Spreadsheet: Graphs and MVC

- **Model, View, Controller is MVC**
  - Model stores and updates state of application
    - Example: calculator, what's the state of a GUI-calculator?
  - When model changes it notifies its views appropriately
    - Example: pressing a button on calculator, what happens?
  - The controller interprets commands, forwards them appropriately to model (usually not to view)
    - Example: code for calculator that reacts to button presses
    - Controller isn't always a separate class, often part of GUI-based view in M/VC

- **MVC is a fundamental design pattern: solution to a problem at a general level, not specific code per se**
  - Patterns part of new foundation of object-orientation
Spreadsheet Model and graphs

- **What happens when we enter a number, string, or formula?**
  - Model changes to record new entry in a cell
    - Propagate change to views
  - Model changes to incorporate dependent calculations
    - What about D10 change when B10 changes below?

- **Graphs: why?**
- **Controller?**
- **View?**
- **Dump command?**
Toward a working SSModel subclass

- How can we associate expressions with cells?
  - Effects of the command \texttt{SET A2 = SUM(B1:B10)}
  - We need to get/set expressions, storage possibilities?
  - How do we notify views when change occurs?

- How can we propagate changes to dependent cells?
  - What associations made for \texttt{SET A2 = SUM(B1:B10)}?
    - Edge from B1 to A2, why?
    - Edge from A2 to B1, why?
    - Given a graph, how do we propagate changes to model?
  - Can we just call View::set for every cell and every view?

- Use cell/row/col functions in util.h for help with model code
Handling exceptions

- **How can we handle unanticipated, but certain to occur errors?**
  - Consider command `SET A2 = AVERG(B5:B1)`
  - Errors can occur deep in parsing code (recursive)
  - Where should error be handled?

- **C++ and Java use exceptions to indicate exceptional condition.**
  - Errors indicated by *throwing* exception
  - Client code *catches* the exception (or program aborts)
  - Client try/catch blocks guard against uncaught exceptions

- **Exceptions in spreadsheet are runtime_error objects**
  - Catch pointer, report using `re->what()` string