A düğümünden başlandığında ağırlığı daha az olan b düğümüne gidilir. **( b=2, c=5, d=3)**

**a -> b**

B düğümünden de aynı şekilde ağırlığ hafif olan tarafa gidilir.  **(c=4, e=8)**

**a -> b-> c**

C düğümünden f ya da d düğümüne gidilir. İkisinin de ağırlıkları birbirlerine eşittir. Fakat bu durumda f düğümüne gidilirse d düğümüne gidilemez o yüzden seçim **d düğümü** olmalıdır.

**a -> b-> c-> d**

D   düğümünden f düğümüne gidilir. Başka seçim yoktur.

**a -> b-> c-> d -> f**

F düğümünden de e düğümü dahah hafif birağırlığa sahip olduğu için seçim e düğümünden yana olur.

**a -> b-> c-> d -> f -> e**

E düğümünden de gezilmemiş ve son düğüm olan g düğümüne gidilerek Dijkstra algoritması tamamlanır.

**a -> b-> c-> d -> f -> e -> g**

 2 + 4 + 1+ 3 + 2 + 2= 14 olur.

A evinden B evine, B evinden de C evine bağlantı olduğuna göre graf türü **ÇEMBER GRAF'dır.**

**N** ev için oluşacak kenar sayısı düğüm sayısı kadardır. Yani kaçdüğüm varsa okadar kenaroluşacağından **N ev için N tane  kenar oluşur.**

**Öncelikle en büyük kenar sayısı 4' e eşit ve kerarlar toplamı çift olduğu için bu graf çizilebilir gibi. Kanıtlama aşamaları da aşağıdaki gibi...**

**ADIM  1**

| graf | |
| --- | --- |
| **d1** | **3** |
| d2 | 2 |
| d3 | 3 |
| d4 | 4 |
| d5 | 2 |

d4  düğümü başlangıç düğümü  seçilirse, (benim oluşturduğum tabloya göre en yüksek kenar sayısına eşit.)

**d4 -> d1**

**d4 -> d2**

**d4 -> d3**

**d4 -> d5**

Olacak şekilde d4'ün kenarlarını kullandık.

**ADIM 2**

| **d1** | **2** |
| --- | --- |
| d2 | 1 |
| d3 | 2 |
| d4 | 0 |
| d5 | 1 |

d4 düğümü en büyük bağlantıya sahip ve tüm düğümler ile bağlantısı olduğu için 2.adımda her düğümden bağlantı sayısını bir azalttık ve d4'ün tüm bağlantınlarını kullandık.

**ADIM 3**

bu adımda, bir sonraki en yüksek bağlantıya sahip olan düğüme bakmalıyız.

d1 ve d3. Burada d1 i seçersek; 2 bağlantı gerekiyor. Bu iki bağlantıyı D2  ve D% ile kullanırsak, d3 ün bağlantılarıaçıkta kalır dolayısıyla; d1'in bir bağlantısını d3'e bir tanesini de d2  ya da d5 ile kullanmamız gerekiyor.

| **d1** | **0** |
| --- | --- |
| d2 | 0 |
| d3 | 1 |
| d4 | 0 |
| d5 | 1 |

**d4 -> d1------------->d2 ve d3**

**d4 -> d2**

**d4 -> d3**

**d4 -> d5**

**ADIM 4**

son durumda d5 ve d3 ün 1 er bağlantısı kalır. bu bağlantıları da birleri ile bağlarsak eğer **grafın oluşabileceğini görebilirz.**

**d4 -> d1------------->d2 ve d3**

**d4 -> d2**

**d4 -> d3--------------> d5**

**d4 -> d5**

 Buradaki bağlantılari tek tek yazarsak eğer;

**d4 -> d1**

**d4 -> d2**

**d4 -> d3**

**d4 -> d5**

**d1 -> d2**

**d1 -> d3**

**d3 -> d5 şeklinde bir sonuç elde ederiz.**

1. toplam = 0 değişkenini oluştur.
2. M değerini oku
3. N değerini oku
4. Topla fonksiyonuna gir
5. i değerini M ye eşitle
6. i değeri N ye eşit olana kadar M+N işlemi yap. ve i yi 1 arttır.
7. i değeri N değerine eşit olduğunda fonksiyondan çık
8. ekrana topla değişkenini bas.

void Topla();     //fonksiyon tanımlanır.

main{

int toplam=0;     // toplam değişkeni tanımlanır.

int M=0;

int N=0;

Toplam( N,M)     //topla fonksiyonu çağrılır.

}

void Topla(int n,int m) //topla fonksiyonunun işlevi

{

int toplam =0;

int n,int m;

    for (int i=m; i<=n; i++)

{     // i değeri m den başlar ve n ye eşit olana kadar 1 er  1er artar.

         topla +=i;      // i değeri m den başlar ve artarak toplam değişkeninde toplanır.

         return toplam;

}

}