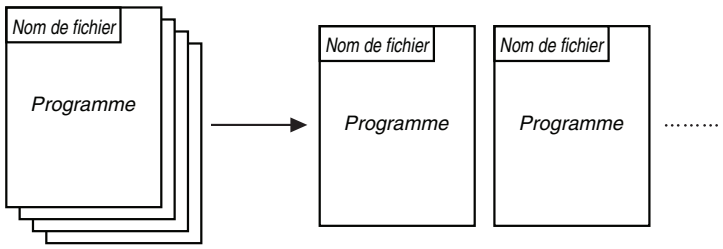
Programmation

20-1 Avant la programmation

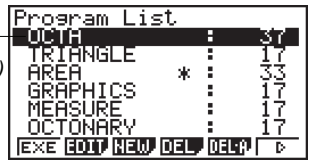
La programmation permet d'effectuer rapidement des calculs complexes et répétitifs. Les commandes et les calculs sont exécutés dans l'ordre qui est utilisé lors des calculs manuels à instructions multiples. Les programmes peuvent être stockés sous des noms de fichiers faciles à rappeler et modifier.



Sélectionnez le symbole **PRGM** sur le menu principal et entrez dans le mode PRGM. Une liste de programmes apparaît alors à l'écran.

Mémoire sélectionnée

(utilisez ▲ et ▼ pour changer de sélection)



- {**EXE**}/{**EDIT**} ... {exécution}/{édition} d'un programme
- {**NEW**} ... {nouveau programme}

P.368

- {**DEL**}/{**DEL·A**} ... effacement {d'un programme particulier}/{de tous les programmes}

P.362

- {**SRC**}/{**REN**} ... {recherche}/{changement} d'un nom de fichier
- {**LOAD**} ... {charge un programme de la bibliothèque de programmes}

*Voir le manuel **indépendant** Banque de données pour les détails.

couleur

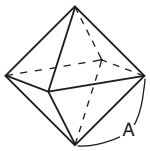
- Si aucun programme n'est stocké dans la mémoire lorsque vous entrez dans le mode PRGM, le message "**No Programs**" apparaît à l'écran et seul le paramètre NEW (F3) est indiqué sur le menu de fonctions.

  Les valeurs à la droite de la liste de programmes indiquent le nombre d'octets utilisés par chaque programme.

20-2 Exemples de programmation

Exemple 1 **Calculer l'aire et le volume de trois octaèdres réguliers ayant les dimensions indiquées sur le tableau suivant**

Stocker la formule de calcul sous le nom de fichier OCTA.



Longueur d'une face (A)	Aire (S)	Volume (V)
7 cm	cm^2	cm^3
10 cm	cm^2	cm^3
15 cm	cm^2	cm^3

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

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

Avant d'entrer une nouvelle formule, enregistrez d'abord le nom de fichier, puis entrez le programme proprement dit.

●Pour enregistrer un nom de fichier

Exemple **Enregistrer le nom de fichier OCTA**

- Le nom de fichier peut contenir au plus huit caractères.

1. Affichez la liste des programmes et appuyez sur [F3] (NEW) pour afficher le menu qui contient les paramètres suivants.

P.360

- {**RUN**}/{**BASE**} ...entrée d'un programme {pour un calcul ordinaire}/{dans une base numérique donnée}
- {**n0**} ... {enregistrement du code d'accès}
- {**SYBL**} ... {menu de symboles}

2. Entrez le nom du fichier.

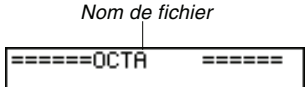
[O] [C] [T] [A]

```
Program Name
[OCTA■   ]
```

- Le curseur change de forme pour indiquer que vous pouvez entrer des caractères alphabétiques.
- Vous pouvez utiliser les caractères suivants pour enregistrer un nom: A à Z, *r*, *θ*, espace, [,], {, }, ', ", ~, 0 à 9, ., +, –, □, □
- Notez cependant que [X,θ,T] et [•] ne peuvent pas être utilisés pour le nom d'un programme contenant des calculs binaires, octaux, décimaux ou hexadécimaux.

- Utilisez [F1] (RUN) pour entrer un programme de calcul ordinaire (à exécuter dans le mode COMP). Avec les calculs qui impliquent un système numérique particulier, utilisez [F2] (BASE). Les programmes qui sont entrés après une pression sur [F2] (BASE) sont indiqués par B à la droite du nom de fichier.
- Appuyez sur [F6] (SYBL) pour afficher un menu des symboles (', ", ~) qui peuvent être entrés.
- Vous pouvez effacer un caractère lors de l'enregistrement du nom de fichier en amenant le curseur sur le caractère que vous voulez supprimer et en appuyant sur [DEL].

3. Appuyez sur [EXE] pour enregistrer le nom de fichier et afficher l'écran de programmation.



- L'enregistrement d'un nom de fichier utilise 17 octets de mémoire.
- L'écran d'enregistrement de nom de fichier reste affiché si vous appuyez sur [EXE] sans entrer de nom de fichier.
- Pour quitter un écran d'enregistrement de nom de fichier et revenir à la liste de programmes sans enregistrer de nom de fichier, appuyez sur [EXIT].
- Quand vous enregistrez le nom d'un programme qui contient des calculs binaires, octaux, décimaux ou hexadécimaux, l'indicateur B est ajouté à la droite du nom de fichier.

●Pour introduire un programme

Le menu de fonctions de l'écran qui est utilisé pour la programmation contient les paramètres suivants.

P.365

- {**TOP**}/{**BTM**} ... {début}/{fin} du programme

P.364

- {**SRC**} ... {recherche}
- {**MENU**} ... {menu de modes}
- {**SYBL**} ... {menu de symboles}

●Pour changer de mode dans un programme

- Appuyez sur [F4] (MENU) quand l'écran de programmation apparaît pour afficher un menu de changement de mode. Vous pouvez utiliser ce menu pour changer de mode en cours de programmation.

P.3

- {**STAT**}/{**MAT**}/{**LIST**}/{**GRPH**}/{**DYNA**}/{**TABL**}/{**RECR**}

  Pour les détails sur chaque mode, voir "Pour sélectionner un symbole" , ainsi que les différentes sections de ce manuel, qui décrivent les possibilités offertes par chaque mode.

- Le menu suivant apparaît quand vous appuyez sur [F4] (MENU) pendant l'introduction d'un programme qui exige une base numérique particulière.
- {**d ~ o**}/{**LOG**}

- Appuyez sur [F6] (SYBL) pour afficher un menu des symboles (', ", ~, ∗, /, #) qui peuvent être entrés dans un programme.

- Appuyez sur [SHIFT] [SETUP] pour afficher un menu des commandes qui peuvent être utilisées pour changer les réglages de l'écran de configuration en cours de programmation.

- {**ANGL**}/{**COOR**}/{**GRID**}/{**AXES**}/{**LABL**}/{**DISP**}/{**P/L**$_{\diamond}$}/{**DRAW**}/{**DERV**}/{**BACK**}/{**FUNC**}/{**SIML**}/{**S-WIN**}/{**LIST**}/{**LOCS**$_{\diamond}$}/{**T-VAR**}/{Σ**DSP**}/{**RESID**}

P.5

Pour les détails sur chacune de ces commandes, voir "Menus de touches de fonction sur l'écran de configuration".

Le menu de touches de fonctions suivant apparaît si vous appuyez sur [SHIFT] [SETUP] lors de l'entrée d'un programme contenant des calculs binaires, octaux, décimaux ou hexadécimaux.

- {**Dec**}/{**Hex**}/{**Bin**}/{**Oct**}

Le contenu proprement dit d'un programme est identique aux calculs manuels. Voici comment l'aire et le volume d'un octaèdre régulier sont calculés lors d'une opération manuelle.

Aire S ..... [2] [×] [SHIFT] [√] [3] [×] <valeur de A> [x^2] [EXE]
Volume V ..... [SHIFT] [√] [2] [÷] [3] [×] <valeur de A> [^] [3] [EXE]

Vous pouvez aussi effectuer ce calcul en affectant la longueur d'une face à la variable A.

Longueur d'une face A
..... <valeur de A> [→] [ALPHA] [A] [EXE]
Aire S ..... [2] [×] [SHIFT] [√] [3] [×] [ALPHA] [A] [x^2] [EXE]
Volume V ..... [SHIFT] [√] [2] [÷] [3] [×] [ALPHA] [A] [^] [3] [EXE]

Si vous entrez simplement le calcul manuel ci-dessus, la calculatrice l'exécutera sans s'arrêter du début à la fin. Les commandes suivantes permettent d'interrompre le calcul pour entrer des valeurs et afficher les résultats intermédiaires.

?: Cette commande interrompt l'exécution d'un programme et affiche un point d'interrogation pour indiquer l'entrée d'une valeur devant être affectée à une variable. La syntaxe de cette commande est la suivante: ? → <nom de la variable>.

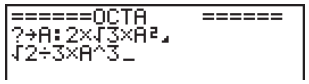
◢: Cette commande interrompt l'exécution d'un programme et affiche le résultat du dernier calcul obtenu ou un texte. Elle correspond à une pression sur [EXE] dans un calcul manuel.

P.369

- Pour tous les détails sur l'utilisation de ces commandes, voir "Commandes de programmation pratiques".

Les exemples suivants indiquent comment utiliser concrètement les commandes ? et ◢.

[SHIFT] [PRGM] [F4](?)[→] [ALPHA] [A] [F6](▷)[F5](:)
[2] [×] [SHIFT] [√] [3] [×] [ALPHA] [A] [x^2]
[F6](▷)[F5](◢)
[SHIFT] [√] [2] [÷] [3] [×] [ALPHA] [A] [^] [3]



[SHIFT] [QUIT] ou [EXIT] [EXIT]

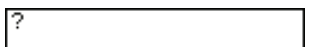
●Pour mettre un programme en route

1. Quand la liste de programmes est à l'écran, utilisez ▲ et ▼ pour mettre le nom du programme que vous voulez exécuter en surbrillance.
2. Appuyez sur [F1] (EXE) ou [EXE] pour exécuter le programme.

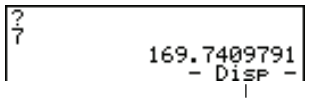
Essayons de mettre en route le programme que nous avons entré précédemment.

Longueur d'une face (A)	Aire (S)	Volume (V)
7 cm	169,7409791 cm^2	161,6917506 cm^3
10 cm	346,4101615 cm^2	471,4045208 cm^3
15 cm	779,4228634 cm^2	1590,990258 cm^3

[F1] (EXE) ou [EXE]

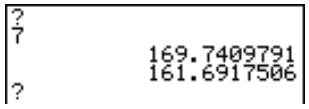


[7] [EXE]
(Valeur de A)

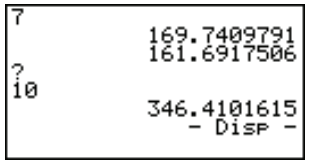


Résultat intermédiaire produit par ◢

[EXE] [EXE]



[1] [0] [EXE]



EXE

```
7
                169.7409791
                161.6917506
?
10
                346.4101615
                471.4045208
```

⋮ ⋮

- Si vous appuyez sur EXE quand le résultat final d'un programme est affiché, tout le programme sera de nouveau exécuté.
- Vous pouvez aussi exécuter un programme dans le **mode RUN** en entrant: Prog "<nom de fichier>" EXE.
- Une erreur se produit si le programme désigné par Prog "<nom de fichier>" ne peut pas être trouvé.

P.377

20-3 Mise au point d’un programme

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’un débogage 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.

```
Ma   ERROR
```

P.436

Quand ce type de message apparaît, appuyez sur ◀ ou ▶ pour afficher le point où l’erreur s’est produite ainsi que le curseur. Contrôlez le “Tableau de messages d’erreur” pour savoir quelles dispositions prendre pour remédier à la situation.

P.360

- Une pression sur ◀ ou ▶ ne permettra pas d’afficher le point où l’erreur s’est produite si le code d’accès est protégé.

●Pour éliminer les erreurs à l’origine de mauvais résultats

P.365

Si le programme aboutit à un résultat qui ne correspond pas à vos attentes, vérifiez le contenu du programme et effectuez les modifications nécessaires. Voir “Édition d’un nom de fichier et d’un programme” pour les détails sur la modification d’un programme.

20-4 Calcul du nombre d'octets utilisés par un programme

Il y a deux types de commandes: les commandes qui utilisent 1 octet* et celles qui utilisent 2 octets* de mémoire.

* Un octet est une unité de mémoire pouvant être utilisée pour le stockage de données.

- Exemple de commandes à 1 octet: sin, cos, tan, log, (,), A, B, C, 1, 2, etc.
- Exemple de commandes à 2 octets: Lbl 1, Goto 2, etc.

Quand le curseur est visible sur un programme, chaque pression sur ◀ ou ▶ le fait avancer d'un octet.

P.24

- Vous pouvez vérifier le volume de mémoire utilisé et le volume restant, quand vous le souhaitez, en sélectionnant le symbole **MEM** sur le menu principal, puis en entrant dans le mode MEM.
  Voir "Statut de la mémoire (MEM)" pour les détails.

20-5 Accès secret

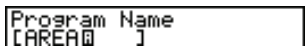
Lorsque vous créez un programme, vous pouvez le protéger avec un code d'accès, pour qu'il ne soit accessible qu'aux personnes qui connaissent le code. Les programmes protégés par un code ne peuvent pas être exécutés si l'on en connaît pas le code.

●Pour enregistrer un code

Exemple **Créer un fichier de programme sous le nom AREA et le protéger par le code CASIO**

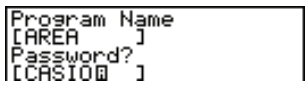
1. Quand la liste de programmes est à l'écran, appuyez sur [F3] (NEW) pour enregistrer le nom de fichier du nouveau programme.

   [F3](NEW)
   [A] [R] [E] [A]

2. Appuyez sur [F5] (π0) puis entrez le code d'accès.

   [F5](π0)
   [C] [A] [S] [I] [O]

P.353

- L'enregistrement d'un code d'accès est identique à l'enregistrement d'un nom de fichier.

3. Appuyez sur [EXE] pour enregistrer le nom de fichier et le code. Vous pouvez maintenant enregistrer le contenu du programme.
   - L'enregistrement d'un code d'accès occupe 16 octets de mémoire.
   - Si vous appuyez sur [EXE] sans enregistrer de code d'accès, seul le nom de fichier est enregistré, sans code.

4. Une fois que vous avez introduit le programme, appuyez sur [SHIFT] [QUIT] pour sortir du fichier et revenir à la liste de programmes. Les fichiers qui sont protégés par un code sont indiqués par un astérisque à la droite du nom de fichier.

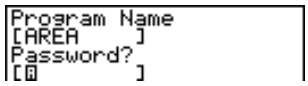
   

●Pour rappeler un programme

Exemple **Rappeler le fichier nommé AREA qui est protégé par le code d'accès CASIO**

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 code d'accès et appuyez sur EXE pour rappeler le programme.
   • Le message "**Mismatch**" apparaît si vous tapez un mauvais code.

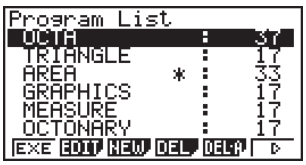
20-6 Recherche d'un fichier

Il existe trois méthodes différentes pour localiser le nom d'un fichier particulier.

●Pour localiser un fichier en faisant défiler les noms

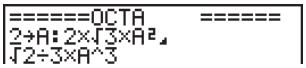
Exemple **Rappeler le programme nommé OCTA en faisant défiler la liste de programmes**

1. Quand la liste de programmes est à l'écran, utilisez ▲ et ▼ pour passer toute la liste des noms de programmes en revue jusqu'à ce que vous trouviez le programme souhaité.



F2

2. Quand la surbrillance est sur le nom de fichier souhaité, appuyez sur F2 (EDIT) pour rappeler le fichier.



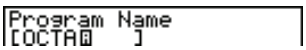
●Pour localiser un fichier par son nom

Exemple **Faire une recherche de nom pour rappeler le programme nommé OCTA**

P.360

1. Quand la liste de programmes est à l'écran, appuyez sur F3 (NEW) et entrez le nom du fichier que vous voulez localiser.
   - Si le fichier que vous recherchez est protégé par un code, vous devez entrer aussi le code d'accès.

   F3 (NEW)
   O C T A



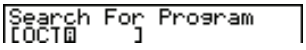
2. Appuyez sur EXE pour rappeler le programme.
   - S'il n'y a aucun programme dont le nom de fichier correspond à celui que vous avez entré, un nouveau fichier est créé à partir de ce nom.

●Pour localiser un fichier par ses initiales

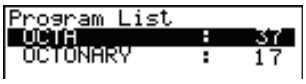
Exemple **Faire une recherche par initiales pour rappeler le programme nommé OCTA**

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

   F6 (▷) F1 (SRC)
   O C T



2. Appuyez sur [EXE] pour commencer la recherche des noms de fichiers.



- Tous les fichiers dont le nom commence par ces caractères sont rappelés.
- Si aucun programme ne commence par les caractères que vous avez entrés, le message "**Not Found**" apparaît à l'écran. Dans ce cas, appuyez sur [EXIT] pour annuler le message d'erreur.

3. Utilisez ▲ et ▼ pour mettre en surbrillance le nom du programme que vous voulez rappeler, puis appuyez sur [F2] (EDIT) pour le rappeler.

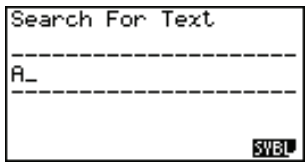
20-7 Recherche de données à l'intérieur d'un programme

Exemple **Rechercher la lettre "A" dans le programme nommé OCTA**

1. Rappeler le programme.

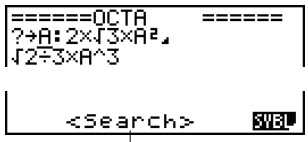
2. Appuyez sur [F3](SRC) et entrez les données que vous recherchez.

[F3](SRC)
[ALPHA] [A]



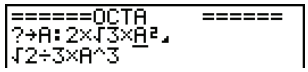
- Vous ne pouvez pas utiliser la commande de retour (↵) ni la commande d'affichage de résultat (◢) pour la recherche de donnée.

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 définie.



Signale que la recherche est en cours.

4. Appuyez sur [EXE] pour localiser la seconde occurrence.



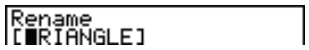
- Si aucune donnée ne correspond à celle que vous avez désignée, le contenu du programme apparaît avec le curseur positionné à l'endroit où vous avez commencé la recherche.
- Lorsque le programme est à l'écran, vous pouvez changer le curseur de place en utilisant les touches de curseur avant de localiser l'occurrence suivante. La recherche s'effectue seulement à partir de la nouvelle position du curseur quand vous appuyez sur [EXE].
- Quand le type de donnée recherché est trouvé, l'entrée d'un caractère ou le déplacement du curseur met fin à la recherche et l'indicateur de recherche disparaît de l'écran.
- Si vous faites une erreur en entrant les caractères que vous recherchez, appuyez sur [AC] pour supprimer votre entrée et recommencez depuis le début.

20-8 Édition d'un nom de fichier et d'un programme

•Pour éditer un nom de fichier

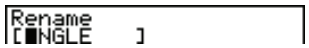
Exemple **Remplacer le nom de fichier TRIANGLE par ANGLE**

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.

   [DEL] [DEL] [DEL]



3. Appuyez sur [EXE] pour enregistrer le nouveau nom et revenir à la liste de programmes.
   - Si, après modification, le nouveau nom de fichier est identique à un nom de programme stocké en mémoire, le message "**Already Exists**" apparaît. Dans ce cas, vous pouvez effectuer une des deux opérations suivantes pour remédier à la situation.
   - Appuyez sur ▶ ou ◀ pour annuler l'erreur et revenir à l'écran d'enregistrement de nom de fichier.
   - Appuyez sur [AC] pour annuler le nom du nouveau fichier et entrer un nouveau nom.

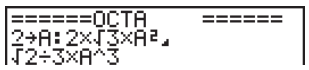
•Pour éditer un programme

1. Recherchez le nom de fichier correspondant au programme que vous voulez modifier.

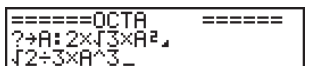
P.20

2. Rappelez le programme.
   - La méthode utilisée pour modifier un programme est comparable à celle utilisée pour l'édition de calculs manuels. Pour les détails, voir "Édition de calculs".
   - Les touches de fonctions suivantes sont également utiles lors de l'édition d'un programme.

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



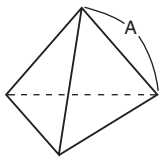
   [F2] (BTM) ..... Positionne le curseur en fin de programme



P.353

Exemple 2 **Utiliser le programme OCTA pour créer un programme qui calcule l'aire et le volume d'un tétraèdre régulier quand la longueur d'une face est connue**

Utiliser TETRA comme nom de fichier.



Longueur d'une face (A)	Aire (S)	Volume (V)
7 cm	cm^2	cm^3
10 cm	cm^2	cm^3
15 cm	cm^2	cm^3

Les formules utilisées pour le calcul de l'aire S et du volume V d'un tétraèdre régulier dont la longueur d'une face est connue sont les suivantes.

$$S = \sqrt{3}\,A^2, \quad V = \frac{\sqrt{2}}{12}\,A^3$$

Faites les opérations suivantes pour introduire le programme.

Longueur d'une face A .. [SHIFT] [PRGM] [F4](?) [→] [ALPHA] [A] [F6](▷) [F5](:)
Aire S ..... [SHIFT] [√] [3] [×] [ALPHA] [A] [x^2] [F6](▷) [F5](◢)
Volume V ..... [SHIFT] [√] [2] [÷] [1] [2] [×] [ALPHA] [A] [^] [3]

Comparez ce programme à celui effectué pour le calcul de l'aire et du volume d'un octaèdre régulier.

Longueur d'une face A .. [SHIFT] [PRGM] [F4](?) [→] [ALPHA] [A] [F6](▷) [F5](:)
Aire S ..... [2] [×] [SHIFT] [√] [3] [×] [ALPHA] [A] [x^2] [F6](▷) [F5](◢)
Volume V ..... [SHIFT] [√] [2] [÷] [3] [×] [ALPHA] [A] [^] [3]

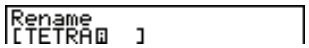
Vous pouvez donc créer le programme TETRA en effectuant les changements suivants dans le programme OCTA.

- Vous supprimez [2] [×] (signalé par un trait ondulé)
- Vous remplacez [3] par [1] [2] (signalé par un trait continu)

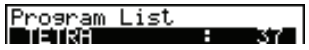
Modifions maintenant OCTA pour obtenir le programme TETRA.

1. Changez d'abord le nom.

[F6](▷) [F2](REN) [T] [E] [T] [R] [A]



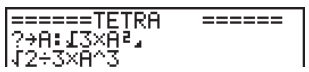
[EXE]



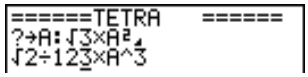
2. Changez ensuite le contenu.

[F2](EDIT)

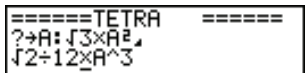
[▶] [▶] [▶] [▶] [DEL] [DEL]



▼ ◀ SHIFT INS 1 2



DEL



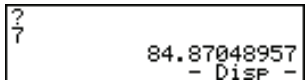
SHIFT QUIT

Mettons maintenant le programme en route.

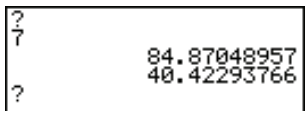
Longueur d'une face (A)	Aire (S)	Volume (V)
7 cm	84,87048957 cm^2	40,42293766 cm^3
10 cm	173,2050808 cm^2	117,8511302 cm^3
15 cm	389,7114317 cm^2	397,7475644 cm^3

F1 (EXE) ou EXE

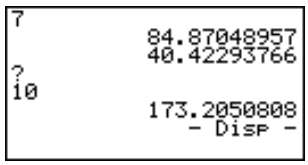
7 EXE
(Valeur de A)



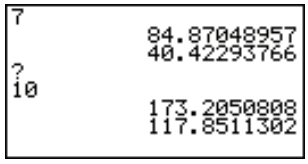
EXE EXE



1 0 EXE



EXE



20-9 Effacement d’un programme

Il existe deux méthodes pour supprimer le nom d'un fichier et le programme correspondant.

●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.

P.26

- Vous pouvez aussi supprimer tous les programmes en **mode MEM**. Voir “Suppression du contenu de la mémoire” pour les détails.

20-10 Commandes de programmation pratiques

Outre les commandes de calcul, la calculatrice offre tout un éventail de commandes d'opérateurs relationnels et de saut qui peuvent être utilisées pour créer des programmes qui faciliteront les calculs.

Menu de programmation

Appuyez sur [SHIFT] [PRGM] pour afficher le menu de programmation.

- {**COM**}/{**CTL**}/{**JUMP**}/{**CLR**}/{**DISP**}/{**REL**}/{**I/O**}
- {**?**} ... {commande d'entrée}
- {◢} ... {commande de sortie}
- { **:** } ... {commande d'instructions multiples}

■ COM (Menu de commandes de boucles et branchements conditionnels)

La sélection de {COM} sur le menu de programmation permet d'afficher les paramètres suivants du menu de fonctions.

- {**If**}/{**Then**}/{**Else**}/{**I-End**}/{**For**}/{**To**}/{**Step**}/{**Next**}/{**Whle**}/{**WEnd**}/{**Do**}/{**Lp-W**} ... commande {If}/{Then}/{Else}/{IfEnd}/{For}/{To}/{Step}/{Next}/{While}/{WhileEnd}/{Do}/{LpWhile}

■ CTL (Menu de commandes de contrôle de programmation)

La sélection de {CTL} sur le menu de programmation permet d'afficher les paramètres suivants du menu de fonctions.

- {**Prog**}/{**Rtrn**}/{**Brk**}/{**Stop**} ... commande {Prog}/{Return}/{Break}/{Stop}

■ JUMP (Menu de commandes de saut)

La sélection de {JUMP} sur le menu de programmation permet d'afficher les paramètres suivants du menu de fonctions.

- {**Lbl**}/{**Goto**} ... commande {Lbl}/{Goto}
- {⇒} ... {commande de saut}
- {**Isz**}/{**Dsz**} ... {saut et incrément}/{saut et décrément}

■ CLR (Menu de commandes d'effacement)

La sélection de {CLR} sur le menu de programmation permet d'afficher les paramètres suivants du menu de fonctions.

- {**Text**}/{**Grph**}/{**List**} ... effacement de {texte}/{graphe}/{liste}

■ DISP (Menu de commandes d'affichage)

La sélection de {DISP} sur le menu de programmation permet d'afficher les paramètres suivants du menu de fonctions.

- {**Stat**}/{**Grph**}/{**Dyna**} ... tracé de {graphe statistique}/{graphe}/{graphe dynamique}

- {**F-Tbl**} ... {menu de commande de Table et Graphe}

Les paramètres qui apparaissent dans le menu précédent sont les suivants.

  - {**Tabl**}/{**G-Con**}/{**G-Plt**} ... commande {DispF-Tbl}/{DrawFTG-Con}/{DrawFTG-Plt}

- {**R-Tbl**} ... {calcul et formule de récurrence}

Les paramètres qui apparaissent dans le menu précédent sont les suivants.

  - {**Tabl**}/{**Web**}/{**an-Cn**}/{Σ**a-Cn**}/{**an-Pl**}/{Σ**a-Pl**} ... commande {DispR-Tbl}/{DrawWeb}/{DrawR-Con}/{DrawRΣ-Con}/{DrawR-Plt}/{DrawRΣ-Plt}

■ REL (Commande d'opérateurs relationnels avec saut conditionnel)

La sélection de {REL} sur le menu de programmation permet d'afficher les paramètres suivants du menu de fonctions.

- {**=**}/{≠}/{**>**}/{**<**}/{≥}/{≤} ... opérateurs relationnels {=}/{≠}/{>}/{<}/{≥}/{≤}

■ I/O (Commandes d'entrée/sortie)

La sélection de {I/O} sur le menu de programmation permet d'afficher les paramètres suivants du menu de fonctions.

- {**Lcte**}/{**Gtky**}/{**Send**}/{**Recv**} ... commande {Locate}/{Getkey}/{Send(}/{Receive(}

- Le menu de fonctions se présente de manière un peu différente lorsqu'un programme contient des calculs binaires, octaux, décimaux ou hexadécimaux, mais les fonctions du menu sont les mêmes.

20-11 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 entrés 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 entrez une commande.
[Crochets]	Les crochets doivent être utilisés pour indiquer des paramètres qui sont optionnels. N'insérez pas de crochets quand vous entrez 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 de base

Commande d'entrée (?)

Fonction: Demande d'entrer une valeur devant être affectée à une variable pendant la programmation.

Syntaxe: ? → <nom de la variable>

Exemple: ? → A ↵

Description:

1. Cette commande interrompt provisoirement l'exécution du programme et vous demande d'entrer une valeur ou une expression qui sera affectée à une variable. Quand la commande d'entrée est exécutée, "?" apparaît à l'écran et la calculatrice attend que la valeur soit entrée.
2. La réponse à cette commande doit être une valeur ou une expression, mais l'expression ne peut pas être une instruction multiple.

Commande de sortie (◢)

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

Description:

1. 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.
2. La commande de sortie 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:

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

Retour (↵)

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

Description:

1. Le retour fonctionne de la même façon que la commande d'instructions multiples.
2. L'utilisation du retour à la place de la commande d'instructions multiples facilite la lecture du programme affiché.

■ Commandes de boucles et branchements conditionnels (COM)

- Définissons a, b, c, d, e... comme étant des instructions.
- Les séparations entre les instructions peuvent être "↵", ":" ou "◢".

Dans les exemples ci-dessous nous utiliserons ":".

- Nous dirons qu'un test est vrai s'il est vérifié et qu'il est faux dans le cas contraire.

Exemple: Si A>3 est vrai pour A = 5.

If ~ Then

Syntaxe: If <condition> : Then a : b : c : d : e...

Si le test est vrai, les instructions a, b, c, d, e... sont exécutées.

Si le test est faux, le programme recommence au tout début du programme.

If ~ Then ~ If End

Syntaxe: If <condition> : Then a : b : c : If End : d : e... etc...

Si le test est vrai, les instructions a, b, c, d, e... sont exécutées.

Si le test est faux, les instructions d, e... sont exécutées.

If ~ Then ~ Else

Syntaxe: If <condition> : Then a : b : c : Else d : e : f... etc...

Si le test est vrai, a, b, c sont exécutées et le programme recommence au tout début du programme.

Si le test est faux, d, e, f... sont exécutées.

If ~ Then ~ Else ~ If End

Syntaxe: If <condition> : Then a : b : c : Else d : e : If End : f : g : etc...

Si le test est vrai, a, b, c, f, g sont exécutées.

Si le test est faux, d, e, f, g sont exécutées.

For~To~Next

Fonction: Cette commande répète tout ce qui se trouve entre l'instruction For (de) et l'instruction Next (suivant). La valeur initiale est affectée à la variable de référence à la première exécution, puis cette variable est incrémentée à chaque exécution. L'exécution se poursuit jusqu'à ce que la valeur de la variable de référence atteigne la valeur finale.

Syntaxe:

For <valeur initiale> → <nom de la variable de référence> To <valeur finale> { ↵ / : / ◢ }

[<instruction> { ↵ / : / ◢ }] Next

Paramètres:

- Nom de la variable de référence: A à Z
- Valeur initiale: valeur ou expression qui produit une valeur (ex. sin x, A, etc.)
- Valeur finale: valeur ou expression qui produit une valeur (ex. sin x, A, etc.)

Description:

1. Quand la valeur initiale de la variable de référence est supérieure à la valeur finale, l'exécution continue à partir de l'instruction suivant Next sans exécuter les instructions entre For et Next.
2. Une instruction For doit toujours avoir une instruction Next correspondante, et l'instruction Next doit toujours venir après l'instruction For qui lui correspond.
3. L'instruction Next définit la fin de la boucle créée par For~Next, et elle doit toujours être incluse. Dans le cas contraire, une erreur se produit.

Exemple: For 1 → A To 10 ↵
A □ 3 → B ↵
B ◢
Next

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 (ex. sin x, A, etc.)
- Valeur finale: valeur ou expression qui produit une valeur (ex. sin x, A, etc.)
- Valeur de l'incrément: valeur numérique (l'omission de cette valeur impose 1 comme incrément)

Description:

1. Cette commande est fondamentalement identique à For~To~Next. La seule différence est que vous pouvez spécifier l'incrément.
2. L'omission de cette valeur impose 1 comme incrément.
3. 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.

Exemple: For 1 → A To 10 Step 0.1 ↵
A □ 3 → B ↵
B ◢
Next

Do~LpWhile

Fonction: Cette commande répète des commandes particulières entre Do et LpWhile tant que sa condition est vraie. Le test est réalisé après les instructions.

Syntaxe:

Do { ↵ / : / ◢ } ~ LpWhile <expression>

Paramètres: Expression

Description:

1. Cette commande répète les commandes contenues dans la boucle tant que sa condition est vraie. Quand la condition devient fausse, l'exécution continue à partir de l'instruction suivant l'instruction LpWhile.
2. 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.

Exemple: Do↵
? → A↵
A □ 2 → B↵
B ◢
LpWhile B >10

While~WhileEnd

Fonction: Cette commande répète des commandes particulières entre While et WhileEnd tant que sa condition est vraie. Le test est réallsé avant les instructions.

Syntaxe:

While <expression> { ↵ / : / ◢ } ~ WhileEnd

Paramètres: Expression

Description:

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

• Il y aura 10 affichages de "GOOD".

Exemple: 10 → A↵
While A > 0↵
A – 1 → A↵
"GOOD"↵
WhileEnd

■ 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:

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

Exemple:
```
While A>0↵
If A > 2↵
Then Break↵
IfEnd↵
WhileEnd↵
A ◢  ←———— Exécuté après l'interruption   si A>2
```

Prog

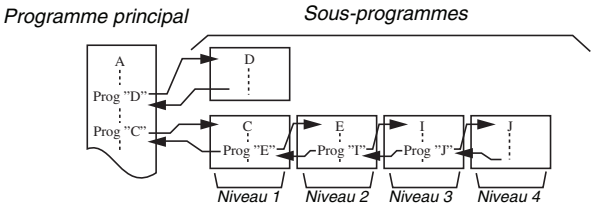
Fonction: Cette commande définit l'exécution d'un autre programme en tant que sous-programme. Dans le mode RUN, cette commande exécute un nouveau programme.

Syntaxe: Prog "nom de fichier" ↵

Exemple: Prog "ABC" ↵

Description:

1. Même quand cette commande se trouve à l'intérieur d'une boucle, elle interrompt immédiatement la boucle et démarre le sous-programme.
2. 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.
3. 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.



4. L'appel d'un sous-programme l'exécute 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.

5. 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.
6. Si le sous-programme correspondant au nom de fichier défini par la commande Prog n'existe pas, une erreur se produit.
7. Dans le **mode RUN**, l'entrée de la commande Prog et sa validation par [EXE] mettent en route le programme désigné 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 d'un programme principal arrête l'exécution de ce programme.

Exemple:

```
Prog "A"          Prog "B"
1 → A↵            For A → B To 10↵
Prog "B"↵         B + 1 → C↵
C ◢               Next↵
                  Return
```

L'exécution du programme dans le fichier A affiche le résultat de l'opération (11).

Stop

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

Syntaxe: Stop ↵

Description:

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

Exemple:

```
For 2 → I To 10↵
If I = 5↵
Then "STOP" : Stop↵
IfEnd↵
Next
```

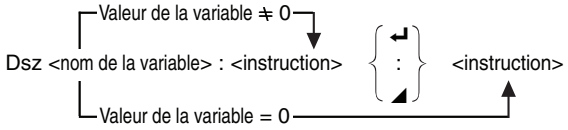
Ce programme compte de 2 à 10. Cependant, quand le compte atteint 5, il termine l'exécution et le message "STOP" est affiché.

■ 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 saute 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 zéro, l'exécution continue avec l'instruction suivante. Si la valeur est égale à zéro, l'exécution saute à l'instruction suivant la commande d'instruction multiple (:), la commande d'affichage de résultat (◢) ou la commande de retour. (↵).

Exemple: 10 → A : 0 → C :
Lbl 1 : ? → B : B+C → C :
Dsz A : Goto 1 : C □ 10
Ce programme demande d'entrer 10 valeurs, puis de calculer la moyenne des valeurs entrées.

Goto~Lbl

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

Syntaxe: Goto <valeur ou variable> ~ Lbl <valeur ou variable>

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

Description:

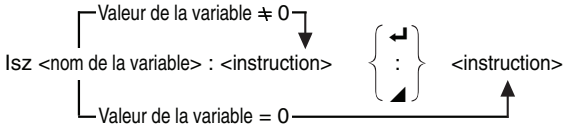
1. Cette commande comprend deux éléments: Goto n (n étant une valeur de 0 à 9) et Lbl n (n étant la valeur définie par Goto). Cette commande fait sauter l'exécution du programme à l'instruction Lbl dont la valeur correspond à celle qui a été spécifiée par l'instruction Goto.
2. Cette commande peut être utilisée pour revenir au début d'un programme ou pour sauter à un endroit quelconque du programme.
3. Cette commande peut être combinée aux sauts conditionnels et aux sauts avec compteurs.
4. S'il n'y a aucune instruction Lbl dont la valeur correspond à celle définie par l'instruction Goto, une erreur se produit.

Exemple: ? → A : ? → B : Lbl 1 :
? → X : A ▯ X + B ◢
Goto 1
Ce programme calcule y = AX + B pour le nombre de valeurs que vous voulez entrer pour chaque variable. Pour abandonner l'exécution de ce programme, appuyez sur [AC].

Isz

Fonction: Cette commande est un saut avec compteur qui incrémente la valeur de la variable de référence d'une unité, puis saute 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.

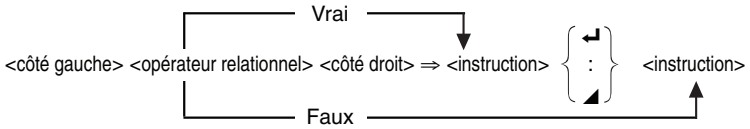
Description:

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 saute 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:

P.387

côté gauche/côté droit: variable (A à Z, r, θ), constante numérique, expression variable (comme A □ 2)

Opérateur relationnel: =, ≠, >, <, ≥, ≤

Description:

1. 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.
2. 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 saute les instructions suivant la commande d'instructions multiples (:), la commande d'affichage (◢) ou la commande de retour (↵).

Exemple: Lbl 1 : ? → A :
A ≧ 0 ⇒ √ A ◢
Goto 1

Avec ce programme, l'entrée de la valeur zéro ou d'une valeur supérieure calcule et affiche la racine carrée de la valeur entrée. L'entrée d'une valeur inférieure à zéro ramène au message d'entrée sans qu'aucun calcul ne soit effectué.

■ 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 efface les données d'une liste.

Syntaxe: ClrList ↵

Description: Cette commande efface le contenu de la liste actuellement sélectionnée (liste 1 à liste 6) pendant l'exécution d'un programme.

ClrText

Fonction: Cette commande efface le texte de l'écran.

Syntaxe: ClrText ↵

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

■ Commandes d'affichage (DISP)

DispF-Tbl, DispR-Tbl

Fonction: Ces commandes affichent des tables numériques.

Syntaxe:

DispF-Tbl ↵

DispR-Tbl ↵

Description:

1. Ces commandes créent des tables numériques pendant l'exécution d'un programme en fonction des conditions définies dans le programme.
2. DispF-Tbl crée une table de fonctions, tandis que DispR-Tbl crée une table de récurrence.

DrawDyna

Fonction: Cette commande exécute un tracé de graphe dynamique.

Syntaxe: DrawDyna ↵

Description: Cette commande exécute un tracé de graphe dynamique pendant le déroulement d'un programme d'après les conditions de tracé qui ont été définies dans le programme.

DrawFTG-Con, DrawFTG-Plt

Fonction: Ces commandes représentent graphiquement des fonctions.

Syntaxe:

DrawFTG-Con ↵

DrawFTG-Plt ↵

Description:

1. Ces commandes représentent graphiquement des fonctions d'après les conditions qui ont été définies dans le programme.
2. DrawFTG-Con produit un graphe à points connectés tandis que DrawFTG-Plt produit un graphe à points séparés.

DrawGraph

Fonction: Cette commande trace un graphe.

Syntaxe: DrawGraph ↵

Description: Cette commande trace un graphe d'après les conditions qui ont été définies dans le programme.

DrawR-Con, DrawR-Plt

Fonction: Ces commandes représentent graphiquement des expressions récurrentes avec $a_n(b_n)$ comme axe vertical et n comme axe horizontal.

Syntaxe:

DrawR-Con ↵

DrawR-Plt ↵

Description:

1. Ces commandes représentent graphiquement des expressions récurrentes avec $a_n(b_n)$ comme axe vertical et n comme axe horizontal d'après les conditions qui ont été définies dans le programme.
2. DrawR-Con produit un graphe à points connectés tandis que DrawR-Plt produit un graphe à points séparés.

DrawRΣ-Con, DrawRΣ-Plt

Fonction: Ces commandes représentent graphiquement des expressions récurrentes avec $\Sigma a_n(\Sigma b_n)$ comme axe vertical et n comme axe horizontal.

Syntaxe:

DrawRΣ-Con ↵

DrawRΣ-Plt ↵

Description:

1. Ces commandes représentent graphiquement des expressions récurrentes avec $\Sigma a_n(\Sigma b_n)$ comme axe vertical et n comme axe horizontal d'après les conditions qui ont été définies dans le programme.
2. DrawRΣ-Con produit un graphe à points connectés tandis que DrawRΣ-Plt produit un graphe à points séparés.

DrawStat

Fonction: Cette commande trace un graphe statistique.

Syntaxe:

DrawStat ↵

Description:

Cette commande trace un graphe statistique d'après les conditions qui ont été définies dans le programme.

DrawWeb

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

Syntaxe: DrawWeb [nom de l'expression récurrente], [nombre de lignes]↵

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

Description:

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

■ Commandes d'entrée/sortie (I/O)

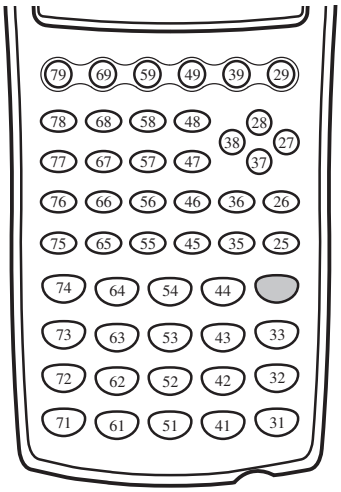
Getkey

Fonction: Cette commande se comporte comme une variable qui prend la valeur correspondant au code de la dernière touche activée.

Syntaxe: Getkey↵

Exemple: Se brancher sur les Lbl 1, Lbl 2 ou Lbl 3, dans une boucle en appuyant sur les touches 1, 2 ou 3

```
Lbl 0
Getkey = 72 => Goto 1
Getkey = 62 => Goto 2
Getkey = 52 => Goto 3
Goto 0
```



- La boucle tournera sur elle-même tant qu'il n'y aura pas d'appui sur une touche.

Locate

Fonction: Cette commande affiche des caractères alphanumériques à un endroit particulier de l'écran de texte.

Syntaxe:

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

Locate <numéro de colonne>, <numéro de ligne>, <nom de variable>

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: valeur numérique
- Nom de variable: A à Z
- Chaîne: chaîne de caractères

Description:

1. Cette commande affiche des valeurs (contenu des variables compris) ou un texte à un endroit particulier de l'écran.
2. 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 CFX"
Ce programme affiche le texte "CASIO CFX" au centre de l'écran.

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

■ Commandes entrées/sorties avec un analyseur (CASIO Data Analyzer)

Receive (

Fonction: Cette commande reçoit les données d'un analyseur (CASIO Data Analyzer).

Syntaxe: Receive (<données>)

Description:

1. Cette commande reçoit les données d'un analyseur (CASIO Data Analyzer).
2. Les données qui peuvent être reçues d'un analyseur (CASIO Data Analyzer) en utilisant cette commande sont les suivantes:
   - Valeurs individuelles affectées aux variables
   - Données matricielles (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)
   - Données graphiques

Send (

Fonction: Cette commande envoie des données à un analyseur (CASIO Data Analyzer).

Syntaxe: Send (<données>)

Description:

1. Cette commande envoie des données à un analyseur (CASIO Data Analyzer).
2. Les données suivantes peuvent être envoyées au moyen de cette commande.
   - Valeurs individuelles affectées aux variables
   - Données matricielles (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)

■ Opérateurs relationnels avec saut conditionnel (REL)

=, ≠, >, <, ≥, ≤

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

Syntaxe:

<côté gauche> <opérateur relationnel> <côté droit> ⇒ <instruction> { ↵ / : / ◢ } <instruction>

Paramètres:

Côté gauche/côté droit:variable (A à Z, r, θ), constante numérique, expression avec variable (comme: A □ 2)

opérateur relationnel: =, ≠, >, <, ≥, ≤

Description:

1. Les six opérateurs relationnels suivants peuvent être utilisés dans la commande de saut conditionnel.

   <côté gauche> = <côté droit> : vrai quand <côté gauche> est égal à <côté droit>

   <côté gauche> ≠ <côté droit> : vrai quand <côté gauche> n'est pas égal à <côté droit>

   <côté gauche> > <côté droit> : vrai quand <côté gauche> est plus grand que <côté droit>

   <côté gauche> < <côté droit> : vrai quand <côté gauche> est plus petit que <côté droit>

   <côté gauche> ≥ <côté droit> : vrai quand <côté gauche> est plus grand que ou égal à <côté droit>

   <côté gauche> ≤ <côté droit> : vrai quand <côté gauche> est plus petit que ou égal à <côté droit>

P.381

2. Voir "⇒ (Code de saut)" pour savoir comment utiliser le saut conditionnel.

20-12 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**
? → X	?
"X =" ? → X	X = ?

- Si le texte est suivi d'une formule de calcul, n'oubliez pas d'insérer une commande d'affichage (◢), un retour à la ligne (↵) ou une commande d'instructions multiples (:) entre le texte et le calcul.
- Si plus de 21 caractères sont entrés, le texte passe automatiquement à la ligne suivante. L'écran défile automatiquement lorsque le texte remplit tout l'écran.

20-13 Utilisation des fonctions de la calculatrice dans un programme

P.80

■ 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 type de programme, veillez à utiliser le **mode MAT** pour entrer la matrice, puis passez dans le **mode PRGM** pour introduire le programme.
- Appuyez sur F4 (MENU) F2 (MAT).

●Pour échanger le contenu de deux lignes (Swap)

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

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

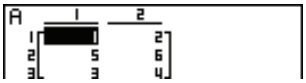
La syntaxe utilisée pour ce programme est la suivante.

Swap A, 2, 3

Nom de la matrice

L'exécution de ce programme produit le résultat suivant.

(Mode MAT)



●Pour calculer un produit scalaire (∗Row)

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

La syntaxe utilisée pour ce programme est la suivante.

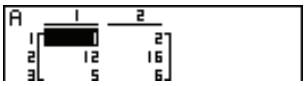
∗Row 4, A, 2

Nom de la matrice

Multiplicateur

L'exécution de ce programme produit le résultat suivant.

(Mode MAT)



●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, en le multipliant par 4, et ajouter le résultat à la ligne 3**

La syntaxe utilisée pour ce programme est la suivante.

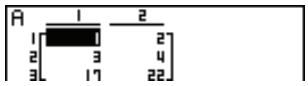
∗Row+ 4, A, 2, 3

Nom de la matrice

Multiplicateur

L'exécution de ce programme produit le résultat suivant.

(Mode MAT)



●Pour additionner deux lignes (Row+)

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

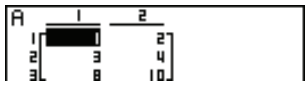
La syntaxe utilisée pour ce programme est la suivante.

Row+ A, 2, 3

Nom de la matrice

L'exécution de ce programme produit le résultat suivant.

(Mode MAT)



■ Utilisation de fonctions graphiques dans un programme

P.112

• MENU **PRGM** EXE F4 (MENU) F4 (GRPH)

Vous pouvez intégrer des fonctions graphiques dans un programme pour tracer des graphes complexes, 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 ↵

• Entrée de la fonction graphique

Y = Type ↵ ..... Définit le type de graphe.

"X^2 – 3" → Y1 ↵

• Tracé de graphe

DrawGraph ↵

• Les commandes soulignées sont obtenues par l'appui sur les touches suivant le numéro correspondant, par exemple ②.

Exemple de programme

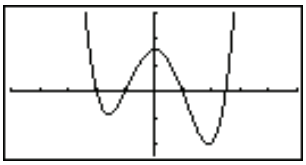
① ClrGraph ↵ ① SHIFT PRGM F6 F1 F2

② View Window –10, 10, 2, –120, 150, 50 ↵ ② SHIFT F3 F1 EXIT

③Y = Type ↵	③ F4 F4 F3 F1
"X ^ 4−X ^ 3−24X² + 4X + 80" → Y1 ↵ (④)	④ VARS F4 F1 EXIT EXIT
⑤G SelOn 1 ↵	⑤ F4 F4 F1 F1 EXIT
⑥Orange G1 ↵	⑥ F4 F2
⑦DrawGraph	⑦ SHIFT PRGM F6 F2 F2

couleur

L'exécution du programme produit le résultat indiqué ici.



■ Utilisation des fonctions de graphe dynamique dans un programme

P.182

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.

• MENU **PRGM** EXE F4 (MENU) F5 (DYNA)

• Plage du graphe dynamique

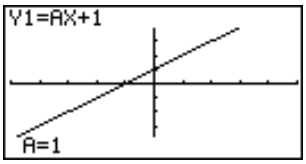
1 → D Start ↵
5 → D End ↵
1 → D pitch ↵

• Les commandes soulignées sont obtenues par l'appui sur les touches suivant le numéro correspondant, par exemple ②.

Exemple du programme

ClrGraph ↵	
View Window −5, 5, 1, −5, 5, 1 ↵	
Y = Type ↵	
"AX + 1" → Y1 ↵ (①)	① VARS F4 F1 EXIT EXIT
②D SelOn 1 ↵	② F4 F5 F1
③D Var A ↵	③ F3
1 → ④D Start ↵	④ VARS F5 F1
5 → ⑤D End ↵	⑤ F2
1 → ⑥D pitch ↵	⑥ F3
⑦DrawDyna	⑦ SHIFT PRGM F6 F2 F3

L'exécution du programme produit le résultat indiqué ici.



■ Utilisation des fonctions de table et graphe dans un programme

• MENU **PRGM** EXE F4 (MENU) F4 (GRPH)

P.206

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 de la plage de la table

1 → F Start ↵

5 → F End ↵

1 → F pitch ↵

• Génération d'une table numérique

DispF-Tbl ↵

• Tracé de graphe

Graphe à points connectés: DrawFTG-Con ↵

Graphe à points séparés: DrawFTG-Plt ↵

Exemple de programme

ClrGraph ↵

ClrText ↵

View Window 0, 6, 1, –2, 106, 2 ↵

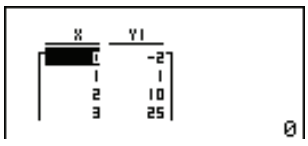
Y = Type ↵

"$3X^2 - 2$" → Y1 ↵

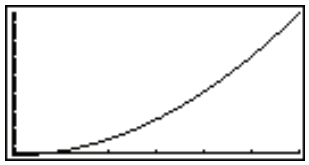
① T SelOn 1 ↵	① F4 F6 F1 F1
0 → ② F Start ↵	② VARS F6 F1 F1
6 → ③ F End ↵	③ F2
1 → ④ F pitch ↵	④ F3
⑤ DispF-Tbl ◢	⑤ SHIFT PRGM F6 F2 F4 F1
⑥ DrawFTG-Con	⑥ SHIFT PRGM F6 F2 F4 F2

L'exécution du programme produit le résultat indiqué ici.

Table numérique



Graphe



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

• MENU **PRGM** EXE F4 (MENU) F6 (▷) F2 (RECR)

P.218

L'intégration de fonctions de table et graphe de récurrence 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 de récurrence sont les suivantes.

• Entrée de la formule de récurrence

a_{n+1} Type ↵ .... Définit 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 ↵

• Tracé de graphe

Graphe à points connectés: DrawR-Con ↵, DrawRΣ-Con ↵

Graphe à points séparés: DrawR-Plt ↵, DrawRΣ-Plt ↵

• Graphe statistique de convergence/divergence (graphe WEB)

DrawWeb a_{n+1}, 10 ↵

Exemple de programme

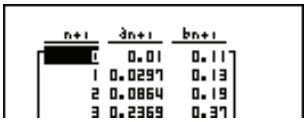
ClrGraph ↵

View Window 0, 1, 1, 0, 1, 1 ↵

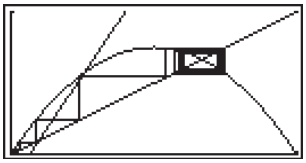
①a_{n+1} Type ↵	① F4 F6 F2 F3 F2 EXIT
"−3②a_n^2 + 3a_n" → a_{n+1} ↵	② F4 F2
"3b_n − 0.2" → b_{n+1} ↵	
0 → ③R Start ↵	③ VARS F6 F2 F2 F1
6 → R End ↵	
0.01 → a_0 ↵	
0.11 → b_0 ↵	
0.01 → a_n Start ↵	
0.11 → b_n Start ↵	
④DispR-Tbl ◢	④ SHIFT PRGM F6 F2 F5 F1
⑤DrawWeb ⑥a_{n+1}, 30	⑤ SHIFT PRGM F6 F2 F5 F2 EXIT EXIT EXIT
	⑥ F4 F6 F2 F4 F3

L'exécution du programme produit le résultat indiqué ici.

Table numérique



Graphe de récurrence



■ Utilisation des fonctions de classement de listes dans un programme

P.234

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

• Ordre ascendant

① ②
SortA (List 1, List 2, List 3)

Listes à classer (six listes au maximum)

① F4 F3 F1 EXIT ② OPTN F1 F1

• Ordre descendant

SortD (List 1, List 2, List 3)

Listes à classer (six listes au maximum)

■ Utilisation de la fonction de résolution dans un programme

Vous pouvez incorporer une fonction de résolution dans un programme.

La syntaxe requise pour l'utilisation de la fonction de résolution dans un programme est la suivante.

Solve ($f(x)$, n, a, b)

Limite supérieure
Limite inférieure
Valeur initiale estimée

Exemple de programme

① Solve ($2X^2 + 7X - 9$, 1, 0, 1) ① OPTN F4 F1

- Dans la fonction $f(x)$, seul X peut être utilisé comme variable dans les expressions. Les autres variables (A à Z, r, θ) sont traitées comme constantes, et la valeur actuellement affectée à la variable est appliquée pendant le calcul.
- L'entrée de la fermeture de parenthèses, de la limite inférieure a et de la limite supérieure b peut être omise.
- Les solutions obtenues lorsqu'on utilise la résolution peuvent contenir des erreurs.
- Notez que vous ne pouvez pas utiliser une valeur de résolution, différentielle, différentielle quadratique, intégration, valeur maximale/minimale ou une expression de calcul de Σ dans un terme du calcul avec résolution.

■ Utilisation de calculs et de graphes statistiques dans un programme

P.250

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 "StatGraph", vous devez définir les conditions suivantes:

- Statut 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 données de fréquence (nom de liste)
- Type de point
- Couleur du graphe

couleur

P.252

Les conditions de tracé du graphe dépendent du type de graphe. Voir "Changement des paramètres d'un graphe".

- La définition caractéristique pour un diagramme de dispersion ou un graphe linéaire xy est la suivante.

  S-Gph1 DrawOn, Scatter, List1, List2, 1, Square, Blue↵

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

- La définition caractéristique d'un graphe pour le marquage de probabilité normale est la suivante.

  S-Gph1 DrawOn, NPPlot, List1, Square, Blue↵

- La définition caractéristique d'un graphe à variable unique est la suivante.

  S-Gph1 DrawOn, Hist, List1, List2, Blue↵

  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.

P.254

  Histogramme: ..... Hist

  Boîte-médiane: ..... MedBox

  Boîte-moyenne: ..... MeanBox

  Distribution normale: ..... N-Dist

  Ligne brisée: ..... Broken

P.254

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

S-Gph1 DrawOn, Linear, List1, List2, List3, Blue ↵

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
Med-Med: ..... Med-Med
Régression quadratique: ..... Quad
Régression cubique: ..... Cubic
Régression quartique : ..... Quart
Régression logarithmique: .... Log
Régression exponentielle: .... Exp
Régression de puissance : ... Power

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

S-Gph1 DrawOn, Sinusoidal, List1, List2, Blue ↵

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

S-Gph1 DrawOn, Logistic, List1, List2, Blue ↵

① SHIFT SET UP F6 F6 F3 F1
② OPTN F1 F1
③ F1 EXIT EXIT
④ F4 F1 F2 F1 EXIT
⑤ F1 F1 EXIT
⑥ F2 F4 EXIT
⑦ EXIT F4 F1
⑧ EXIT F5 F1
⑨ SHIFT PRGM F6 F2 F1

Exemple de programme

ClrGraph ↵

① S-Wind Auto ↵

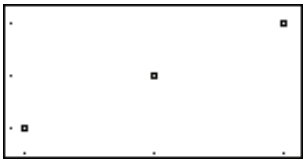
{1, 2, 3} → List 1 ↵ ②

{1, 2, 3} → List 2 ↵ ③

④ S-Gph1 DrawOn ⑤, Scatter ⑥, List1, List2, 1, Square ⑦, Blue ⑧ ↵

⑨ DrawStat

L'exécution de ce programme produit le diagramme de dispersion indiqué ici.



■ Exécution de calculs statistiques

• Calcul statistique à variable unique

①1-Variable List 1, List 2

Données de fréquence (Frequency)

Données de l'axe x (XList)

① F4 F1 F6 F1

```
1-Variable
x̄    =2.33333333
Σx   =14
Σx²  =36
xσn  =0.74535599
xσn-1=0.81649658
n    =6
```

• Calcul statistique à variable double

2-Variable List 1, List 2, List 3

Données de fréquence (Frequency)

Données de l'axe y (YList)

Données de l'axe x (XList)

```
2-Variable
 x̄   =2
 Σx  =6
 Σx² =14
 xσn =0.81649658
 xσn-1=1
 n   =3
```

• Calcul statistique de régression

①LinearReg List 1, List 2, List 3

*Type de calcul**

Données de fréquence (Frequency)

Données de l'axe y (YList)

Données de l'axe x (XList)

① F4 F1 F6 F6 F1

```
LinearReg
   a =1
   b =0
   r =1
   r²=1
y=ax+b
```

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

LinearReg ..... régression linéaire

Med-MedLine . calcul Med-Med

QuadReg..... régression quadratique

CubicReg ..... régression cubique

QuartReg ..... régression quartique

LogReg ..... régression logarithmique

ExpReg ..... régression exponentielle

PowerReg ..... régression de puissance

• Calcul statistique de régression sinusoïdale

SinReg List 1, List 2

Données de l'axe y (YList)

Données de l'axe x (XList)

• Calcul statistique de régression logistique

LogisticReg List 1, List 2

Données de l'axe y (YList)

Données de l'axe x (XList)

■ Création d'une liste indicée

Vous pouvez réaliser une liste indicée en utilisant la fonction Seq qui pourra créer une liste de D éléments.

Exemple **Constituer une liste de variables indicées**

Les modèles de calculatrices CASIO ne disposant pas de la fonction List pouvaient utiliser des variables indicées du type Z [I].

Nous allons comparer 2 programmes permettant de constituer une liste de D variables indicées.

Dans le programme "ancien", la variable indicée est Z [I].

Dans le programme "nouveau", la variable indicée est List1 [I].

Manuellement
Defm D
0 → I
Lbl 1
1 + I → I
"Val"? → Z [I]
I<D → Goto 1
End

⇒

"Dim"? → D Seq (0, X, 1, D, 1) → List1 ou bien D → Dim List 1	
Lbl 1	For 1 → I To D
1 + I → I	
"Val"? → List1 [I]	"Val"? → List1 [I]
I<D → Goto 1	
End	Next