CSCI36200 **Student Name:** ____________________
Spring 2015 **Student ID:** ____________________
Bonus Homework Assignment 3
Due Date: April 29 (hand in BEFORE class)

Instructions:

This homework assignment contains 4 pages (including this cover page) and 3 questions.

Please **PRINT** your name in the "Student Name" box, and your student ID (as it appears in Canvas) in the "Student ID" box and on the running header of each page. Make sure your handwriting is **UNDERSTANDABLE**. If the TA or the graders cannot recognize your name, ID or your handwriting when they do the grading, you will lose 20 points (nonnegotiable!). Please **STAPLE** all the pages in right order, otherwise you will lose 10 points (nonnegotiable!)

Please **print** out the sheets and answer the questions on the sheets like you do in an exam.

Grade Table (for TA/grader use only)

Question	Points	Score
1	40	
2	10	
3	20	
Total:	70	

1. (40 points) Read the code of function one_pass() and all_passes() in Algrithm 1. The merge function

   merge(int list [ ], int sorted [ ], int i, int m, int n)

   in Algorithm 1 is to merge two sorted sub-arrays list[i], ..., list[m], and list[m+1], ..., list[n], into the array sorted[ ]. Both the array list[i], ..., list[m], and list[m+1], ..., list[n] are already sorted in increasing order and after merging, the array sorted[ ] is also in increasing order.

   1. Run the all_passes code on an input list 4, 1, 5, 10, 8, 9, 6, 12, that is, int list[ ] = { 4, 1, 5, 10, 8, 9, 6, 12 }, and n = 8. Use a big value for MAX_SIZE as necessary. Demonstrate the results step by step. You need to show the results (how list and extra look before and after calling one_pass) for line 1, 2, 3, 4 and 5. For each iteration in the for loops, you need to explicitly show the results. You also need to show the results after all_passes is done. (20 points)
   2. What does all_passes() do? (5 points) What does one_pass() do? (5 points). Explain why and how all_passes() is able to do the job using while loops, length, merge and one_pass() (10 points).

Algorithm 1 all passes

```
void one_pass(int list[ ], int sorted [ ], int n, int length){
    int i = 0, j = 0;
    for (i = 0; i ≤ n - 2 * length; i += 2 * length)
        merge(list, sorted, i, i + length - 1, i + 2 * length - 1); // 1
    if (i + length < n)
        merge(list, sorted, i, i + length - 1, n - 1); // 2
    else
        for (j = i; j < n; j ++)
            sorted[j] = list[j]; // 3
}
void all_passes(int list[ ], int n){
    int length = 1;
    int extra[MAX_SIZE];
    while (length < n) {
        one_pass(list, extra, n, length); // 4
        length *= 2;
        one_pass(extra, list, n, length); // 5
        length *= 2;
    }
}
```

2. (10 points) Demonstrate the results of quicksort on { 4, 1, 5, 10, 8, 9, 6, 12 } step by step (i.e., swap the pivot, split the array, etc). Always use the first number in the array that is to be sorted (in increasing order) as the pivot.

3. (20 points) Demonstrate the results of single source shortest paths from v0 step by step (i.e., fill out the tables as we demonstrated in the class) in the following graph.

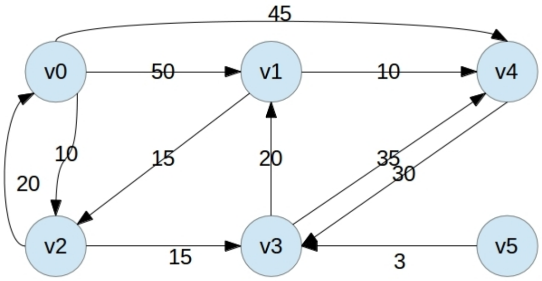

Figure 1: a weighted directeg graph