CARPENTIER Simon, CARPENTIER François TD2 TPD

STEP 1

After analyzing the algorithms, we begin the modification. First, we ensure that it only calculates collisions for balls in motion.

```
def computeCollisions(balls):
    '''
    Computes the collisions between a collection of balls and a single ball
    among them, identified by their list index
    '''

    collisions = []
    for i in range(len(balls)):
        p = balls[i]
        for j in range (i+1, len(balls)):
            if i!=j :
                # To avoid considering collisions with themselves
                q = balls[j]
                # Test collision between the balls p and q
                if circleCollision((p["x"],p["y"]),(q["x"],q["y"])):
                    collisions.append((i, j))
```

This algorithm has the disadvantage of calculating collisions for all balls, so we will modify it to only consider those in motion.

```
def computeCollisions(balls):
    '''
    Computes the collisions between a collection of balls and a single ball
    among them, identified by their list index
    '''

    collisions = []
    for i in range(len(balls)):
        p = balls[i]
        for j in range (i+1, len(balls)):
            if i!=j :
                # To avoid considering collisions with themselves
                q = balls[j]
                # To test mooving balls only
                if ((p["dx"] != 0) or (p["dy"] != 0) or (q["dx"] != 0) or (q["dy"] != 0)):
                    # Test collision between the balls p and q
                    if circleCollision((p["x"],p["y"]),(q["x"],q["y"])):
                        collisions.append((i, j))

    return collisions
```

With these modifications, we are sure that the collisions calculated are from moving ball, because we are looking for balls with a direction different than 0, so they have a direction and so they are mooving.