```
1. // ************************************************
2. // ***** TP Question 1 : voir question du TD
3. // ************************************************
4.
5.
6. // ************************************************
7. // ***** TP Question 2 et 3
8. // ************************************************
9. // ************************** Sampling ***************************************************
   ***
10.void draw_sampling()
11.{
12.    int i,j;
13.    float a, b;
14.    Complex c;
15.    Complex center = make_complex( DIMW/2, DIMW/2);
16.    color(255,0,0);
17.    const int MAX = 10;
18.    for(i=0;i<MAX;++i)
19.    {
20.        a = float(DIMW) * i/MAX;
21.        for(j=0;j<MAX;++j)
22.        {
23.            b = float(DIMW) * j/MAX;
24.            c = make_complex(a,b);
25.            circleFill( c.x, c.y, 2);
26.        }
27.    }
28.
29.}
30.
31.// ************************** Sampling EXPO ***************************************
   ********
32.void draw_sampling_expo()
33.{
34.    int i,j;
35.    float r, theta;
36.    Complex c;
37.    Complex center = make_complex( DIMW/2, DIMW/2);
38.    color(255,0,0);
39.    const int MAX = 20;
40.    for(i=0;i<MAX;++i)
41.    {
42.        r = 0.5f*DIMW * i/MAX;
43.        for(j=0;j<MAX;++j)
44.        {
45.            theta = 2.f*M_PI*j/MAX;
46.            c = center + make_complex_expo(r,theta);
47.            circleFill( c.x, c.y, 1);
48.        }
49.    }
50.
51.}
52.
53.
54.// ************************************************
55.// ***** TP Question 8 Ã  10 : voir question du TD
56.// ************************************************
57.
58.
59.
60.
61.// ************************************************
62.// ***** TP Question 4 Ã  7
63.// ************************************************
64.// ************************** BIRD ****************************************************

65.struct Bird
66.{
67.    Complex c;      // centre de l'oiseau
68.    float angle;    // angle des ailes en degrÃ©
```

```
69. };
70.
71. void init(Bird& b)
72. {
73.     b.c = make_complex(DIMW/2, DIMW/2);
74.     b.angle = 15;           // angle au repos en degrÃ©
75. }
76.
77. void draw(Bird& b)
78. {
79.     color(255,255,0);
80.     circleFill( b.c.x, b.c.y, 2);
81.
82.     Complex right = b.c + make_complex(20,0);       // extremite de l'aile droite
83.     Complex left = b.c + make_complex(-20,0);       // extremite de l'aile gauche
84.     right = rotate( right, b.c.x, b.c.y, b.angle);  // tourner l'extremitÃ© de l'aile d
    'un angle b.angle
85.     left = rotate( left, b.c.x, b.c.y, -
    b.angle);   // tourner l'extremitÃ© de l'aile d'un angle -b.angle
86.
87.     // dessine les deux ailes
88.     line(b.c.x, b.c.y, left.x, left.y);
89.     line(b.c.x, b.c.y, right.x, right.y);
90. }
91.
92. void update(Bird& b)
93. {
94.     const float d = 0.1f;
95.     if (b.c.y > 3) b.c.y-=d;            // fait tomber l'oiseau Ã  chaque fois
96.
97.     if (isKeyPressed(SDLK_LEFT))   if (b.c.x > 0) b.c.x-=d;
98.     if (isKeyPressed(SDLK_RIGHT))  if (b.c.x < DIMW) b.c.x+=d;
99.     if (isKeyPressed(SDLK_UP))
100.          {
101.              b.c.y += 2.f*d;                 // fait remonter l'oiseau si KEY_UP
102.              float t = elapsedTime();
103.              b.angle = 20.f*cos(50.f*t);     // modifie l'angle des ailes, oscille ent
    re -20 et +20 Â° (le 50*t sert Ã  accÃ©lÃ©rer le mouvement)
104.          }
105.          else b.angle = 15.f;
106.      }
107.
108.
109.
110.      // ************************************************
111.      // ***** TP Question 11 Ã  14
112.      // ************************************************
113.      // ============================  JULIA ============================================
    ===
114.      const int MAXITE = 200;
115.      int suite_julia(float borne, int maxIte, Complex Z0, Complex C)
116.      {
117.          Complex Zn=Z0;
118.          int i;
119.          i=0;
120.          do
121.          {
122.              Zn = Zn*Zn+C;
123.              i++;
124.          }while ( (norm(Zn)<borne) && (i<maxIte) );
125.          return i;
126.      }
127.
128.      void couleur_julia(int n, unsigned char& r, unsigned char& g, unsigned char& b)
129.      {
130.          float c = (float(n))/MAXITE;
131.          r = c*255;
132.          g = 128 + c*128;
133.          b = 64  + c*(127+64);
134.      }
135.
```

```
136.    void draw_julia()
137.    {
138.        int i,j,n;
139.        unsigned char r,g,b;
140.        Complex C = make_complex(0.32, 0.043);
141.        float x,y;
142.        for(i=0;i<DIMW;i+=1)
143.        {
144.            for(j=0;j<DIMW;j+=1)
145.            {
146.                x = ((float(i))/DIMW)*3 - 1.5;
147.                y = ((float(j))/DIMW)*3 - 1.5;
148.                n = suite_julia( 2, MAXITE, make_complex(x,y), C );
149.                couleur_julia( n, r,g,b);
150.                put_pixel(i,j,r,g,b);
151.            }
152.        }
153.    }
154.
155.
156.
157.
158.    int main(int , char** )
159.    {
160.        bool stop=false;
161.        winInit("Complex numbers are cool!!!!", DIMW, DIMW);
162.        backgroundColor( 10, 20, 120 );
163.
164.        Menu menu;
165.        menu_add( menu, "Sampling reg");
166.        menu_add( menu, "Sampling expo");
167.        menu_add( menu, "SolarSystem");
168.        menu_add( menu, "Bird");
169.        menu_add( menu, "Polygon");
170.        menu_add( menu, "Julia");
171.
172.        SolarSystem ss;
173.        init(ss);
174.
175.        Bird b;
176.        init(b);
177.
178.        Polygon po;
179.        init(po);
180.
181.        while( !stop )
182.        {
183.            setKeyRepeatMode(true);
184.            winClear();
185.            //cout<<menu_select(menu)<<endl;
186.            switch( menu_select(menu) )
187.            {
188.                case 0:
189.                    draw_sampling();
190.                    break;
191.                case 1:
192.                    draw_sampling_expo();
193.                    break;
194.                case 2 :
195.                    draw(ss);
196.                    update(ss);
197.                    break;
198.                case 3 :
199.                    draw(b);
200.                    update(b);
201.                    break;
202.                case 4:
203.                    draw_polygon(po);
204.                    break;
205.                case 5:
206.                    draw_julia();
```

```
207.                     break;
208.                 }
209.                 menu_draw( menu );
210.                 stop = winDisplay();
211.             }
212.             winQuit();
213.             return 0;
214.         }
```