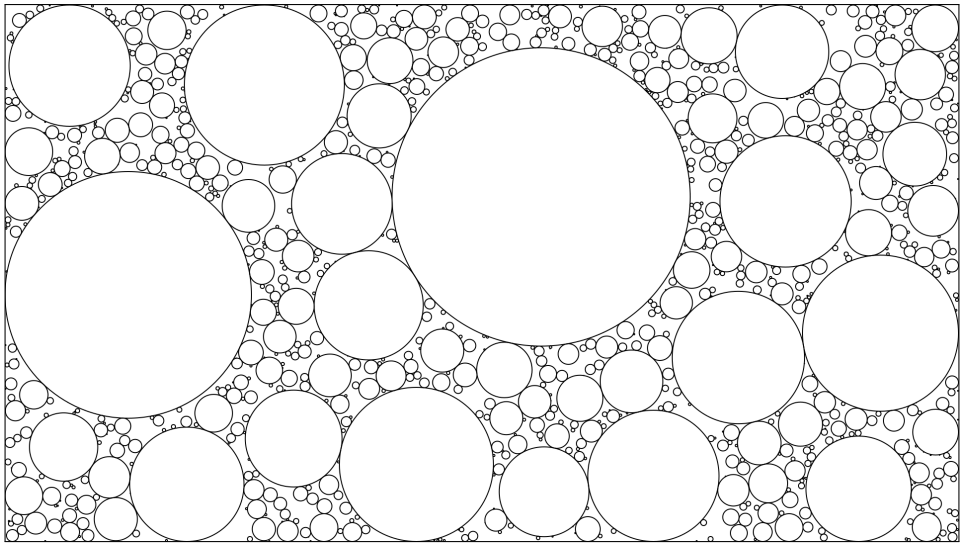
Objectif

Cette activité conçoit algorithmiquement une image de fond d'écran. Celle-ci est conçue partiellement de manière aléatoire. Ci-dessous un exemple de résultat attendu :



Voici le code créant cette image :

Les 1 000 cercles de l'image :

- ne se chevauchent pas ;
- ne contiennent pas d'autres cercles ;
- sont intégralement dans l'image ;
- ont un diamètre maximal de 300 pixels.

Code Python

```python
import fond_ecran as fe

mon_fond = fe.Fond(couleur=(1, 1, 1))

mon_fond.tracer_cercles(1000, (0, 0, 0))
mon_fond.sauvegarder("fond_cercles.png")
```

Ce résultat est celui qui est obtenu dans cette activité. On pourra modifier de manière plus personnelle (voir section `Pour aller plus loin`). De plus, on s'intéressera à la manière d'automatiser le placement du fond d'écran avec le système d'exploitation (voir section `Avec le système d'exploitation`)

Module et classes

On utilisera le module `pycairo` qui permet de créer des images en dessinant des formes et du texte. Attention ce n'est pas un module de traitement d'images comme peut l'être `PIL(low)`.

Le principe est de créer un objet `Surface` sur laquelle on va appliquer un objet `Context`, sorte de calque graphique de dessin, sur lequel on va dessiner.

Documentation : `https://pycairo.readthedocs.io/`

De plus, on créera :

- une classe `Fond` gérant la conception globale du fond ;
- une classe `Cercle` gérant le positionnement, le rayon et le tracé de chacun des cercles (instances) créés.

composé avec LuaLaTeX sous linux debian à 1668692480

Classe Cercle

Cette classe permet de définir et dessiner des cercles.

Cercle
centre rayon
appartient_disque(point) trace_cairo(ctx, couleur) distance(xy0, xy1)

Attributs

- `centre` : `Tuple[int, int]`, position du centre du cercle en pixels
- `rayon` : `int`, rayon du cercle en pixels

Méthodes

- `appartient_disque(point: Tuple[int, int])⟶bool`
  Vérifie qu'un point donné appartient au disque (intérieur et frontière du cercle).
- `trace_cairo(ctx: cairo.Context, couleur: Tuple[float, float, float])⟶None`
  Trace le cercle dans le contexte graphique précisé en utilisant une couleur donnée.
- `distance(xy0: Tuple[int, int], xy1: Tuple[int, int])⟶float`
  Méthode de classe (c'est pour cela que c'est souligné) calculant la distance entre deux points donnés.

Classe Fond

Cette classe permet de créer une image considérée comme un fond d'écran.

Cercle
largeur hauteur surface ctx cercles
est_dans_un_cercle(point) rayon_max_bords(position) rayon_max_cercles(position) tracer_cercles(effectif, couleur, max_admis) sauvegarder(nom_fichier_png)

Attributs

- `largeur` : `int`, largeur de l'image en pixels
- `hauteur` : `int`, hauteur de l'image en pixels
- `surface` : `cairo.ImageSurface`, l'objet image
- `ctx` : `cairo.Context`, l'objet sur lequel on dessine
- `cercles` : `List[Cercle, ...]`, liste des objets cercles déjà tracés

Méthodes

- `est_dans_un_cercle(point: Tuple[int, int])⟶bool`
  Vérifie qu'un point donné est dans un des cercles déjà tracés.
- `rayon_max_bords(position: Tuple[int, int])⟶int`
  Renvoie la plus petite distance aux quatre bords de l'image de la position passée en argument. C'est le rayon d'un cercle qui ne déborde pas de l'image.
- `rayon_max_cercles(position: Tuple[int, int])⟶int`
  Renvoie le plus grand rayon de cercle possible en étudiant la position passée en argument par rapport aux autres cercles déjà tracés, en évitant que le nouveau cercle chevauche les autres.
- `tracer_cercles(effectif: int, couleur: Tuple(float, float, float), max_admins: int⟶None`
  Fonction principale de la classe, permet de lancer le traitement pour générer l'image.
  `max_admis` désigne le rayon qu'on ne veut en aucun cas dépassé, on en fait un argument optionnel avec une valeur par défaut à 300 px.
- `sauvegarder(nom_fichier_png: str)⟶None`
  Permet l'enregistrement de l'image dans un fichier PNG. Mettre l'extension ".png" dans le nom de fichier.

Composé avec LuaLaTeX sous linux debian à 1668692481