```c
#include "nuitsBlanches.h"

//@ Les fonctions...ah j'aurais aime dire les methodes(la POO me manque!)
//creation du point
Point point(int x,int y){
    Point retour;
    retour.x=x;
    retour.y=y;
    return retour;
}
//creation du triangle
Triangle triangle(Point p1,Point p2,Point p3){
    Triangle retour;
    retour.s=p1;
    retour.b1=p2;
    retour.b2=p3;
    return retour;
}
//calcul des coordonnees du point milieu entre p1 et p2
Point pointMilieu(Point p1,Point p2){
    Point procheX;
    Point loinX;
    Point procheY;
    Point loinY;
    (Min(p1.x,p2.x)==p1.x)?(procheX=p1):(procheX=p2);
    (Max(p1.x,p2.x)==p1.x)?(loinX=p1):(loinX=p2);
    (Min(p1.y,p2.y)==p1.y)?(procheY=p1):(procheY=p2);
    (Max(p1.y,p2.y)==p1.y)?(loinY=p1):(loinY=p2);
    return point(procheX.x+(loinX.x-procheX.x)/2,procheY.y+(loinY.y-procheY.y)/2);
    //une figure suffit pour comprendre cette formule!!!
}

Triangle* diviser(Triangle t){
    Triangle* retour=(Triangle*)malloc(4*sizeof(Triangle));
    if (retour==NULL) exit(EXIT_FAILURE);
    *retour=triangle(t.s,pointMilieu(t.s,t.b1),pointMilieu(t.s,t.b2));
    *(retour+1)=triangle(t.b1,pointMilieu(t.b1,t.s),pointMilieu(t.b1,t.b2));
    *(retour+2)=triangle(t.b2,pointMilieu(t.b2,t.s),pointMilieu(t.b2,t.b1));
    *(retour+3)=triangle(pointMilieu(t.s,t.b1),pointMilieu(t.s,t.b2),pointMilieu(t.b1,t.
b2));
    return retour;
}
//desin d'un point
void dpoint(Point p,Uint32 couleur){
    SDL_Rect k = {p.x, p.y, 1, 1};
    SDL_FillRect(SDL_GetVideoSurface(), &k, couleur);
}
//dessin d'une ligne
void trace(Point p1,Point p2,Uint32 couleur){
    double x = p1.x, y = p1.y;
    double inc_x = p2.x - p1.x, inc_y = p2.y - p1.y;
    int m = Max(abs(inc_x), abs(inc_y));
    inc_x /= m;
    inc_y /= m;

    for(; m >= 0; m--)
    {
        dpoint(point((unsigned int)x, (unsigned int)y), couleur);
        x += inc_x;
        y += inc_y;
    }
}
//dessinons un triangle
void dTriangle(Triangle t,Uint32 couleur){
    trace(t.s,t.b1,couleur);
    trace(t.s,t.b2,couleur);
```

```
    trace(t.b1,t.b2,couleur);
    SDL_Flip(SDL_GetVideoSurface());
}
//Nombre de triangles contenus dans la boule!!!
long nbTriangles(int rang){
    int retour=0,i;
    for (i=0;i<=rang;i++){
        retour+=pow(4,i);
    }
    return retour;
}
//Verifie si un nombre est une puissance de 4
int powOf4(long nbre){
    long t=nbre;
    int i=0;
    while (t>0){
        t-=pow(4,i);
        i++;
    }
    return t==0;
}
//Enfin la derniere phase de calcul....et la plus interessante: LE DESSIN DE LA BOULE!
void chaineT(Triangle ancetre,int rang){
    if (rang<0) exit(EXIT_FAILURE);
    Triangle *retour=NULL,*passeur=NULL;
    long nb=nbTriangles(rang);
    retour=(Triangle*)malloc(nb*sizeof(Triangle));
    passeur=(Triangle*)malloc(4*sizeof(Triangle));
    if (retour==NULL||passeur==NULL) exit(EXIT_FAILURE);
    int posAv=1,posAr=1;
    passeur=diviser(ancetre);
    *retour=ancetre;
    rang--;
    if (rang>=0){
    int i;
    for(i=1;i<=4;i++){
        *(retour+i)=*(passeur+(i-posAv));
    }
    rang--;
    posAv+=4;
    }
    int i;
    while (rang>=0){
        passeur=diviser(*(retour+posAr));
        for(i=posAv;(i-posAv)<=4;i++){
            *(retour+i)=*(passeur+(i-posAv));
        }
        posAr++;
        posAv+=4;
        if(powOf4(posAv)) rang--;
    }
    if (nb==1){//s'il n'ya qu'un triangle dans la pile!
        dTriangle(*(retour),NOIR);
        free(retour);
    }
    else{
        int i=0;
        while (i!=nb-1) {
            dTriangle(*(retour+i),NOIR);
            i++;
        }
    }
}
```