ECE 36800 Assignment #6

Original Due: 1:00 PM, Tuesday, November 4
Extended: 1:00 PM, Thursday, November 6

Description

In this assignment, you will implement a very simple version of collision detection. Your collision detector will load a "scene" consisting of a set of points in two-dimensional space, and then respond to a number of "collision queries". Each query will present you with the center and radius of a circle, and then ask you how many points in the scene collide with that circle.

Your program should take as a command-line argument the name of a text file describing the scene, and then respond to queries on standard input. The text file will contain a list of points, with one point on each line. Each point will be represented by two integers, the x- and the y-coordinate respectively, separated by a space. A query will consist of three integers: the x- and y-coordinates of the center, and the radius.

Consider the following points.txt file:

```
100 -100
100 100
-100 100
-100 -100
100 0
0 100
-100 0
0 -100
```

An sample run of the program is shown below:

```
$ ./a6 points.txt
> 0 0 100
4
> 0 0 142
8
> 50 50 100
3
```

Grading

For this assignment, you will be graded time and correctness, in addition to the usual memory management. Most test cases will be relatively small, consisting of up to 100 points in the scene and up to 10 queries; for these, you will get full points just by returning the correct responses. Other test cases will be much larger, consisting of scenes with several million points, and potentially several thousand queries. In addition to checking for correctness, we will time these test cases, and your score will be based on your run time. As always, a 50% penalty will be applied to any test case with a memory leak or memory error. You should put a comment at the start of each method describing its functionality.

You should not be printing additional things other than the expected outputs.

You may submit to Gradescope as often as you'd like before the deadline. Only your active submission (by default the most recent) will be counted. While the score given to you by Gradescope is likely a good indicator of your final grade on the assignment, we reserve the right to add additional test cases after the submission deadline.

Submission

Submit any source/header files with your implementation, as well as a Makefile that builds a target called `a6`, to Gradescope. DO NOT include executables in your submission. DO NOT put your files inside a folder. Note that to receive points, your submission must work on `eceprog`.