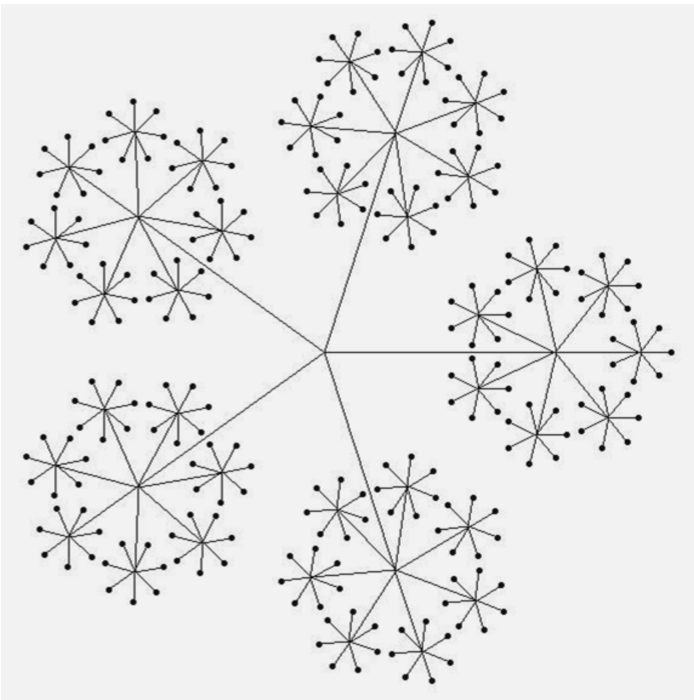
Dessin d'un arbre étoilé

On veut fabriquer un arbre présentant les régularités suivantes : à partir de la racine située à l'étage 0 partent n_0 branches, de chaque nœud de l'étage 1 partent n_1 branches, de chaque nœud de l'étage 2 partent n_2 branches aboutissant à n_2 nœuds terminaux. A cause des dimensions limitées de l'écran, on n'ira pas plus loin.

1) Combien cet arbre possède-il de nœuds terminaux ?

Il y a n_0 nœuds à l'étage 1, puis à chaque fois n_1 nœuds à l'étage 2, et enfin à chaque fois n_2 nœuds à l'étage 3, soit au total $n_0\, n_1\, n_2$ nœuds terminaux.

2) Programmer le dessin de cet arbre sous forme circulaire, les successeurs de chaque nœud étant disposés régulièrement sur un cercle, comme ci-dessous pour $n_0 = 5$, $n_1 = 7$, $n_2 = 7$:



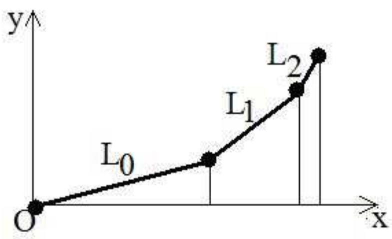
On va numéroter chaque nœud terminal par un mot de longueur 3, de la forme $a_0a_1a_2$, avec $0 \le a_0 < n_0$, $0 \le a_1 < n_1$, $0 \le a_2 < n_2$. En effet, les branches issues de la racine sont numérotées de 0 à n_0 - 1 en tournant dans le sens inverse des aiguilles d'une montre, puis celles issues de chaque nœud de l'étage 1 sont numérotées de 0 à $n_1 - 1$ en partant de la direction donnée par la branche provenant du nœud précédent, etc. On se donne aussi les longueurs L_0, L_1 et L_2 des branches issues des étages successifs, en les choisissant de plus en plus petites de façon que les étoiles finales ne s'entremêlent pas.

Les coordonnées d'un nœud terminal noté $a_0a_1a_2$, c'est-à-dire de numéro :
$n_1n_2\, a_0 + n_2\, a_1 + a_2$, sont alors :

$$\begin{cases} x = L_0 \cos\alpha_0 + L_1 \cos\alpha_1 + L_2 \cos\alpha_2 \\ y = L_0 \sin\alpha_0 + L_1 \sin\alpha_1 + L_2 \sin\alpha_2 \end{cases}$$

avec

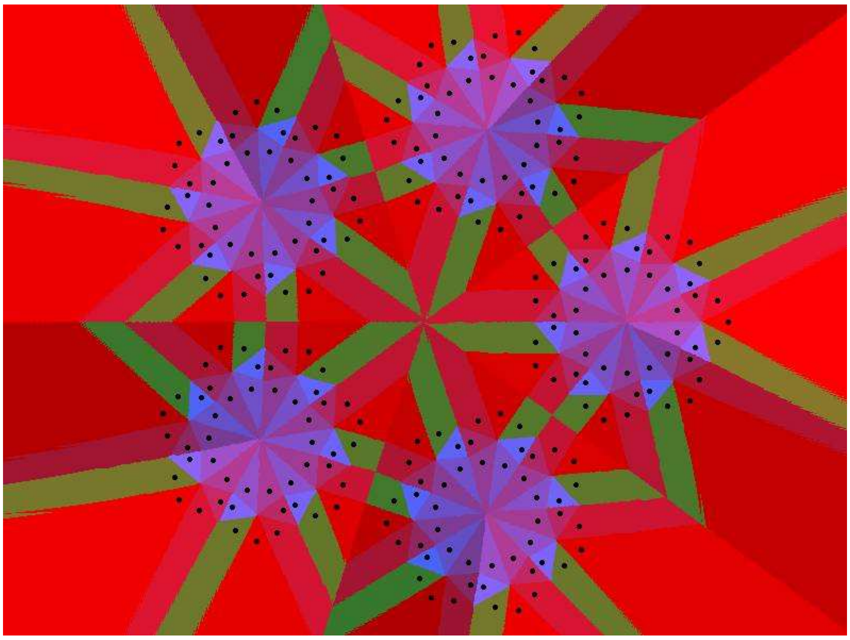
$$\alpha_0 = a_0 \frac{2\pi}{n_0},\ \alpha_1 = \alpha_0 + a_1 \frac{2\pi}{n_1},\ \alpha_2 = \alpha_1 + a_2 \frac{2\pi}{n_2}.$$



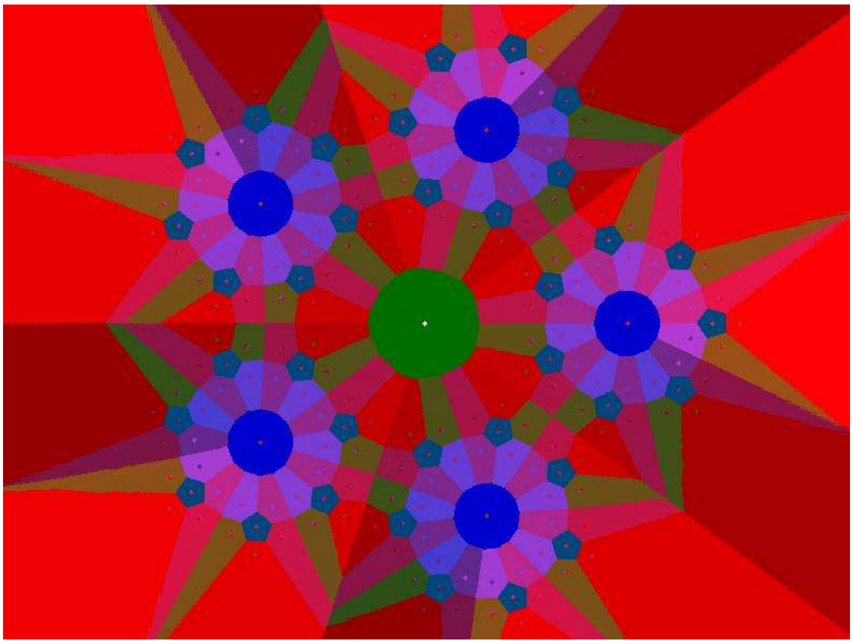
Le programme en découle.

```
#define deuxpi 6.28318
#define xorig 400
#define yorig 300

int xc[1000], yc[1000],a[3], L[3],n[3],numero;
float angle[3];
int N; Uint32 color[1000];
            SDL_Surface * ecran; Uint32 rouge,blanc,noir;

int main(int argc, char ** argv)
{   int i, j,k;
   SDL_Init(SDL_INIT_VIDEO);
   ecran=SDL_SetVideoMode(800,600,32, SDL_HWSURFACE|SDL_DOUBLEBUF);
   blanc=SDL_MapRGB(ecran->format,255,255,255);
   noir=SDL_MapRGB(ecran->format,0,0,0);
   SDL_FillRect(ecran,0,blanc);

   n[0]=5; n[1]=7; n[2]=7;
   L[0]=190; L[1]=70; L[2]= 25;

   for(i=0;i<n[0]; i++) for(j=0; j<n[1]; j++) for(k=0;k<n[2]; k++)
     { a[0]=i; a[1]=j; a[2]=k;
       numero=i*n[1]*n[2]+j*n[2]+k;
       angle[0]=(float)a[0]*deuxpi/(float)n[0];
       angle[1]=angle[0]+(float)a[1]*deuxpi/(float)n[1];
       angle[2]=angle[1]+(float)a[2]*deuxpi/(float)n[2];
       xc[numero]=xorig+(float)L[0]*cos(angle[0])+(float)L[1]*cos(angle[1])
                        +(float)L[2]*cos(angle[2]);
       yc[numero]=yorig-(float)L[0]*sin(angle[0])-(float)L[1]*sin(angle[1])
                       -(float)L[2]*sin(angle[2]);
       ligne(xorig,yorig,xorig+L[0]*cos(angle[0]),yorig-L[0]*sin(angle[0]),noir);
       ligne(xorig+L[0]*cos(angle[0]),yorig-L[0]*sin(angle[0]),
               xorig+L[0]*cos(angle[0])+L[1]*cos(angle[1]),
               yorig-L[0]*sin(angle[0])-L[1]*sin(angle[1]),noir);
       ligne(xorig+L[0]*cos(angle[0])+L[1]*cos(angle[1]),
              yorig-L[0]*sin(angle[0])-L[1]*sin(angle[1]),  xc[numero],yc[numero],noir);
      color[numero]=SDL_MapRGB(ecran->format,
                         255-20*a[2]-10*a[1]-5*a[0], 20*a[2], (50*a[2])%256);
     }
    N=n[0]*n[1]*n[2];
    for(i=0;i<N;i++)  for(j=0; j<=rdebut; j++) cercle(xc[i],yc[i],j,noir);
    SDL_Flip(ecran); pause(); return 0;
}
```

3) Utiliser la méthode des cercles grossissants pour avoir les cellules de Descartes-Voronoï des nœuds terminaux.



4) Construire les cellules de Descartes-Voronoï de tous les nœuds de l'arbre, et pas seulement celles des nœuds terminaux.



Ici $n_0 = 5$, $n_1 = 7$, $n_2 = 5$