Feuille d'exercices n°0 de l'option d'informatique.

Voici un TD pour réactiver quelques neurones et reprendre quelques habitudes de programmations.

Quelques rappels pour utiliser les graphismes dans Caml.

Taper `#open "graphics";;` pour charger la bibliothèque de fonctions graphiques,
puis `open_graph "";;` pour ouvrir la fenêtre graphique,

Pour la suite, on donne une liste de fonctions graphiques :

- `clear_graph : unit -> unit` pour nettoyer la fenêtre graphique,
- `moveto : int -> int -> unit` pour déplacer le point courant,
- `plot : int -> int -> unit` pour dessiner un point,
- `set_color : color -> unit` pour changer la couleur courante (on dispose de couleurs prédéfinies : black, white, red, green, blue, yellow, cyan, magenta),
- `size_x : unit -> int` et `size_y : unit -> int` retourne la taille de la fenêtre graphique,

S'il vous manque une fonction, vous pouvez toujours aller faire un tour du côté de l'aide

1. Étant donnée une liste l de couples (n, x), écrire une fonction **limite l** qui retourne le quadruplet $(n_{min}, n_{max}, x_{min}, x_{max})$ donnant les valeurs minimales et maximales de n et de x dans la liste l.
Exemple donné avec la fonction de l'exercice 3 :

```
#limite(suite (fun x ->2.85 *. x *. (1.-. x)) 0.5 50 55);;
- : float * float * float * float = 50.0, 55.0, 0.649101931506, 0.64914054996
```

2. Écrire une fonction **trace** qui à partir d'une liste de couples (x_n, y_n) de type `float * float` trace le nuage de points correspondant dans la fenêtre graphique.
Exemple : Voir figure 1

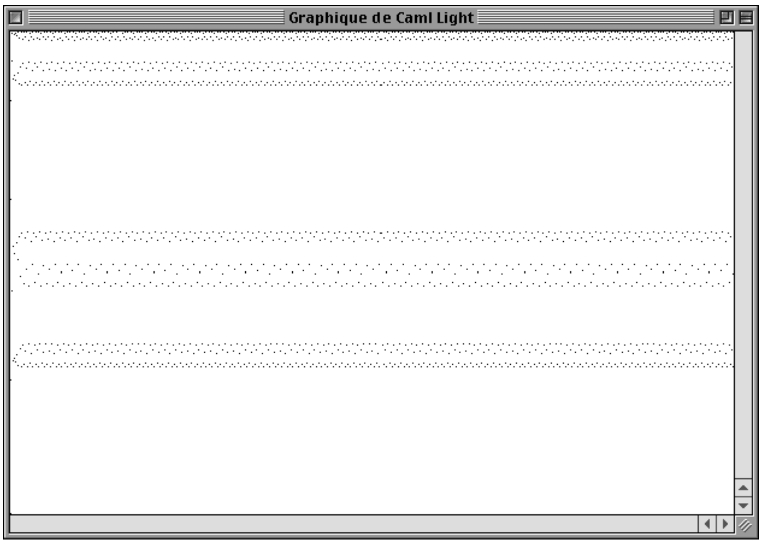

Fig. 1 – *trace(suite (fun x ->3.56995 *. x *. (1.-. x)) 0.1 0 2000);;*

3. On considère une fonction f de $\mathbb{R}$ dans $\mathbb{R}$ et la suite récurrente $(u_n)_{n\in\mathbb{N}}$ définie par

$$u_0 \quad \text{et} \quad \forall n \in \mathbb{N}, u_{n+1} = f(u_n) \quad (*)$$

Écrire une fonction **suite f u0 n m** qui retourne la liste des couples de réels (k, u_k) pour k compris entre n et m (la suite $(u_n)_{n\in\mathbb{N}}$ vérifiant la relation de récurrence $(*)$).
Exemple : `#suite (fun x ->2.85 *. x *. (1.-. x)) 0.5 50 55;;`

```
- : (float * float) list =
[50.0, 0.649101931506; 51.0, 0.64914054996; 52.0, 0.649107724619; 53.0, 0.649135626408;
54.0, 0.649111910067; 55.0, 0.649132069087]
```

4. Partant du segment $[OI]$ qui est l'ordre 0 de la courbe, on remplace ce segment par quatre disposés comme sur le schéma ci-contre. (sur lequel $AB = BC = BD = CD = DE = OI/3$) en faisant coïncider A avec O et E avec I; on a alors la courbe à l'ordre 1. Ainsi, à chaque ordre successif, on obtient une ligne

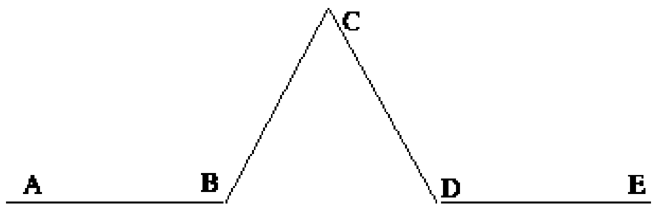

FIG. 2 – *courbe à l'ordre 1*

polygonale commençant en O et se finissant en I, contenant au total $4^n + 1$ points, et l'on obtient l'ordre suivant en remplaçant chaque segment par une reproduction du schéma en envoyant A sur la première extrémité et E sur la seconde. La limite de cette construction est appelé la courbe de Van Koch.
Écrire une fonction qui détermine les coordonnées des trois points intermédiaires B, C et D en fonctions des extrémités A et E.

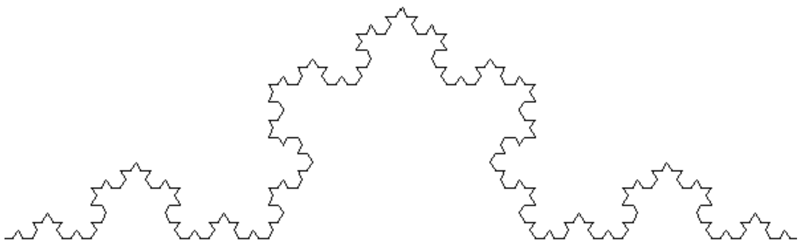

FIG. 3 – *Courbe de Von Koch pour* $n = 4$

Écrire un fonction Calm **Von Koch** qui détermine l'ensemble des sommets de la courbe de Von Koch limité à un ordre n quelconque.
Tracer la courbe obtenue pour $n = 4$ (cf figure 3) (on pourra écrire une fonction **tracebis** qui par rapport à la fonction **trace** précédente, relier les points entre eux et considère une échelle isotrope en utilisant la fonction `lineto x y` qui trace un segment de droite reliant le point courant et le point de coordonnées (x, y). Le point de coordonnées (x, y) dévient alors le nouveau point courant).

5. ([1]) Soit c un nombre complexe, on définit par récurrence une suite de parties de $\mathbb{C}$ par $E_0 = \{0\}$ et pour tout entier n, $E_{n+1} = \{z \in \mathbb{C} / z^2 + c \in E_n\}$.

- démontrer que $E = \bigcup\limits_{n \in \mathbb{N}} E_n$ est inclus dans le disque de centre 0 et de rayon $\dfrac{1}{2} + \sqrt{|c| + \dfrac{1}{4}}$

- écrire une fonction ayant pour paramètre c et N qui trace $\bigcup\limits_{k=0}^{N} E_k$ (on affectera une couleur différente à chaque E_k)

- tracer l'ensemble précédent pour $N = 15$ et $c = -1$.

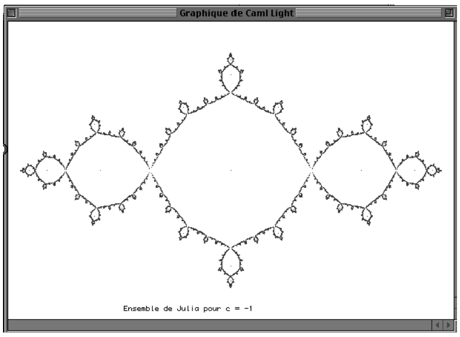

FIG. 4 – *Ensemble de Julia pour $c = -1$ et $n = 15$*

6. Soit c un nombre complexe de module inférieur strict à 1 et, pour chaque $n \in \mathbb{N}$, soit E_n l'ensemble des sommes $\sum\limits_{0 \leqslant k \leqslant n} e_k . c^k$ avec $e_k \in \{0, 1\}$ pour tout k.

- démontrer que la réunion $E = \bigcup\limits_{n \in \mathbb{N}} E_n$ est incluse dans le disque de centre 0 et de rayon $\dfrac{1}{1 - |c|}$.

- programmer la fonction qui donne la représentation graphique de $E_N = \bigcup\limits_{k=0}^{N} E_k$ avec comme paramètre N et c (on changera de couleur pour faire apparaître les $E_{k+1} \setminus E_k$).

- tracer l'ensemble précédent avec $N = 15$ et $c = -1/3 + i/\sqrt{3}$.

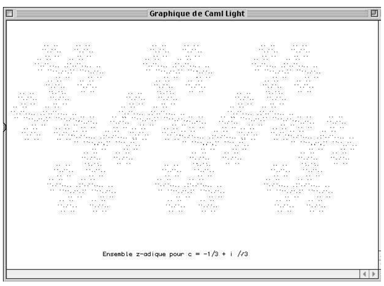

FIG. 5 – *Ensemble des z-adique pour $N = 15$ et $c = -1/3 + i/\sqrt{3}$*

1. Pour cet exercice ainsi que le suivant, on sera amené à choisir une représentation des nombres complexes et à programmer les fonctions d'addition, de soustraction, de multiplication et de racine carrée dans l'ensemble des nombres complexes. On pourra définir un complexe sous la forme `type complexe = {re : float; im : float};;`. La création d'un complexe se fait comme dans l'exemple suivant : `let a = {re = 0.3;im = 3.2};;` et ensuite les parties réelle et imaginaire de `a` sont obtenues avec : `a.re` et `a.im`.