Turkey Java Community
Tech Talk

2+2‘yi toplayalım

Java’da 2 sayıyı kaç farklı şekilde toplayabilirsiniz?

Furkan Şahin KULAKSIZ
Software Developer
Co Founder of Turkey Java Community

NEDEN BÖYLE BİR ŞEY YAPIYORUZ

1) VERİMLİLİK

Algoritmalar, problemleri çözme ve bilgisayarla yürütülebilecek talimatları oluşturma konusunda çok önemlidir. İyi tasarlanmış bir algoritma, daha az bilgisayar kaynağı (bellek, işlemci zamanı vb.) kullanır, bu da genellikle daha hızlı ve verimli bir yazılım anlamına gelir.

1) VERİMLİLİK

Algoritmalar, problemleri çözme ve bilgisayarla yürütülebilecek talimatları oluşturma konusunda çok önemlidir. İyi tasarlanmış bir algoritma, daha az bilgisayar kaynağı (bellek, işlemci zamanı vb.) kullanır, bu da genellikle daha hızlı ve verimli bir yazılım anlamına gelir.

2) PROBLEM ÇÖZME YETENEĞİ

Algoritmik düşünme, programcıların belirli bir problemi ayrıntılı bir şekilde anlamalarını ve karmaşık problemleri basit ve yönetilebilir parçalara ayırmalarını sağlar. Bu, problem çözme yeteneğini artırır ve genellikle daha etkili ve hatasız kodların oluşturulmasına yardımcı olur.

3) MANTIKSAL DÜŞÜNME VE ANALİTİK BECERİLER

Algoritmik sorun çözme, genel mantıksal düşünme ve analitik becerileri geliştirir. Bu yetenekler, bir programcının her türlü yazılım projesinde yararlı olacaktır, çünkü çoğu programlama görevi, bir problemi anlamayı, bir çözüm tasarlamayı ve bu çözümü kod olarak uygulamayı gerektirir.

1) Normal Yollarla Toplama İşlemi

```
System.out.println(a + b);
```

Class’tan bir instance oluşturularak yapılan toplama işlemi

Math Kütüphanesi kullanılarak toplama işlemi

BigDecimal Class’ı kullanılarak toplama işlemi

Class’ta topla() adında bir method oluşturduğumuzu düşünelim ve bu method üzerinden sayılarımızı toplayalım.

Peki ya methodumuz PRIVATE ise.?

2) REFLECTION

Reflection API runtime’da bir sınıfın davranışını inceleyen bir yapıdır.

3) MERGE LISTS

4 ELEMANLI BİR LİSTE

0	0	0	0
0. Index	1. Index	2. Index	3. Index

6 ELEMANLI BİR LİSTE

0	0	0	0	0	0
0. Index	1. Index	2. Index	3. Index	4. Index	5. Index

10 ELEMANLI BİRLEŞTİRİLMİŞ LİSTE

0	0	0	0	0	0	0	0	0	0
0	1	2	3	4	5	6	7	8	9

4) BRUTE FORCE

Brute Force yaklaşımı, bütün olasılıkları tek tek deneyerek sonuca gitmenizi sağlayan bir yaklaşımdır.

Basit bir yaklaşımı vardır. Tutarlıdır. Uygulaması kolaydır.

Yavaş çalışan bir tekniktir.

5) BITWISE OPERATIONS

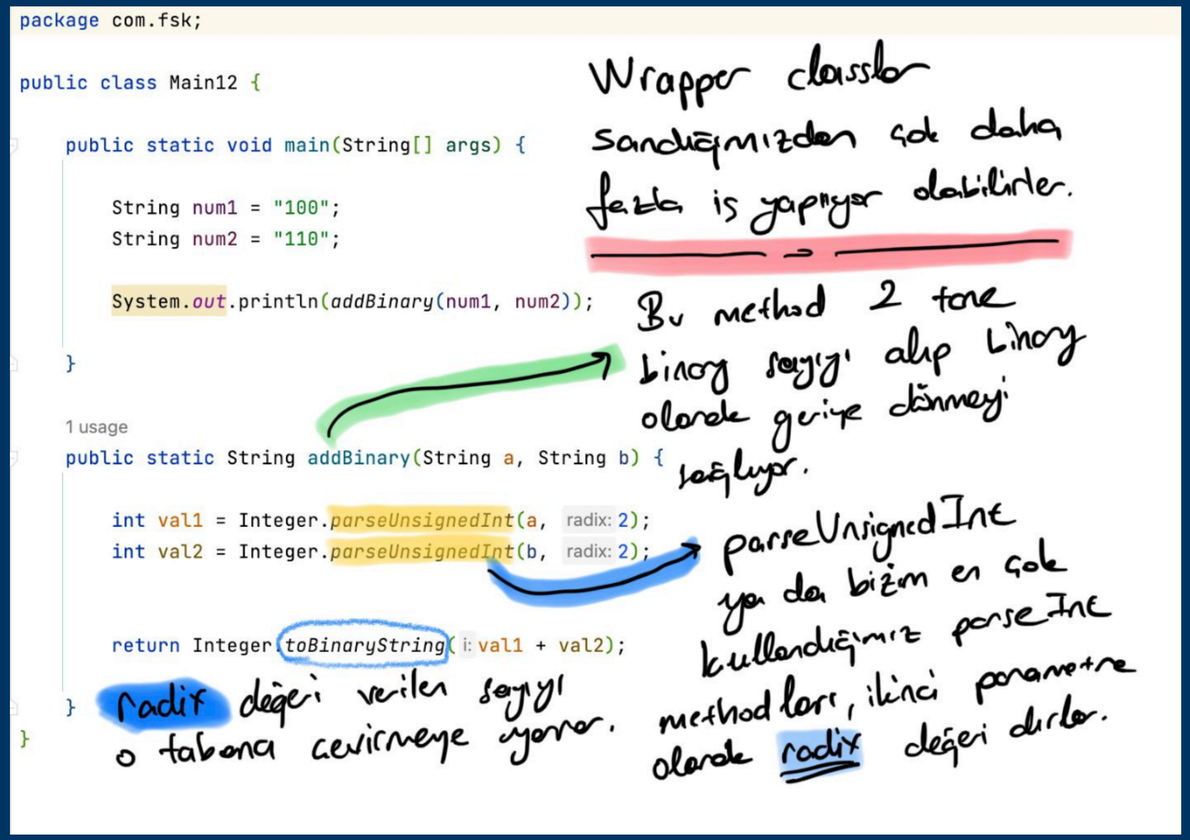
Bit seviyesinde işlem yapmamızı sağlayan operatörlerdir.

Düşük seviyeli programlama dillerinde daha yaygındır.

Performans Optimizasyonu
Sıkıştırma ve Şifreleme Algoritmaları
Donanımla Doğrudan Etkileşim

5) BITWISE OPERATIONS

https://www.linkedin.com/posts/frknshnklksz_java-activity-7028111112176906240-NZ7R?utm_source=share&utm_medium=member_desktop



5) BITWISE OPERATIONS

Bitwise OR (|)

Önermelerden 2’si de 0 ise / yanlış ise / false ise sonuç 0’dır. Bunun dışında tüm sonuçlar 1 ‘ dir.

X (1. ÖNERME)	Y (2. ÖNERME)	SONUÇ
1	1	1
1	0	1
0	1	1
0	0	0

5) BITWISE OPERATIONS

Bitwise AND (&)

Önermelerden 2’si de 1 ise / doğru ise / true ise sonuç 1’dir. Bunun dışında tüm sonuçlar 0 ‘ dır.

X (1. ÖNERME)	Y (2. ÖNERME)	SONUÇ
1	1	1
1	0	0
0	1	0
0	0	0

5) BITWISE OPERATIONS

Bitwise XOR (^)

Önermeler birbirlerinin zıttıysa; yani iki önermeden birisi doğru, diğeri yanlış ise yada, birisi yanlış diğeri doğru ise 1 döndürür. Eğer önermelerden ikisi de doğru ya da ikisi de yanlış ise o zaman 0 döndürür.

X (1. ÖNERME)	Y (2. ÖNERME)	SONUÇ
1	1	0
1	0	1
0	1	1
0	0	0

5) BITWISE OPERATIONS

Bitwise COMPLEMENT (~)

Bu işaret, uygulandığı sayının 2’lik tabandaki karşılığının tam tersi bitlerini alır. Yani bit 0 ise 1’e, 1 ise 0’a çevirir.

~1010 = 0101

5) BITWISE OPERATIONS

Bitwise LEFT SHIFT (<<)

Bu işaret, uygulandığı sayının 2’lik tabandaki karşılığını sola doğru kaydırır.

5) BITWISE OPERATIONS

Bitwise RIGHT SHIFT (>>)

Bu işaret, uygulandığı sayının 2’lik tabandaki karşılığını sağa doğru kaydırır.

6) BINARY SEARCH

Elimizde önce sırasız bir dizi var. Aradığımız sayı da (target) 38 olsun.

8	91	56	72	2	12	5	23	38	16
0. Index	1. Index	2. Index	3. Index	4. Index	5. Index	6. Index	7. Index	8. Index	9. Index

6) BINARY SEARCH

1.Adım, Diziyi Sırala

2	5	8	12	16	23	38	56	72	91
0. Index	1. Index	2. Index	3. Index	4. Index	5. Index	6. Index	7. Index	8. Index	9. Index

6) BINARY SEARCH

2. Adım, Low, High ve Mid değerlerini belirle

LOW				MID					HIGH
2	5	8	12	16	23	38	56	72	91
0. Index	1. Index	2. Index	3. Index	4. Index	5. Index	6. Index	7. Index	8. Index	9. Index

6) BINARY SEARCH

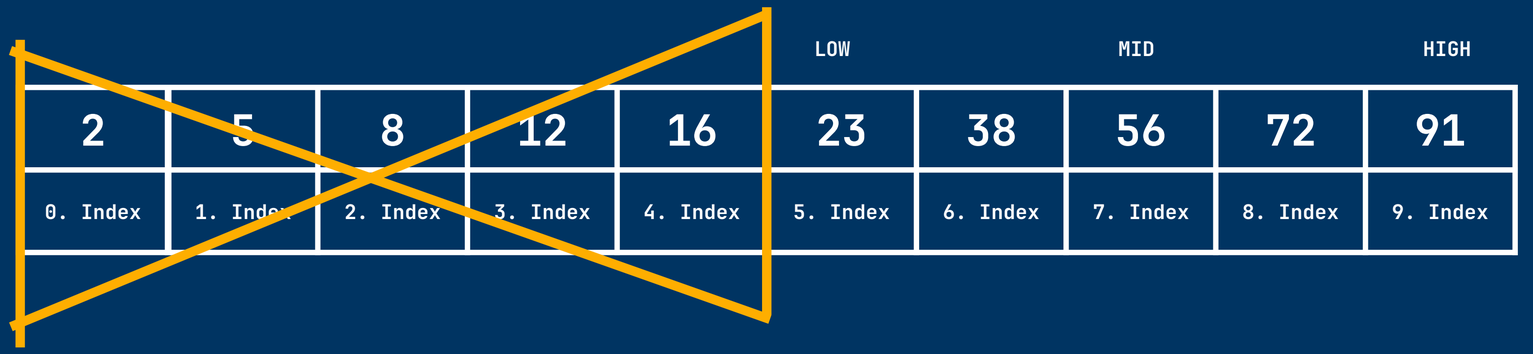
3. Adım

arr[mid] < target
arr[mid] > target
arr[mid] = target

LOW			MID						HIGH
2	5	8	12	16	23	38	56	72	91
0. Index	1. Index	2. Index	3. Index	4. Index	5. Index	6. Index	7. Index	8. Index	9. Index

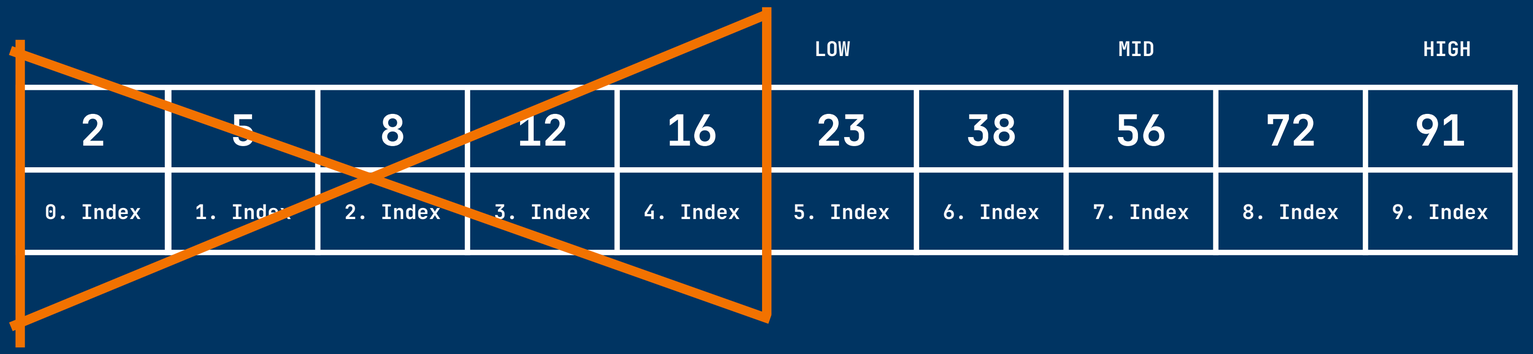
6) BINARY SEARCH

4.Adım, 3. Adımdan sonra LOW, HIGH ve MID değerlerini güncelle.



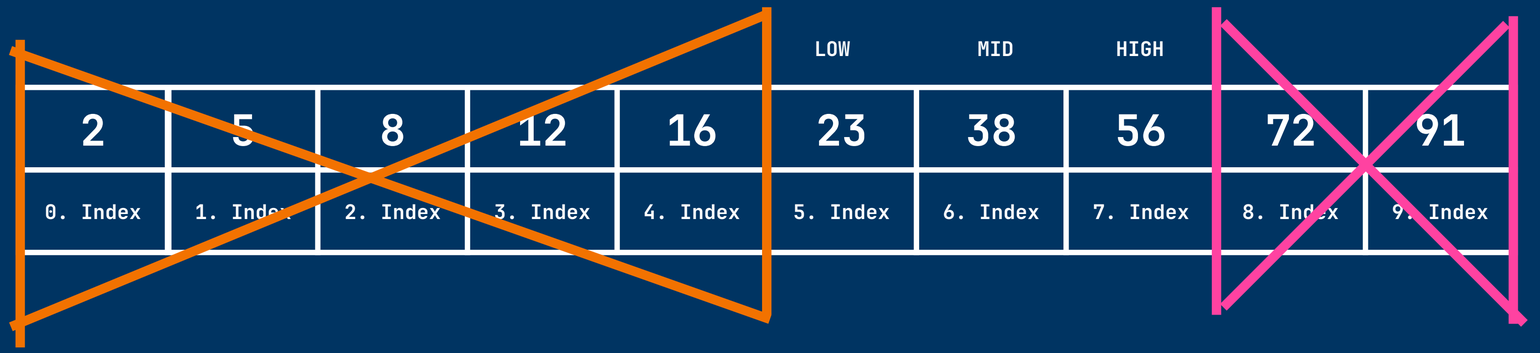
6) BINARY SEARCH

5.Adım, 3. Ve 4. Adımdan sonra LOW, HIGH ve MID değerlerini güncelle.



6) BINARY SEARCH

5.Adım, 3. Ve 4. Adımdan sonra LOW, HIGH ve MID değerlerini güncelle.

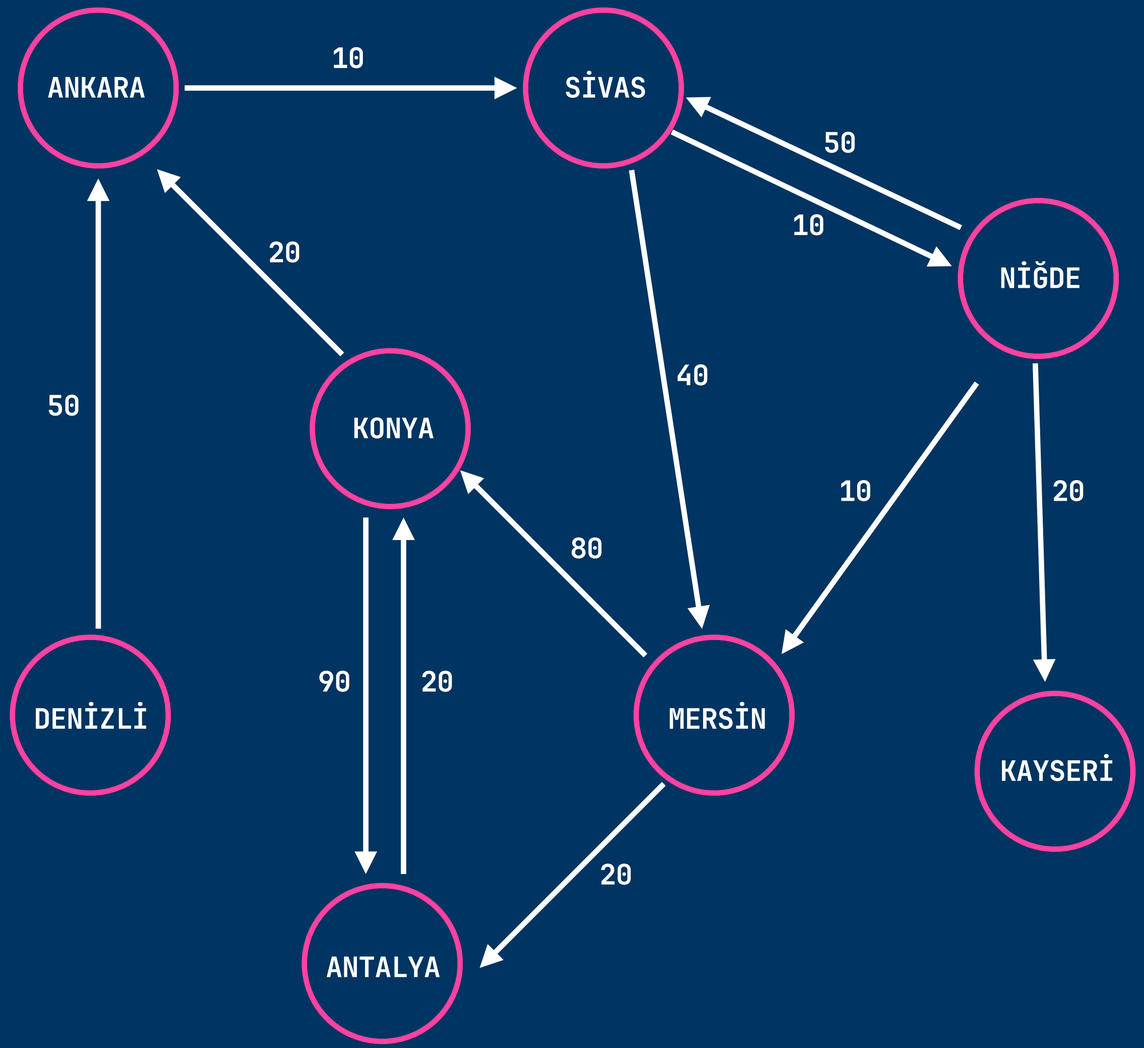


7) DJIKSTRA ALGORITHM

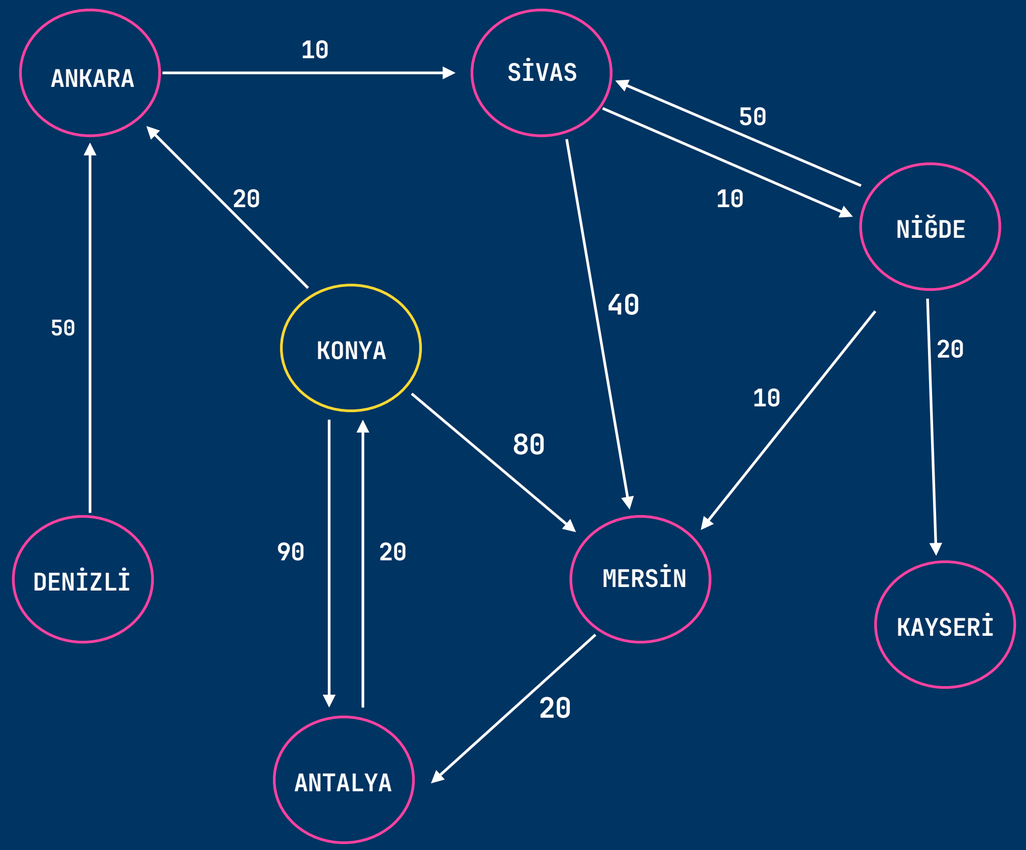
En kısa yol algoritmasıdır.

Bu algoritmanın temel amacı, bulunduğumuz noktadan diğer tüm noktalara en kısa yolu bulma amacı taşır.

7) DJIKSTRA ALGORITHM



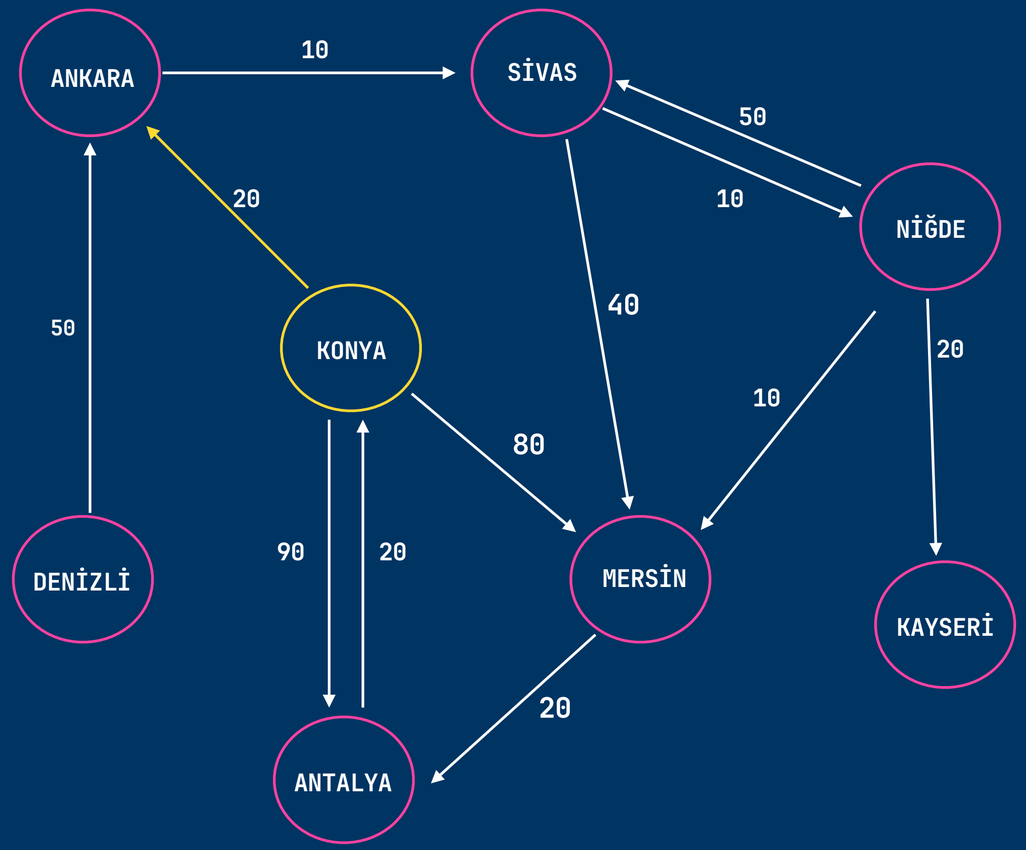
7) DJIKSTRA ALGORITHM



Başlangıç düğümü seçilir ve bunun maliyeti 0 olur.

DÜĞÜM	KONYA	ANKARA	NİĞDE	MERSİN	DENİZLİ	SİVAS	ANTALYA	KAYSERİ
KONYA	0	∞	∞	∞	∞	∞	∞	∞

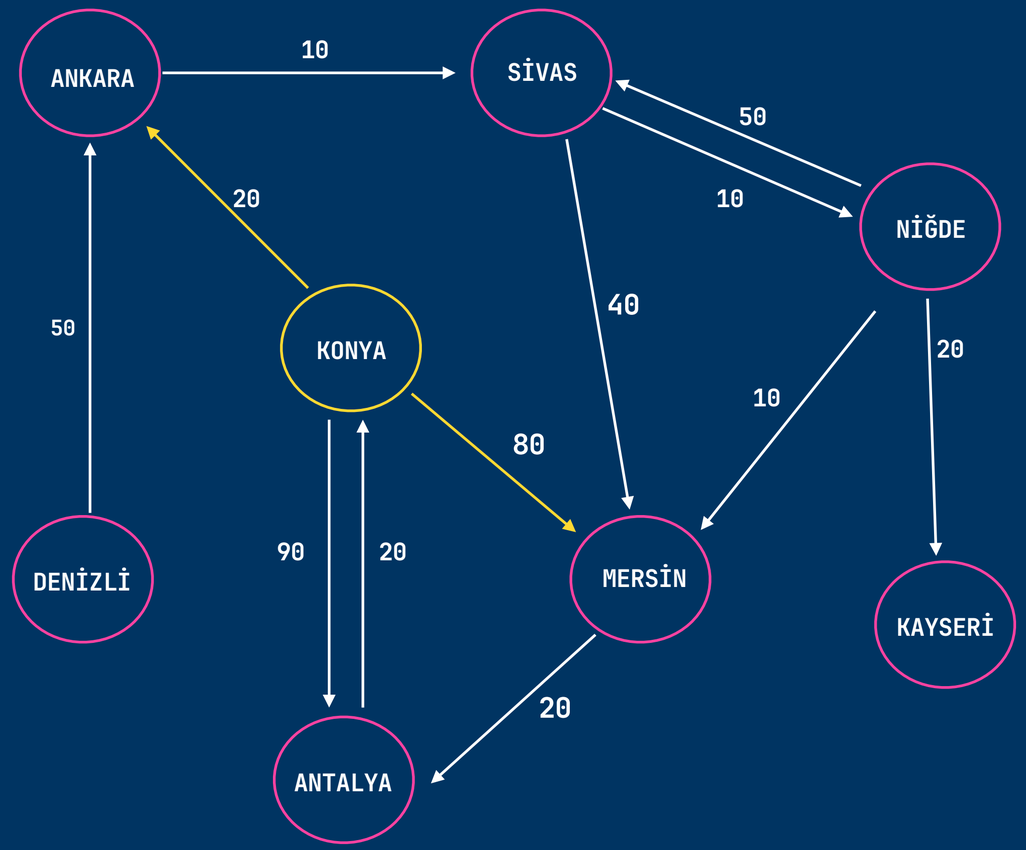
7) DJIKSTRA ALGORITHM



Gidilecek düğümün yol maliyeti ile, o anki maliyet Karşılaştırılır. Küçük olan ile güncellenir.

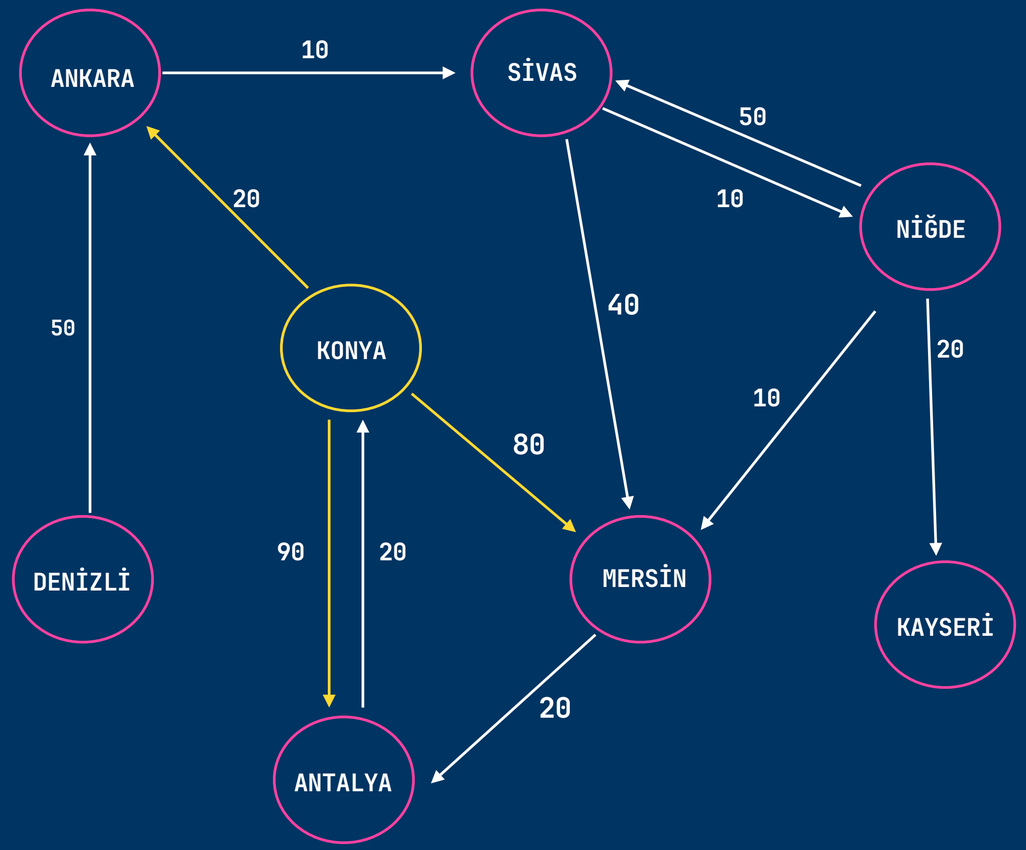
DÜĞÜM	KONYA	ANKARA	NİĞDE	MERSİN	DENİZLİ	SİVAS	ANTALYA	KAYSERİ
KONYA	0	~~∞~~ 20 P: KONYA	∞	∞	∞	∞	∞	∞

7) DJIKSTRA ALGORITHM



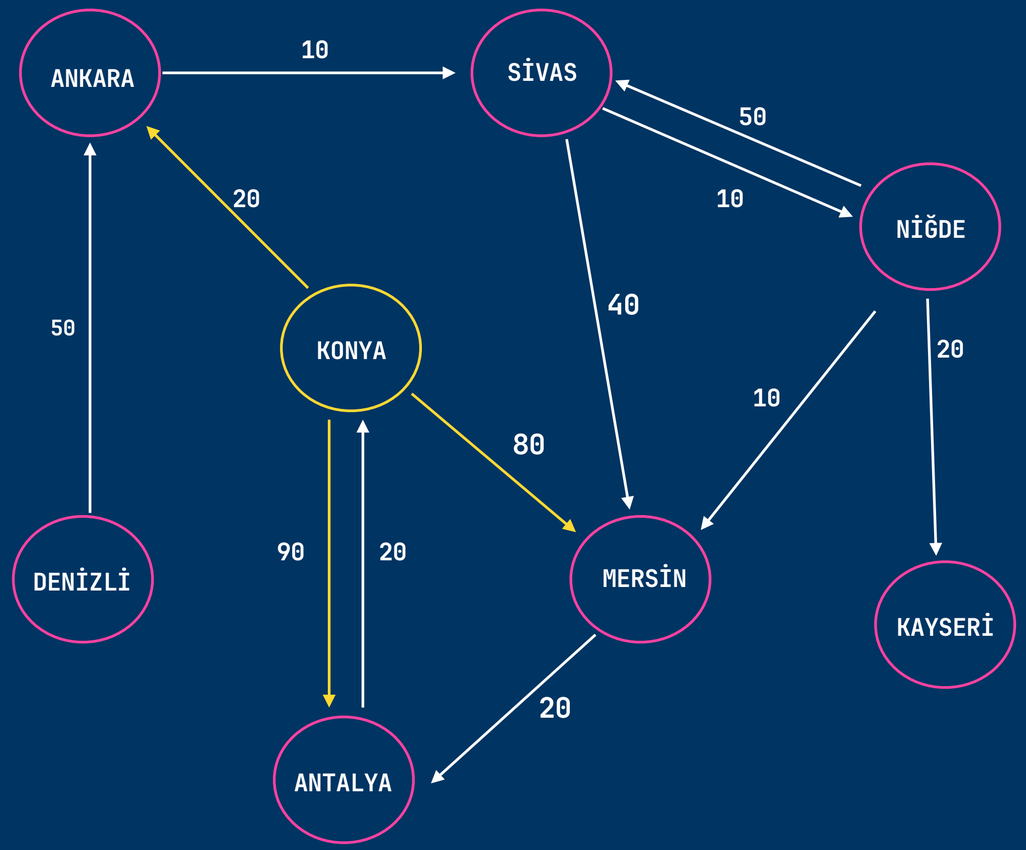
DÜĞÜM	KONYA	ANKARA	NİĞDE	MERSİN	DENİZLİ	SİVAS	ANTALYA	KAYSERİ
KONYA	0	~~∞~~ 20 P: KONYA	∞	~~∞~~ 80 P: KONYA	∞	∞	∞	∞

7) DJIKSTRA ALGORITHM



DÜĞÜM	KONYA	ANKARA	NİĞDE	MERSİN	DENİZLİ	SİVAS	ANTALYA	KAYSERİ
KONYA 0	0	~~∞~~ 20 P: KONYA	∞	~~∞~~ 80 P: KONYA	∞	∞	~~∞~~ 90 P: KONYA	∞

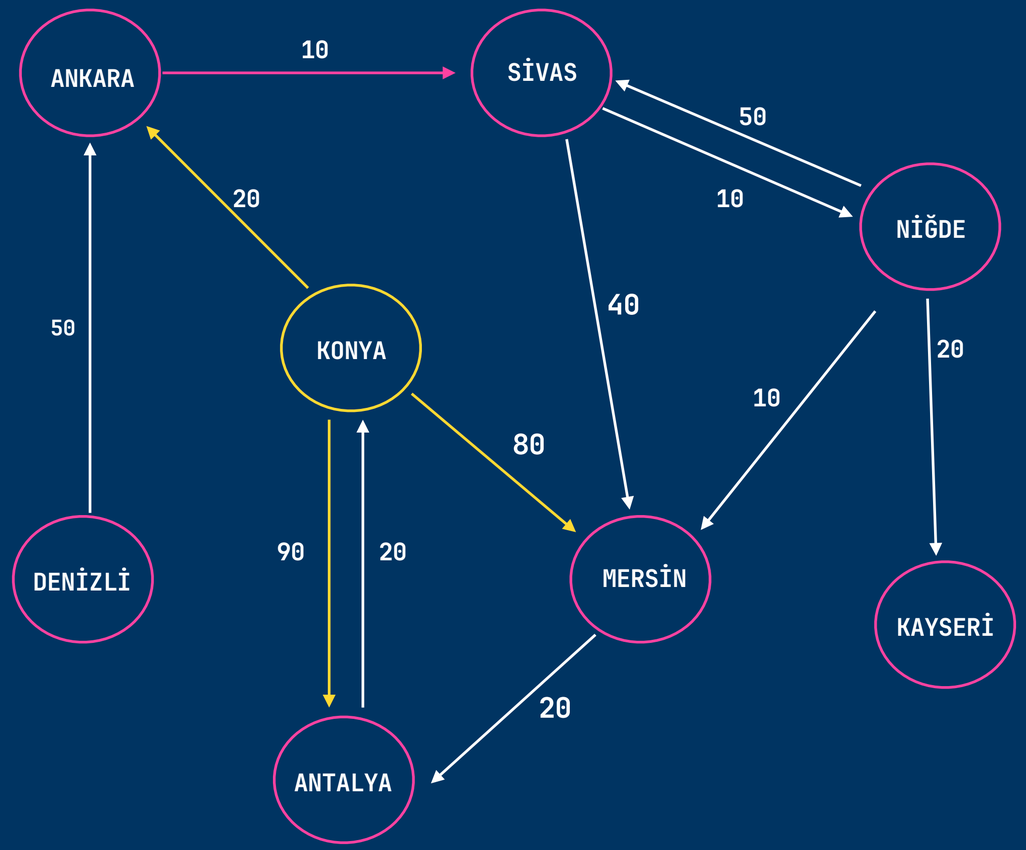
7) DJIKSTRA ALGORITHM



DÜĞÜM	KONYA	ANKARA	NİĞDE	MERSİN	DENİZLİ	SİVAS	ANTALYA	KAYSERİ
KONYA 0	0	~~∞~~ 20 P: KONYA	∞	~~∞~~ 80 P: KONYA	∞	∞	~~∞~~ 90 P: KONYA	∞
ANKARA 20	0	20 P: KONYA	∞	80 P: KONYA	∞	∞	90 P: KONYA	∞

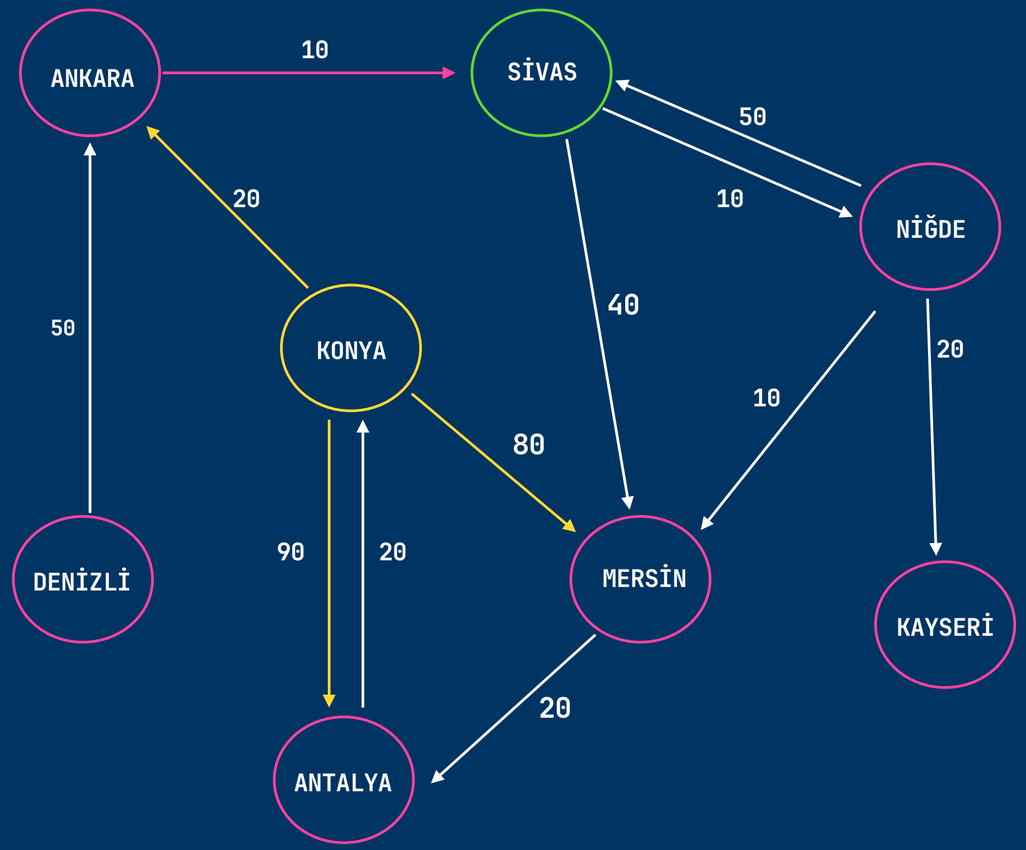
7) DJIKSTRA ALGORITHM

Parent Düğüm işlendikten sonra, ilgili satırdaki en az maliyetli olana geçilir.
Tek şart, düğümün ziyaret edilmemiş olması gerekmektedir.
Ve maliyeti elimizde tutmamız lazım.



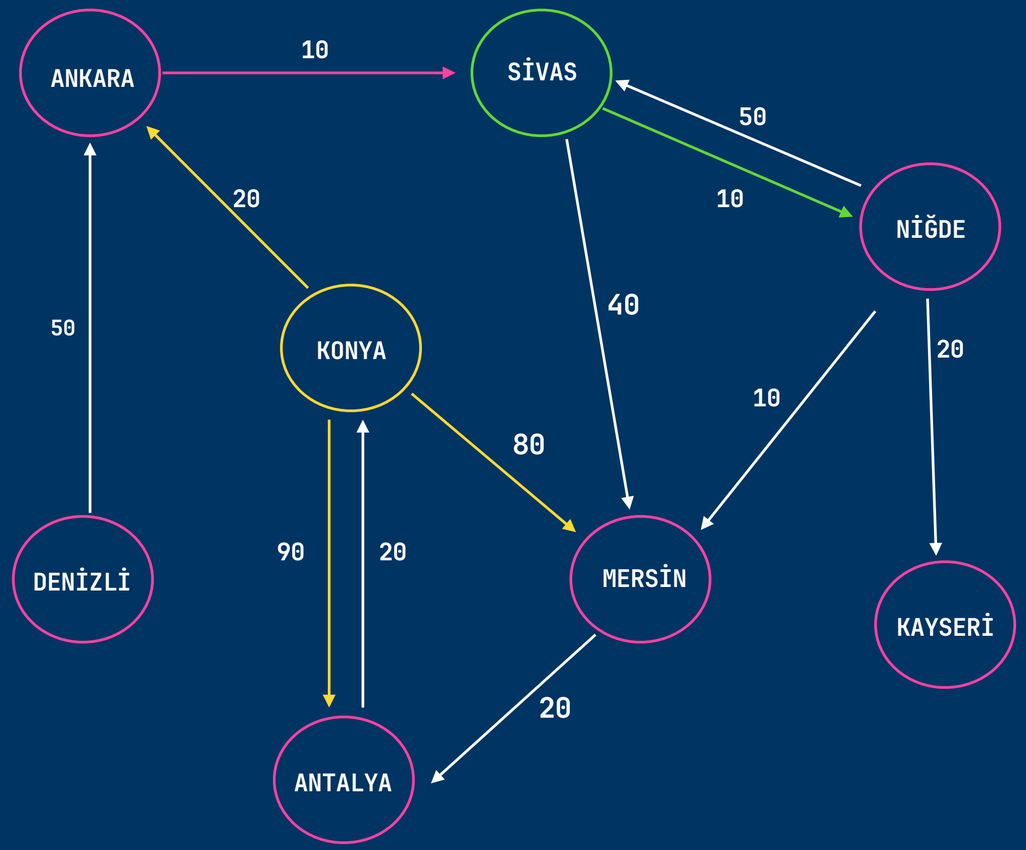
DÜĞÜM	KONYA	ANKARA	NİĞDE	MERSİN	DENİZLİ	SİVAS	ANTALYA	KAYSERİ
KONYA 0	0	~~∞~~ 20 P: KONYA	∞	~~∞~~ 80 P: KONYA	∞	∞	~~∞~~ 90 P: KONYA	∞
ANKARA 20	0	20 P: KONYA	∞	80 P: KONYA	∞	~~∞~~ 20 + 10 P: ANKARA	90 P: KONYA	∞

7) DJIKSTRA ALGORITHM



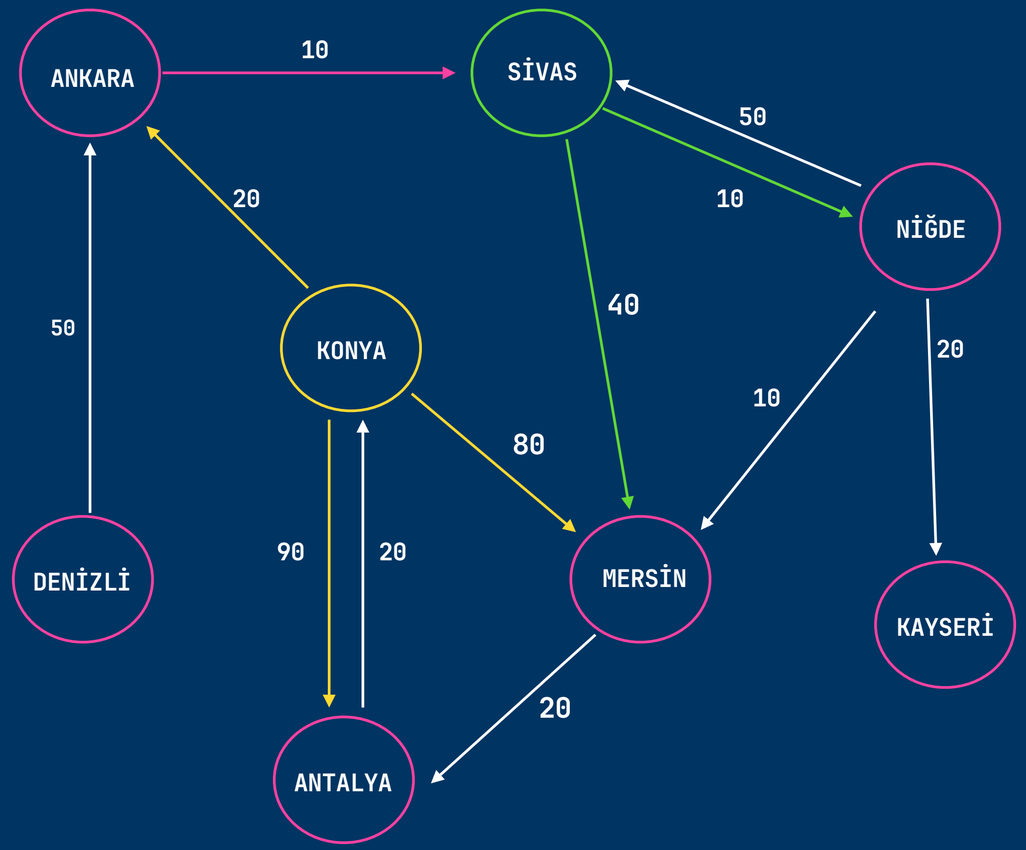
DÜĞÜM	KONYA	ANKARA	NİĞDE	MERSİN	DENİZLİ	SİVAS	ANTALYA	KAYSERİ
KONYA 0	0	~~∞~~ 20 P: KONYA	∞	~~∞~~ 80 P: KONYA	∞	∞	~~∞~~ 90 P: KONYA	∞
ANKARA 20	0	20 P: KONYA	∞	80 P: KONYA	∞	~~∞~~ 20 + 10 P: ANKARA	90 P: KONYA	∞
SİVAS 30	0	20 P: KONYA	∞	80 P: KONYA	∞	30 P: ANKARA	90 P: KONYA	∞

7) DJIKSTRA ALGORITHM



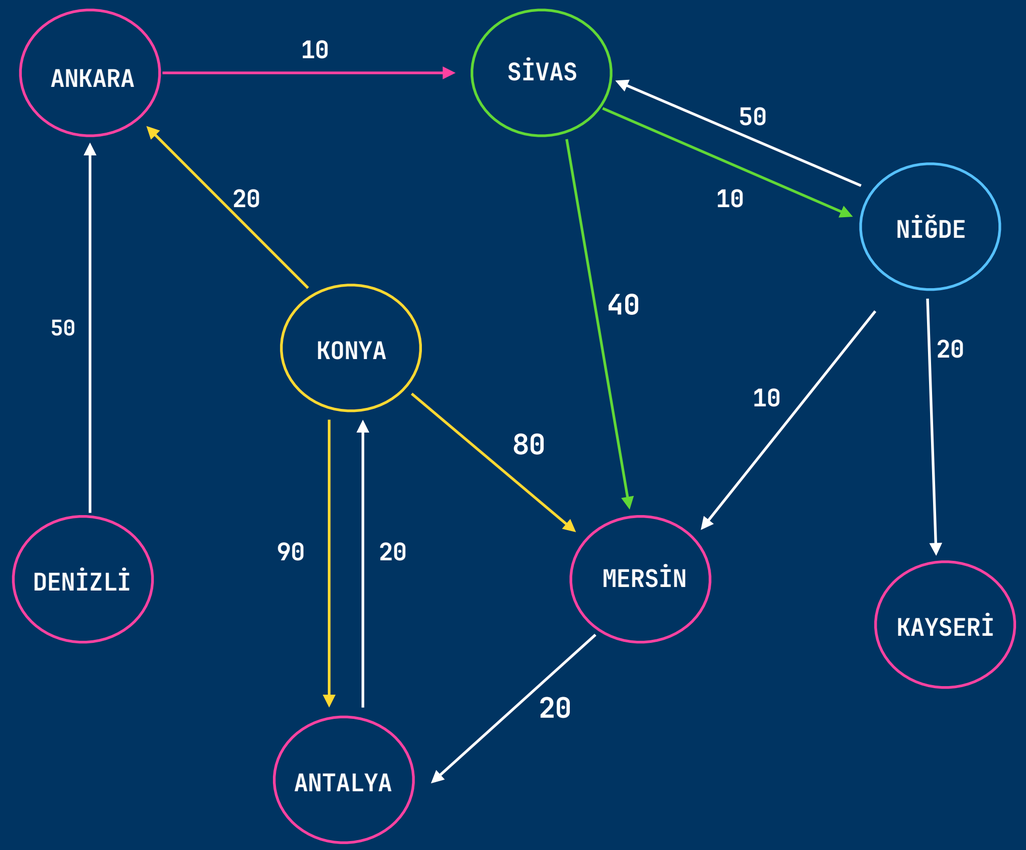
DÜĞÜM	KONYA	ANKARA	NİĞDE	MERSİN	DENİZLİ	SİVAS	ANTALYA	KAYSERİ
KONYA 0	0	~~∞~~ 20 P: KONYA	∞	~~∞~~ 80 P: KONYA	∞	∞	~~∞~~ 90 P: KONYA	∞
ANKARA 20	0	20 P: KONYA	∞	80 P: KONYA	∞	~~∞~~ 20 + 10 P: ANKARA	90 P: KONYA	∞
SİVAS 30	0	20 P: KONYA	~~∞~~ 30 + 10 P: SİVAS	80 P: KONYA	∞	30 P: ANKARA	90 P: KONYA	∞

7) DJIKSTRA ALGORITHM



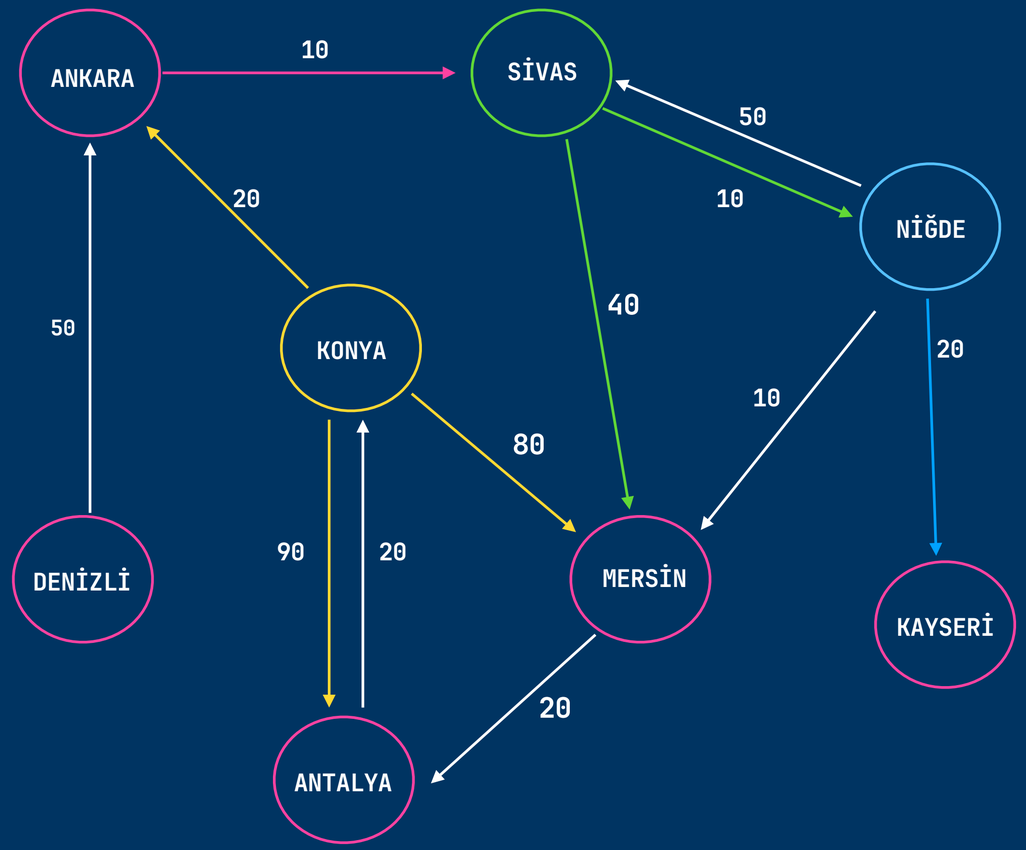
DÜĞÜM	KONYA	ANKARA	NİĞDE	MERSİN	DENİZLİ	SİVAS	ANTALYA	KAYSERİ
KONYA 0	0	~~∞~~ 20 P: KONYA	∞	~~∞~~ 80 P: KONYA	∞	∞	~~∞~~ 90 P: KONYA	∞
ANKARA 20	0	20 P: KONYA	∞	80 P: KONYA	∞	~~∞~~ 20 + 10 P: ANKARA	90 P: KONYA	∞
SİVAS 30	0	20 P: KONYA	~~∞~~ 30 + 10 P: SİVAS	~~80~~ 30 + 40 P: SİVAS	∞	30 P: ANKARA	90 P: KONYA	∞

7) DJIKSTRA ALGORITHM



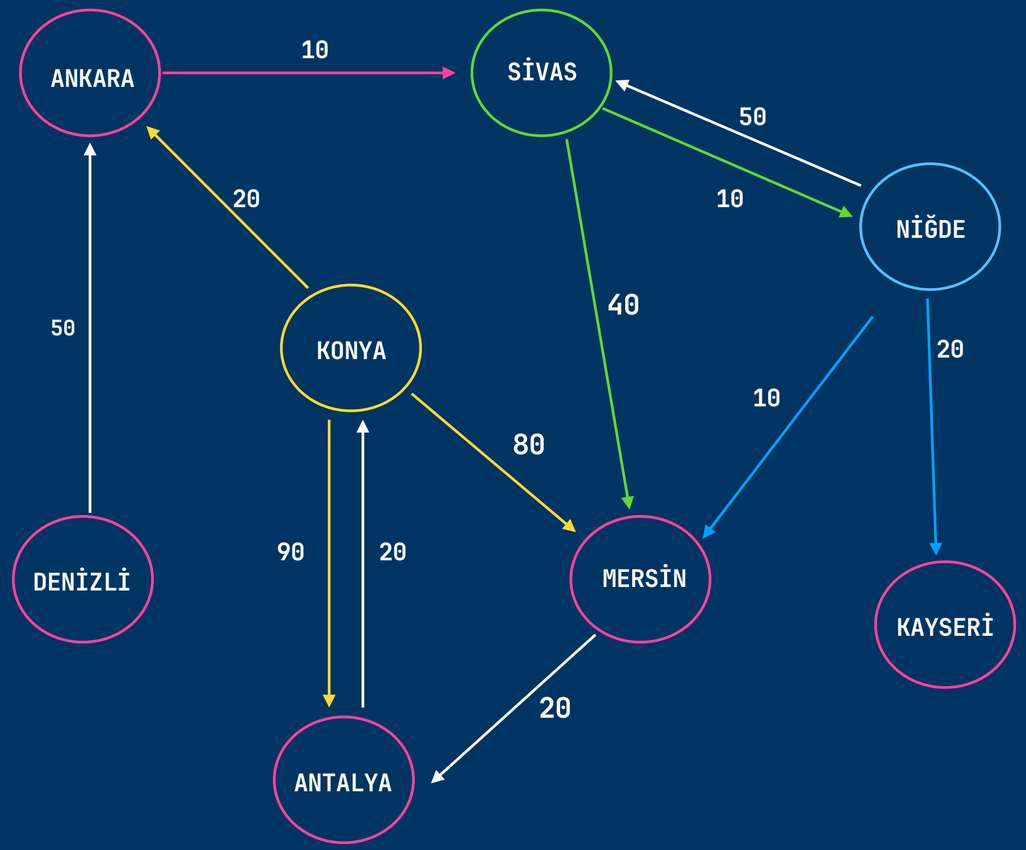
DÜĞÜM	KONYA	ANKARA	NİĞDE	MERSİN	DENİZLİ	SİVAS	ANTALYA	KAYSERİ
KONYA 0	0	~~∞~~ 20 P: KONYA	∞	~~∞~~ 80 P: KONYA	∞	∞	~~∞~~ 90 P: KONYA	∞
ANKARA 20	0	20 P: KONYA	∞	80 P: KONYA	∞	~~∞~~ 20 + 10 P: ANKARA	90 P: KONYA	∞
SİVAS 30	0	20 P: KONYA	~~∞~~ 30 + 10 P: SİVAS	~~80~~ 30 + 40 P: SİVAS	∞	30 P: ANKARA	90 P: KONYA	∞
NİĞDE 40	0	20 P: KONYA	40 P: SİVAS	70 P: SİVAS	∞	30 P: ANKARA	90 P: KONYA	∞

7) DJIKSTRA ALGORITHM



DÜĞÜM	KONYA	ANKARA	NİĞDE	MERSİN	DENİZLİ	SİVAS	ANTALYA	KAYSERİ
KONYA 0	0	~~∞~~ 20 P: KONYA	∞	~~∞~~ 80 P: KONYA	∞	∞	~~∞~~ 90 P: KONYA	∞
ANKARA 20	0	20 P: KONYA	∞	80 P: KONYA	∞	~~∞~~ 20 + 10 P: ANKARA	90 P: KONYA	∞
SİVAS 30	0	20 P: KONYA	~~∞~~ 30 + 10 P: SİVAS	~~80~~ 30 + 40 P: SİVAS	∞	30 P: ANKARA	90 P: KONYA	∞
NİĞDE 40	0	20 P: KONYA	40 P: SİVAS	70 P: SİVAS	∞	30 P: ANKARA	90 P: KONYA	~~∞~~ 60 P: NİĞE

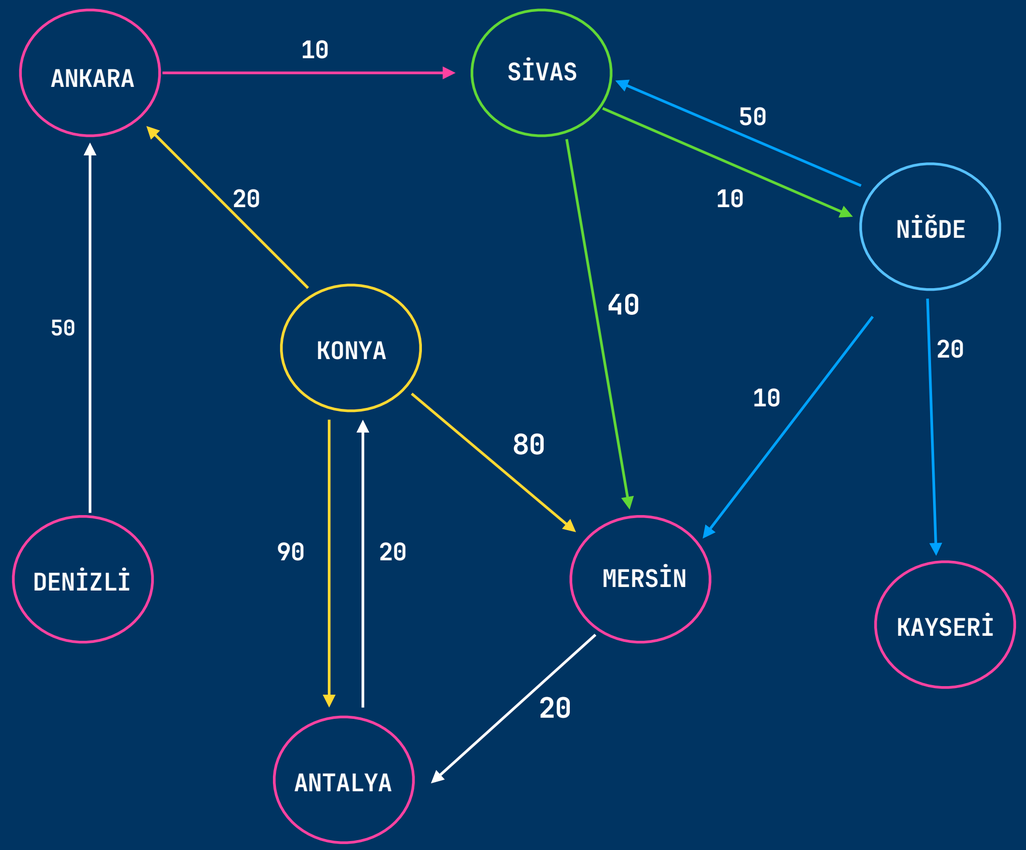
7) DJIKSTRA ALGORITHM



DÜĞÜM	KONYA	ANKARA	NİĞDE	MERSİN	DENİZLİ	SİVAS	ANTALYA	KAYSERİ
KONYA 0	0	~~∞~~ 20 P: KONYA	∞	~~∞~~ 80 P: KONYA	∞	∞	~~∞~~ 90 P: KONYA	∞
ANKARA 20	0	20 P: KONYA	∞	80 P: KONYA	∞	~~∞~~ 20 + 10 P: ANKARA	90 P: KONYA	∞
SİVAS 30	0	20 P: KONYA	~~∞~~ 30 + 10 P: SİVAS	~~80~~ 30 + 40 P: SİVAS	∞	30 P: ANKARA	90 P: KONYA	∞
NİĞDE 40	0	20 P: KONYA	40 P: SİVAS	~~70~~ 50 P: NİĞDE	∞	30 P: ANKARA	90 P: KONYA	~~∞~~ 90 P: NİĞDE

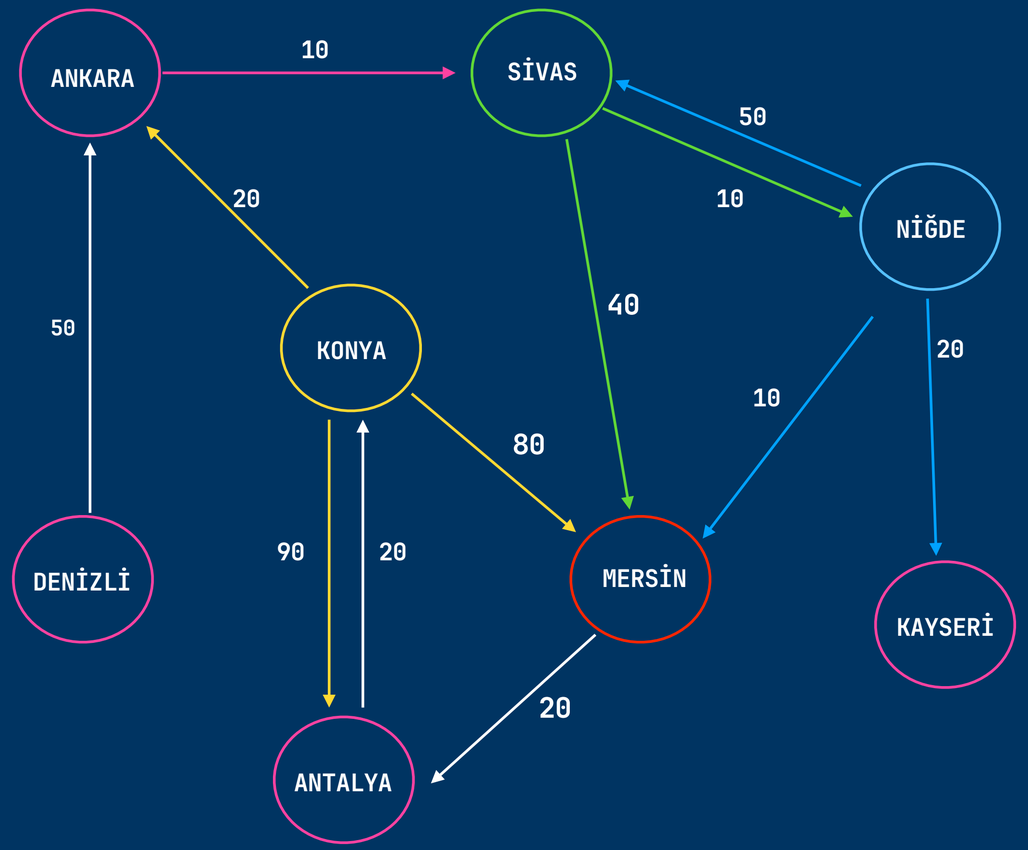
7) DJIKSTRA ALGORITHM

SİVAS'IN MALİYETİNDEN DOLAYI GÜNCELLEME OLMAZ



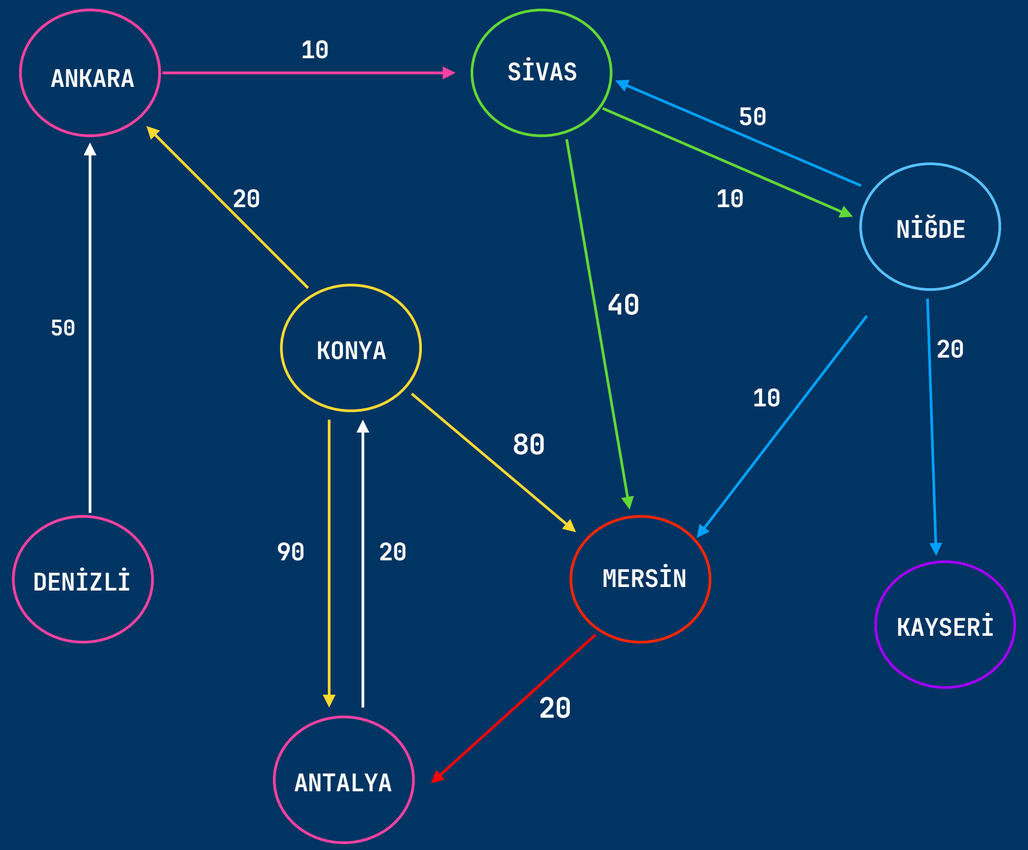
DÜĞÜM	KONYA	ANKARA	NİĞDE	MERSİN	DENİZLİ	SİVAS	ANTALYA	KAYSERİ
KONYA 0	0	~~∞~~ 20 P: KONYA	∞	~~∞~~ 80 P: KONYA	∞	∞	~~∞~~ 90 P: KONYA	∞
ANKARA 20	0	20 P: KONYA	∞	80 P: KONYA	∞	~~∞~~ 20 + 10 P: ANKARA	90 P: KONYA	∞
SİVAS 30	0	20 P: KONYA	~~∞~~ 30 + 10 P: SİVAS	~~80~~ 30 + 40 P: SİVAS	∞	30 P: ANKARA	90 P: KONYA	∞
NİĞDE 40	0	20 P: KONYA	40 P: SİVAS	~~70~~ 50 P: NİĞDE	∞	30 P: ANKARA	90 P: KONYA	~~∞~~ 60 P: NİĞDE

7) DJIKSTRA ALGORITHM



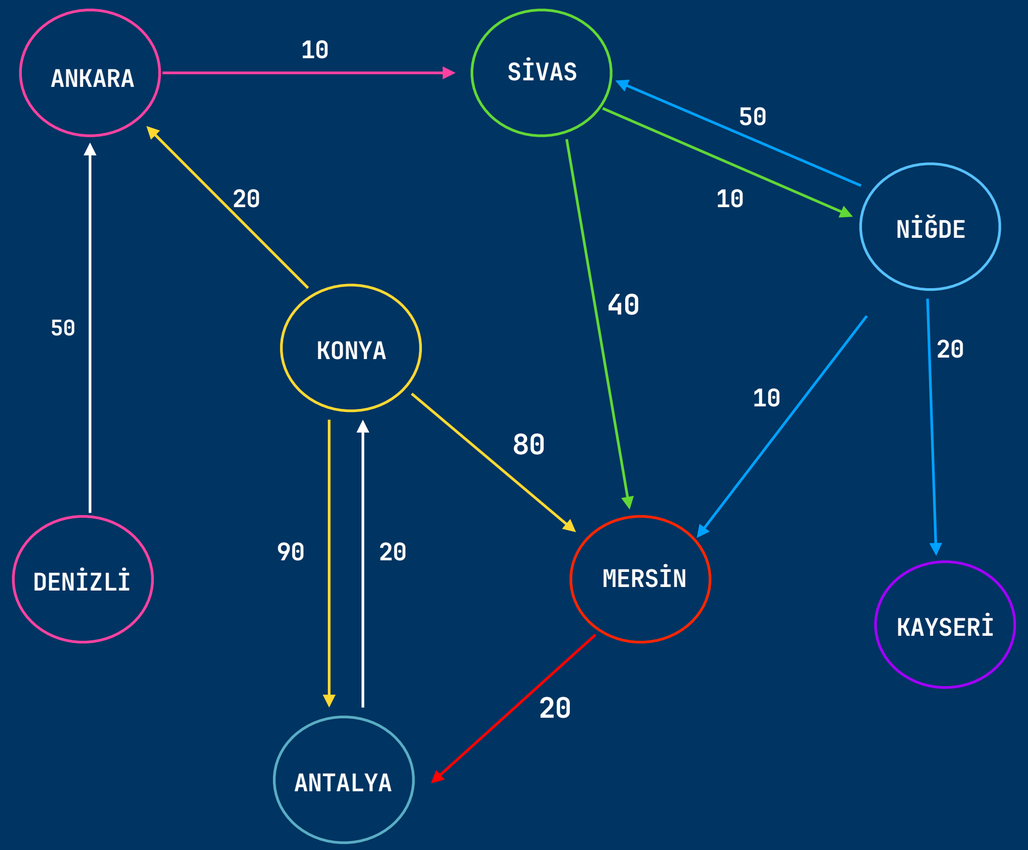
DÜĞÜM	KONYA	ANKARA	NİĞDE	MERSİN	DENİZLİ	SİVAS	ANTALYA	KAYSERİ
KONYA 0	0	~~∞~~ 20 P: KONYA	∞	~~∞~~ 80 P: KONYA	∞	∞	~~∞~~ 90 P: KONYA	∞
ANKARA 20	0	20 P: KONYA	∞	80 P: KONYA	∞	~~∞~~ 20 + 10 P: ANKARA	90 P: KONYA	∞
SİVAS 30	0	20 P: KONYA	~~∞~~ 30 + 10 P: SİVAS	~~80~~ 30 + 40 P: SİVAS	∞	30 P: ANKARA	90 P: KONYA	∞
NİĞDE 40	0	20 P: KONYA	40 P: SİVAS	~~70~~ 50 P: NİĞDE	∞	30 P: ANKARA	90 P: KONYA	~~∞~~ 60 P: NİĞDE
MERSİN 50	0	20 P: KONYA	40 P: SİVAS	50 P: NİĞDE	∞	30 P: ANKARA	90 P: KONYA	60 P: NİĞDE

7) DJIKSTRA ALGORITHM



DÜĞÜM	KONYA	ANKARA	NİĞDE	MERSİN	DENİZLİ	SİVAS	ANTALYA	KAYSERİ
KONYA 0	0	~~∞~~ 20 P: KONYA	∞	~~∞~~ 80 P: KONYA	∞	∞	~~∞~~ 90 P: KONYA	∞
ANKARA 20	0	20 P: KONYA	∞	80 P: KONYA	∞	~~∞~~ 20 + 10 P: ANKARA	90 P: KONYA	∞
SİVAS 30	0	20 P: KONYA	~~∞~~ 30 + 10 P: SİVAS	~~80~~ 30 + 40 P: SİVAS	∞	30 P: ANKARA	90 P: KONYA	∞
NİĞDE 40	0	20 P: KONYA	40 P: SİVAS	~~70~~ 50 P: NİĞDE	∞	30 P: ANKARA	90 P: KONYA	~~∞~~ 60 P: NİĞDE
MERSİN 50	0	20 P: KONYA	40 P: SİVAS	50 P: NİĞDE	∞	30 P: ANKARA	~~90~~ 70 P: MERSİN	60 P: NİĞDE
KAYSERİ 60	0	20 P: KONYA	40 P: SİVAS	50 P: NİĞDE	∞	30 P: ANKARA	70 P: MERSİN	60 P: NİĞDE

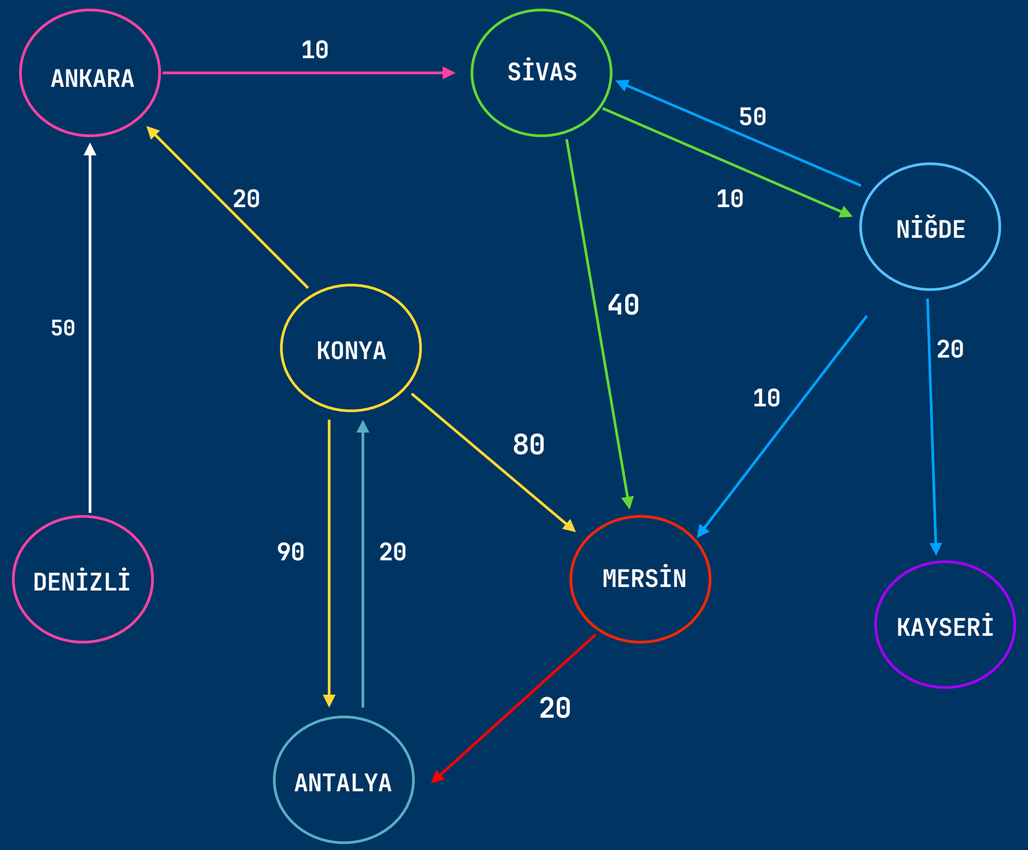
7) DJIKSTRA ALGORITHM



DÜĞÜM	KONYA	ANKARA	NİĞDE	MERSİN	DENİZLİ	SİVAS	ANTALYA	KAYSERİ
KONYA 0	0	~~∞~~ 20 P: KONYA	∞	~~∞~~ 80 P: KONYA	∞	∞	~~∞~~ 90 P: KONYA	∞
ANKARA 20	0	20 P: KONYA	∞	80 P: KONYA	∞	~~∞~~ 20 + 10 P: ANKARA	90 P: KONYA	∞
SİVAS 30	0	20 P: KONYA	~~∞~~ 30 + 10 P: SİVAS	~~80~~ 30 + 40 P: SİVAS	∞	30 P: ANKARA	90 P: KONYA	∞
NİĞDE 40	0	20 P: KONYA	40 P: SİVAS	~~70~~ 50 P: NİĞDE	∞	30 P: ANKARA	90 P: KONYA	~~∞~~ 60 P: NİĞDE
MERSİN 50	0	20 P: KONYA	40 P: SİVAS	50 P: NİĞDE	∞	30 P: ANKARA	~~90~~ 70 P: MERSİN	60 P: NİĞDE
KAYSERİ 60	0	20 P: KONYA	40 P: SİVAS	50 P: NİĞDE	∞	30 P: ANKARA	70 P: MERSİN	60 P: NİĞDE
ANTALYA 70	0	20 P: KONYA	40 P: SİVAS	50 P: NİĞDE	∞	30 P: ANKARA	70 P: MERSİN	60 P: NİĞDE

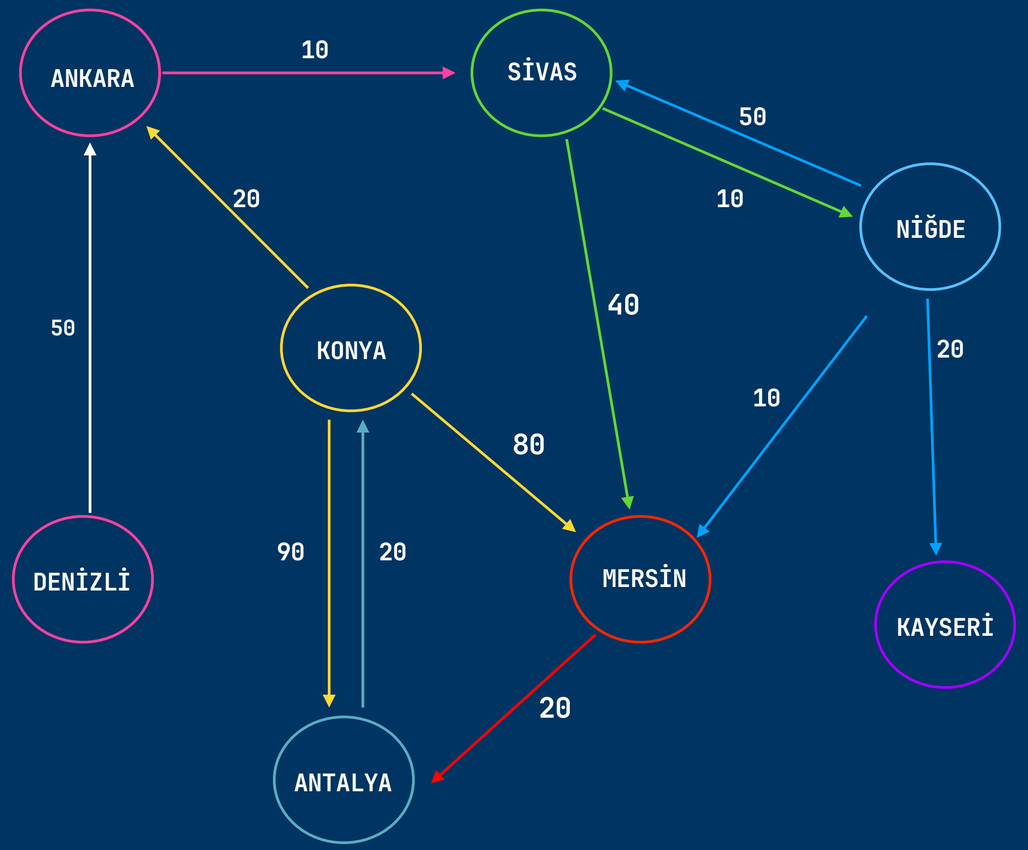
7) DJIKSTRA ALGORITHM

Bulunduğum yer Konya
Maaliyeti 0
Bu yüzden güncelleme olmaz



DÜĞÜM	KONYA	ANKARA	NİĞDE	MERSİN	DENİZLİ	SİVAS	ANTALYA	KAYSERİ
KONYA 0	0	~~∞~~ 20 P: KONYA	∞	~~∞~~ 80 P: KONYA	∞	∞	~~∞~~ 90 P: KONYA	∞
ANKARA 20	0	20 P: KONYA	∞	80 P: KONYA	∞	~~∞~~ 20 + 10 P: ANKARA	90 P: KONYA	∞
SİVAS 30	0	20 P: KONYA	~~∞~~ 30 + 10 P: SİVAS	~~80~~ 30 + 40 P: SİVAS	∞	30 P: ANKARA	90 P: KONYA	∞
NİĞDE 40	0	20 P: KONYA	40 P: SİVAS	~~70~~ 50 P: NİĞDE	∞	30 P: ANKARA	90 P: KONYA	~~∞~~ 60 P: NİĞDE
MERSİN 50	0	20 P: KONYA	40 P: SİVAS	50 P: NİĞDE	∞	30 P: ANKARA	~~90~~ 70 P: MERSİN	60 P: NİĞDE
KAYSERİ 60	0	20 P: KONYA	40 P: SİVAS	50 P: NİĞDE	∞	30 P: ANKARA	70 P: MERSİN	60 P: NİĞDE
ANTALYA 70	0	20 P: KONYA	40 P: SİVAS	50 P: NİĞDE	∞	30 P: ANKARA	70 P: MERSİN	60 P: NİĞDE

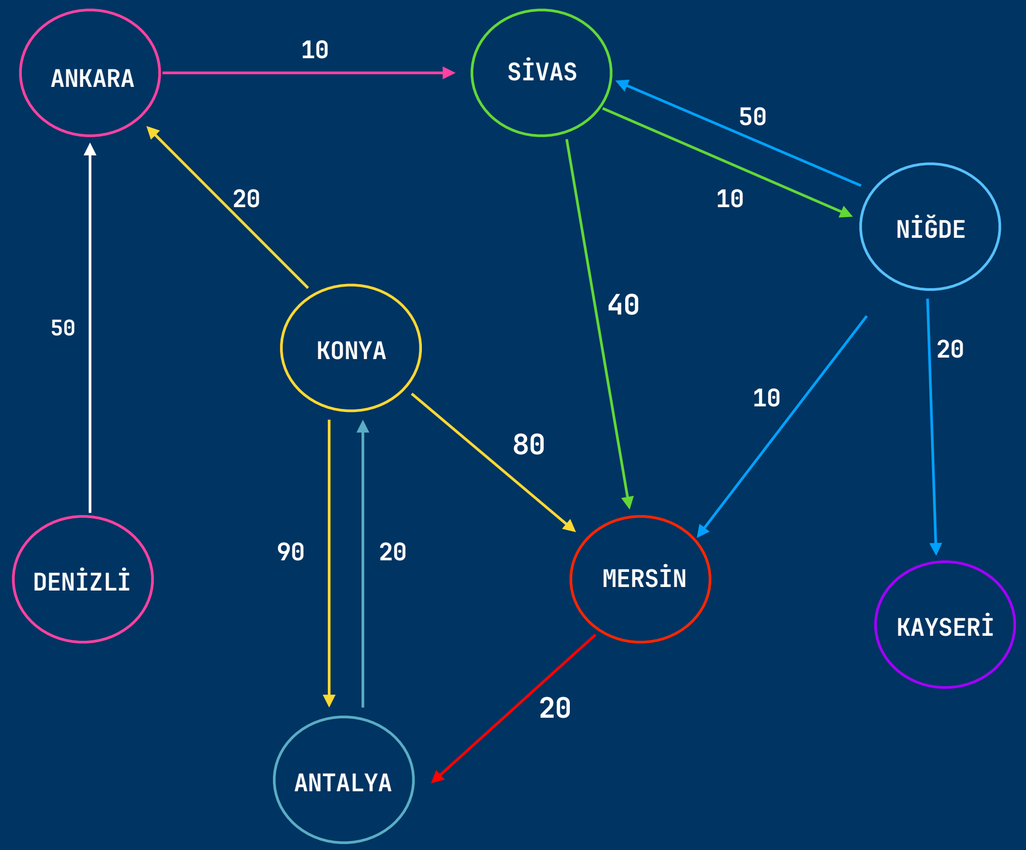
7) DJIKSTRA ALGORITHM



DÜĞÜM	KONYA	ANKARA	NİĞDE	MERSİN	DENİZLİ	SİVAS	ANTALYA	KAYSERİ
KONYA 0	0	~~∞~~ 20 P: KONYA	∞	~~∞~~ 80 P: KONYA	∞	∞	~~∞~~ 90 P: KONYA	∞
ANKARA 20	0	20 P: KONYA	∞	80 P: KONYA	∞	~~∞~~ 20 + 10 P: ANKARA	90 P: KONYA	∞
SİVAS 30	0	20 P: KONYA	~~∞~~ 30 + 10 P: SİVAS	~~80~~ 30 + 40 P: SİVAS	∞	30 P: ANKARA	90 P: KONYA	∞
NİĞDE 40	0	20 P: KONYA	40 P: SİVAS	~~70~~ 50 P: NİĞDE	∞	30 P: ANKARA	90 P: KONYA	~~∞~~ 60 P: NİĞDE
MERSİN 50	0	20 P: KONYA	40 P: SİVAS	50 P: NİĞDE	∞	30 P: ANKARA	~~90~~ 70 P: MERSİN	60 P: NİĞDE
KAYSERİ 60	0	20 P: KONYA	40 P: SİVAS	50 P: NİĞDE	∞	30 P: ANKARA	70 P: MERSİN	60 P: NİĞDE
ANTALYA 70	0	20 P: KONYA	40 P: SİVAS	50 P: NİĞDE	∞	30 P: ANKARA	70 P: MERSİN	60 P: NİĞDE

DENİZLİYE ULAŞAN HERHANGİ BİR YOL OLMADIĞI İÇİN DENİZLİ ADIMI ATLANIR VE SONUCA ETKİ ETMEZ

7) DJIKSTRA ALGORITHM



SONUÇ OLARAK;

KONYA - ANKARA: 20

KONYA - SİVAS: 30

KONYA - MERSİN: 50

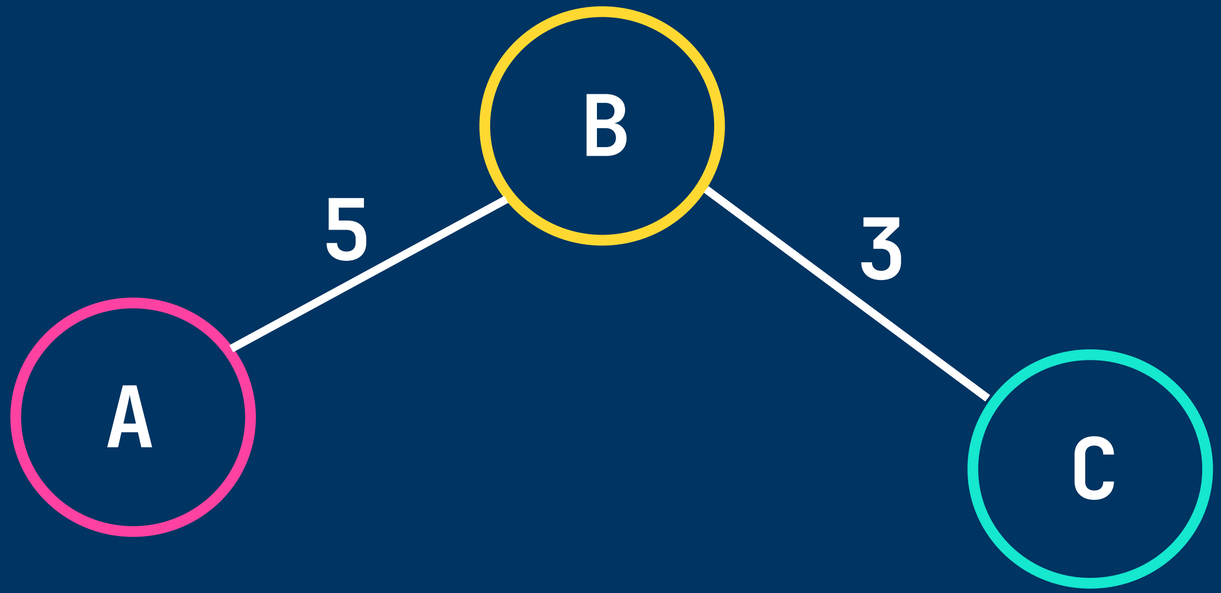
KONYA - ANTALYA: 70

KONYA - NİĞDE: 40

KONYA - KAYSERİ: 60

KONYA - DENİZLİ: SONSUZ

7) DJIKSTRA ALGORITHM



LEETCODE LİNKİ

https://leetcode.com/problems/add-two-integers/solutions/1968134/21-different-ways-to-solve-this-problem/

TEŞEKKÜRLER