CSE 321 – HW3 RAPOR

1. **Devrim Arabaları :**

   Türkiye'nin ilk yerli otomobilleri devrim arabalarının kısıtlı bir sürede 23 türk mühendisi tarafından tamamen yerli üretilen parçalarla yapılması gurur verici. Fakat bu araçların üretiminin durdurulması gerçekten üzücü. Devrim Arabaları, Türkiye'de yapılmaz denen bir çok şeyin, hatta herşeyin yapılabildiğini gösteriyor. Türkiye herşeyi üretebilecekken, sürekli dışa bağımlı olması ve bazı şeyler için yapılması imkansız denmesi herşeye engel oluyor. Aslında Türkiye herşeyini kendi üretebilecek güçtedir. Bu filmde mühendislerin ne kadar kısa süreleri olursa olsun, gecelerini gündüzlerine katıp, nasıl başarıya ulaştıklarını görüyoruz. Fakat sonrasında ise bir benzin eksikliği yüzünden Devrim Arabaları'nın gündemden düşmesi, insanların olayı çarpıtması ve sonrasında üretimden kaldırılması üzücü bir durum. Yerli araba üretimi gibi, yerli uçak üretimi veya herhangi bir alanda gerekli olan ürünlerimizi kendimiz üretebilecek yetkinliğe sahipken bunu yapmamamızın yanlış olduğunu anlatıyor bu film.

2. Complexity :

helper isimli fonksiyon R.A sayısını kullanarak(r) permütasyonların hepsini bulduğundan, bu fonksiyonun complexity'si θ(r!) dir. Sonrasında bu r! farklı olasılıkta, n(kurs sayısı) kadar toplama işlemi yapıp tempe atma işlemi olduğundan(her birinde r eleman var, ama n tanesi kullanılacak),
complexity θ(n*r!) dir.

3.

Algoritma :

Öncelikle mapOfGTU ve start node inputları alan ve node vectoru döndüren bir bfs fonksiyonu yazdım.Bunu connectedCount isimli fonksiyonun içinde çağırıp, döndürdüğü vectordeki değerlerin hepsini işaretledim. Her bfs çağırışında count'u arttırdım. Böylelikle connected Graph sayısını elde ettim. findMinimumCostToLabifyGTU içerisinde ise connected sayısı * x + yol sayısı(vertex sayısı - connected sayısı)* y formülünü return ettim.

Worst Case :

Worst Case' de 1 tane complete graph olmasıdır. Bu durumda connectedCount fonksiyonu v kadar kesin dönecektir. Döngü içerisinde ise visit edilmemiş node'lar için bfs

yapacaktır. İlk bfs yapışında, bfs bütün node'ları gezecektir. Bunu yaparken, edge'ler yerine adjacency list'in value vectorlerini alıp(komuşular) işlediği için v defa dönecektir(bütün edgelere göre döngü yapsaydı, V+E olacaktı). connectedCount fonksiyonunda alınan bu bfs vectorunu(bütün vertexleri içerir), v kez dönen bir döngüyle visited olarak işaretleyecektir. Bir kez bütün vertexler işaretlendiği için, ana döngü devam ederken bir daha bfs ve işaretleme işlemi yapılmayacaktır. Bu nedenle en dış döngü v defa dönmeyi tamamlayacaktır. Dolayısıyla worst case = (en dış)v + (1 kez dfs)v + (1 kez işaretleme)v ' den
$\theta(3v) = \theta(v)$' dir.

4.

Insertion Sort

12 , 34 , 54 , 2 , 3

12, 34 , 54 , 2 , 3

12 , 34, 54 , 2 , 3

2 , 12 , 34 , 54 , 3

2 , 3 , 12 , 34 , 54

Shell Sort

12 , 34 , 54 , 2 ,3 (gap : 2)

12 , 34 , 54 , 2 ,3

12 , 2 , 54 , 34 , 3

12 , 2 , 3 ,34 , 54

3 , 2 , 12 , 34 , 54 (gap : 1)

2 , 3 , 12 , 34 , 54

Shell Sort'un, Insertion Sort'a göre avantajı : Birbirinden uzak noktalardan eleman swap ettiğimiz zaman(aradakiler sıralı), Shell sort gap'e göre yaparken, Insertion Sort birer birer swap ederek işlemi tamamlayacak.Shell Sort'un yaptığı swap işlemi sonucunda, elemanın doğru pozisyona yakın olma derecesi, Insertion Sort'unkinden daha yüksektir.

- Best case'de Shell Sort $\Omega(n\log n)$ iken, Insertion Sort $\Omega(n)$'dir.
- Insertion Sort, online sort algoritması iken, Shell Sort online sort algoritması değildir.