Etan Soclof

Max Rectangle

1. Brute force: Iterate through every pair of points and calculate area of every possible rectangle to find the maximum area. The performance is O(n^2).

2. This greedy algorithm considers rectangles of largest width first by considering pairs of points at either end of the array. Note that there can only be a larger rectangle if there is another point of greater height than the point with the minimum height of the two being considered. Continue iterating toward the center of the array and ascertain if there is a rectangle of greater area by considering points of greater height than the minimum height of the two being considered.

3.

```
array = array of coordinates
int start = pointer to beginning of array
int end = pointer to end of array
maxArea = 0
left;
right;

while (start != end) {

    if (area of rectangle made by points array[start] and array[end]
> maxArea){
        maxArea = area of rectangle made by points array[start] and
array[end]
        left = start
        right = end
    }

    //Increment pointer to coordinate of minimum height
    if array[start] > array[end] {
        end--
    } else {
        start++
    }

}

return left, right, maxArea
```

4.
Feasibility - Two coordinates are returned which by definition represent a rectangle.
Optimality - The first coordinates considered - *a* and *b* - are the first and last coordinates in the array and therefore represent a rectangle of the largest width. For simplicity, assume that the y-coordinate (representing height) of *a* is less than the y-coordinate of *b*.
Every other possible pair of coordinates will represent a rectangle of lesser width. Therefore, the only way that there is another pair of coordinates c and d that represent a rectangle of greater area than this rectangle is if the coordinate of minimum height of *c* and *d* is of greater height than the coordinate of minimum height of the pair of *a* and *b*. It follows, therefore, that

there is no other combination of coordinates containing coordinate *a* that can represent a rectangle of a larger area because:

1. If the other coordinate's height is less than the height of *a*, the height of the rectangle represented by both coordinates is less than the height of the rectangle represented by *a* and *b*.

2. If the other coordinate's height is greater than or equal to the height of *a*, the rectangle formed by both coordinates will be of equal height as the rectangle formed by *a* and *b*. Because this new rectangle is of a lesser width than the rectangle formed by *a* and *b*, its area is less.

Therefore, coordinate *a* can be eliminated from consideration as it cannot be part of another rectangle of greater area that the rectangle formed by *a* and *b*.

Move to the next index of the array in the array closer to the center of the array to consider the rectangle of the next greatest width. Recursively repeat.