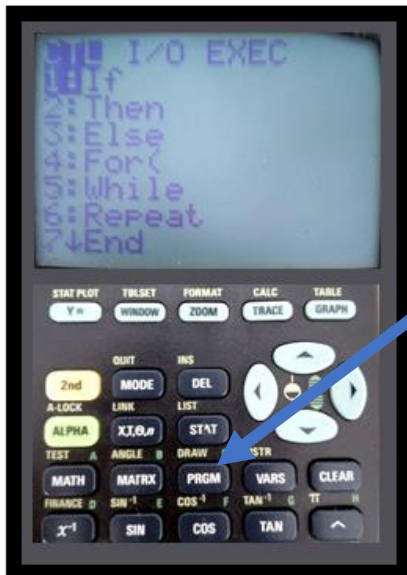


Programmer en *TI-Basic*, langage des calculatrices TI

1) Comment accéder aux fonctions de programmer



Pour utiliser des fonctions pour programmer ou des mots clefs, il faut (pendant l'édition d'un programme) appuyer le bouton **PRGM** et on aura alors accès à un menu avec 3 pages :

- **CTL** (« Control ») : Pour les boucles, les conditions et le mot clef End
- **I/O** (« Input/Output ») : pour demander des valeurs ou pour les montrer
- **EXEC** (« Execute ») : on ne l'utilisera pas

2) Programmer

Imprimer le tableau suivant (*page 1*) pour vous aider à programmer en faisant des exercices et à apprendre le langage de la calculatrice (imprimer page suivante).

3) S'exercer

Essayer de traduire les algorithmes de la *page 2* dans votre calculatrice pour apprendre le langage. Certains entre eux sont pris de sujet du bac.

Description	Dans le programme	Exemple
Demander une valeur	Prompt <i>En I/O</i>	: Prompt A : Prompt B, C
Afficher une valeur	Disp <i>En I/O</i>	: Disp «Hello World» : Disp A
Donner une valeur à une variable	Valeur → Variable <i>Touche : STO→</i>	: 3→A : B^2-4*A*C→D
Instruction conditionnelle simple « Si » <i>Si condition →Executer ligne suivante si condition vraie</i>	If *condition* <i>*Ligne à réaliser si condition vrai*</i> <i>En CTL</i>	: 2→A : If A>2 : Disp A : 4→A : If A<4 : Disp 1
Instruction conditionnelle « Si Alors Sinon » <i>Si condition Alors →Executer ligne suivante si condition vraie Sinon →Executer ligne suivante si condition fausse Fin du Si</i>	If *condition* Then <i>*Ligne à réaliser si condition vrai*</i> Else <i>*Ligne à réaliser si condition fausse*</i> End <i>En CTL</i>	: If A > 0 : Then : Disp «POSITIF» : Else : Disp «PAS POSITIF» : End
Boucle conditionnelle « Tant que » (<i>plus utilisé au bac</i>) <i>Tant que condition →Executer ligne suivante tant que la condition est vraie Fin du Tant que</i>	While *condition* <i>*Ligne à réaliser tant que condition vrai*</i> End <i>En CTL</i>	: 4→A : While A > 0 : Disp A : A-1→A : End Résultat: 4 3 2 1
Boucle inconditionnelle « Pour » (<i>moins utilisé au bac</i>) <i>Pour variable allant de n_1 à n_2 →Executer ligne suivante répétitivement et à chaque fois augmenter/diminuer de 1 la variable jusqu'à atteindre n_2 Fin du Pour</i>	For(*variable*, *n_1*, *n_2*) <i>*Ligne à réaliser pour la variable allant de n_1 à n_2*</i> End <i>En CTL</i>	: 10→N : For(I,0,N) : Disp I : End Résultat: 1 2 ... 9 10

Initialisation:	U prend la valeur 1000 I prend la valeur 0
Entrée:	Demander N
Traitement:	Tant que $I < N$ U prend la valeur $0,95 \times U$ I prend la valeur $I + 1$ Fin du Tant que
Sortie:	Afficher U

Variables :	<i>k</i> et <i>p</i> sont des entiers naturels <i>u</i> est un réel
Entrée :	Demander la valeur de <i>p</i>
Traitement :	Affecter à <i>u</i> la valeur 5 Pour <i>k</i> variant de 1 à <i>p</i> Affecter à <i>u</i> la valeur $0,5u + 0,5(k - 1) - 1,5$ Fin de pour
Sortie :	Afficher <i>u</i>

Variables:	i, N, A : nombres
Entrée:	Saisir la valeur de N
Initialisation:	Affecter à i la valeur 0 Affecter à A la valeur 25
Traitement:	Tant que $i < N$ Affecter à i la valeur de $i + 1$ Affecter à A la valeur de $1,05 \times A - 0,1$ Fin Tant que
Sortie:	Afficher A

Entrée:	Saisir la valeur de n
Traitement:	Si la partie décimale de $n/2$ est nulle [Math>Num>fpart(/] Alors Afficher « n est pair » Sinon Afficher « n est impair » Fin du Si