BLM2512 Veri Yapıları ve Algoritmalar Projesi

Graf Yapılarında Çokgen Analizi

Öğrenci Adı Soyadı : *Berat Can KARAKAŞ*

Öğrenci No : *21011102*

Dersin Eğitmeni : *Mine Elif KARSLIGİL*

Video Linki : *https://youtu.be/UFvgLMl9cRI*

1 - Problem Tanımı ---

Bu problemde kullanıcıdan N (Düğüm sayısı) ve M (Kenar Sayısı) alınarak , dosyadan okunan komşuluk listesi ile bir grafın gerçeklenmesi ve bu grafın içinde Depth First Search (DFS) algoritması kullanılarak oluşturulabilecek tüm farklı şekillerin tüm farklı varyasyonlarının sayısını ayrı ayrı ekrana yazdıran bir programın kodlanması istenmektedir .

2 - Problem Çözümü ---

Bu problemin çözümünde ilk olarak kullanıcıdan dosyanın ismi alınır . Dosya var mı yok mu kontrolünden sonra dosyanın içinden uygun formattaki veriler okunur ve bu veriler ışığında komşuluk matrisi oluşturulmaya başlanır . ilk sütun birinci düğümü , ikinci sütun 2. düğümü ve son sütun ise kenarın ağırlığını oluşturmaktadır . Düğümler tek bir harften oluştuğu için bunlar alphabet dizisi içindeki yerleri yanı sıraları alınarak int e çevrilir (örn : A ilk eleman olduğu için 0 , C 3. eleman olduğu için 2 yi temsil etmektedir .) Bu şekilde her bir kenar için düğüm oluşturulur . Bu şekilde alınan veriler bir graf oluşturulduktan sonra içindeki adjList dizisine aktarılır . Bu dizinin her bir gözü o düğümün kimlerle bağlı olduğunu gösteren bir adres tutmaktadır ve bu adresler bir linkli liste yapısı ile birbirine bağlıdırlar . (örn : adjListin 0. gözü A düğümü ile komşu olan tüm düğümleri birbirine bağlı şekilde tutmaktadır . Dosyadan okuma işleminin tamamlanmasının ardından şekil oluşturmak için DFS algoritmasını kullanmaya başlayabiliriz . DFS fonksiyonumuz özyinelemeli şekilde çalışmaktadır . DFS fonksiyonunun içine ilk olarak tüm yol düğümlerinin adreslerini tutucak bir dizi tanımlanıp gönderilir .

Burada oluşturulabilecek en küçük şekil 3 gen olduğu için dizinin 0. gözüne düğüm sayısı 3 olan yollar eklenmiştir , diğer şekiller de artan şekilde bu dizide yerini almıştır . Gezilen her bir düğüme bir daha gitmemesi için visited dizisi oluşturulup gidilen düğümler bu visited dizisi içinde 1 yapılmıştır. Ama yolun tamamlanabilmesi için yolun içerdiği son düğüm başlangıç düğümüne eşit olması gerekmektedir . Bu yüzden visited ı 1 ise ilk düğüm son düğüme eşit mi kontrolü ve en küçük oluşturulabilecek şekil 3 gen olacağı için düğüm sayısı 4 ten büyük mü kontrolü yapılmaktadır . Bu şartları sağlayan yollar ekleme işlemine bir adım daha yaklaşmaktadır . Bu ekleme işlemi içinde daha önce bu yolun eklenip eklenmediği kontrol edilmektedir . Bunun için tüm yoların bulunduğu dizinin içinden aynı boyuta sahip olan yollar ile bir kıyasa girmektedir . Bu kıyasta eklenmek istenen tolun ve kontrol edilecek yolun hot vectorleri oluşturulur . Graftaki toplam düğüm sayısı boyutunda bir dizi oluşturulup daha sonra o düğüm var olup olmamasına göre düğüm olduğu indis 1 yapılmıştır . Eğer benzer bir hot vectore başka bir yol sahip ise o yol daha önce eklenmiş denip ekleme işlemi yapılmamaktadır . Eğer benzeşim bulunmaz ise o düğüm eklenir . Ekleme işleminde de bir yol düğüm (struct) oluşturulur ve bunun içindeki dizi kısmına bizim yolumuz eklenir nexti de null yapılır . Ve en başta verilen bu yolları tutan adres dizisine 3 gen ise 0 indis 4 gen ise 1. indise eklenicek şekilde yerleştirilir . Eğer daha önce bir ekleme yapılmış ise var olan yol düğümünün nexti null olan bir düğüm bulana kadar sona gidilir ve yeni yol düğümü oraya eklenir ve bu şekilde tüm şekiller bulunur . En son olarak program sonlanmadan tüm şekillerin toplamı ve bu şekillerin yolları ayrı ayrı ekrana yazdırılıp programımız sonlandırılır .

3 - Örnek Çıktılar ---

-girdi-

```
A B 2
A C 3
B C 4
B D 5
C D 6
C E 7
D E 8
```

-çıktı-

```
-- Adjacency List -

0 -> 2,3  1,2
1 -> 3,5  2,4  0,2
2 -> 4,7  3,6  1,4  0,3
3 -> 4,8  2,6  1,5
4 -> 3,8  2,7

--------------
Sekil Sayisi -> 6

1. 3 gen -> A C B A , uzunluk -> 9
2. 3 gen -> B D C B , uzunluk -> 15
3. 3 gen -> C E D C , uzunluk -> 21

1. 4 gen -> A C D B A , uzunluk -> 16
2. 4 gen -> B D E C B , uzunluk -> 24

1. 5 gen -> A C E D B A , uzunluk -> 25
```

-girdi-

```
D A 1
C B 4
C A 3
A B 2
```

-çıktı-

```
-- Adjacency List -

0 -> 1,2  2,3  3,1
1 -> 0,2  2,4
2 -> 0,3  1,4
3 -> 0,1

--------------
Sekil Sayisi -> 1

1. 3 gen -> A B C A , uzunluk -> 9
```

-girdi-

```
A B 2
D B 5
E F 6
G F 7
E G 8
C A 1
C B 3
C D 4
```

-çıktı-

```
-- Adjacency List -

0 -> 2,1  1,2
1 -> 2,3  3,5  0,2
2 -> 3,4  1,3  0,1
3 -> 2,4  1,5
4 -> 6,8  5,6
5 -> 6,7  4,6
6 -> 4,8  5,7

--------------
Sekil Sayisi -> 4

1. 3 gen -> A C B A , uzunluk -> 6
2. 3 gen -> B C D B , uzunluk -> 12
3. 3 gen -> E G F E , uzunluk -> 21

1. 4 gen -> A C D B A , uzunluk -> 12
```

-girdi-

```
A B 1
C D 2
```

-çıktı-

```
-- Adjacency List -

0 -> 1,1
1 -> 0,1
2 -> 3,2
3 -> 2,2

--------------
Sekil Sayisi -> 0
```