# Assignment 3

## Discussion Questions (5 points)

1. In general, it is recommended to "filter early and often" in your Pig scripts. Why might this be beneficial for performance when Pig compiles the script into a MapReduce job?
2. Although Pig Latin is often called a procedural language, it lacks some features that are common in other procedural languages like Perl, including control structures (if/then/else).
    a. Why might control structures present a problem for Pig?
    b. If such control flow logic is needed, how could you incorporate it into a Pig script?

## Assignment (Total 15 points)

The Resources section contains two CSVs containing historical NYSE stock price and dividends data from 1970-2010 (for stock symbols starting with N)[1]:

The NYSE_daily_prices_N.csv data contains the following fields:

1. exchange
2. stock_symbol
3. date

[1] Infochimps "NYSE Data 1970-2010": http://www.infochimps.com/datasets/nyse-daily-1970-2010-open-close-high-low-and-volume

4. stock_price_open
5. stock_price_high
6. stock_price_low
7. stock_price_close
8. stock_volume
9. stock_price_adj_close

The NYSE_dividends_N.csv data contains:

1. exchange
2. stock_symbol
3. date
4. dividends

**(5 points)**

Load the above CSVs into Pig relations, and write a Pig script to join the price data with the dividends data, by both stock symbol and date.

**(5 points)**

Generate a new relation that projects the price change (closing price minus opening price) for each joined row, and retains the dividends.

**(5 points)**

Determine the big paying days by filtering on the rows where the *price change is greater than or equal to 1.0* and the *dividends is greater than or equal to 0.25*. Group these results and generate an aggregate count of these days by stock symbol.  Order it in descending order by count.

Submit the script and stored output.