UNIVERSITÉ PARIS SUD

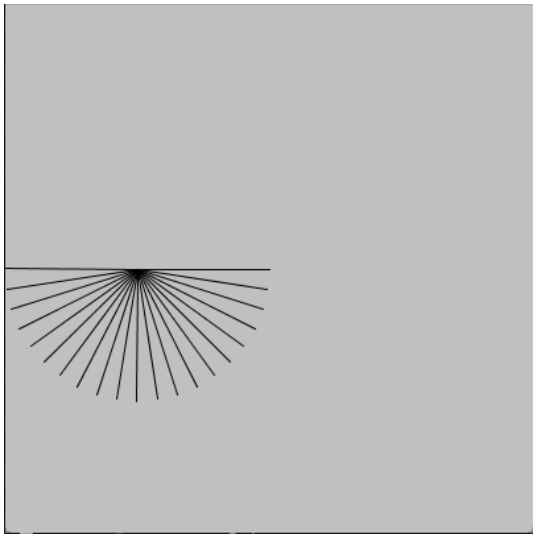
Licence MPI. Info 112 (IGOR)
F. Vernier
Poly. de cours (éventuellement annoté) **autorisé !**

Examen 2018-2019, session 1 (2h, barème /20) 10/01/2019

1. Dessin (7 pts)

Dans cet examen, il vous est demandé de fournir un certain nombre d'éléments nécessaires à la réalisation d'applications écrites en langage Processing. Nous partirons du programme ci-dessous qui donne le résultat ci-contre.

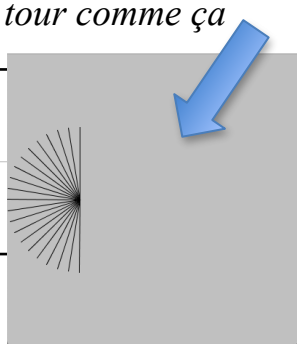
```
1.  void setup(){
2.    size(400, 400);
3.  }
4.
5.  void draw(){
6.    background(192, 192, 192);
7.    float R = 100;
8.    int N = 20;
9.    strokeWeight(1);noFill();
10.   for (int i=0; i<=N; i++){
11.     float x0 = width/4 +R*cos(i*PI/N);
12.     float y0 = height/2+R*sin(i*PI/N);
13.     line (width/4, height/2, x0, y0);
14.   }
15. }
```



1.1. ***(1pt)*** *Comment s'appelle le principe de math mis en oeuvre aux lignes 11 et 12 ?*

La trigonométrie

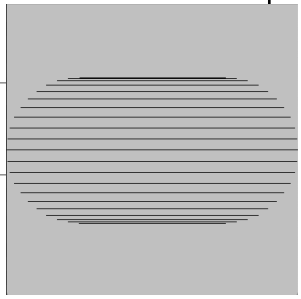
1.2. ***(1 pt)*** *Changer les lignes 11 et 12 pour tourner la figure d'un quart de tour comme ça*



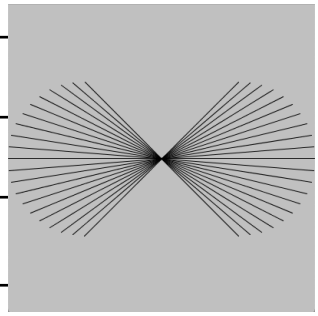
```
float x0 = width/4 +R*cos(i*PI/N+PI/2);
float y0 = height/2+R*sin(i*PI/N+PI/2);
```

1.3. ***(2 pts)*** *Quelles lignes rajouter ensuite pour remplacer la ligne 13 et obtenir le résultat ci-dessous ?*

```
float x1 = 3*width/4+R*cos(i*PI/N-PI/2);
float y1 = height/2 +R*sin(i*PI/N+PI/2);
line (x0, y0, x1, y1);
```

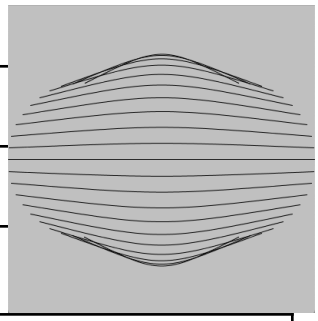


1.4. ***(1 pt)*** *Que modifier (expliquer sans re-écrire tout le code) pour obtenir l'effet noeud papillon ci-contre ?*



```
On remplace i par (N-i) dans le calcul
 de x1 et y1
```

1.5. ***(2 pts)*** *En repartant de la question 1.2 comment dessiner une courbe de bézier qui donne un effet de tonneau bombé (vous calculerez la position d'un unique point de contrôle décalé par rapport au centre)*

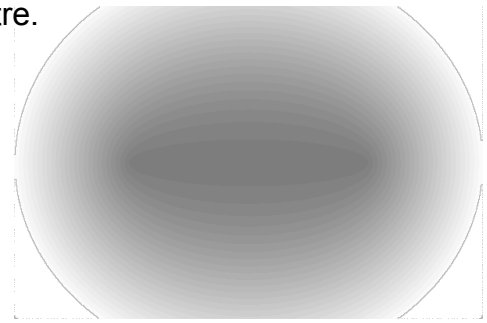


```
float ydecalle = (height/2)+(y0-height/2)*1.5;
bezier(x0, y0, width/2, ydecalle,
      width/2,ydecalle, x1, y1);
```

Indication *: la fonction bezier (x0, y1, x1, y1, x2, y2, x3, y3) trace une courbe de x0,y0 vers x3,y3 avec x1,y1 le point de contrôle qui gère la tangente en x0,y0 et x2,y2 qui gère la tangente en x3, y3*

2. Pixels (6 pts)

Nous partirons du programme ci-dessous qui donne le résultat ci-contre.



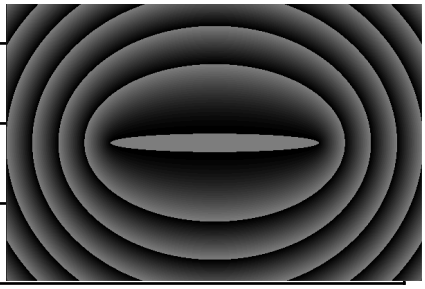
```
16. void setup(){
17.   size(600, 400);
18. }
19.
20. void draw(){
21.   loadPixels();
22.    for (int j=0; j<height; j++) {
23.      for (int i=0; i<width; i++) {
24.        float d1 = dist(1*width/4, height/2, i, j) ;
25.        float d2 = dist(3*width/4, height/2, i, j) ;
26.        int   v  = Math.round(d1+d2)*255/width;
27.        pixels[i+j*width] = color(v);
28.      }
29.    }
30.    updatePixels();
31. }
```

2.1. ***(1pt)*** *Comment s'appelle le principe de math mis en oeuvre à la ligne 26 ?*

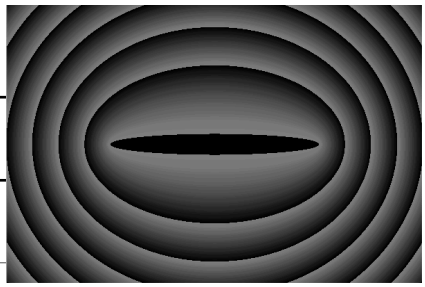
Règle de trois / règle des proportions / proportionnalité

2.2. ***(3 pts)*** *en utilisant une division entière ou un modulo transformez la couleur des pixels pour obtenir un effet de dégradé répétitif avec une luminosité entre 0 et 128 environ*

```
pixels[i+j*width] = color(4*(v%32));
color((v/32)*16); // BOF = paliers
color(64*(1+cos(64*v/255.0))); // :-(
```



2.3. ***(1 pts)*** *Comment inverser les couleurs comme ci-contre ?*



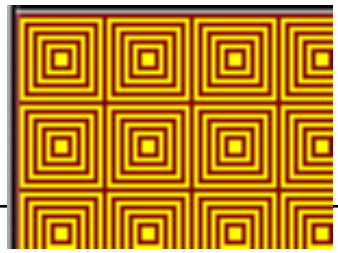
```
color(128-4*(v%32));
```

2.4. ***(1 pts)*** *En supposant que nous disposons d'une variable t qui démarre à 0 et s'incrémente de 1 à chaque affichage comment faire une animation qui donne l'impression que les ovales grandissent comme des vagues dans une flaque d'eau (de nouveaux ovales doivent se créer au centre)?*

```
color(128-4*((v-t)%32));
```

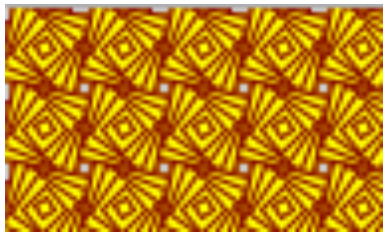
3. Boucles imbriquées de dessin (7 pts)

3.1. ***(3pt)*** *Ecrivez la fonction draw() qui dessine le motif de carrés 20px * 20px suivant. Chaque carré a un bord rouge et un fond jaune. Les carrés remplissent tout l'écran et chaque zone de 20px * 20px comporte 5 carrés imbrique*



```
stroke(128, 0, 0); fill(255, 255, 0);
for (int j=0; j<height; j+=20){
 for (int i=0; i<width; i+=20){
  for (int k=0; k<5; k++){
   rect (i+k*2, j+k*2, 20-4*k, 20-4*k);
  }
 }
}
```

3.2. *Re-écrivez entièrement le code pour que les carrés imbriqués tournent de PI/16 a chaque imbrication. Pour atteindre ce résultat nous vous proposons de rajouter rectMode(CENTER); au debut de la fonction dessin qui modifie le comportement de la fonction rect(x,y, w, h) qui dessine alors un rectangle centré sur le point x, y et toujours de taille w*h* ***(4pts)***



```
void draw(){
  rectMode(CENTER);
  stroke(128, 0, 0);
  fill(255, 255, 0);
  for (int j=0; j<height; j+=20){
    for (int i=0; i<width; i+=20){
      pushMatrix();
      translate(i+10, j+10);
      for (int k=0; k<5; k++){
        rotate(PI/16);
        rect (0, 0, 20-4*k, 20-4*k);
      }
      popMatrix();
    }
  }
}
```