CSE 321 - Introduction to Algorithm Design
Homework 03

1. If I write my thought about this film in English , I may not manage to convey my thoughts to you. So I write this part of the homework in Turkish.

   ”Bu film gibi gerçek hayatdan ve geçmişten uyarlanarak hazırlanan filmler her zaman hoşuma gitmiştir. Bu film de geçen bir söz var ”Türkiye’de hiçbir başarı cezasız kalmaz.”. Bu sözün gerçekliğini çok iyi bir şekilde gösteren bir film olmuştur. Filmde de gösterildiği gibi Türkiye’deki mühendislerin bir araya gelerek güzel şeyler ortaya çıkarabilcekleri görülüyor ama her zaman bu işleri baltalayan ya da yapılan bu zor ama imkansız olan işler üzerinde dışarıdan bilinçsizce yorumlar yapan insalar olmuştur. Ama önemli olan bu insanlara karşı pes etmek değil bu insanlara karşı güzel işler ortaya çıkarak savaşmak gerekir. Belki işin sonuda kaybedebilirsin ama sonuçta savaşmış olacaksın ve bu sana hep birseyler katacaktır. Türkiye’de eğer iyi ve büyük bir iş yapmak istiyorsan arkandan seni her zaman destekleyecek büyük birilerinin olması gerçeğini de göstermek istemiştir bu film. Yani özetlemek gerekirse film güzel gerçekleri iyi bir şekilde gün yüzüne çıkarmış ama gerçekler kötüdür. Yani ’Türkiye’de hiçbir başarı cezasız kalmaz.’ ”

2. Firstly, I found all the possible results by using permutation then I calculated the cost(min. work times per week) of these possible results and I choose the result that has minimum cost.
   Complexity analysis of the algorithm:
   Let say that number of RA is **n** and number of courses **m**.

   - **Best Case:**
     For the best case m= n.
     Permutation function complexity is n!
     And the last for loop is executed n*n*(n-1) times.
     So the best case is

     $$B(n) = n! + n * n * (n-1) = \theta(n!) \tag{0.1}$$

   - **Worst Case:**
     For the worst case n is greater than m : So n - m iteration.
     Permutation function complexity is n!
     And the last for loop is executed n*n*(n-1) times.
     So the worst case is

     $$W(n) = (n-m) + n! + n * n * (n-1) = \theta(n!) \tag{0.2}$$

   - **Average Case:**
     The average case is

     $$A(n) = \theta(n!) \tag{0.3}$$

3. Firstly , I made the control that if the cost of building a lab is less than repairing a road or not. If it is , I calculate the result by calculating (building a lab) * (number of departments). If it's not then I created 2 array(parentArr and identified). Then I made BFS and I found the parentArr that tells the connected graphs. After I found the parentArr , the minimum cost is calculated by using this array.
Worst case analysis of the algorithm:
Let say that department number is **n** , road number is **m**.
The worst case of the BFS function is

$$W(n) = O(n + m) \tag{0.4}$$

This function is working n times to find connected graphs. So the worst case of the all algorithm is

$$W(n) = O(n^2 + m * n) = O(n^2) \tag{0.5}$$

4. The list is 12 , 34 , 54 , 2 , 3
Insertion Sort :
Assume that i= 1 for the second element of the list. (i is smaller than size of the list)

- **12** - 34 - 54 - 2 - 3 : i= 1. Second element (34) is not smaller than first element (12) so no need to moving.
- **12 - 34** - 54 - 2 - 3 : i= 2. Third element (54) is not smaller than previous sorted part so again no need to move any element.
- **12 - 34 - 54** - 2 - 3 : i= 3. 2 will be moved to the beginning of the list and other elements (12 - 34 - 54) will move 1 step to the right from their current position.
- **2 - 12 - 34 - 54** - 3 : i= 4. 3 will be moved to the position that is after 2 and other elements (2 - 12 - 34 - 54) will move 1 step to the right from their current position.
- **2 - 3 - 12 - 34 - 54** : Sorted list.

Shell Sort :
Assume that gap= (size/2)= 2.

- 12 - 34 - 54 - 2 - 3 : Compare 34 and 2 and swap.
- 12 - **2** - 54 - **34** - 3 : Compare 54 and 3 and swap.
- 12 - 2 - **3** - 34 - **54** : Compare 12 and 3 and swap.
- **3** - 2 - **12** - 34 - 54 : Now gap=1 and apply insertion sort algorithm. Compare 3 and 2 and swap.
- **2 - 3 - 12 - 34 - 54** : Sorted list.

Advantages of shell sort compared to insertion sort :

- Shell sort is faster than insertion sort.

- More efficient to swap elements that are far away from each other.
- Shell sort is more efficient for small lists.