DSC 40B - Homework 05

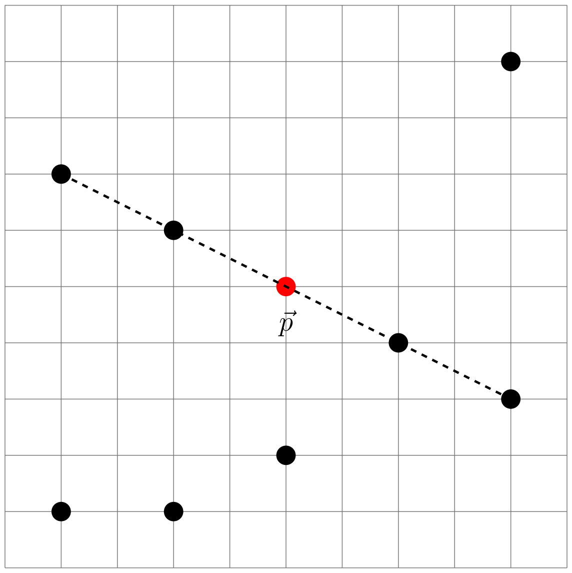
Due: Wednesday, May 8

Write your solutions to the following problems by either typing them up or handwriting them on another piece of paper. Unless otherwise noted by the problem's instructions, show your work or provide some justification for your answer. Homeworks are due via Gradescope at 11:59 p.m.

Programming Problem 1.

Consider the problem of finding collinear points. Suppose you are given a set X of n points in $\mathbb{R}^2$, along with a query point $\vec{p} \in \mathbb{R}^2$. Your goal is to calculate the maximum number of points in X that fall on the same straight line containing $\vec{p}$. That is, considering all of the (infinitely) many lines containing $\vec{p}$, find the one that contains the most points from X.

For example, in the picture below, the black points are those in X and the red point represents $\vec{p}$. The maximum number of points in X that falls on a line containing $\vec{p}$ is four (those on the dashed line).



In a file named `max_points_on_line.py`, write a function `max_points_on_line(X, p)` which accepts the following arguments:

- `X`: A list of points, each point being a 2-tuple of integers.
- `p`: A query point represented as a 2-tuple of integers.

Your function should return the maximum number of points in `X` that fall on the same straight line containing $\vec{p}$. Your answer should be an integer. If `X` is empty, return `0`. If a point $\vec{x}$ in `X` is such that $\vec{x} = \vec{p}$, you should consider to be on the same line as $\vec{p}$ (even if there are no other points).

Your algorithm should take expected time that is linear in the number of points in `X`.

Example (using the points in the picture above):

```
>>> X = [
... (1, 1),
... (3, 1),
... (5, 2),
... (3, 6),
... (1, 7),
... (7, 4),
... (9, 3),
... (9, 9)
... ]
>>> p = (5, 5)
>>> max_points_on_line(X, p)
4
```