PROGRAMCILAR İŞ BAŞINA

©Ali Galip Bayrak

ODTÜ Bilgisayar Topluluğu'nun Bilim ve Teknik için hazırladığı bu sayfayla, bilgisayar bilimlerinin temel problemlerini tanıtmayı amaçlıyoruz. Bu problemler için herhangi bir dilde yazacağınız çözüm kodunu bteknik@tubitak.gov.tr adresine yollayabilirsiniz. Her ay sonunda o ayın çözümlerine ve yapılan değerlendirme sonucu topladığınız puanlara web sitemizden (www.biltek tubitak.gov.tr) ulaşabilirsiniz. Yıl sonunda en fazla puan toplayan yarışmacıya özel bir ödül vereceğiz. İlgilenenler için ODTÜ Bilgisayar Topluluğu'nun web sitesi: http://www.cclub.metu.edu.tr/biltek

Yazı

Çiftçi Oğuz, tarlasını kazmakla uğraştığı sıcak bir yaz gününde tarlasının farklı yerlerinde çok eski, iki kutu bulur. Kutuların içerisinde birer tane kağıt vardır. Kağıtlarda yazılanlara anlam veremeyen Oğuz, köyün bilgesi Umut'un yanına gider. Umut kağıtları uzun süre incedikten sonra "Bu kağıtlarda yazan yazıların en uzun ortak kısımlarını bulursak sorunu çözmüş oluruz" der. Sizden istenen bu konuda Umut ve Oğuz'a yardımcı olacak programı yazmanız.

Varsayımlar

- Kağıtlardaki yazıların uzunlukları *m* ve *n* dir ($1 \le n,m \le 1000$).
- Yazılarda sadece İngiliz alfabesinin küçük harfleri vardır.

Girdi

- Girdiler "yazi.gir" isimli dosyadan okunacaktır.
- İlk satırda şehir sayısını ifade eden *m* ve *n* verilecektir.
- Takip eden satırda uzunluğu *m* olan yazı verilecektir.
- Sonraki satırda uzunluğu n olan yazı verilecektir.

Çıktı

- Çıktılar "yazi.cik" isimli dosyaya yazılacaktır.
- İlk satırda en uzun ortak kısmın uzunluğunu ifade eden bir adet tamsayı bulunacaktır.
- İkinci satırda en uzun ortak kısım bulunacaktır.

Örnek

yollar.gir:
15 10
okadakabasakali
okabasaoka

yollar.cik:
6
kabasa

*Ortak kısım, tek parça halinde bulunmalıdır. Örneğin "alirveli" ve "aliveli" sözcüklerinin en uzun ortak kısımları "veli" dir, "aliveli" değil.

Yazı 2

Yıllar sonra çiftçi Oğuz'un oğlu yine tarlada çalışırken benzer şekilde iki adet kutu bulur. Bu sefer kutulardaki yazılar düz yazı şeklinde değil, karelerden oluşan büyük bir dikdörtgenin her karesine bir harf gelecek şekildedir. Oğuz, babasının başına gelen olayı hatırlar ve hemen köyün bilgesi Umut'un yanına gider. Umut biraz inceledikten sonra bu kez en büyük ortak kareyi çıkarmak gerektiğini bulur.

Varsayımlar

- Birinci kağıttaki yazı *mxn*'lik bir dikdörtgende, ikinci kağıttaki yazı *pxq*'luk bir dikdörtgendedir ($1 \le n,m,p,q \le 200$). *m* ve *p* satır sayılarını, *n* ve *q* sütun sayılarını ifade etmektedir.
- Yazılarda sadece İngiliz alfabesinin küçük harfleri vardır.

Girdi

- Girdiler "yazi2.gir" isimli dosyadan okunacaktır.
- İlk satırda dikdörtgenlerin boyutlarını ifade eden *m,n,p,q* sırayla verilecektir.
- Takip eden *m* satırın herbirisinde *n* tane harf bulunacaktır.
- Takip eden *p* satırın herbirisinde *q* tane harf bulunacaktır.

Çıktı

- Çıktılar "yazi2.cik" isimli dosyaya yazılacaktır.
- Tek satırda en büyük ortak karenin boyutlarını ifade eden bir adet tamsayı bulunacaktır.

Örnek

yazi2.gir:

```
5 6 4 4
a t l m y y
t k a y m m
l v e n t t
y s a r y t
m n n n v v
k a y g
v e n h
s a r h
n l l n
```

yazi2.cik:

```
3
```

a	t	l	m	y	y
t	k	a	y	m	m
l	v	e	n	t	t
y	s	a	r	y	t
m	n	n	n	v	v

k	a	y	g
v	e	n	h
s	a	r	h
n	l	l	n

En büyük ortak kare:
kay
ven
sar

Geçen Sayımızdaki Soruların Çözümleri

Yollar 1

Bu problem bilgisayar biliminde "Minumum Spanning Tree (MST)" yani "en küçük kapsar ağaç" olarak bilinir. Kruskal'ın algoritması ve Prim'in algoritması bu problemi çözmek için en sık kullanılan algoritmalardır. Bunlardan Kruskal'ın algoritması şu şekildedir:

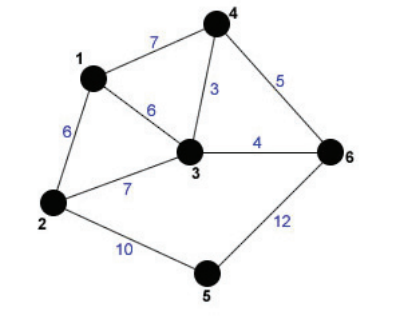
0. Bütün yolları siyaha boyayalım.

1. Her şehir için ayrı birer küme oluşturalım ve bunları numaralandıralım.

2. Bütün yolları uzunluklarına göre küçükten büyüğe sıralayalım.

3. Sıralanmış olan yollardan sırası geleni alalım (ilk başta en küçüğü, daha sonra bir sonrakini...). Eğer bu yolun iki ucundaki şehir aynı kümede değil ise bu yolu kırmızıya boyayalım ve bu iki uçtaki şehirlerin kümelerini birleştirelim. Bu işlemi bütün yollar bitene kadar uygulayalım (aslında tüm şehirleri aynı kümede toplayana kadar uygulamak yeterli).

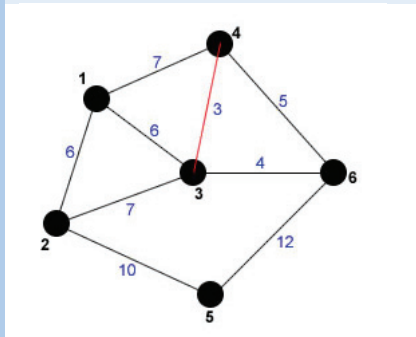
Örneğimiz üzerinde algoritmayı uygulayacak olursak:



İlk olarak bütün yolları siyaha boyayalım. Her şehrin kümesi kendi numarası olsun, yani 1. şehir 1. kümede, 2. şehir 2. kümede vb. Bütün yolları sıralarsak:

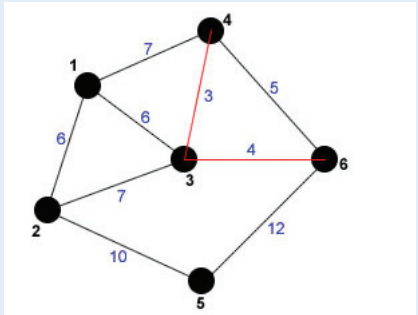
3 (3. şehirle 4. şehir arası), 4 (3-6 arası), 5 (4-6 arası), 6 (1-3 arası), 6 (1-2 arası), 7 (1-4 arası), 7 (2-3 arası), 10 (2-5 arası), 12 (5-6 arası).

1. küme = {1} 2. küme = {2}
3. küme = {3} 4. küme = {4}
5. küme = {5} 6. küme = {6}



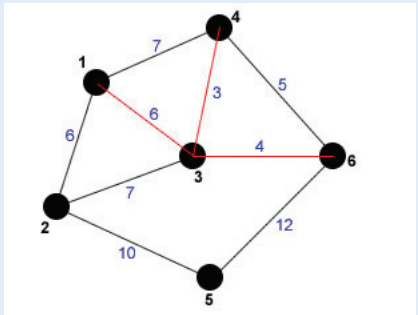
Şimdi sırayla yolları alalım. En kısa olan yol, 3. şehri 4. şehre bağlayan 3 uzunluktaki yol. Bu şehirler farklı kümelerde olduğu için bu yolu kırmızıya boyayalım. 3. ve 4. şehirlerin kümelerini birleştirelim, örneğin iki şehrin de yeni kümesi 3 olsun (4 de yapabiliriz, tek önemli olan aynı kümede olmaları).

1. küme = {1} 2. küme = {2}
3. küme = {3, 4} 4. küme = {}
5. küme = {5} 6. küme = {6}



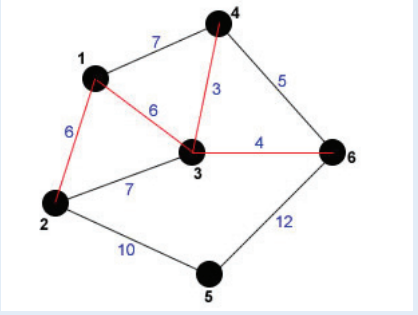
Sıradaki yol, 3. şehri 6. şehre bağlayan 4 uzunluktaki yol. Bu şehirler farklı kümelerde olduğu için bu yolu kırmızıya boyayalım. 3. ve 6. şehirlerin kümelerini birleştirelim, örneğin 6. şehri de 3. kümeye alalım.

1. küme = {1} 2. küme = {2}
3. küme = {3, 4, 6} 4. küme = {}
5. küme = {5} 6. küme = {}



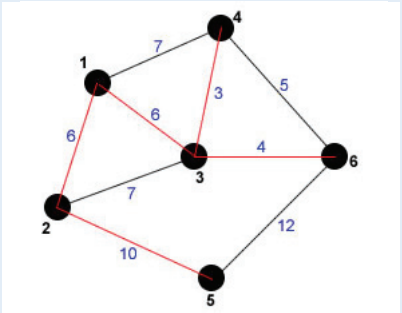
Sıradaki yol, 4. şehri 6. şehre bağlayan 5 uzunluktaki yol. Bu şehirler aynı kümede olduğu için (3. kümede) devam edelim. Sıradaki yol, 3. şehri 1. şehre bağlayan 6 uzunluktaki yol. Bu şehirler farklı kümelerde olduğu için bu yolu kırmızıya boyayalım. 3. ve 1. şehirlerin kümelerini birleştirelim, örneğin 3., 4. ve 6. şehirlerin kümesini de 1 yapalım.

1. küme = {1, 3, 4, 6} 2. küme = {2}
3. küme = {} 4. küme = {}
5. küme = {5} 6. küme = {}



Sıradaki yol, 1. şehri 2. şehre bağlayan 6 uzunluktaki yol. Bu şehirler farklı kümelerde olduğu için bu yolu kırmızıya boyayalım. 1. ve 2. şehirlerin kümelerini birleştirelim, örneğin 2. şehri de 1. kümeye alalım.

1. küme = {1, 2, 3, 4, 6} 2. küme = {}
3. küme = {} 4. küme = {}
5. küme = {5} 6. küme = {}



Sıradaki yol, 1. şehri 4. şehre bağlayan 7 uzunluktaki yol. Bu şehirler aynı kümede olduğu için devam edelim. Aynı şekilde 3. ve 2. şehirler de aynı kümede olduğu için uzunluğu 7 olan diğer yolu da atlayalım. Sıradaki yol 2. ve 5. şehirleri bağlayan 10 uzunluktaki yol. Bu yolu da kırmızıya boyayıp kümeleri birleştirelim. İşlemi burada bitirebiliriz (tüm şehirler aynı kümede). Devam edersek uzunluğu 12 olan yolun kırmızıya boyanamayacağını görebiliriz.

1. küme = {1, 2, 3, 4, 5, 6} 2. küme = {}
3. küme = {} 4. küme = {}
5. küme = {} 6. küme = {}

Biraz incelediğimiz zaman göreceğimiz üzere, bizden istenilen, kırmızıya boyalı yolların sayısı ve bu yollar. Bu algoritmayı kullanarak çözüme ulaşabiliriz. Kırmızıya boyalı yolların sayısı her zaman *şehir sayısı – 1* kadardır. Çünkü algoritmadaki boyama işini *şehir sayısı - 1* kez uygularsak bütün şehirleri aynı kümede toplamış oluruz ve açıkça görülebilir ki bu kümedeki bütün şehir ikilileri arasında bir yol vardır.

Yollar 2

İlk sorunun çözümünü biraz değiştirerek bu soruya da çözüm üretebiliriz. Şöyle ki:

0. Daha önceden yenilenmiş yolları kırmızıya, diğerlerini siyaha boyayalım.

1. Her şehir için ayrı birer küme oluşturalım ve bunları numaralandıralım. Daha sonra kırmızı ile birbirine bağlanmış şehirlerin kümelerini birleştirelim.

2. Siyah yolları, uzunluklarına göre küçükten büyüğe sıralayalım.

3. Sıralanmış olan yollardan sırası geleni alalım (ilk başta en küçüğü, daha sonra bir sonrakini...). Eğer bu yolun iki ucundaki şehir aynı kümede değil ise bu yolu kırmızıya boyayalım ve bu iki uçtaki şehirlerin kümelerini birleştirelim. Bu işlemi bütün yollar bitene kadar uygulayalım.