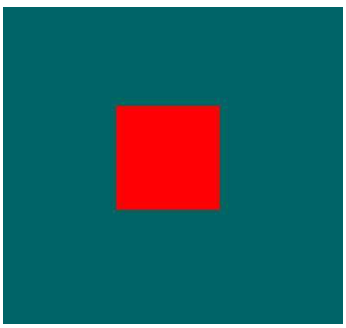
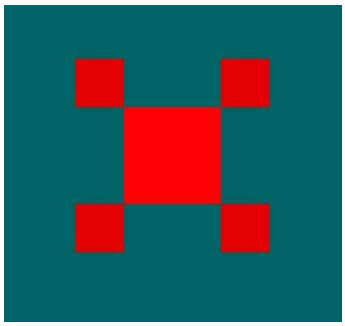
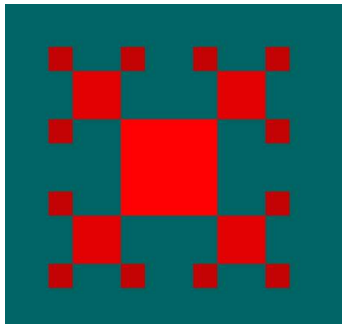
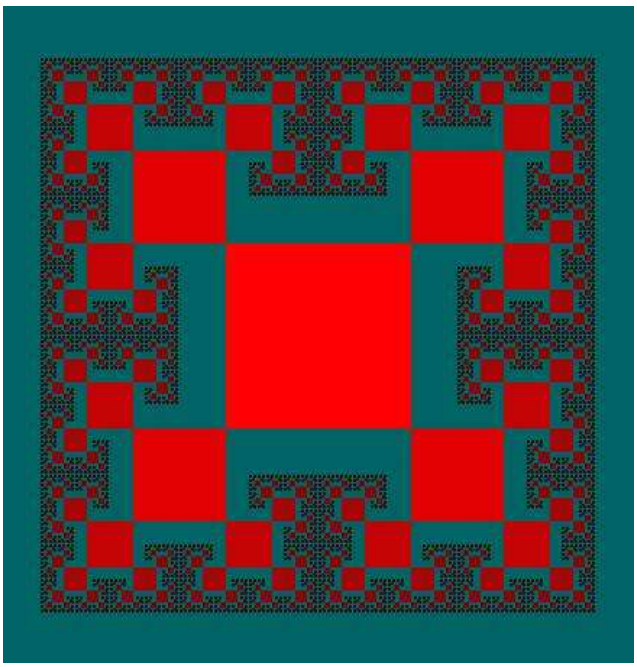
Polder

Imaginons que l'on veuille construire une île artificielle, en procédant de façon mécanique et répétitive. On commence par construire une plateforme carrée de terre. Puis en chaque coin de ce carré qui s'ouvre sur la mer, on bâtit des carrés de côté moitié moindre que le carré précédent, ce qui donne quatre nouveaux carrés accolés au premier. Puis pour chaque carré obtenu, on fait de même en plaçant des carrés de côté moitié en chacun de leurs coins où la mer est présente. Ce procédé est répété un certain nombre de fois, de façon qu'à la fin les carrés de la bordure finale soient quasiment juxtaposés. On obtient ainsi un carré dont le côté est le triple du carré initial, avec en son intérieur des zones d'eau qui restent emprisonnées, et dont on peut attendre qu'elles s'assèchent.




…



étape finale

Quelques considérations théoriques

- Prenons comme unité de longueur le demi-côté du carré initial, et projetons tous les carrés sur une droite horizontale. En allant vers la droite à partir du centre du carré, on obtient une longueur finale égale à 1 + 1 + 1/2 + 1/4 + 1/8 + … = 3. A partir du carré initial de longueur 2 on arrive à un carré final de longueur 6

- En descendant étage de carrés identiques par étage de carrés, à partir du carré central, on trouve deux carrés de côté moitié au-dessous, puis quatre carrés au-dessous, etc. Ces carrés occupent un tiers de la longueur totale, et il en est ainsi à chaque descente. Si les trous occupent toujours les deux-tiers de la longueur 6, chacun a une longueur qui tend vers 0. D'où à la fin une poussière de points dense sur la bordure finale avec une densité de 1/3.
- La zone marine emprisonnée dans le carré final occupe les 4/9 de la surface totale. Pour obtenir ce résultat, il suffit de chercher la surface occupée par les carrés de terre, soit : $4 + 4 . 1 + 4 . 3 . (1/4) + 4 . 3 . 3. (1/16) + \ldots$

$= 4 + 4 (1 + 3/4 + (3/4)^2 + (3/4)^3 + \ldots) = 4 + 4 / (1 – 3/4) = 20,$

la proportion des terres est égale à 20 / 36 = 5 / 9.

Programme

Dans le repère de l'écran, un carré est defini par les coordonnées *x*, *y* de son centre et par la longueur *c* de son demi-côté. Le programme principal se contente d'appeler la fonction récursive *carre*(*x, y, c, n*) avec ici le point (*x,y*) au centre de l'écran, ainsi que *c* = 64 et *n* = 0, *n* étant associé au nombre de rappels récursifs de la fonction. Cette fonction, qui agit sous réserve que *n*<8, commence par dessiner le carré concerné puis se rappelle quatre fois aux coins de ce carré, sous réserve que ces futurs carrés aient leur centre dans la mer, et cela avec l'indice de récursion augmenté de 1. Les nouveaux carrés sont pour centre $x \pm 3c/2$, $y \pm 3c/2$. Pour éviter le cumul de petites erreurs d'arrondi, le demi-côté *c* est mis en flottants, Le carré a son coin en haut à gauche défini par ses coordonnées :

position.x = *x* - (int)(*c*+0.5), *position.y* = *y* - (int)(*c*+0.5)

où l'on a pris l'entier le plus proche de *x* – *c*. On lui donne une couleur rouge en dégradé au fil des récursions, puis on créé sa surface, on la remplit avec cette couleur, et l'on fait le *blit* indispensable pour pouvoir l'afficher par dessus la fenêtre de l'écran elle-même coloriée en bleu au départ.

```
#include <SDL/SDL.h>
void pause();  /* fonction déjà utilisée, à récupérer, ainsi que la suivante */
Uint32 getpixel(int xe, int ye);
void carre(int x, int y, float c, int n);
SDL_Surface *ecran, * ccarre;   Uint32 bleu,vert;
int main (int argc, char ** argv )
{
   SDL_Init(SDL_INIT_VIDEO);
   ecran=SDL_SetVideoMode(800, 600, 32,SDL_HWSURFACE);
   bleu=SDL_MapRGB(ecran->format,0,100,100); /* couleur de la mer */
   SDL_FillRect(ecran,NULL,bleu); /* coloriage de l'écran en bleu */
   carre(800/2, 600/2, 64.,0); /* la fonction recursive essentielle */
   SDL_Flip(ecran); pause(); return 0;
}
void carre(int x, int y, float c, int n)
{ SDL_Rect position;   Uint32 couleur;
  if (n<8)
   {
      position.x= x-(int)(c+0.5); position.y=y-(int)(c+0.5);
      couleur=SDL_MapRGB(ecran->format,255-30*n,2*n,2*n);
```

```
        ccarre=SDL_CreateRGBSurface(SDL_HWSURFACE,2*(int)(c+0.5),
                                                2*(int)(c+0.5),32,0,0,0,0);
        SDL_FillRect(ccarre,NULL,couleur);
        SDL_BlitSurface(ccarre,NULL,ecran,&position);
        if (getpixel(x-3.*c/2.,y-3.*c/2.)==bleu)  carre(x-3.*c/2.,y-3.*c/2.,c/2.,n+1);
         if (getpixel(x-3.*c/2.,y+3.*c/2.)==bleu)  carre(x-3.*c/2.,y+3.*c/2.,c/2.,n+1);
         if (getpixel(x+3.*c/2.,y-3.*c/2.)==bleu) carre(x+3.*c/2.,y-3.*c/2,c/2.,n+1);
         if(getpixel(x+3.*c/2.,y+3.*c/2.)==bleu) carre(x+3.*c/2.,y+3.*c/2.,c/2.,n+1);
    }
}
```