Project 1

- Choose either project 1A or 1B.
- **Deadline: February 26, 2015**.
- Submit the group reports in the lecture. Submit the source code on Blackboard by 11:59 PM (end of the day). For the source code submission. I expect one submission per team, but mention the team members clearly in your submission.
- Please stick to the teams that I assigned you with.

Project 1A

Develop a program to maintain two lists of homework assignments: assigned assignments and completed assignments.

Your program should support the following technical requirements:

Technical Requirements

- (Weight: 10%) Keep track of the assignments using doubly-linked lists.
- (Weight: 10%) Use iterators to iterate through the lists.
- (Weight: 5%) Display the assignments (the assigned and completed assignments)
- (Weight: 10%) Add a new assignment (to the assigned list or the completed list).
  Don't add the assignment in these cases:
  The given dates (assigned or due date) are invalid.
  The due date is less or equal to the assigned date.
  An assignment with the given assigned list already exists in the assigned list.
- (Weight: 5%) Complete an assignment with a given assigned date and completion date. You first need to find the assignment with the given assigned date. When the assignment is completed, remove it from the assigned list and add it to the completed list. If the assignment is completed after the due date, mark it as late.
  Don't complete the assignment in these cases:
  The assignment with the given assigned list doesn't exists in the assigned list.
  The given assigned or completion dates are invalid.
- (Weight: 10%) Edit an assignment in the assigned list (e.g. edit the due date or the description). You first need to find the assignment with the given assigned date.
  Don't edit the assignment in these cases:
  The assignment with the given assigned list doesn't exists in the assigned list.
  The given assigned or due dates are invalid.
- (Weight: 10%) Keep the assigned assignments ordered by due date (in ascending order).
- (Weight: 10%) On demand, count the number of late assignments.
- (Weight: 15%) Your program should be menu-based. Hence, the user can choose which command to run(display assignments, add assignment, edit due date, edit description, complete assignment, number of late assignments, save, exit)
- (Weight: 10%) Initially you should read assignments from a file, and populate your assigned list.
- (Weight: 5%) Write a function that overwrites the assignment file to reflect the changes (e.g. newly added, edited assigned and completed assignments). The function can be called on demand.

Facts

- An assignment has these attributes:
  - **`dueDate.`** Since there is no direct way to define date and time in C++, you are welcome to use a `Date` I wrote (it has year, month, day components). You can find a project that shows examples of how to use the `Date` class here: https://www.dropbox.com/sh/1bfc5ayty5kl16x/AAD-bO28NQX7RaFQj39ZRQV1a?dl=0
  - **`description`**. (`string` type)
  - **`assignedDate`**. The date the assignment was assigned. This is also a Date type. *This value is unique for assignments in the assigned list. No two assignments have the same assignedDate.*
  - **`status:`** an enum type that tells us whether the assignment is assigned, completed, or late.
- The assigned list has all the assignments with the "assigned" status. The completed list has all the assignments with the "completed" or "late" status.
- The file that keeps track of the assignments is a text file, and it looks like this:

  ```
  2-11-2015, linked lists, 2-2-2015, late
  3-10-2015, stacks, 3-1-2015, assigned
  ```

  Each line is an assignments, and the information is structured like this: due date, description, assigned date, status
- You are welcome to use the source code on Blackboard (e.g. `StringTokenizer, list, Ordered_List`). You can also use the C++ built-in `list` class.

Project 1B

We can represent a polynomial as an ordered list of terms, where the terms are ordered by their exponents. For example:

$$3x^4 + 2x^2 + 3x + 7 \; added \; to \; 2x^3 + 4x + 5 \; \; is \; \; 3x^4 + 2x^3 + 2x^2 + 7x + 12$$

Technical Requirements

- (Weight: 10%) Use linked lists to keep track of the polynomials.
- (Weight: 10%) Use iterators to iterate through the lists.
- (Weight: 30%) Ask the user enter the polynomials in a user-friendly fashion. For instance, $3x^3 - x^2 + 1$ should be entered as: 3X^3 – 1X^2 + 1 (You may assume there is only one variable (X) and there are no spaces).

  Here are examples of some polynomials:

What the user enters	How the polynomial is stored
-X+5+X^2-10	$x^2 - x - 5$
5X-5X^-2+10-5X+X^2	$x^2 + 10 - 5x^{-2}$

  When you are reading a polynomial, sort the terms by exponent in descending order. Further, if the polynomial has terms with equal exponents, make sure you add the coefficients of that term. Hence, the polynomial should be sorted by term exponent, and there shouldn't be multiple terms with the same exponent. Further, don't keep track of terms with zero coefficients.
- (Weight: 10%) You should define a class Term that contains the exponent and coefficient. This class should implement comparison operators (e.g. <) to compare the exponents.
- (Weight: 20%) Add two polynomials.
- (Weight: 10%) Store the result in a new list, and show it to the user in a user-friendly fashion.
- (Weight: 10%) Your program should be menu-based. Hence, the user can choose which command (e.g. enter first polynomial) to run.

Facts and Assumptions

- A polynomial is a list (linked list) of terms.
- The coefficients and powers can be more than one digit.
- You are welcome to use the source code on Blackboard (e.g. `list`). You can also use the C++ built-in list class.
- You may assume the user enters one polynomial at a time.