Il faut avoir lu le document d'introduction pour aborder ce document.

1. Écrire le squelette de la classe `Cercle` avec :
   - son constructeur qui prend deux arguments : `centre` et `rayon` qui seront stockés dans deux attributs (voir document d'introduction) ;
   - ses méthodes, explicitées dans le document d'introduction (on mettra pour l'instant le mot-cle `pass` dans le corps).
2. À partir de la formule suivante, programmer en langage Python la méthode de classe `distance(xy0, xy1)` qui calcule la distance entre deux points du plan :

   Dans un repère orthonormé, si on considère deux points $A(x_A ; y_A)$ et $B(x_B ; y_B)$, on a alors :

   $$AB = \sqrt{(x_B - x_A)^2 + (y_B - y_A)^2}$$

   Rappel : `distance(xy0: Tuple[int, int], xy1: Tuple[int, int])⟶float`

   Cette méthode prendra sa place au-dessus du constructeur.
3. Un point M appartient au disque de centre I et de rayon r si et seulement si $IM \leqslant r$.

   À partir de cette propriété, écrire la méthode `appartient_disque(point)`.

   Rappel : `appartient_disque(point: Tuple[int, int])⟶bool`
4. Voici un récapitulatif des commandes utiles pour tracer un cercle avec `pycairo` (et nos contraintes). Dans ce qui suit, `ctx` désigne un objet `cairo.Context` sur lequel il est possible de dessiner :

   ctx.set_source_rgb Cette commande permet de changer de couleur, elle prend trois arguments : les canaux RVB de la couleur à utiliser. Les valeurs sont comprises entre 0 et 1. Elle ne renvoie rien.

   Par exemple, `ctx.set_source_rgb(1, 0, 0)` signifie qu'on utilisera dorénavant du rouge pour les tracés.

   ctx.arc Cette commande définit (et met dans la file des tracés à exécuter) un arc de cercle. Elle ne renvoie rien. Elle prend 5 arguments :
   a) l'abscisse du centre
   b) l'ordonnée du centre
   c) le rayon
   d) l'angle de départ (en radians)
   e) l'angle de fin (en radians)

   Par exemple, la commande `ctx.arc(100, 100, 50, 0, math.pi)` définit un demi-cercle dont le centre est au point de coordonnées $(100 ; 100)$ et dont le rayon vaut 50.

   ctx.stroke Cette commande demande de tracer avec la couleur dernièrement choisie la ou les figures qui sont actuellement dans la file des tracés à exécuter.

   Elle ne prend pas d'argument, elle ne renvoie rien.

   En utilisant ces commandes, écrire la méthode `trace_cairo`.

   Rappel : `trace_cairo(ctx: cairo.Context, couleur: Tuple[float, float, float])⟶None`

compose avec LuaLaTeX sous linux debian à 1668693011