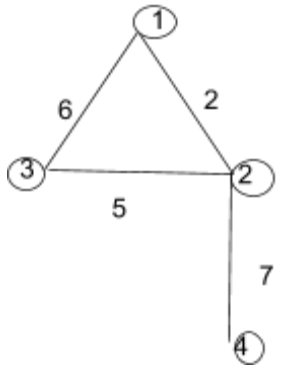
CS575 Programming Assignment 3
Due at 11:59PM April 6 (Submit through blackboard)

1. Implement the following minimum spanning tree algorithms.

(a) [30%] Implement Prim's algorithm. To implement a priority queue in Prim's algorithm, simply use a 1-dimensional array.

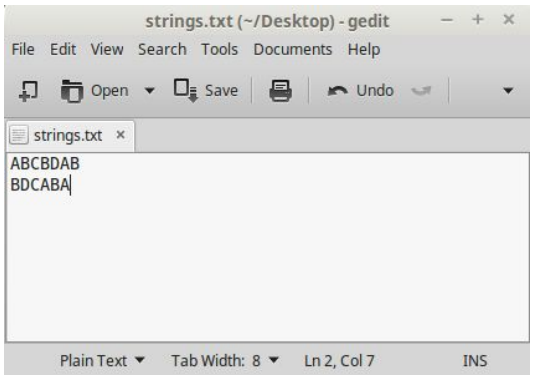
(b) [30%] Implement Kruskal's algorithm using the find3() and union3() functions discussed in class. (You will get zero, if you use other algorithms unless you can formally prove that your methods are correct and more efficient than find3() and union3().) For sorting the edges in the input graph, use the *randomized* quicksort algorithm.

In your Prims' and Kruakal's programs, print the edges in a minimum spanning for the following graph. Pass the adjacency matrix of the graph below as a file named "graph.txt". Make sure your programs work correctly for any other graph passed as "graph.txt" too. In the adjacency matrix value 999 is used to represents no direct edge between two vertices.

Graph diagram (sample)	graph.txt
	0,2,6,999 2,0,5,7 6,5,0,999 999,7,999,0

2. [30%] Find the longest common subsequence (LCS) between two input character strings x and y. Implement the LCS algorithm discussed in Chapter 7. Pass x and y in a text file named "strings.txt". Print a LCS for x=ABCBDAB and y=BDCABA. (Write the first string x in the first line and the second string y in the second line in "strings.txt".) Make sure your program successfully finds a LCS for any other pair of strings passed as "strings.txt".

strings.txt will look like as below:



3. [10%] Coding style: Write meaningful comments, while making your code structured, easy to read, and robust.

Instructions for project submission:

1. Your code has to work correctly in **remote.cs.binghamton.edu**. You can develop it in any environment as you wish but make sure you test it and run it in the retome.cs.binghamton.edu and package it from the same server before submitting. If your code does not work in the mentioned server you will receive no grade points for the respective algorithm. No exception will be considered.

2. Project name should be like this: <language code>_<userid>_proj3.tar.gz, all lowercase letters and **tar.gz** file format
   Example: **c_psaha4_proj3.tar.gz** or **cpp_psaha4_proj3.tar.gz**
   Points will be deducted for bad project naming.

3. Your source code file name have to be exactly as below:
   prim.c, kruskal.c and lcs.c (if you are using CPP then use cpp extension instead of c)

4. Project should **not contain** any **object, executable, readme, input** files. So, make sure you clean it before submitting it. Points will be deducted for presence of unwanted files. While testing,TA will put graph.txt and string.txt into your project folder to test your projects.
   So your project package will look like as below:

5. "**make**" command will **only compile** all three source code files and it **should not run** any of the program. After executing "**make**" command it should create three object files with below mentioned names.
   prim.out, kruskal.out, lcs.out
6. Prim's algorithm should be executable by "**./prim.out graph.txt"** command, Kruskal's algorithm should be executable by "**./kruskal.out graph.txt"** command and LCS should be executable by "**./lcs.out string.txt"** command. If its not running by these commands then expect zero points for respective algorithms.

7. Please don't apply your common sense and intuition for packaging, strictly follow mentioned instructions.
8. Make sure you don't look into others code and write it by your own. Copying from internet sources is strictly not allowed. If you are found doing so, you will receive zero points for the algorithm.
9. If BlackBoard says your submission is late it will be considered as late and 10% will be deducted.
10. Contact the TA (psaha4@binghamton.edu) for any clarification needed.