2143 OOP - Test 1

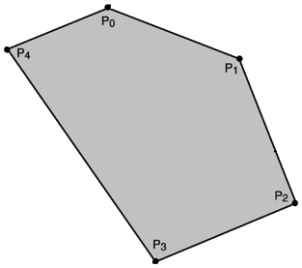
Name: _____

Instructions

- Use pencil only
- Write your name at the top of all pages turned in.
- Staple pages together at the top left corner.
- Make sure your pages are in order, with questions also in order.
- Handwriting that is illegible (messy, small, not straight) will lose points.
- Indentation matters. Keep code aligned correctly.
- Failure to comply will result in loss of letter grade.
- All answers will be written on the paper provided, and not directly on the test.

Background:

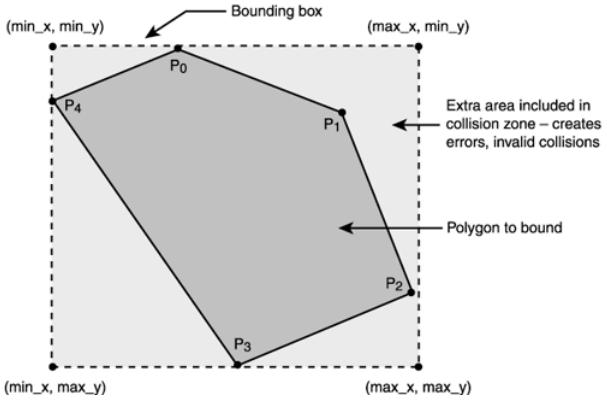
- A polygon is a 2D shape made up of 3 or more sides.



- Each side can also be thought of as a line with a beginning point (x_1,y_1) and an ending point (x_2,y_2).
  - For example, the polygon above has Points: P_0, P_1,P_2,P_3,P_4
  - It also has Lines: (P_0, P_1),(P_1, P_2),(P_2,P_3),(P_3,P_4),(P_4,P_0).
- The length of a line (or distance between two points can be calculated using the following formula:

$$D = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

- Remember that:
  - Square root is `sqrt(some number)`
  - Exponentiation is `pow(2,3)` or 2^3.
- A bounding box of a polygon can be found by finding the 4 extreme `x,y` values



General

- You are going to write 3 class definitions
- Each class should have methods to set / get the data members of the class.
- Each class definition should build on the previous class.
- Do not implement any methods until asked, definitions only.

Question 1:

What are the 3 Central Principles of OOP?

Question 2 - Point Class

- Write a class that represents a point (x,y).
- X and Y are integer values.

Question 3 - Line Class

- Write a class that represents a line (x_1,y_1) ,(x_2,y_2).
- Your class should have 2 constructors:
    - One that takes 4 values x_1,y_1,x_2,y_2
    - One that takes 2 values P_1,P_2.
- This class can return the length of its line.

Question 4 - Polygon Class

- Write a class that represents a polygon.
- Your polygon can have between `3` and `N` sides.
- Your class should have multiple constructors:
    - One that initializes an empty polygon
    - One that accepts an array of points $[P_1,P_2,...,.P_n]$.
    - One that accepts an array of lines.
- This class can return the perimeter of the polygon.
- This class can return the area of a bounding box of the polygon.

Question 5 - Implementation

- Implement the perimeter method of the polygon.

Question 6 - Bonus

- Implement the bounding box method of the polygon.