Ali Mirabzadeh
305179067
CS180 – HW5
3.

a. The correct answer is 3; however, the algorithm returns 2 if we do the following graph with (v1,v2), (v1,v3), (v2,v5), (v3,v4), (v4,v5). It goes (v1,v2) and (v2,v5). The correct answer would be form (v1,v2), (v2,v4) and (v4,v5)

b. We use DP to solve this problem. Therefore, we use optimal subproblems, call it M.
We assign, M[1]= 0, path from node one to itself and for all other assign to infinity.
Then for all edges (i,j) if the current edge is not infinity then
Check if M< M[current edge] +1
    M = M[current edge] +1
After checking for all the edges
M[j]= M
After the first loop
Return M[n] which would be length of longest path

Ali Mirabzadeh
305179067
CS180 – HW5
4.

a. counter example:
M=10

	1	2	3
NY	1	4	1
SF	10	1	15

The algorithm would chose NY, SF, NY: 1+1+1+13 ; however the actual answer would be NY, NY, NY: 1+4+1+0= 6

B.
M=10

	1	2	3	4
NY	1	50	1	50
SF	50	1	50	1

NY,SF, NY,SF + 3M= 34 and it moves 3 times. Other plans at least pay 50 so not optimal.

c. We use DP and define two OPT one ending in SF and one in NY.

$OPT_{NY}(0) = OPT_{SF}(0)=0$
Then for each month i=1,2,3,…n
$OPT_{NY}(i)$= cost of current month for NY + min($OPT_{NY}(i-1)$, M+ $OPT_{SF}(i-1)$)
$OPT_{SF}(i)$= cost of current month for SF + min (OPT_{SF}(i-1, M+ $OPT_{NY}(i-1)$)
End the loop
Then return the smaller of OPT

Ali Mirabzadeh
305179067
CS180 – HW5
6.

Based on the question, we can see that the last word in a line is w_n.
I used DP to solve this problem. I define OPT[i] to be the optimal value on the set of words. Also, for any i less than or equal to j, let $S_{i,j}$ denotes to the slack of line containing the words $w_{i,}$……, w_j. Also, $S_{i,j}$ to be infinity if these words exceed total length L.
As mentioned the optimal solution must begin the last line somewhere at word $w_{j,}$ and solve subproblems recursively. Hence, we would have the following.

$$OPT[N] = \min \ S_{i,n}^2 + OPT[j-1]$$

Here you can see the complete algorithm

First, you compute all $S_{i,j}$
Then, set OPT[0]=0
For k=1,2,3…,n
    OPT[k] = min ($S_{i,n}^2$ + OPT[j-1])
End for
Return the OPT[N]

Ali Mirabzadeh
305179067
CS180 – HW5
12.

I again use DP. This time OPT[j] would denote the minimum cost of solution on servers 1,2,…,n. Note that we place a copy of that file at server j. We want to search for possible places to put the highest copy of the file before j. Say in the optimal solution this at position i. Then the cost for all server up to i is OPT(i). Then we calculate all the sum of the access costs for i+1 through j which then become $({}^{j-1}{}_{2})=0+1+\ldots.+(j-i-1)$. Also c_j would be the cost at the server j. The value OPT can be built in order of increasing j.

$$OPT(j) = c_{j\,+} \min\left(({}^{j-1}{}_{2}) + OPT(i)\right) ;\ OPT(0)=0 \ \&\ ({}^{1}{}_{2})=0$$