Problem #3

Fibonacci Numbers

The lecturer gives pseudo code for three different implementations for calculating Fibonacci numbers:

- A naïve implementation that is correct, and horribly slow (exponential run-time)
- A memoized recursive implementation using memoization as an optimization
- A "bottom up" iterative implementation that is a further optimization to the first two

Using the pseudo-code presented in the video, implement all three algorithms. Then from method main, call each one to calculate the Fibonacci number for the values, {10, 20, 30, 40, 50}. For each implementation, output the number of nanoseconds each implementation took to calculate each of the values. Use these values to create a table in a format similar to the following:

	10	20	30	40	50
naive					
recursive					
bottom-up					

You can create the table by hand if you wish. However, extra credit can be earned by creating a program generated plot.

When you are finished, upload an image of your data table, along with the .java or any OPPs based code file(s).

Fibonacci Implementation and efficiency based in neno second

Implementations	Nth number			
	10	20	30	40
naive	8644	364091	5215040	65243513
recursive	24848	5941	3783	11343
bottom-up	2701	1621	1080	2161