Atefe Rajabi

40230563

HW1 minimax

1. Generate Affine Functions (Section 1):

- The code defines a function `generate_affine_functions` that generates a specified number of random affine functions.

- Each affine function is represented as a dictionary with 'slope' and 'intercept' properties.

2. Plot Affine Functions (Section 1):

- The code uses Matplotlib to plot the generated affine functions.

- It sets x-axis ticks explicitly, ensuring precision of 1 between -10 and 10.

- The `x` values for the plot range from -5 to 5.

- Each affine function is plotted as a line on the graph.

3. Sort Functions Based on Slope Ascending, Intercept Descending (Section 2):

- The `affine functions` list is sorted based on two criteria: slope in ascending order and intercept in descending order.

- This sorting order is achieved by using the `sorted` function with a custom key function.

- The time complexity of this part is O(n log n), where 'n' represents the number of affine functions.

4. Finding Min of Maximums (Section 3):

- The `minimax` function calculates the minimum of maximum values for the sorted affine functions.

- It uses a stack `S` to keep track of points and iterates through the sorted functions to find min-max points.

- The minimum of maximums is determined based on the second element of the tuples in `S`.

- The result is returned as the minimum value of maximums.

- The time complexity of the function mentioned in the article, as stated in the problem document, is O(n).

5. Overall Report:

- The code successfully generates, visualizes, sorts, and analyzes a set of random affine functions.

- It finds the minimum of maximum values based on specific criteria.

- The precision of the x-axis ticks has been set to 1, which ensures a clear representation of the functions in the plot.

- The overall time complexity is O(n log n + n), where O(n log n) is for sorting n affine functions, and O(n) accounts for the minimax function.