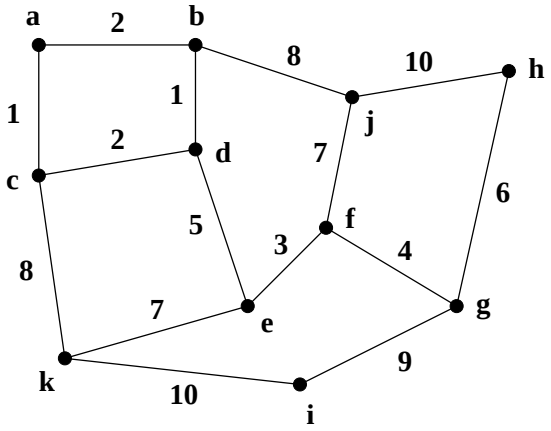
University of Nevada, Las Vegas Computer Science 477/677 Fall 2015

Assignment 6: Due October 20, 2015

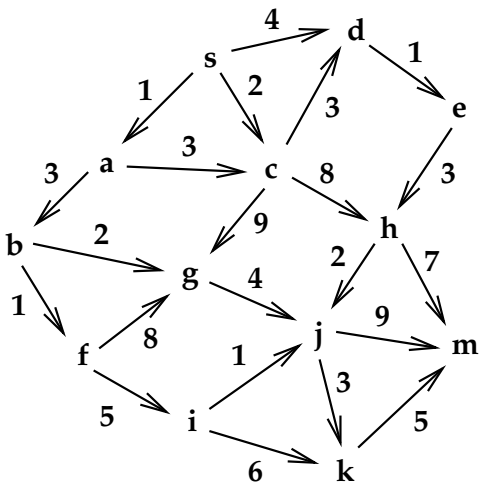
Name:_____

You are permitted to work in groups, get help from others, read books, and use the internet. But the handwriting on this document must be your own. You may attach extra sheets, using a stapler.

1. A connected weighted graph G is illustrated in the figure below. Find the minimum spanning tree of G using Kruskal's algorithm. Use the union-find data structure, in the manner shown in class, and show the forest at each step.



2. A weighted directed acyclic graph G is illustrated in the figure below.



(a) Write the array of in-neighbor lists of G. Be sure to include the weights.

(b) Using that array as input, find a topological order of the vertices of G. Show your work. When you correct something, do not erase it; simply cross it out and write the correction next to it.

(c) Using that topological order and dynamic programming, find the minimum weight path from **s** to **m**. Show the arrays of best weights and back pointers. When you correct an entry of one of those arrays, do not erase it; instead, simply cross it out.

3. It is confusing to think about the loop invariant of a for loop. It helps to first replace it with an equivalent while loop. For example the for loop

```
for(int i=1; i< n; i++){
  whatever;
}
```

is equivalent to:

```
int i = 1;
while(i < n){
  whatever;
  i++;
}
```

A loop invariant need not be expressed as an equation; it can be written as a sentence. For example consider the following code:

```
int s = 0;
for (int i = 0; i < n; i++){
  s = s + A[i];
  }


int m = A[0];
for (int i = 1; i < n; i++){
  if (A[i] > m)
    m = A[i];
  }
```

The first loop invariant is, "**s** is the sum of the first **i - 1** entries of the array **A**," and the second loop invariant is, "**m** is the maximum of the first **i - 1** entries of the array **A**."

The following is code for sorting an array **A** of length n, again with indices ranging from 0 to $n-1$.

```
for (int i = 0; i < n; i++){
  int j = i;
  for (int k = i; k < n; k++){
    if (A[k] < A[j])
      j = k;
    }
  swap(A[i],A[j]);
  }
```

(a) Rewrite the code, replacing each for loop by an equivalent while loop.

(b) Write two loop invariants, one for each loop. Indicate exactly where in the code each invariant holds.