Le module kandinsky - Corrigé

NUMWORKS

Cette fiche a été rédigée par **Philippe Moutou**. Il enseigne au lycée Henri IV à Paris.

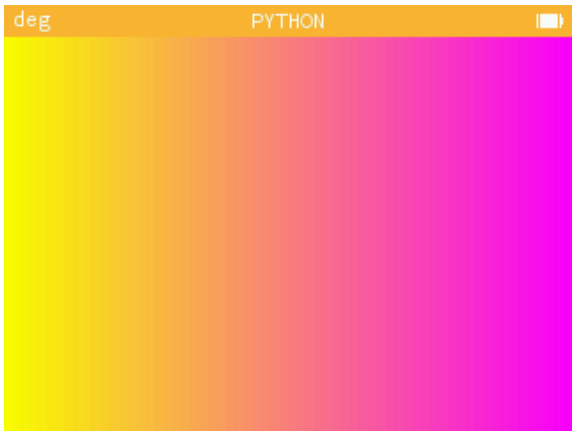
1 Exercice

Écrire un programme qui réalise un dégradé horizontal entre deux couleurs quelconques.

L'idée ici est de couvrir l'écran avec des pixels dont la couleur est identique sur une même colonne verticale (x constant) mais change sur les lignes horizontales (y constant), passant progressivement de `c1` à gauche à `c2` à droite. Pour obtenir le dégradé, on calcule les différences `dr`, `dg` et `db` entre les composantes rgb de `c2` et `c1` et on ajoute aux composantes de `c1` la part de ces différences correspondant à l'avancement dans la ligne (donc au numéro de la colonne).

```
from kandinsky import *
def couleur():
  dr=c2[0]-c1[0]
  dg=c2[1]-c1[1]
  db=c2[2]-c1[2]
  return color(c1[0]+i*dr//320,c1[1]+i*dg//320,c1[2]+i*db//320)

c1=(255,255,0)
c2=(255,0,255)
for i in range(320):
  col=couleur()
  for j in range(222):
    set_pixel(i,j,col)
```



Voici le résultat pour un dégradé entre le jaune et le magenta.

Pour prolonger cette idée, puisque la hauteur de l'écran n'est pas utilisée ici (sauf pour étaler la couleur), je vais dégrader verticalement la couleur d'une colonne entre sa valeur actuelle qui sera appliquée sur la ligne du haut et une troisième couleur (pour compléter mon exemple, je prendrai du cyan) qui ne sera atteinte que sur la ligne du bas. J'applique exactement le même principe que précédemment, calculant dans un premier temps la couleur du haut (dégradé entre `c1` et `c2`) dans les variables `r`, `g` et `b` et, dans un second temps, dégradant la couleur obtenue avec `c3`.

```
from kandinsky import *
def couleur():
  r=c1[0]+i*(c2[0]-c1[0])//320
  g=c1[1]+i*(c2[1]-c1[1])//320
  b=c1[2]+i*(c2[2]-c1[2])//320
  r+=j*(c3[0]-r)//320
  g+=j*(c3[1]-g)//320
  b+=j*(c3[2]-b)//320
  return color(r,g,b)

c1=(255,255,0)
c2=(255,0,255)
c3=(0,255,255)
for i in range(320):
  for j in range(222):
    col=couleur()
    set_pixel(i,j,col)
```

2 Autre exercice

Écrire un programme qui place des points aléatoirement dans l'écran, la couleur de ces points étant fonction de la distance au centre de l'écran, le point de coordonnées `(160,111)`.

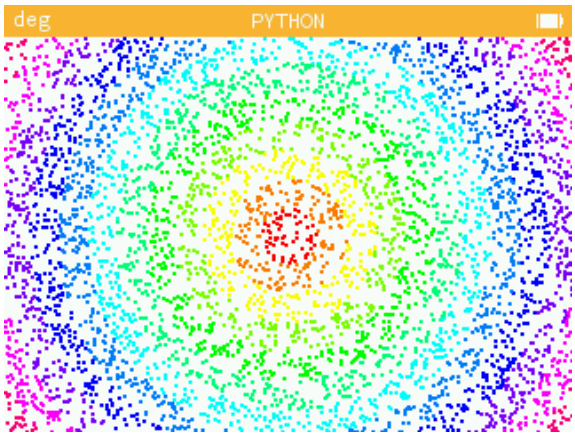
```
from kandinsky import *
from random import *
from math import *

def couleur(x,y):
  d=int(sqrt((x-160)**2+(y-111)**2)/16.25)
  return color(c[d][0],c[d][1],c[d][2])
```

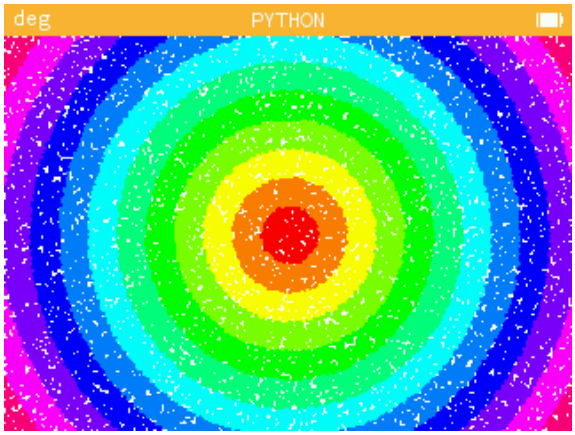
```
def tirage(n):
  for i in range(n):
    x=randint(0,320)
    y=randint(0,222)
    c=couleur(x,y)
    set_pixel(x,y,c)
    set_pixel(x+1,y,c)
    set_pixel(x,y+1,c)
    set_pixel(x+1,y+1,c)

c=[[255,0,0],[255,127,0],[255,255,0],[127,255,0],[0,255,0],[0,255,127],[0,255,255],[0,127,255],[0,0,255],[1
```

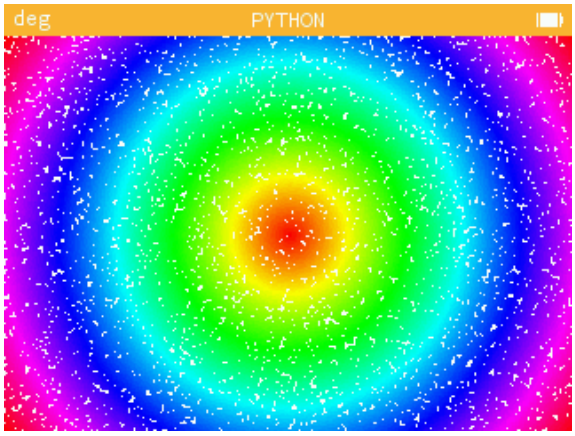
Le programme détermine si la couleur du point appartient à l'une des douze couleurs de notre cercle chromatique. La distance maximum au centre de l'écran étant 195 environ, la division par 16.25 permet de ramener les coordonnées aléatoires à l'une de ces douze couleurs. J'ai dessiné des points plus gros, formés de quatre pixels, pour qu'ils soient plus visibles.



Essai pour `tirage(5000)`.



Essai pour `tirage(50000)`.



Pour prolonger ce travail, on peut obtenir un passage progressif d'une couleur à l'autre, un dégradé sans frontière : éliminons les six couleurs intermédiaires qui sont seulement les milieux des dégradés successifs et adaptons le module qui permettait d'obtenir un dégradé dans l'exercice précédent. Le résultat est alors un peu différent.

```
from kandinsky import *
from random import *
from math import *

def degrade(c1,c2,k):
  dr=c2[0]-c1[0]
  dg=c2[1]-c1[1]
  db=c2[2]-c1[2]
  return color(c1[0]+int(k*dr),c1[1]+int(k*dg),c1[2]+int(k*db))

def tirage(n):
  for i in range(n):
    x=randint(0,320)
    y=randint(0,222)
    c=couleur(x,y)
    set_pixel(x,y,c)
    set_pixel(x+1,y,c)
    set_pixel(x,y+1,c)
    set_pixel(x+1,y+1,c)

def couleur(x,y):
  d=sqrt((x-160)**2+(y-111)**2)/32.5
  return degrade(c[int(d)],c[(int(d)+1)%6],d-int(d))

c=[[255,0,0],[255,255,0],[0,255,0],[0,255,255],[0,0,255],[255,0,255]]
```