Premiers programmes avec la calculatrice Graph 90+E

1 Prise en main de l'environnement

— **Entrer dans le module Khicas** :
Allumer la calculatrice puis choisir **Khicas** dans le menu principal.

— **Choisir le langage de programmation Python :**
Cliquer sur **F6 (Fich,Cfg)** puis descendre avec les flèches au **choix 8 Python**. Si la case est cochée, le langage utilisé est déja Python, cliquer donc sur **EXIT** pour revenir à la page initiale de Khicas. Si la case n'est pas cochée, cliquer sur **EXE** puis **F1**. Recliquer sur **F6 (Fich,Cfg)** pour vérifier que le **choix 8 Python** est maintenant coché.

— **Créer un ficher Tortue nommé test.py :**
Cliquer sur **F6(Fich,Cfg)**, sélectionner avec les flèches **choix 1 nouveau script** puis cliquer sur **EXE**, entrer le nom **test**, cliquer sur **F1** (tortue). Un fichier vient d'être créé, il contient pour le moment une seule ligne *efface ;*.

— **Exécuter le fichier :**
Cliquer sur **F6(Fich,Cfg)**, sélectionner avec les flèches **choix 1 tester syntaxe** puis cliquer sur **EXE**. Soit un message indique qu'il y a une **erreur de syntaxe** à corriger, soit le programme s'exécute et le résultat est affiché. Cliquer alors sur **EXIT** pour revenir au programme.

— **Créer un premier dessin :**
Depuis le programme, se placer à la ligne qui suit *efface ;*. Cliquer sur **F4 CATALOG**, sélectionner **choix 19 Tortue shift QUIT** puis cliquer sur **F2 EXEMPL1**. Le programme contient maintenant une deuxième ligne *avance 40*. Executer à nouveau le programme pour voir l'effet de cette commande sur la Tortue. Tester d'autres commandes à partir du catalogue (par exemple **F4 CATALOG** puis **choix 19 Tortue shift QUIT** puis **choix 9 pas de cote**).

— **Revenir à la page initiale de Khicas :**
Depuis le programme, cliquer sur **F6(Fich,Cfg)** puis sélectionner **choix 9 Quitter** et enfin cliquer sur **F1** pour sauvegarder.

— **Reouvrir le fichier test.py :**
Depuis la page initiale de Khicas, cliquer sur **F6(Fich,Cfg)** puis sélectionner **choix 2 Editer script** et sélectionner **test.py**.

2 Mise en pratique

2.1 Exercice 1

Créer un fichier **tracergrille.py** qui trace le quadrillage d'une grille rectangulaire de taille $m \times n$. A noter que pour un meilleur affichage, chaque case sera un carré de coté $u = 20$ (et non un carré de coté $u = 1$).

Indication 1 : Une possibilité est de tracer toutes les lignes horizontales du quadrillage puis toutes les lignes verticales.

Indication 2 : Pour répéter plusieurs fois une ou plusieurs commandes, on utilisera une boucle **for** (disponible dans **F4 CATALOG** puis **choix 18 Programmes PRGM** puis **choix 1 boucle for**) qui fonctionne avec **in range** (disponible dans **F4 CATALOG** puis **choix 18 Programmes PRGM** puis **choix 8 range (a,b)**). Par exemple, tester le code suivant :

```
for ind in range(0,4) :
    avance 100
    recule 100
    pas_de_cote 10
```

2.2 Exercice 2

Ajouter au quadrillage précédent un trou en coloriant en noir une case désignée par le numéro de sa ligne et de sa colonne (numérotation commençant à 0).

Indication : On utilisera la commande Tortue **rectangle_plein**.

2.3 Exercice 3

1) Créer un fichier **tracerpolygones.py** qui trace un polygone régulier à k cotés (avec k un entier quelconque supérieur ou égal à 3).

Indication : Si $k = 3$, on tracera un triangle équilatéral en faisant 3 fois avancer et pivoter de $\frac{360}{3}^o$ la tortue. Si $k = 4$, on tracera un carré en faisant 4 fois...

2) Modifier le programme **tracerpolygones.py** pour tracer, les uns à coté des autres, les m premiers polygones réguliers à k cotés (où m est un entier donné supérieur ou égal à 1).

Indication 1 : Lorsque $m = 3$, on veut tracer un triangle équilatéral ($k = 3$), un carré ($k = 4$) et un pentagone régulier ($k = 5$). Par extension, donner la plage des entiers que doit parcourir k lorsque m est un entier non nul quelconque.

Indication 2 : Pour l'implémentation, on pourra faire deux boucles **for** imbriquées l'une dans l'autre.