		**n**
	I%		**I%**
	PV		**PV**
	PMT		**PMT**
	FV		**FV**
	P/Y		**P/Y**
	C/Y		**C/Y**
Str			**Str_**

Touche SHIFT VARS (PRGM)

Niveau 1	Niveau 2	Niveau 3	Commande
COM	If		**If_**
	Then		**Then_**
	Else		**Else_**
	I-End		**IfEnd**
	For		**For_**
	To		**_To_**
	Step		**_Step_**
	Next		**Next**
	Whle		**While_**
	WEnd		**WhileEnd**
	Do		**Do**
	Lp-W		**LpWhile_**
CTL	Prog		**Prog_**
	Rtrn		**Return**
	Brk		**Break**
	Stop		**Stop**
JUMP	Lbl		**Lbl_**
	Goto		**Goto_**
	⇒		**⇒**
	Isz		**Isz_**
	Dsz		**Dsz_**
	Menu		**Menu_**
?			**?**
◢			**◢**
CLR	Text		**ClrText**
	Grph		**ClrGraph**
	List		**ClrList_**
	Mat		**ClrMat_**
DISP	Stat		**DrawStat**
	Grph		**DrawGraph**
	Dyna		**DrawDyna**
	F-Tbl	Tabl	**DispF-Tbl**
		G-Con	**DrawFTG-Con**
		G-Plt	**DrawFTG-Plt**
	R-Tbl	Tabl	**DispR-Tbl**
		Phase	**PlotPhase**
		Web	**DrawWeb_**
		an-Cn	**DrawR-Con**
		Σa-Cn	**DrawR Σ-Con**
		an-Pl	**DrawR-Plt**
		Σa-Pl	**DrawR Σ-Plt**
REL	=		**=**
	≠		**≠**
	>		**>**
	<		**<**
	≥		**≥**
	≤		**≤**
I/O	Lcte		**Locate_**
	Gtky		**Getkey**
	Send		**Send(**
	Recv		**Receive(**
	S38k		**Send38k_**
	R38k		**Receive38k_**
	Open		**OpenComport38k**
	Close		**CloseComport38k**
:			**:**
STR	Join		**StrJoin(**
	Len		**StrLen(**
	Cmp		**StrCmp(**
	Src		**StrSrc(**
	Left		**StrLeft(**
	Right		**StrRight(**
	Mid		**StrMid(**
	E▶S		**Exp▶Str(**
	Exp		**Exp(**
	Upr		**StrUpr(**
	Lwr		**StrLwr(**
	Inv		**StrInv(**
	Shift		**StrShift(**
	Rot		**StrRotate(**

Touche SHIFT MENU (SET UP)

Niveau 1	Niveau 2	Niveau 3	Commande
ANGL	Deg		**Deg**
	Rad		**Rad**
	Gra		**Gra**
COOR	On		**CoordOn**
	Off		**CoordOff**
GRID	On		**GridOn**
	Off		**GridOff**
AXES	On		**AxesOn**
	Off		**AxesOff**
LABL	On		**LabelOn**
	Off		**LabelOff**
DISP	Fix		**Fix_**
	Sci		**Sci_**
	Norm		**Norm_**
	Eng	On	**EngOn**
		Off	**EngOff**
		Eng	**Eng**
S/L	—		**S-L-Normal**
	▬		**S-L-Thick**
	·····		**S-L-Broken**
	·····		**S-L-Dot**
DRAW	Con		**G-Connect**
	Plot		**G-Plot**

DERV	On		**DerivOn**
	Off		**DerivOff**
BACK	None		**BG-None**
	Pict		**BG-Pict_**
FUNC	On		**FuncOn**
	Off		**FuncOff**
SIML	On		**SimulOn**
	Off		**SimulOff**
S-WIN	Auto		**S-WindAuto**
	Man		**S-WindMan**
LIST	File		**File_**
LOCS	On		**LocusOn**
	Off		**LocusOff**
T-VAR	Rang		**VarRange**
	List		**VarList_**
ΣDSP	On		**ΣdispOn**
	Off		**ΣdispOff**
RESID	None		**Resid-None**
	List		**Resid-List_**
CPLX	Real		**Real**
	a+bi		**a+bi**
	r∠θ		**r∠θ**
FRAC	d/c		**d/c**
	ab/c		**ab/c**
Y•SPD	Norm		**Y=DrawSpeedNorm**
	High		**Y=DrawSpeedHigh**
DATE	365		**DateMode365**
	360		**DateMode360**
PMT	Bgn		**PmtBgn**
	End		**PmtEnd**
PRD	Annu		**PeriodsAnnual**
	Semi		**PeriodsSemi**
INEQ	And		**IneqTypeAnd**
	Or		**IneqTypeOr**
SIMP	Auto		**SimplfyAuto**
	Man		**SimplfyMan**
Q1Q3	Std		**Q1Q3TypeStd**
	OnD		**Q1Q3TypeOnData**

Touche SHIFT

Niveau 1	Niveau 2	Niveau 3	Commande
ZOOM	Fact		**Factor_**
	Auto		**ZoomAuto**
V-WIN	V-Win		**ViewWindow_**
	Sto		**StoV-Win_**
	Rcl		**RclV-Win_**
SKTCH	Cls		**Cls**
	Tang		**Tangent_**
	Norm		**Normal_**
	Inv		**Inverse_**
	GRPH	Y=	**Graph_Y=**
		r=	**Graph_r=**
		Parm	**Graph(X,Y)=(**
		X=c	**Graph_X=**
		G-∫dx	**Graph_∫**
		Y>	**Graph_Y>**
		Y<	**Graph_Y<**
		Y≥	**Graph_Y≥**
		Y≤	**Graph_Y≤**

		X>	**Graph_X>**
		X<	**Graph_X<**
		X≥	**Graph_X≥**
		X≤	**Graph_X≤**
	PLOT	Plot	**Plot_**
		Pl-On	**PlotOn_**
		Pl-Off	**PlotOff_**
		Pl-Chg	**PlotChg_**
	LINE	Line	**Line**
		F-Line	**F-Line_**
	Crcl		**Circle_**
	Vert		**Vertical_**
	Hztl		**Horizontal_**
	Text		**Text_**
	PIXL	On	**PxlOn_**
		Off	**PxlOff_**
		Chg	**PxlChg_**
	Test		**PxlTest(**
	STYL	—	**SketchNormal_**
		—	**SketchThick_**
		·····	**SketchBroken_**
		·····	**SketchDot_**

Programme BASE

Touche F4 (MENU)

Niveau 1	Niveau 2	Niveau 3	Commande
d~o	d		**d**
	h		**h**
	b		**b**
	o		**o**
LOG	Neg		**Neg_**
	Not		**Not_**
	and		**and**
	or		**or**
	xor		**xor**
	xnor		**xnor**
DISP	▶Dec		**▶Dec**
	▶Hex		**▶Hex**
	▶Bin		**▶Bin**
	▶Oct		**▶Oct**

Touche SHIFT VARS (PRGM)

Niveau 1	Niveau 2	Niveau 3	Commande
Prog			**Prog_**
JUMP	Lbl		**Lbl_**
	Goto		**Goto_**
	⇒		**⇒**
	Isz		**Isz_**
	Dsz		**Dsz_**
	Menu		**Menu_**
?			**?**
◢			**◢**
REL	=		**=**
	≠		**≠**
	>		**>**
	<		**<**
	≥		**≥**
	≤		**≤**
:			**:**

Touche SHIFT MENU (SET UP)

Niveau 1	Niveau 2	Niveau 3	Commande
Dec			**Dec**
Hex			**Hex**
Bin			**Bin**
Oct			**Oct**

	Niveau 3	Niveau 4	Commande
*1	Exp	ae^bx	**Exp(ae^bx)**
		ab^x	**Exp(ab^x)**
*2	MARK	▫	**Square**
		×	**Cross**
		▪	**Dot**
	STICK	Leng	**StickLength**
		Hztl	**StickHoriz**
	%DATA	%	**%**
		Data	**Data**
	None		**None**
*3	X	ax+b	**LinearReg(ax+b)**
		a+bx	**LinearReg(a+bx)**
*4	EXP	ae^bx	**ExpReg(a·e^bx)**
		ab^x	**ExpReg(a·b^x)**
*5	NORM	NPd	**NormPD(**
		NCd	**NormCD(**
		InvN	**InvNormCD(**
	t	TPd	**tPD(**
		TCd	**tCD(**
		Invt	**InvTCD(**
	CHI	CPd	**ChiPD(**
		CCd	**ChiCD(**
		InvC	**InvChiCD(**
	F	FPd	**FPD(**
		FCd	**FCD(**
		InvF	**InvFCD(**
	BINM	BPd	**BinomialPD(**
		BCd	**BinomialCD(**
		InvB	**InvBinomialCD(**
	POISN	PPd	**PoissonPD(**
		PCd	**PoissonCD(**
		InvP	**InvPoissonCD(**
	GEO	GPd	**GeoPD(**
		GCd	**GeoCD(**
		InvG	**InvGeoCD(**
	H•GEO	HPd	**HypergeoPD(**
		HCd	**HypergeoCD(**
		InvH	**InvHyperGeoCD(**
*6	Z	1-S	**OneSampleZTest_**
		2-S	**TwoSampleZTest_**
		1-P	**OnePropZTest_**
		2-P	**TwoPropZTest_**
	t	1-S	**OneSampleTTest_**
		2-S	**TwoSampleTTest_**
		REG	**LinRegTTest_**
	CHI	GOF	**ChiGOFTest_**
		2-WAY	**ChiTest_**
	F		**TwoSampleFTest_**
	ANOV	1-W	**OneWayANOVA_**
		2-W	**TwoWayANOVA_**
*7	TEST	p	**p**
		z	**z**
		t	**t**
		chi	**χ^2**
		F	**F**
		$\hat{p}$	**$\hat{p}$**
		$\hat{p}_1$	**$\hat{p}_1$**
		$\hat{p}_2$	**$\hat{p}_2$**
		df	**df**
		s_e	**s_e**
		r	**r**
		r^2	**r^2**
		pa	**pa**
		Fa	**Fa**
		Adf	**Adf**
		SSa	**SSa**
		MSa	**MSa**
		pb	**pb**
		Fb	**Fb**
		Bdf	**Bdf**
		SSb	**SSb**
		MSb	**MSb**
		pab	**pab**
		Fab	**Fab**
		ABdf	**ABdf**
		SSab	**SSab**
		MSab	**MSab**
		Edf	**Edf**
		SSe	**SSe**
		MSe	**MSe**
	INTR	Left	**Left**
		Right	**Right**
		$\hat{p}$	**$\hat{p}$**
		$\hat{p}_1$	**$\hat{p}_1$**
		$\hat{p}_2$	**$\hat{p}_2$**
		df	**df**
	DIST	p	**p**
		xInv	**xInv**
		x1Inv	**x1Inv**
		x2Inv	**x2Inv**
		zLow	**zLow**
		zUp	**zUp**
		tLow	**tLow**
		tUp	**tUp**

8. Bibliothèque de programmes

• Vérifiez le nombre d'octets libres dans la mémoire avant d'essayer d'utiliser un programme.

Nom du programme	Décomposition en facteurs premiers

Description

Ce programme divise continuellement un nombre naturel par des facteurs jusqu'à ce que tous ses facteurs premiers soient produits.

But

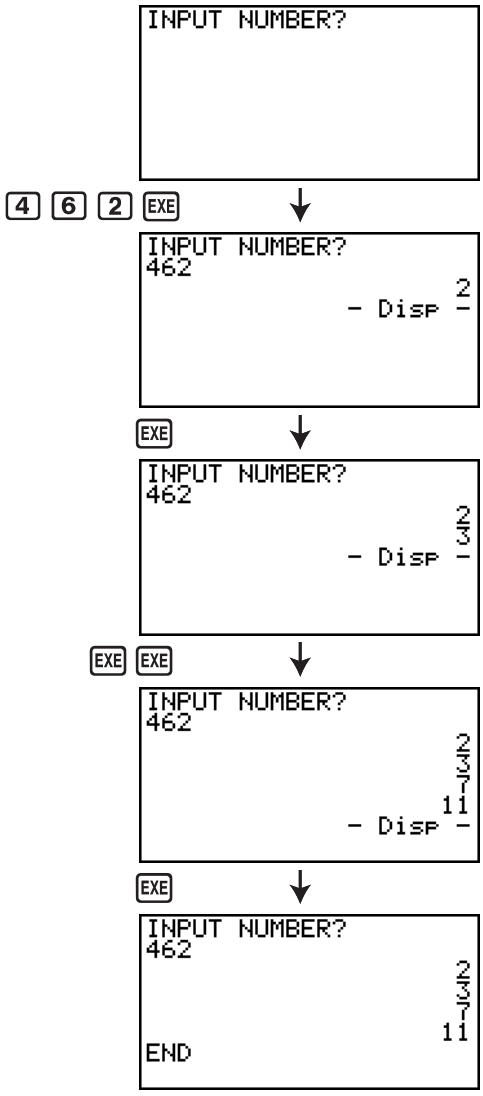
Ce programme accepte la saisie d'un nombre naturel A et le divise par B (2, 3, 5, 7…) pour trouver les facteurs premiers de A.

• Si une division a un reste nul, alors le résultat de l'opération sera affecté à A.

• L'opération précédente se répète jusqu'à ce que B > A.

Exemple **462 = 2 × 3 × 7 × 11**

```
ClrText↵
"INPUT NUMBER"?→A↵
2→B↵
Do↵
While Frac (A÷B)=0↵
B◢
A÷B→A↵
WhileEnd↵
If B=2↵
Then 3→B↵
Else B+2→B↵
IfEnd↵
LpWhile B≤A↵
"END"
```



Nom du programme Ellipse

Description

Ce programme affiche une table de valeurs basée sur la saisie des deux foyers d'une ellipse, de la somme des distances d'un point quelconque de l'ellipse à chacun des foyers et du pas d'incrémentation de X.

Y1 : Ordonnée d'un point de l'ellipse situé dans le plan supérieur

Y2 : Ordonnée d'un point de l'ellipse situé dans le plan inférieur

Y3 : Distance entre un point de l'ellipse et un foyer

Y4 : Distance entre le même point de l'ellipse et l'autre foyer

Y5 : Somme de Y3 et Y4

Puis, le programme trace les foyers et les valeurs dans Y1 et Y2.

But

Ce programme illustre que la somme des distances d'un point quelconque de l'ellipse à chacun des foyers est constante.

```
AxesOff↵
Do↵
ClrText↵
"FOCUS (C,0),(-C,0)"↵
"C="?→C↵
"SUM DISTANCE"?→D↵
LpWhile 2Abs C≥D Or D≤0↵
D÷2→A↵
√(A²-C²)→B↵
Y=Type↵
"B√(1-X²÷A²)"→Y1↵
"-Y1"→Y2↵
"√((X-C)²+Y1²)"→Y3↵
"√((X+C)²+Y1²)"→Y4↵
"Y3+Y4"→Y5↵
For 1→E To 20↵
If E≤5↵
Then T SelOn E↵
Else T SelOff E↵
IfEnd↵
Next↵
-Int A→F Start↵
Int A→F End↵
"F pitch"?→F pitch↵
DispF-Tbl◢
ClrGraph↵
1.2A→Xmax↵
-1.2A→Xmin↵
1.2B→Ymax↵
-1.2B→Ymin↵
T SelOff 3↵
T SelOff 4↵
T SelOff 5↵
DispF-Tbl↵
DrawFTG-Plt↵
PlotOn C,0↵
PlotOn -C,0◢
"END"
```