Lab 3: Iterative Algorithms

Learning Outcomes

- Design an algorithm for a given computational problem statement
- Justify the correctness of an algorithm
- Perform asymptotic complexity analysis of the run time of an algorithm
- Generate test cases for an algorithm
- Correctly implement an algorithm from pseudocode
- Design and execute benchmarks for an algorithm

Overview

For this lab, you are given the description of several problems. You will pick one problem to work on. You will be expected to think about the problem and come up with a fundamental insight or rule that will allow you to solve the problem. Once achieved, you will be able to write and analyze an algorithm using the techniques you learned in class.

Instructions

Choose **one** of the problems described below. Submit a report containing the following:

1. A paragraph describing a "decision rule" that can be applied to solve to the computational problem. Provide at least 2 illustrations (test cases) that demonstrate how the rule is applied.

2. High-level pseudocode for an algorithm that uses that rule to solve the computational problem for any input

3. Provide an explanation and justification for why your algorithm is correct (1-3 paragraphs)

4. Perform an analysis of the worst-case run time using asymptotic notation.

5. A table of your test cases, the answers you expect, and the answers returned by running your implementation of the algorithm.

6. A table and graph from benchmarking your implementation on problem instances of different sizes. The benchmarks should support your theoretically derived run time.

7. Attach all your source code and test cases in an appendix.

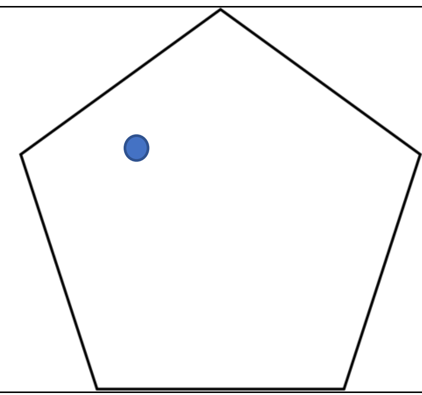
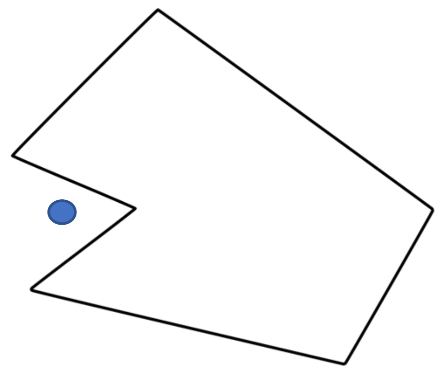
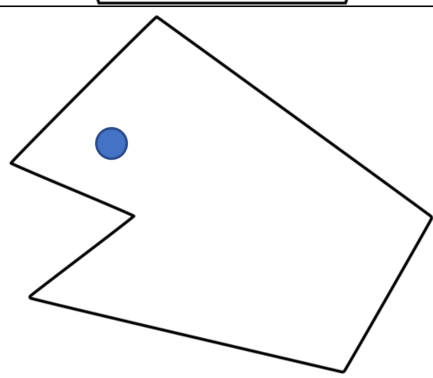
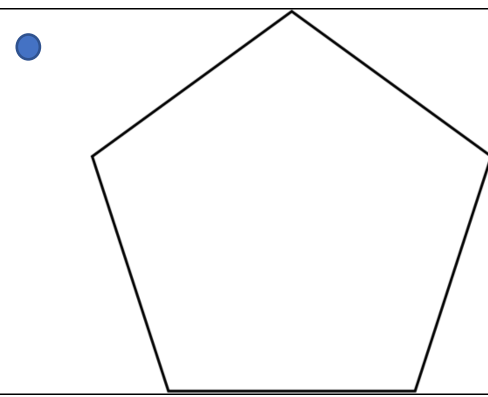
In addition to the lab report, you will be required to provide a working implementation of your algorithm. Your implementation should be accompanied by a handful of both simple and more complex test cases that you came up with.

Problem 1: Determine if a Point is Located Inside a Polygon

Input:

- A sequence $<p_1, p_2, \ldots, p_n>$ of $n \geq 3$ 2D points. Each point is a pair of x and y coordinates. The points correspond to the vertices of a simple (non-intersecting) polygon. The polygon is connected by line segments between each adjacent pair of points, including a line segment from the last point to the first point.
- The x and y coordinates for a single point distinct from the vertex points.

Output: A Boolean value indicating whether the point is located inside the polygon.

Inside	Outside
	
	

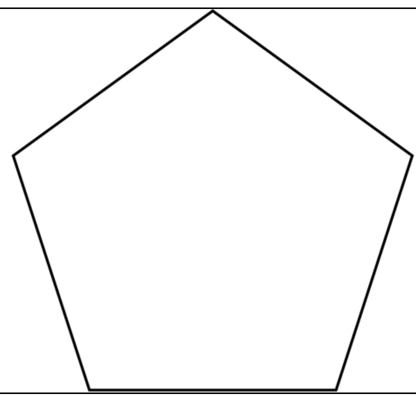
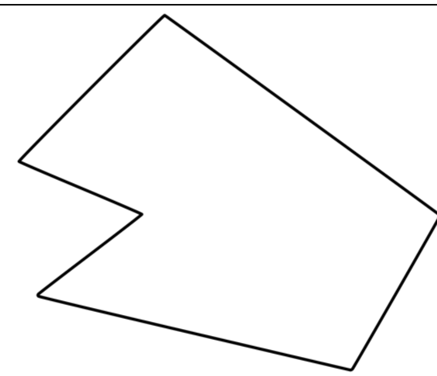
Hints:

- Draw examples of polygons with points located inside and outside the polygons. For each point, try drawing horizontal and vertical line segments that start at each point and continue infinitely in **one direction**. Consider any crossings between the line segments and the edges of the polygon.

Problem 2: Determine if a Polygon is Convex

Input: A sequence $<p_1, p_2, \ldots, p_n>$ of $n \geq 3$ 2D points. Each point is a pair of x and y coordinates. The points correspond to the vertices of a simple (non-intersecting) polygon. The polygon is connected by line segments between each adjacent pair of points, including a line segment from the last point to the first point.

Output: True if the polygon is convex. False otherwise.

Convex	**Concave**
	

Hints:

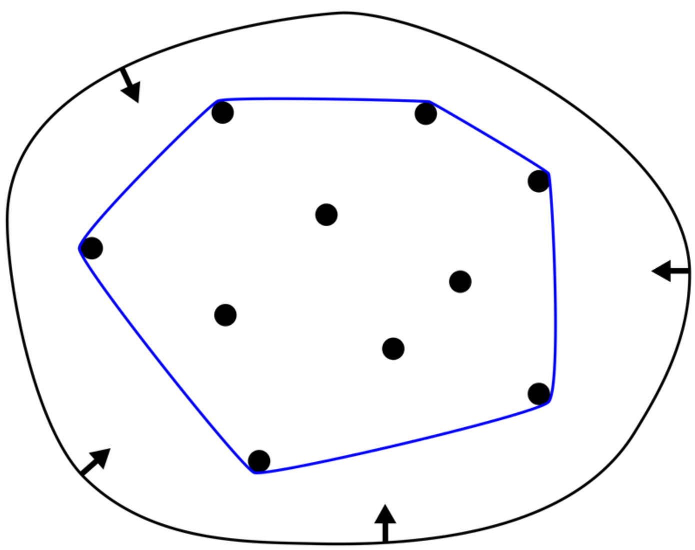
- Draw multiple examples of convex and concave examples of polygons. Consider the angles.
- Approach this problem as a proof by contradiction problem. You only need to find one example of a violation of the decision rule.

Problem 3: 2D Convex Hull

Assume you are given a set of points (the black dots below). The convex hull (blue) is the subset of points that form a convex polygon and encloses the original set of points.

Input: A sequence <p_1, p_2, …, p_n> of $n \geq 3$ 2D points. Each point is a pair of x and y coordinates.

Output: A subsequence of the input points that form the convex hull.



Hints:

- Sort the points by angle so you can process them in clockwise or counterclockwise order.
- Use a stack.
- Whenever a new point is being considered, see if it violates a requirement of convexity. If so, remove points so that convexity is restored.

<u>Submission Instructions</u>

Submit the lab report as a PDF, all source code in a zip file, and upload to Canvas.

Rubric

		Full Credit	Partial Credit	No Credit
Lab report writing and presentation quality	20%			
Decision Rule	10%	The decision rule is correct for all possible inputs that conform to the problem description	The described rule is correct for most inputs.	The decision rule is not correct for some common cases.
High-level Pseudocode	10%			
Justification of Correctness	10%	Uses techniques described in class to provide a solid and convincing argument that the algorithm is correct.	Provides a somewhat convincing argument that the algorithm is correct.	Argument contains one or more serious flaws.
Asymptotic run time analysis	10%	Analysis is correct for the provided pseudocode	Analysis is mostly correct except for minor flaws	Analysis is significantly flawed
Algorithm Implementation	10%	Implementation is faithful to the pseudocode description above and correct.	Implementation is mostly faithful to the pseudocode description above and correct for most inputs.	Implementation is not faithful to the pseudocode or not correct for some common inputs.
Test Cases	10%	Test cases consider a range of problem sizes and complexities and potential edge cases.	Limited number of test cases or only testing obvious or simple cases	
Benchmarks	10%	Benchmark experiments were set up and implemented correctly.	Benchmark experiments, implementations, or results are mostly correct.	Benchmark experiments, implementations, or results are flawed.
Algorithm run time	10%	Benchmark results agree with the theoretical run time analysis.	Benchmark results mostly agree with the theoretical run time analysis.	Benchmark results disagree significantly with the theoretical run time analysis.