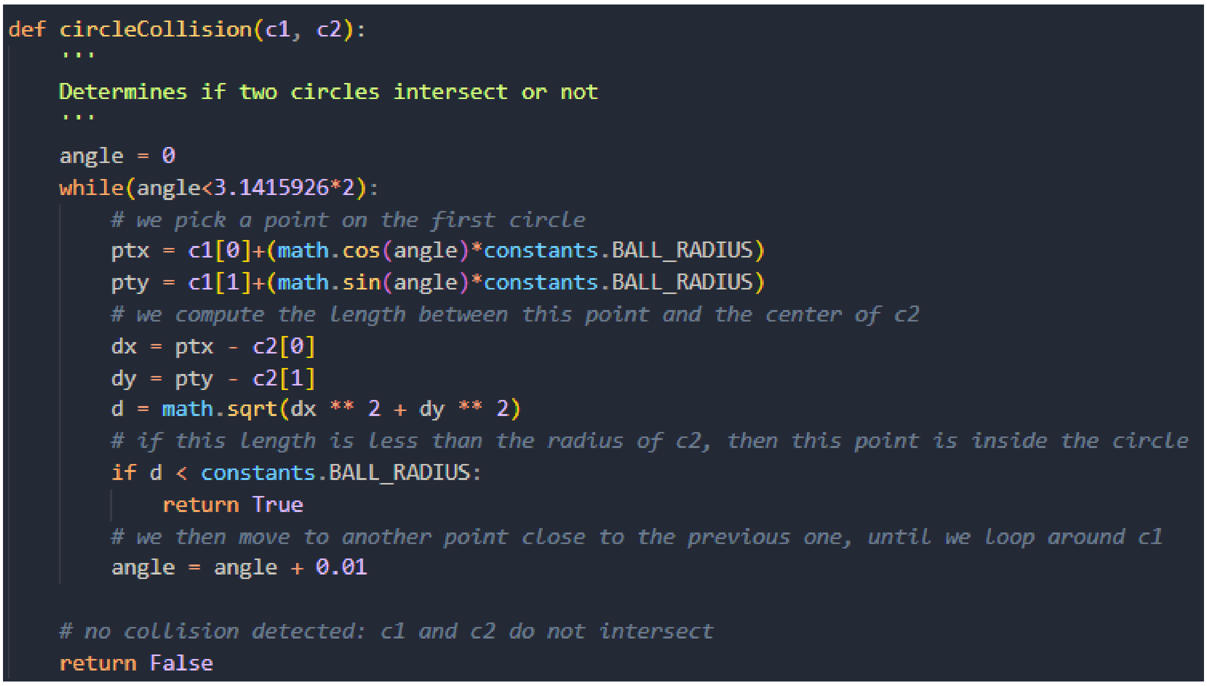
CARPENTIER Simon, CARPENTIER François TD2 TPD

STEP 2

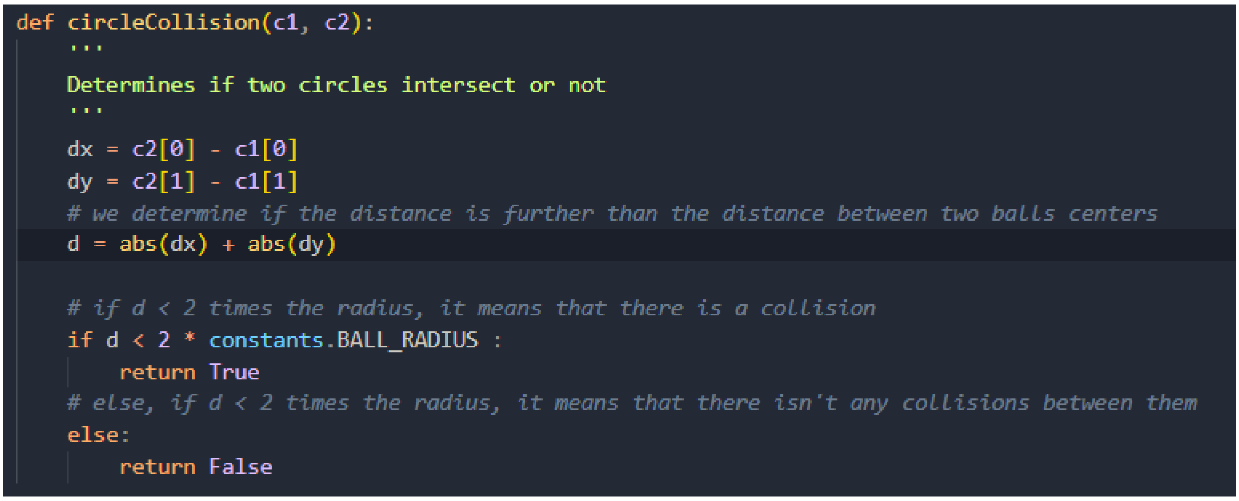
In this part, we modify the algorithm to calculate at every instant whether a point on a ball is in contact with the nearest ball. We will modify this to have much fewer calculations to perform.



```
def circleCollision(c1, c2):
    '''
    Determines if two circles intersect or not
    '''
    angle = 0
    while(angle<3.1415926*2):
        # we pick a point on the first circle
        ptx = c1[0]+(math.cos(angle)*constants.BALL_RADIUS)
        pty = c1[1]+(math.sin(angle)*constants.BALL_RADIUS)
        # we compute the length between this point and the center of c2
        dx = ptx - c2[0]
        dy = pty - c2[1]
        d = math.sqrt(dx ** 2 + dy ** 2)
        # if this length is less than the radius of c2, then this point is inside the circle
        if d < constants.BALL_RADIUS:
            return True
        # we then move to another point close to the previous one, until we loop around c1
        angle = angle + 0.01

    # no collision detected: c1 and c2 do not intersect
    return False
```

We will modify this to have much fewer calculations to perform.



```
def circleCollision(c1, c2):
    '''
    Determines if two circles intersect or not
    '''
    dx = c2[0] - c1[0]
    dy = c2[1] - c1[1]
    # we determine if the distance is further than the distance between two balls centers
    d = abs(dx) + abs(dy)

    # if d < 2 times the radius, it means that there is a collision
    if d < 2 * constants.BALL_RADIUS :
        return True
    # else, if d < 2 times the radius, it means that there isn't any collisions between them
    else:
        return False
```

By doing this, we compare whether the distance between the centers of two balls is greater or less than twice the radius of a ball, which means we are checking whether two balls are in contact. This will greatly reduce the number of calculations the algorithm will perform.

For this, we use the abs() function, which is the absolute value function, to avoid negative distance problems.