Hugh A. Miles II

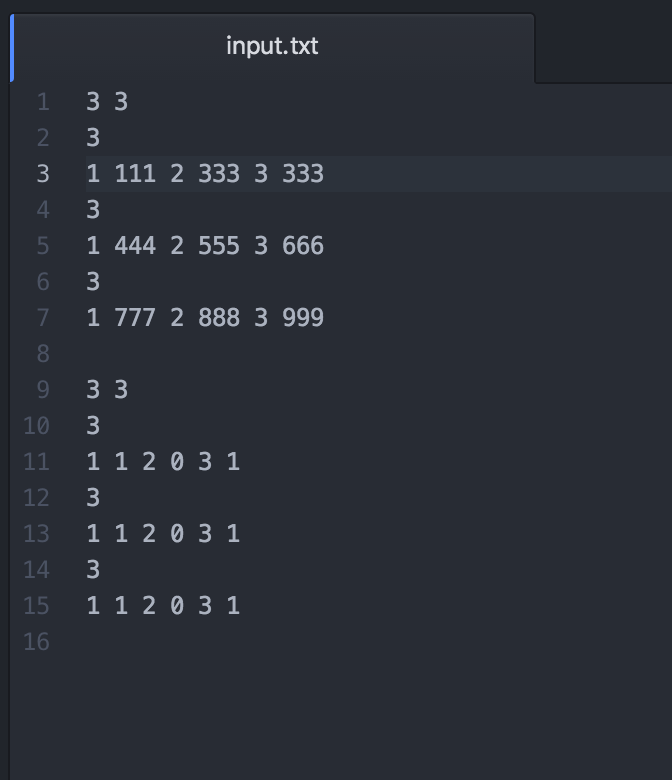
UFID: 78686913

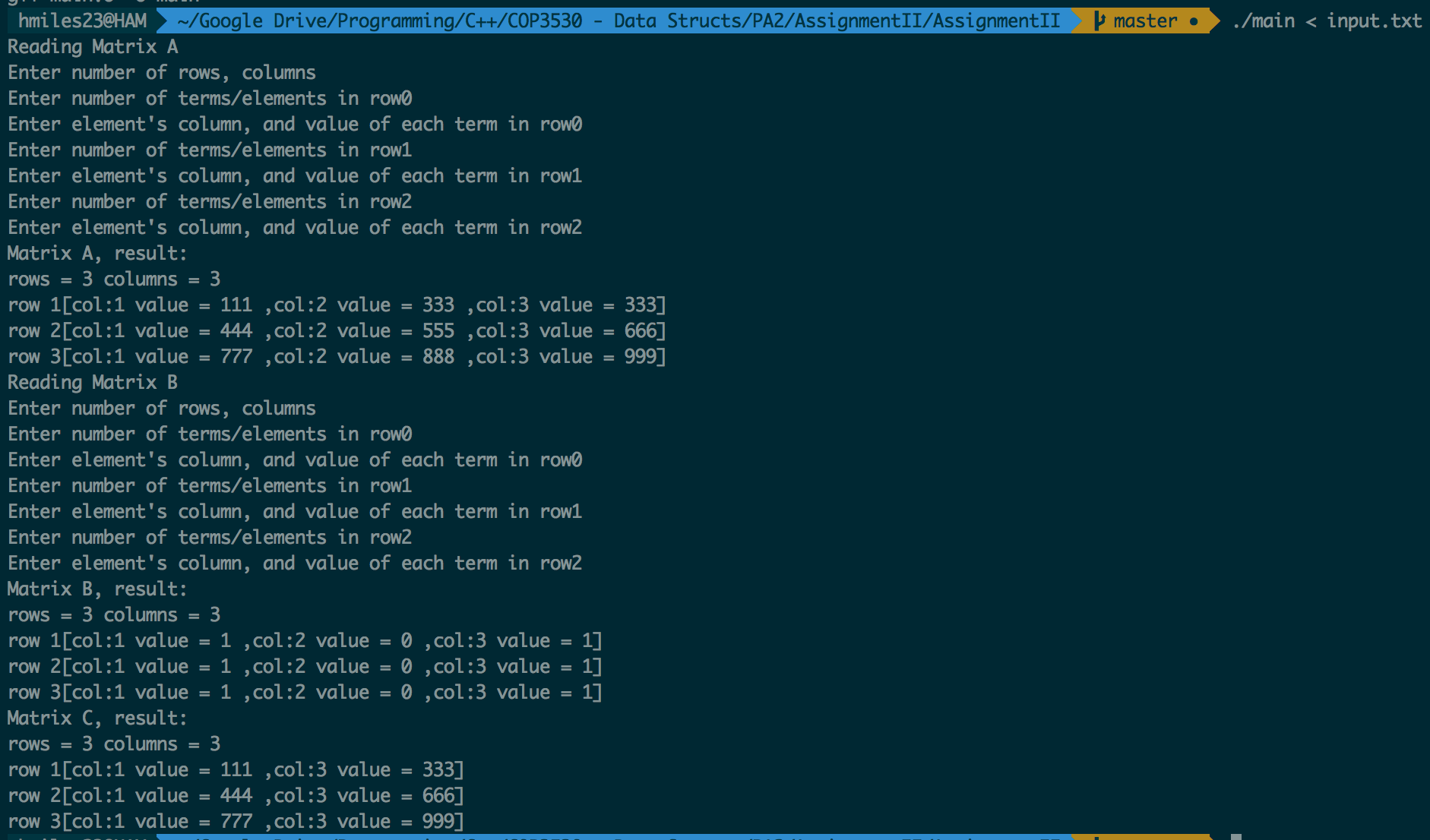
Section: 13A8

09/21/16

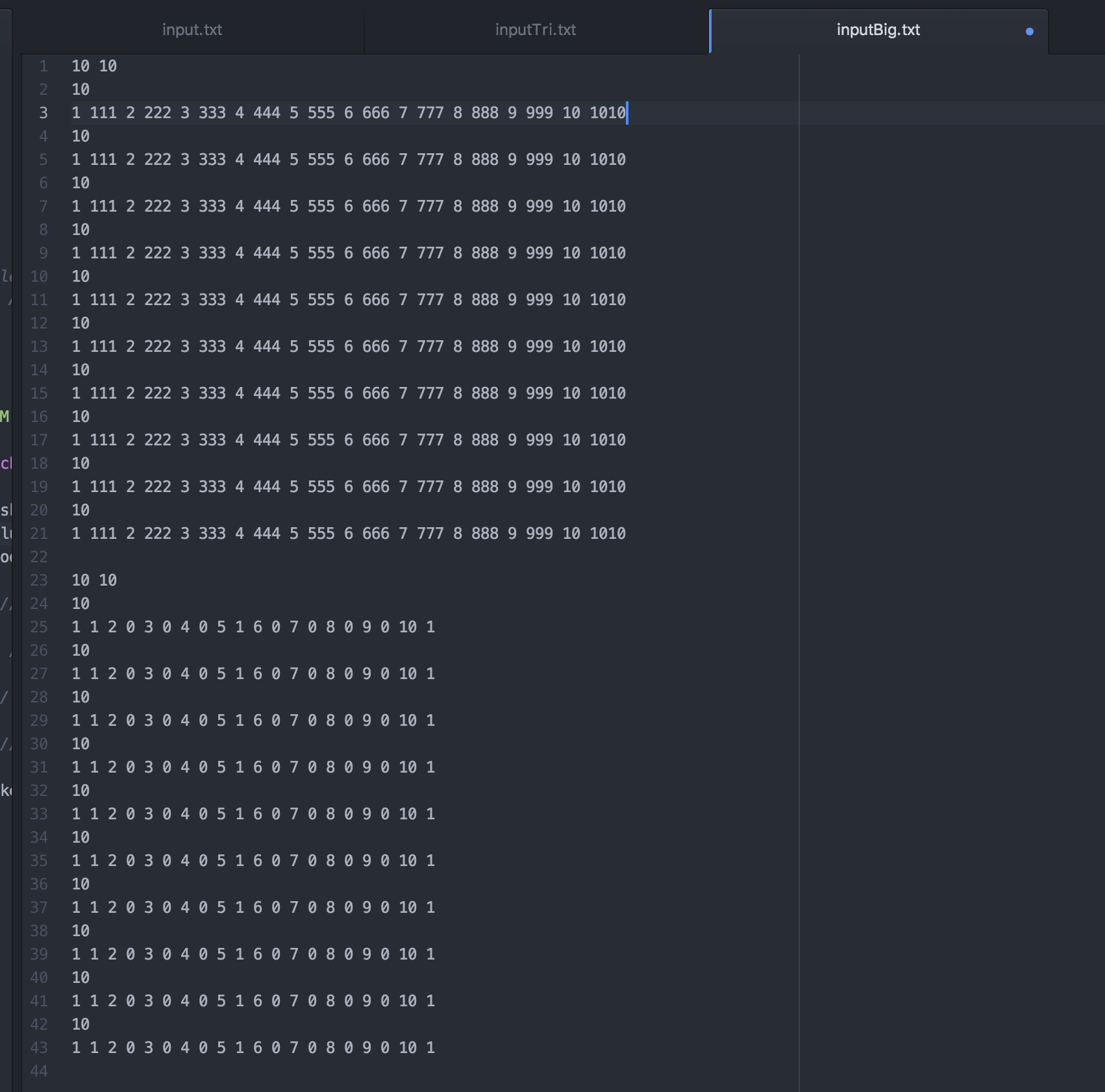
COP3530

Sparse Matrix

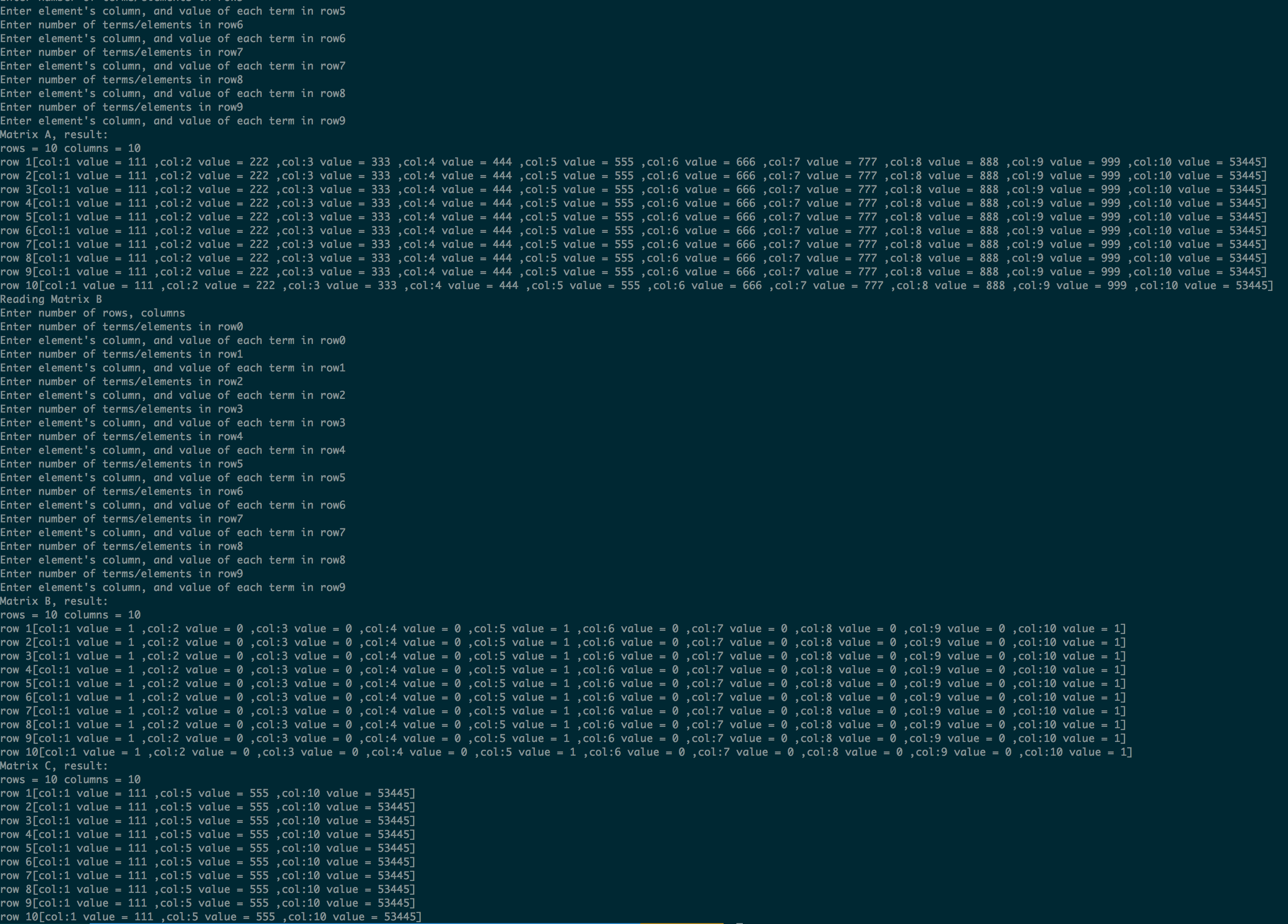
* Summary
  + My Sparse Matrix class incorporates an array of Linked List with the end of each list pointing to NULL. This is a template class so the Nodes can hold any type based upon the type template class at its instantiation. Each node has a value, col, and index.
* Structure
  + Node<type>
    - Constructor (type initValue, int strIndex)
      * In the constructor you need to indicate the value to be set, and where the Node is being placed with the strIndex.
    - Int col
      * Location in the list
    - Type value
      * The data value stored in side the node.
    - Node<type>\* nextNode
      * Address for the next Node in the list
  + LinkedList
    - Head<type>
      * Pointer to start of the list (index = 0)
    - Int length
      * Keeps track of the number of Nodes in the current list.
  + Sparse Matrix
    - read()
      * read cin for the user to create
    - print()
      * prints the matrix sequentially for nicer reading
    - mask(SparseMatrix<int> b , SparseMatrix <bool> b)
      * takes the b matrix and mask with all its values and with union it fillups matrix with the values needed
    - setRow
      * setter for creating the SparseMatrix rows dynamically
    - setCol
      * setter for col length
* Methods
  + Insert (type value, int index)
    - Allow users to insert element at a certain spot in the current list that is less than the this.length + 1.
  + getNode(int col)
    - Finds the node inside the list with the match col value
  + Print
    - Prints entire list sequentially.
  + Helper Methods
    - Append(type value)
      * Adds node with the value to the end of the list
* Test Cases
  + Small
    - Input 
    - Output



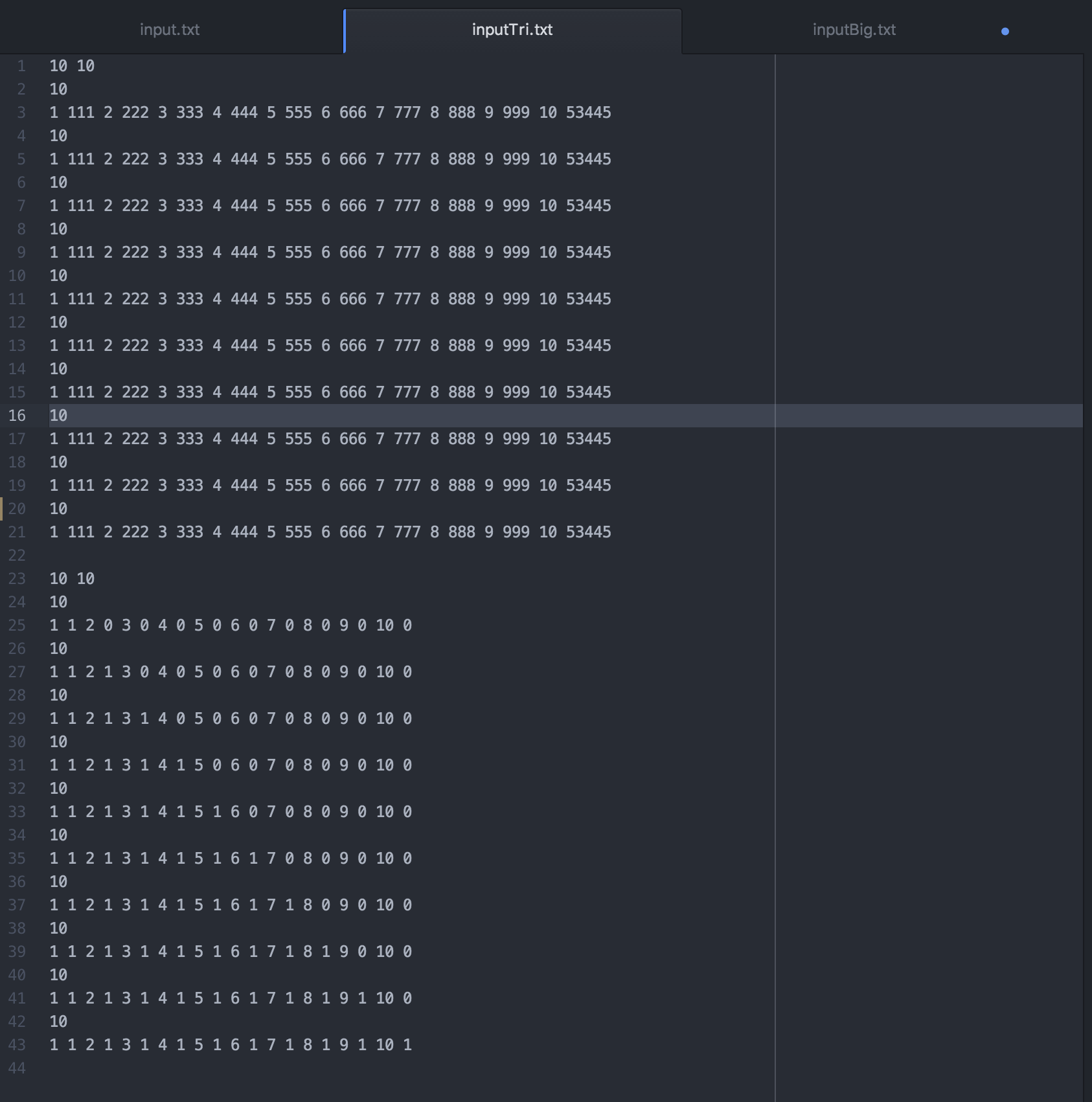
* Big
  + Input



* + Output



* Triangle
  + Input



* + Output

