In [3]:

```
# setup
from IPython.core.display import display,HTML
display(HTML('<style>.prompt{width: 0px; min-width: 0px; visibility: collapse}</style>'))
display(HTML(open('rise.css').read()))

# imports
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
sns.set(style="whitegrid", font_scale=1.5, rc={'figure.figsize':(12, 6)})
```

CMPS 2200

Introduction to Algorithms

Midterm Review

Module 1: Model of Computation and Asymptotic Notation

Lecture Topics:

- Asymptotic Notation (O, Ω, o, ω)
- Work, Span, Speedup, Amdahl's Law
- Functional languages and SPARC
- Computational costs, Parallelism, Work Efficiency, Scheduling

Homework 1

- Asymptotic analysis
- SPARC to Python
- Longest run (a divide-and-conquer algorithm, actually)

Lab 1

- Comparison of the empirical performance linear and binary search.

Lab 2

- Calculating the values of recurrences

Skills

- Knowledge of our model of computation (parallelism, speedup, work, span, work efficiency)
- Know the difference between functional and imperative expressions
- Translate from SPARC and Python
- Given a function of n, derive asymptotic bounds

Module 2: Recurrences

Lecture Topics:

- Why do we care about recurrences?
- Tree Method
- Brick Method
- Example: Integer Multiplication

Homework 2

- More recurrences using the brick method.
- Comparing the asymptotic performance (work, actually) of 3 algorithms
- Implement Karatsaba-Ofman

Lab 3

- Practice the tree method and the brick method.

Lab 4

- `map` and `reduce` for word occurrences and sentiment analysis

Skills

- Write a recurrence that captures the behavior of a given algorithm
- Understand the computation graph for an algorithm execution
- Application of the tree and brick methods to analyze a given algorithm

Module 3: Sequences

Lecture Topics:

- Abstract Data Types
- Operations on sequences

- Reduce and Iterate
- `scan` (using contraction)

Module 4: Divide-and-Conquer

Lecture Topics:

- Reductions, Search Spaces and Brute-Force
- Divide and Conquer Framework
  - Correctness using induction
  - Running time using recurrences
- Examples:
  - MergeSort
  - Karatsaba-Ofman
  - `reduce`
  - `scan`
  - eTSP
  - MCSS

Homework 3

- Finding an element in an unsorted list using Divide and Conquer
- Parenthesis matching

Lab 5

- Count Sort

Skills

- Understand how sequence operations such as `reduce` and `scan` are parallelized
- Applying sequence operations to solve problems efficiently
- Provide a brute force search algorithm along with its work/span for a given algorithm
- Understand reductions and how they relate to problem complexity
- Devise divide and conquer algorithms and prove their correctness and running time