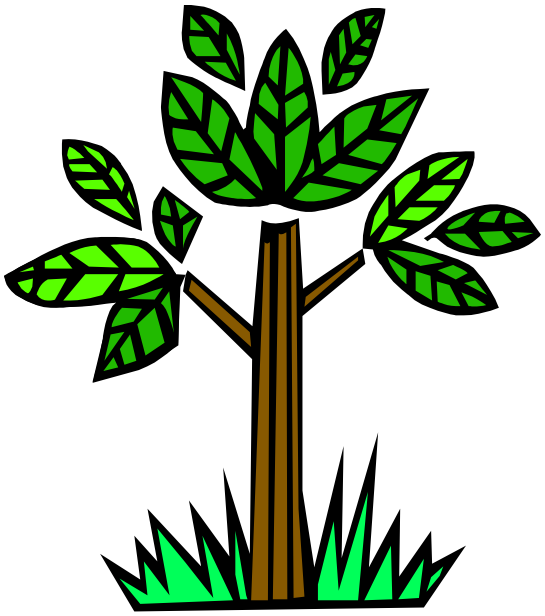
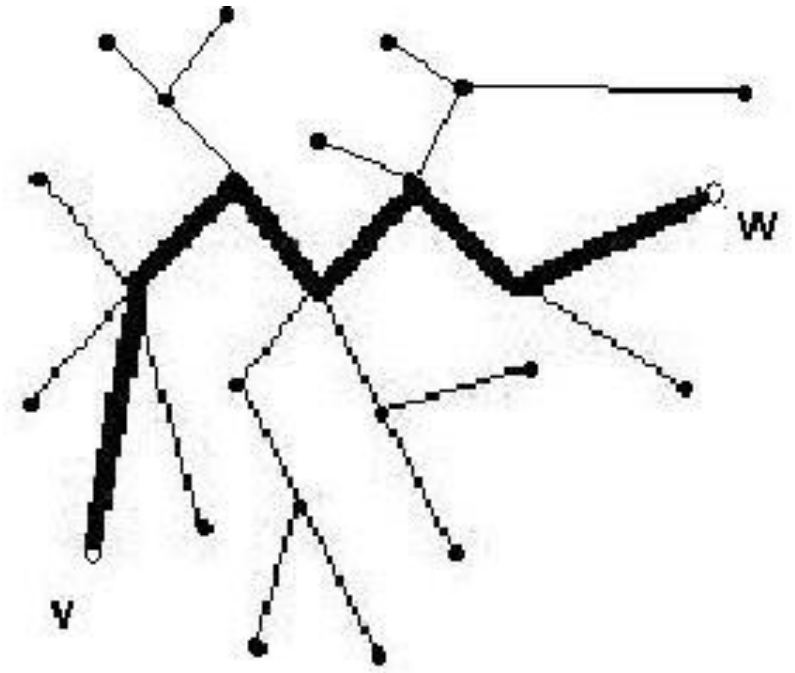
BÖLÜM 4
TREES (AĞAÇLAR)



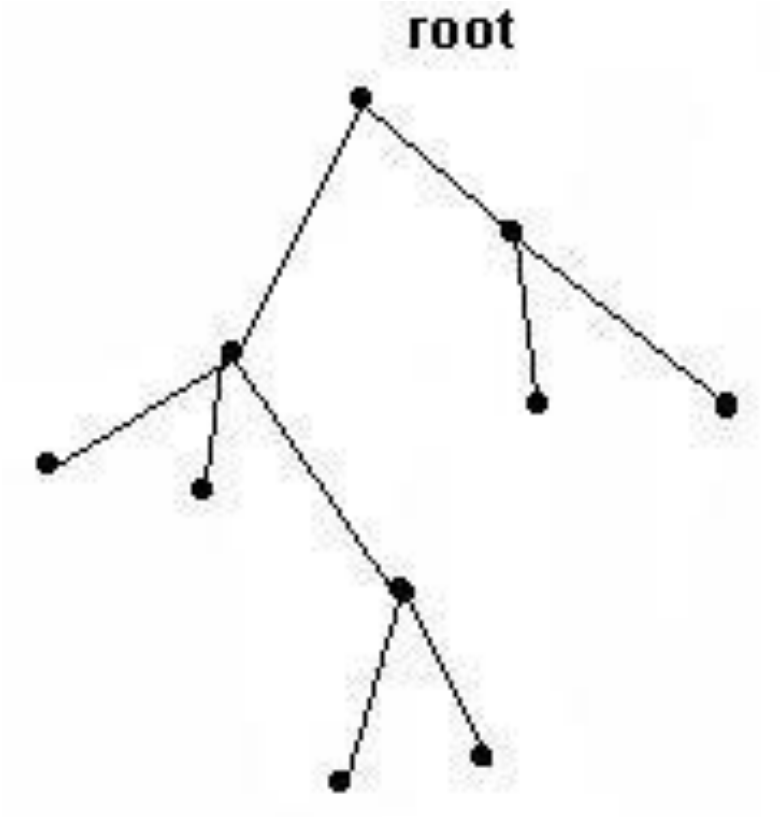
GİRİŞ



Bir ağaç

- ***v*** ve ***w*** düğüm çiftinden oluşan basit bir graftır
- ***v***'den ***w***'ya tek bir yol vardır
- dairesel bir yapı içermez

Köklü Ağaç (Rooted tree)



Düğümlerinden biri kök olarak (root) belirlenmiş bir ağaçtır

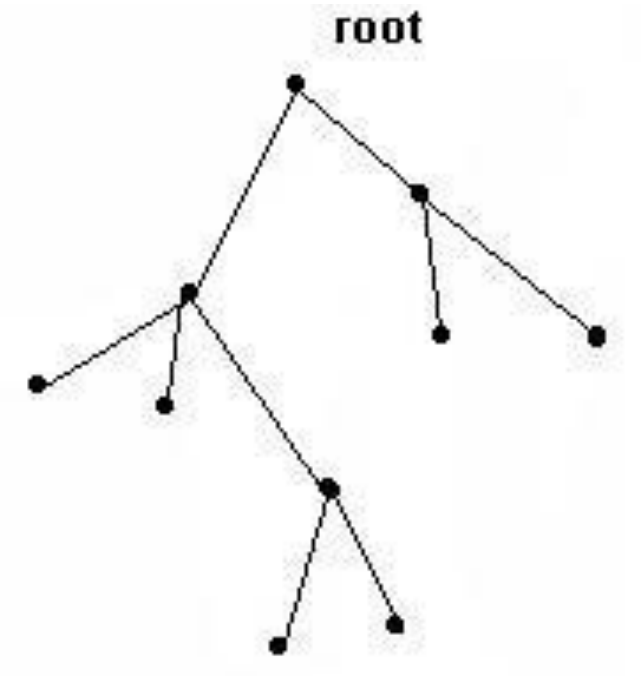
Düğüm seviyesi ve ağacın yüksekliği

T köklü bir ağaç olsun:

- Bir düğümün seviyesi, o düğümün *l(v)* bulunduğu yerden ağacın köküne olan yolun uzunluğudur
- Bir ağacın yüksekliği, o ağacın maksimum seviyeye sahip olan düğümünün değerine eşittir

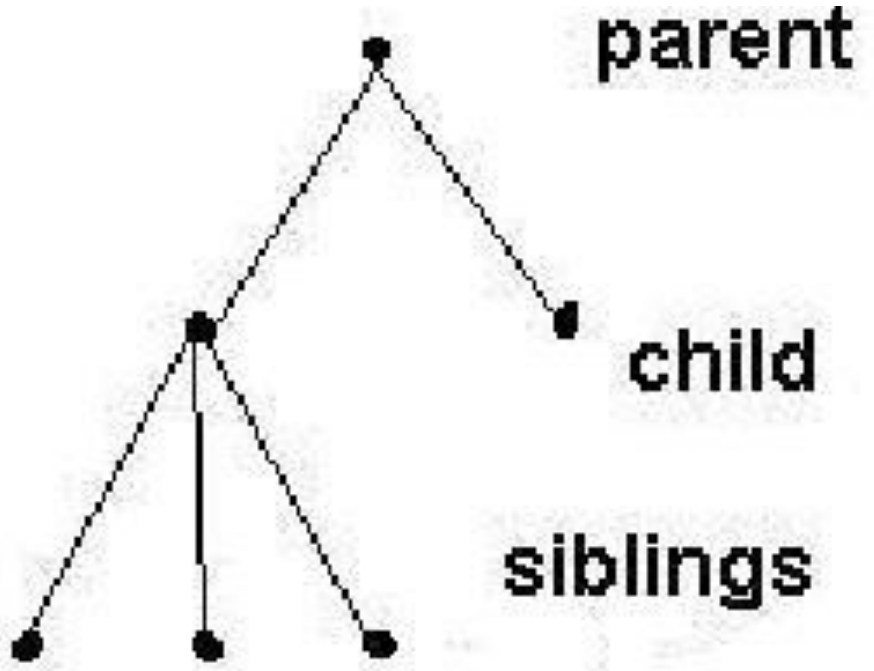
$$h = \max_{v \in V(T)} \{ l(v) \}$$

- Örnek:
  - sağdaki ağacın yüksekliği 3

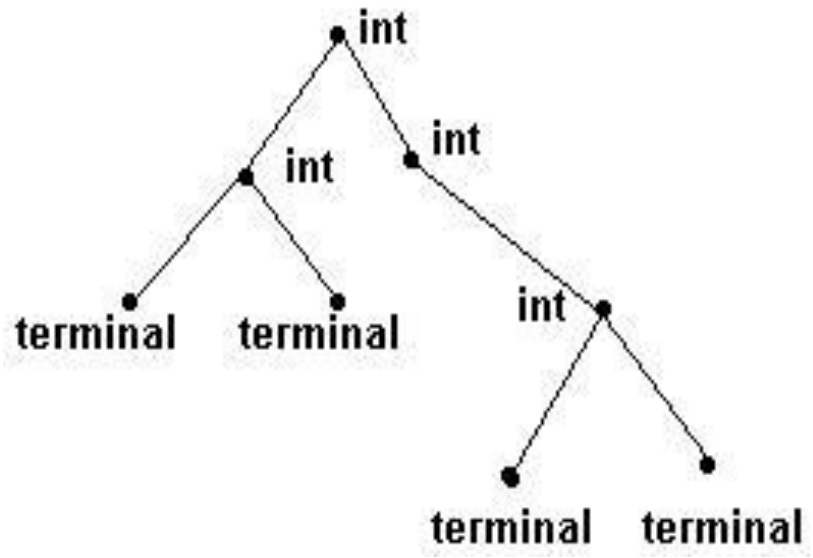


Terminoloji

- Parent (ana/baba)
- Ancestor (ata)
- Child (çocuk)
- Descendant (torun)
- Siblings(kardeş düğüm)
- Terminal vertices (uç düğüm)
- Internal vertices (ara düğüm)
- Subtrees (alt ağaçlar)



Ara (internal) ve Uç (terminal) Düğümler

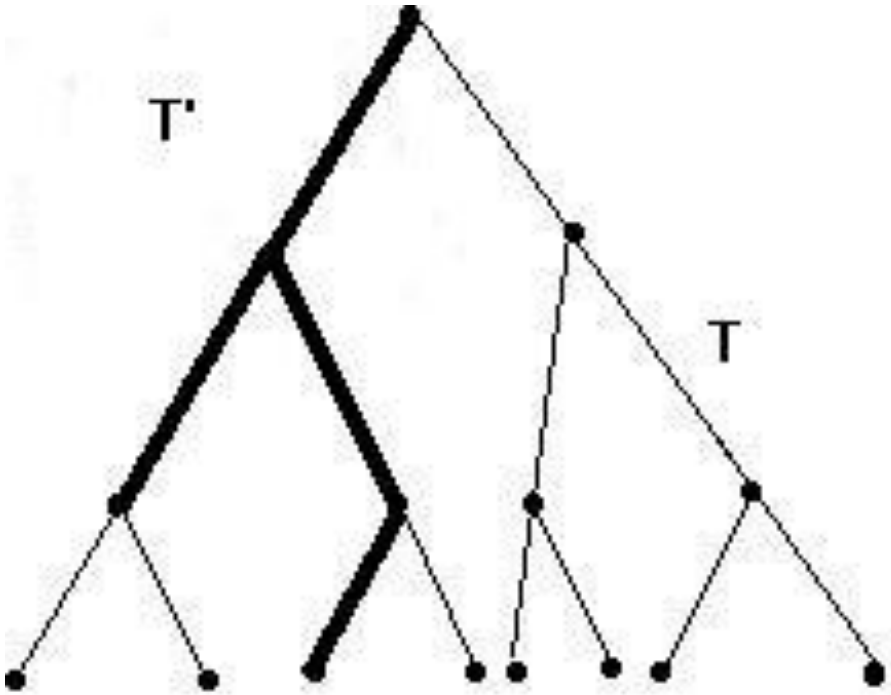


- En az bir çocuğa sahip olan düğümler ara (internal) düğümlerdir
- Hiç çocuğu olmayan düğümler uç (terminal) düğümlerdir
- Örnekteki ağaçta 4 adet internal ve 4 adet terminal düğüm mevcuttur

Alltağaçlar (Subtrees)

Bir **T** ağacının altağacı (subtree) **T'** olup

- $V(T') \subseteq V(T)$ and
- $E(T') \subseteq E(T)$



Ağaçların Karakteristiği

Theorem

Eğer **T**, *n* düğümlü bir graf ise aşağıdakiler doğrudur:

a) T bir ağaçtır

b) T bağlantılı (connected) ve acyclic

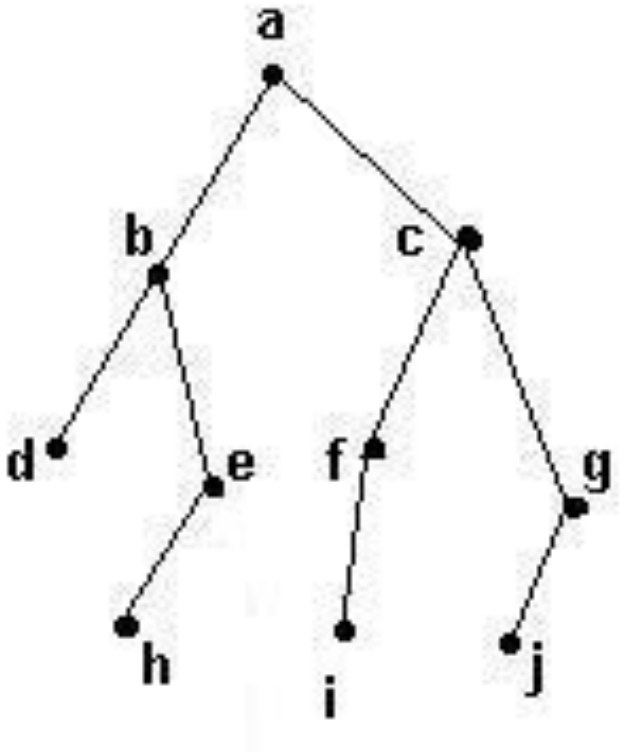
- (“acyclic” = döngü mevcut değil)

c) T bağlantılı ve n-1 kenara sahiptir

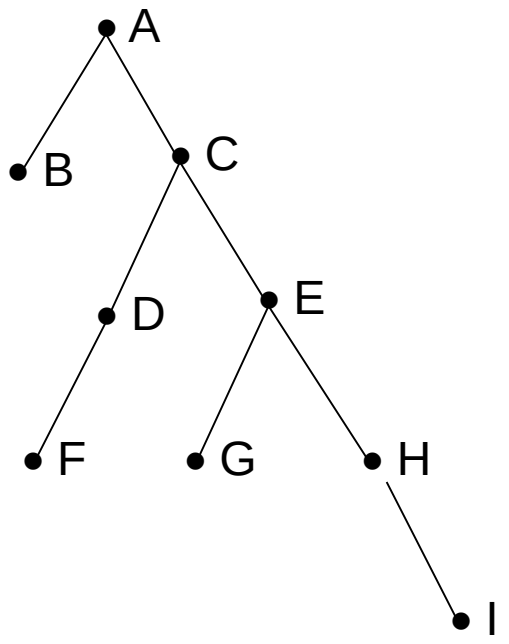
d) T acyclic ve n-1 kenara sahiptir

İkili Ağaçlar (Binary trees)

Bir ikili ağaçta, her bir düğüm en fazla iki çocuğa sahiptir. Bir düğüm 0,1 veya 2 çocuğa sahip olabilir

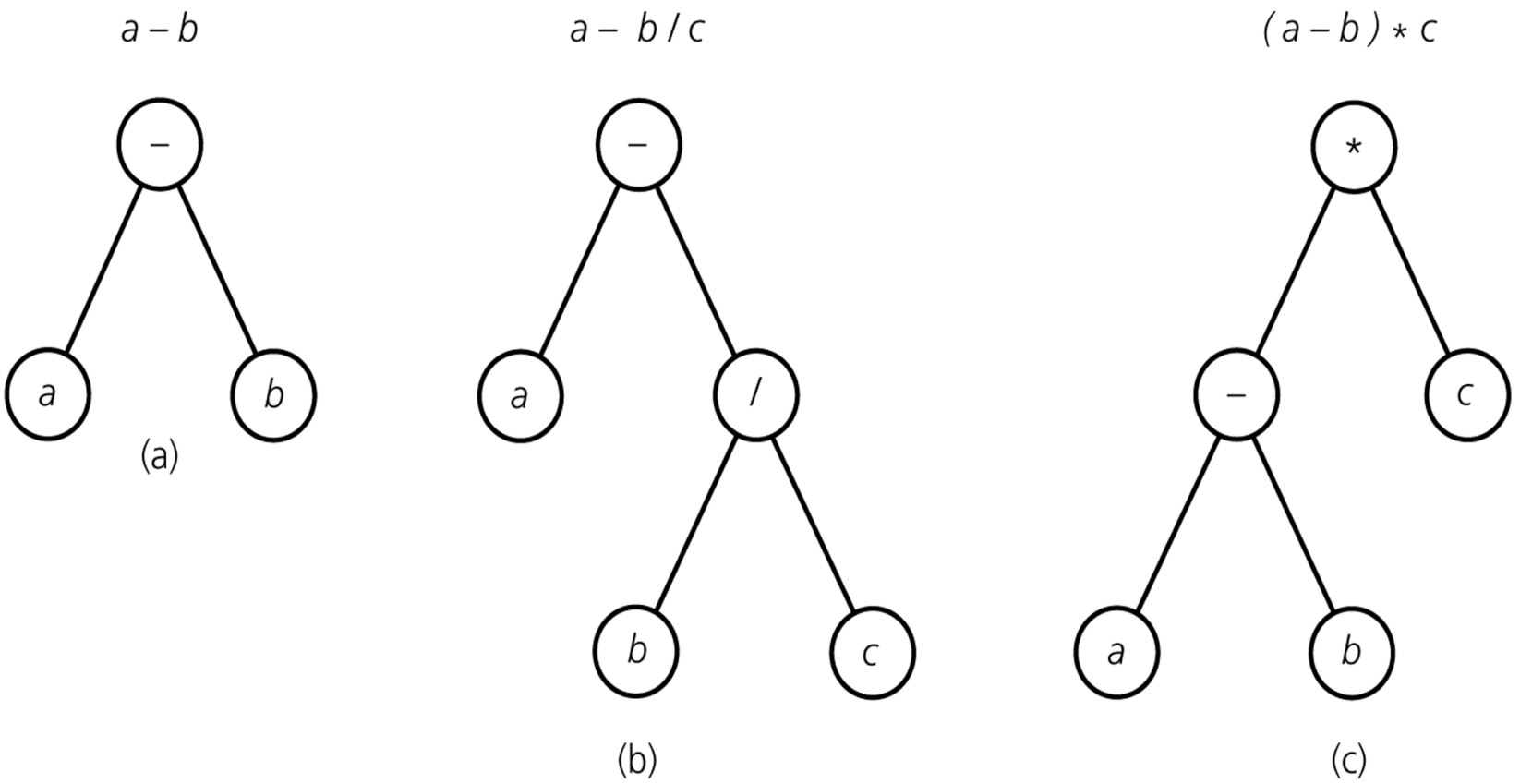


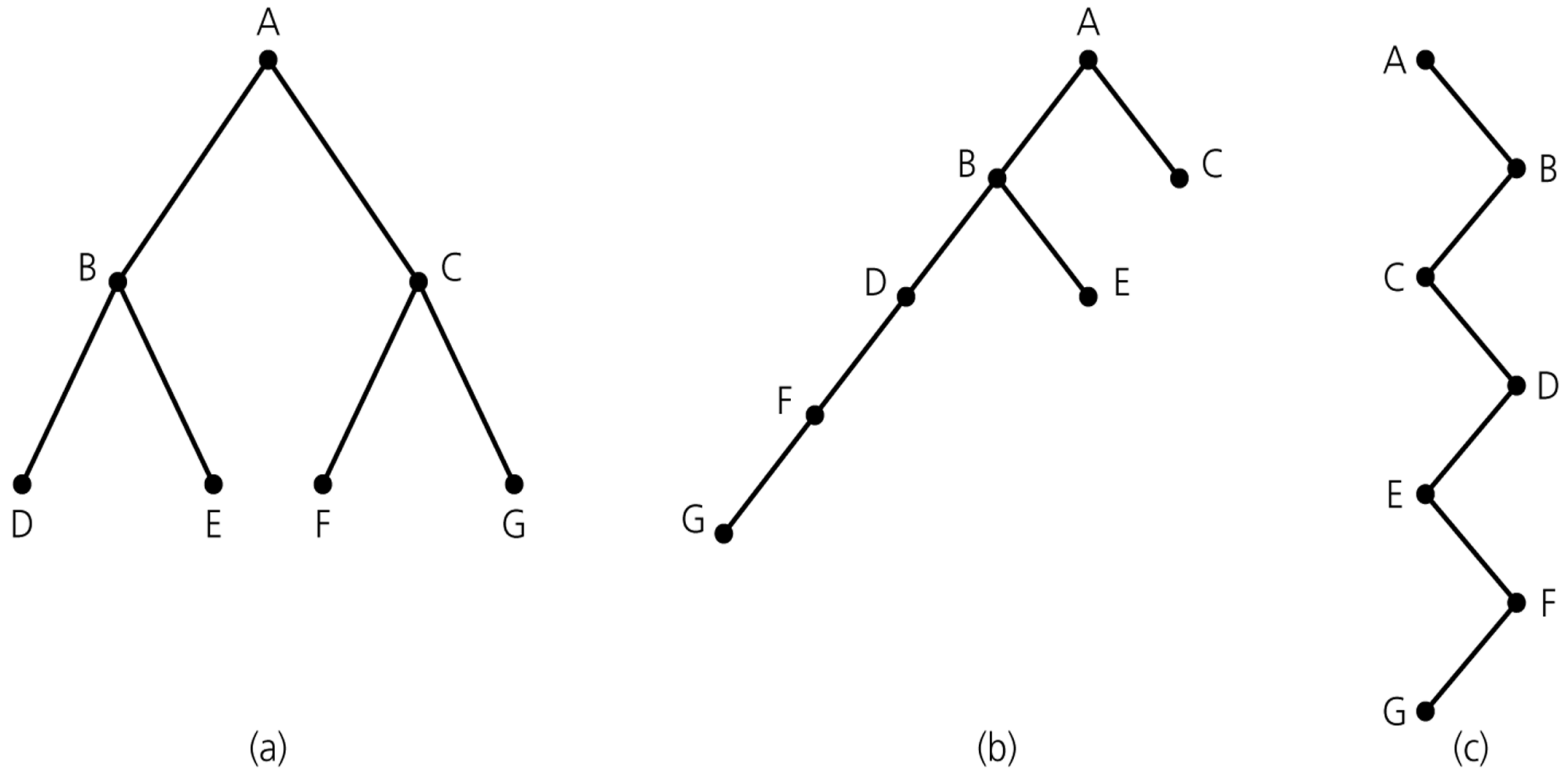
Binary Tree -- Örnek



- A is the root.
- B is the left child of A, and C is the right child of A.
- D doesn’t have a right child.
- H doesn’t have a left child.
- B, F, G and I are leaves.

Binary Tree – Matematiksel İfadelerin Gösterilimi





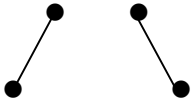
Aynı düğüm sayısına sahip, farklı yüksekliklerdeki İkili Ağaçlar

Aynı düğüm sayısına sahip farklı İkili Ağaçlar

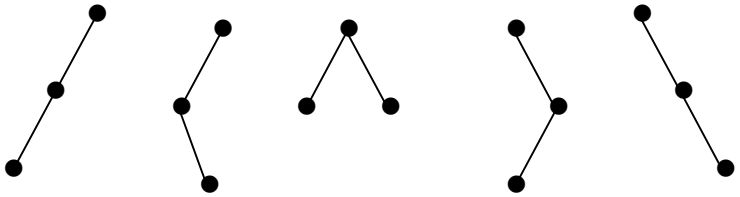
n=0 ➔ empty tree

n=1 ➔ (1 tree)

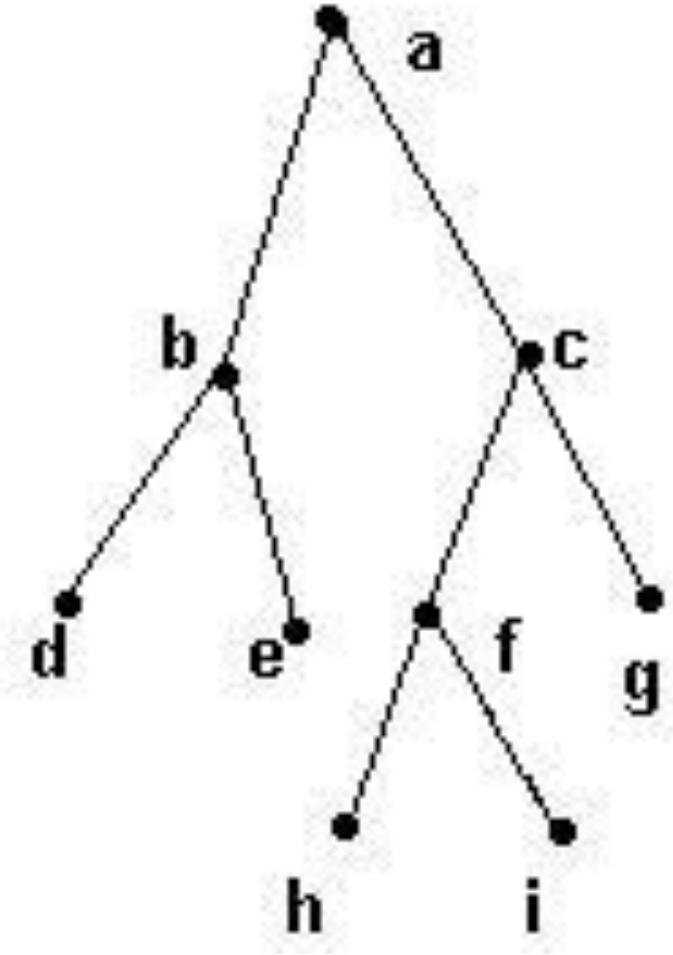
n=2 ➔ (2 trees)



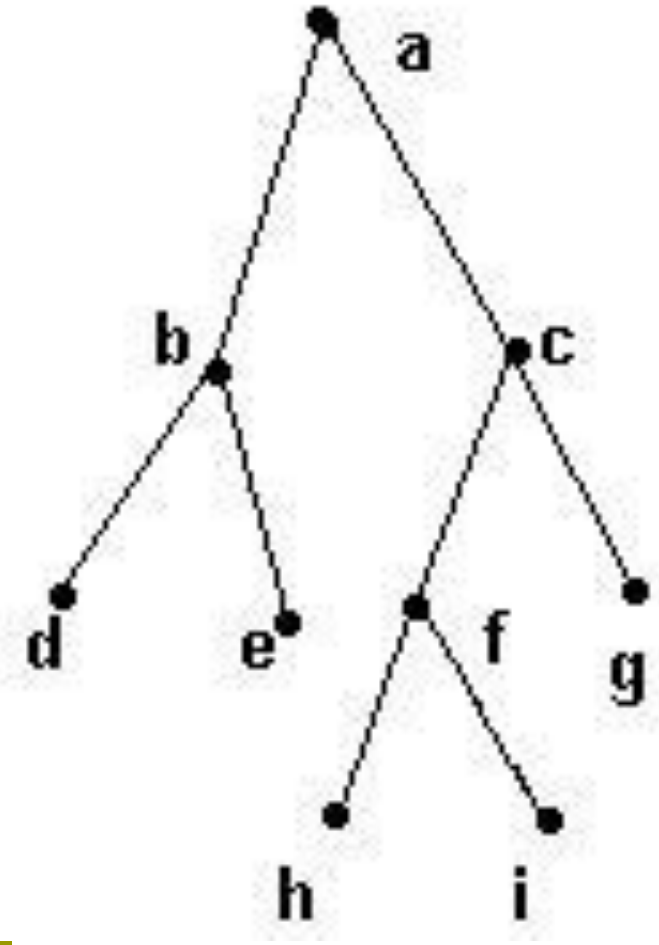
n=3 ➔ (5 trees)



Tam İkili Ağaç (Full binary tree)



Bir *full* binary tree ‘de uç düğümler hariç her düğüm iki çocuğa sahiptir.



Tam ikili ağaç (full binary tree)'da :

* ***t*** adet uç düğüm varsa ağacın yüksekliği ***lg(t)≤h***

* ***i*** adet ara düğümü varsa ***(i+1)*** adet uç düğümü vardır

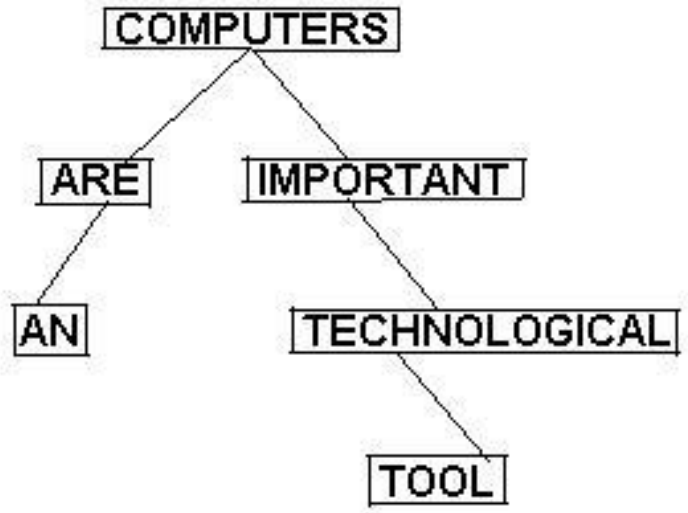
* ***i*** adet ara düğümü varsa toplam düğüm sayısı ***(2i+1)***

- Örnek: k = 4 internal vertices (a, b, c ve f) ve 5 terminal vertices (d, e, g, h and i) toplamda 9 düğüm

İkili Arama Ağaçları (Binary Search Trees)

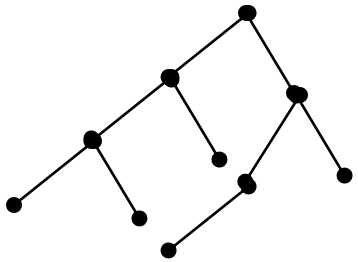
- Veri düğümlere yerleştirilir
- Veri alfabetik sırada ağaca yerleştirilir. Düğümün sol tarafındaki veri, o düğümdeki veriden daha küçüktür
- Ve düğümün sağ tarafındaki veri de o düğümdeki veriden daha büyüktür

- Örnek: "Computers are an important technological tool"

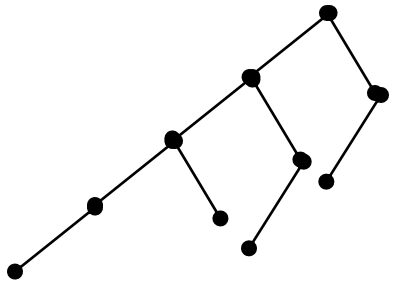


Dengeli Ağaç (Balanced Tree)

Yüksekliği ***h*** olan bir ağacın yapraklarının seviyesi ***h*** veya ***h-1*** ise dengeli ağaç adını alır



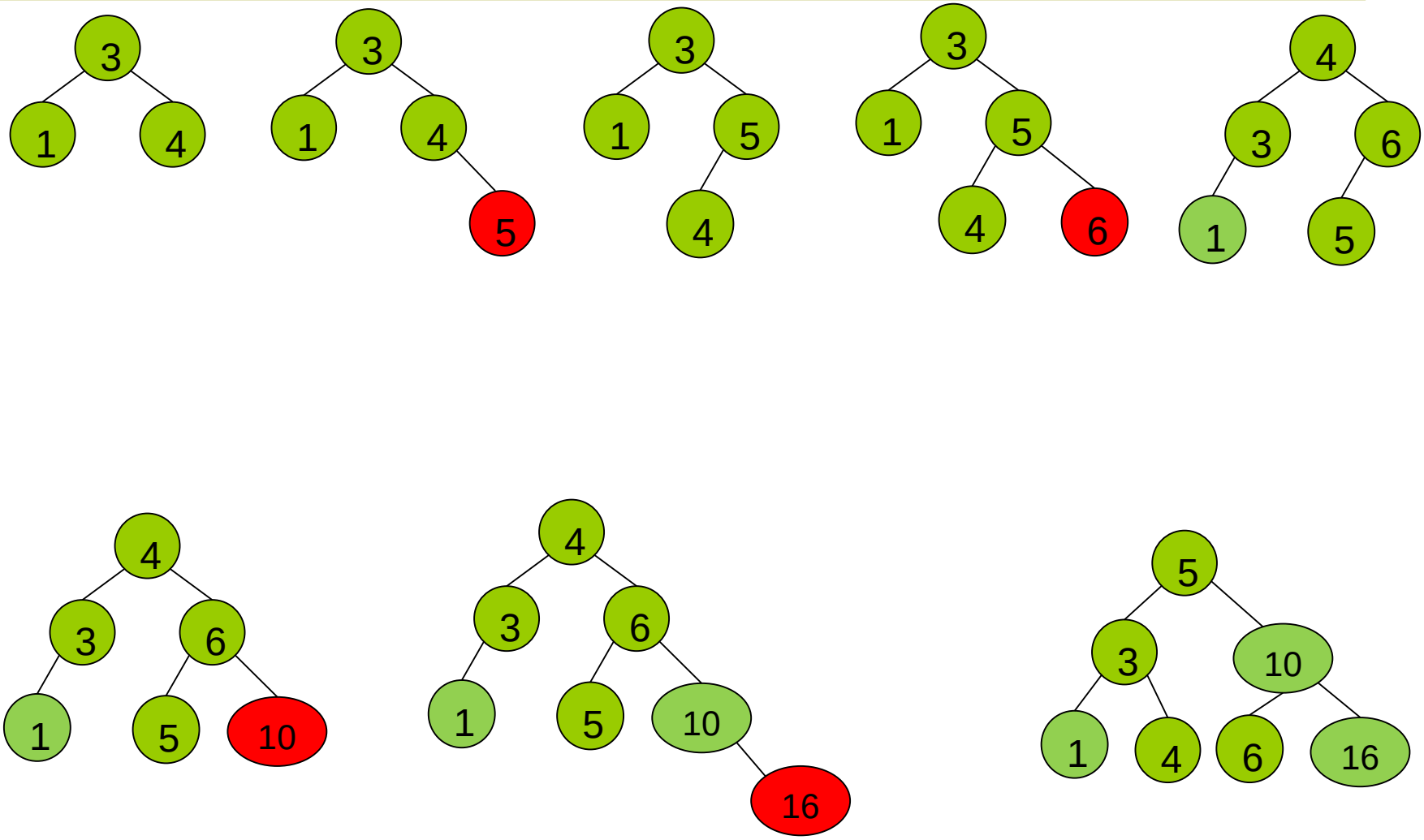
Dengeli ağaç



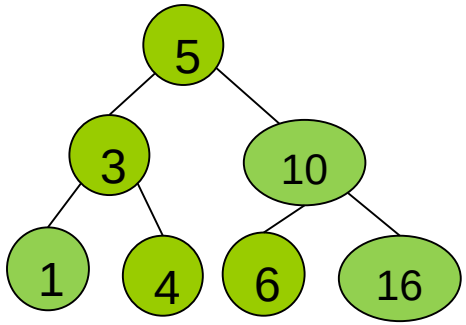
Dengesiz ağaç

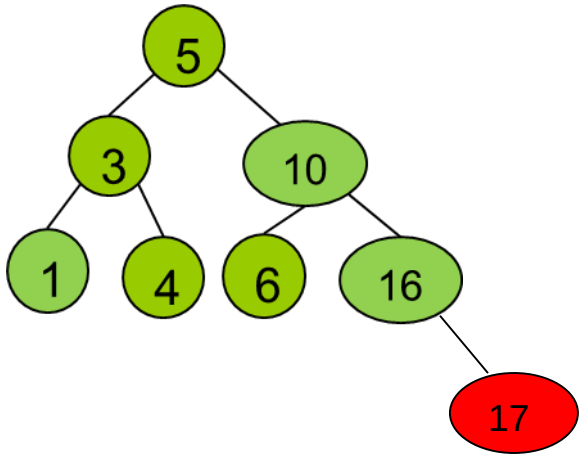
Örnek : Binary Search Tree Oluşturma

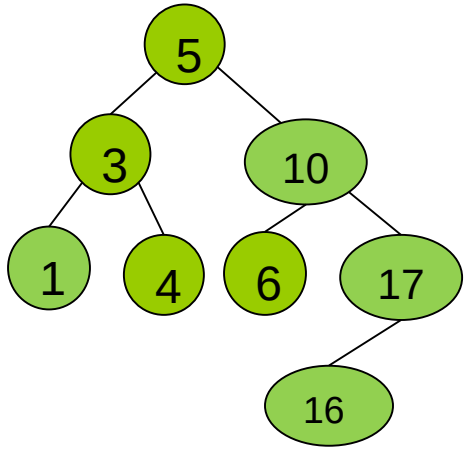
1 3 4 5 6 10 16 17 19

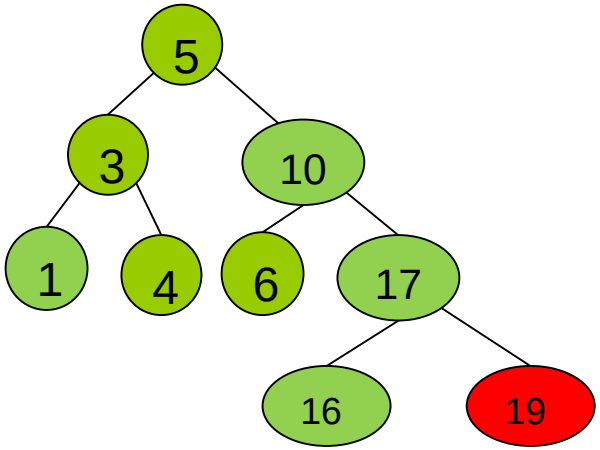


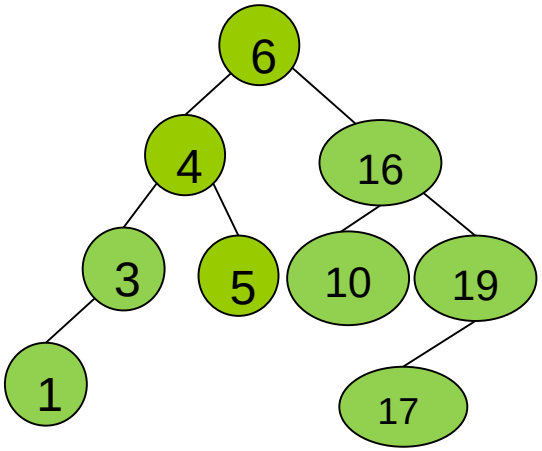
1 3 4 5 6 10 16 17 19



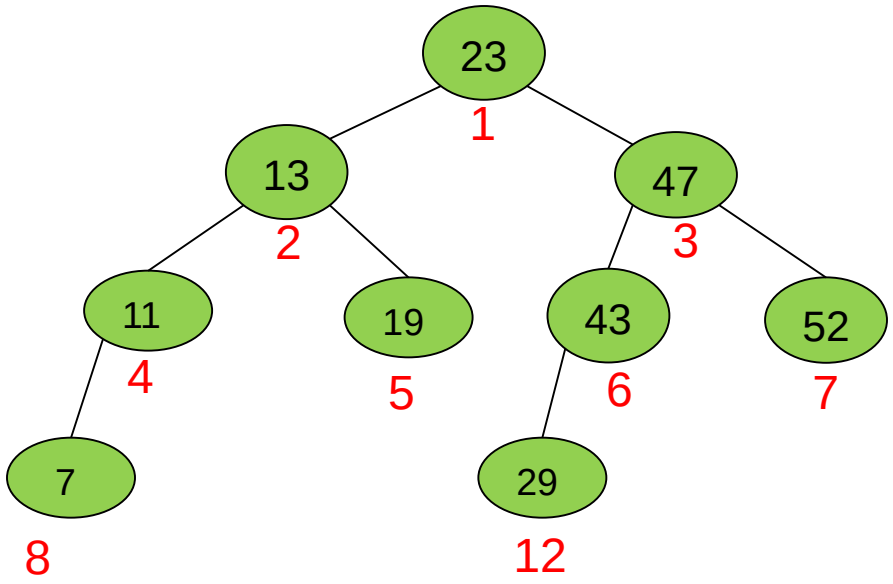








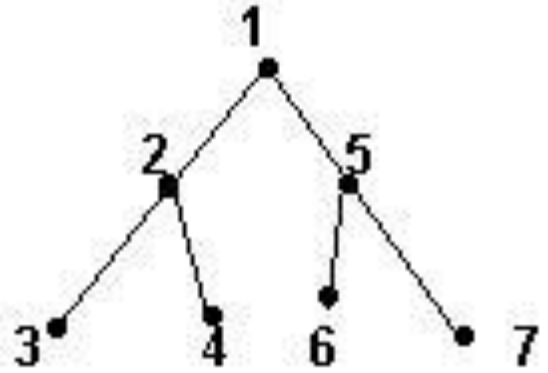
Binary Search uygun şekilde verilmiş olan **23 13 19 47 11 43 7 29 52** diziyi belirli bir kurala göre bir ağaca yerleştirelim.



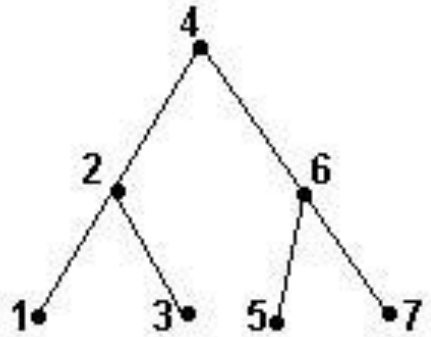
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
23	13	47	11	19	43	52	7	0	0	0	29	0	0	0

Ağacın Düğümlerinin Listesi (Tree Traversals)

- 1: Pre-order traversal
  (önden sıralı)

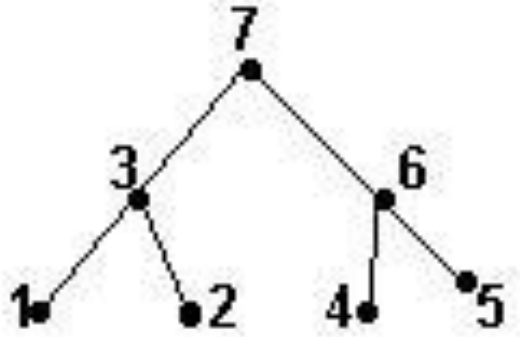


- 2: In-order traversal
  (içten sıralı)

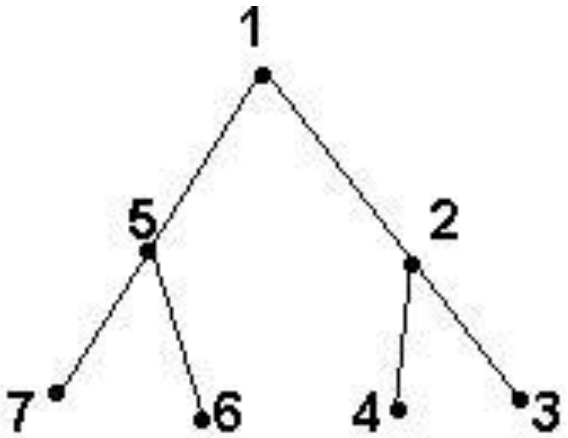


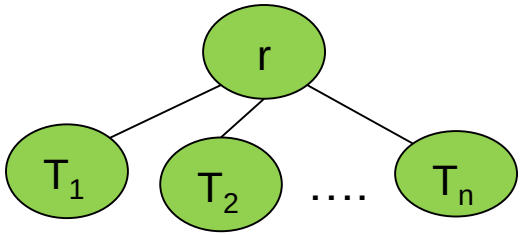
devam....

- 3: Post-order traversal
  (sondan sıralı)

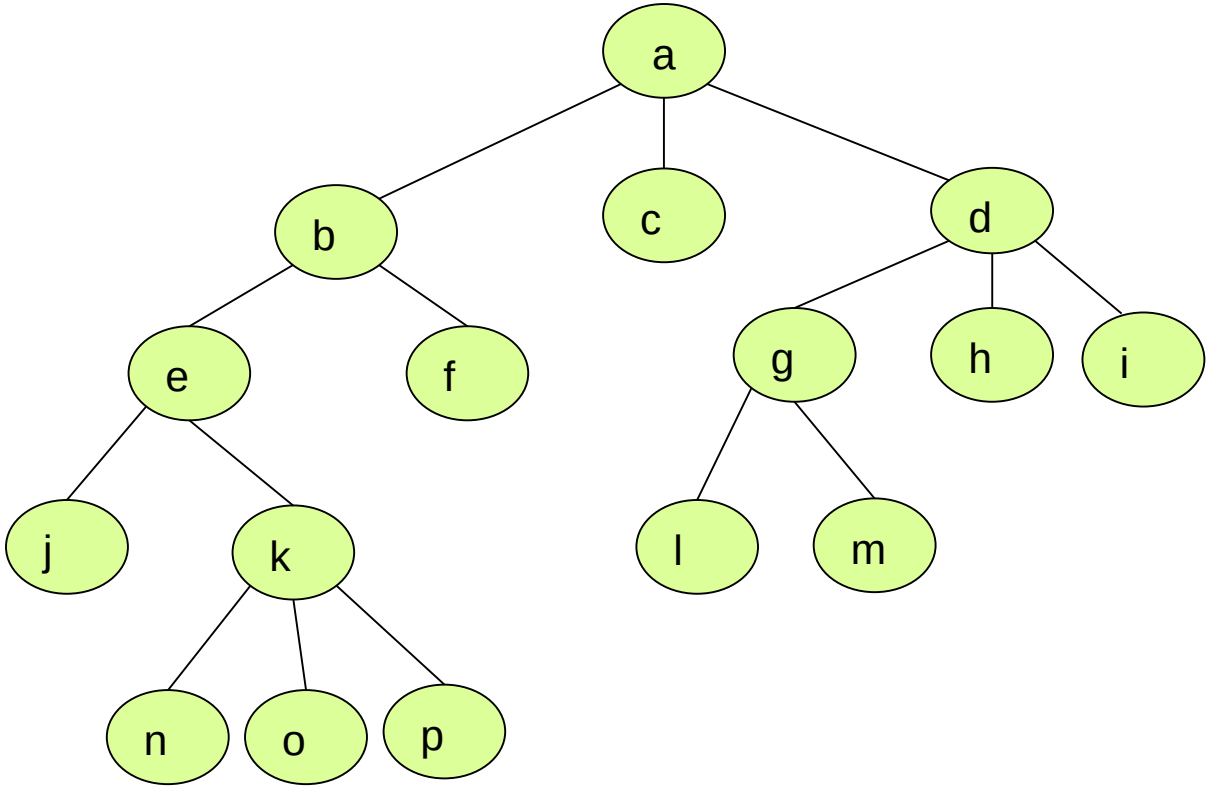


- 4: Reverse post-order traversal (ters önden sıralı)

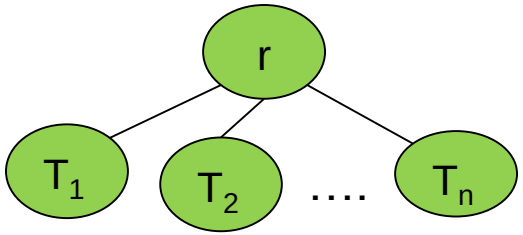




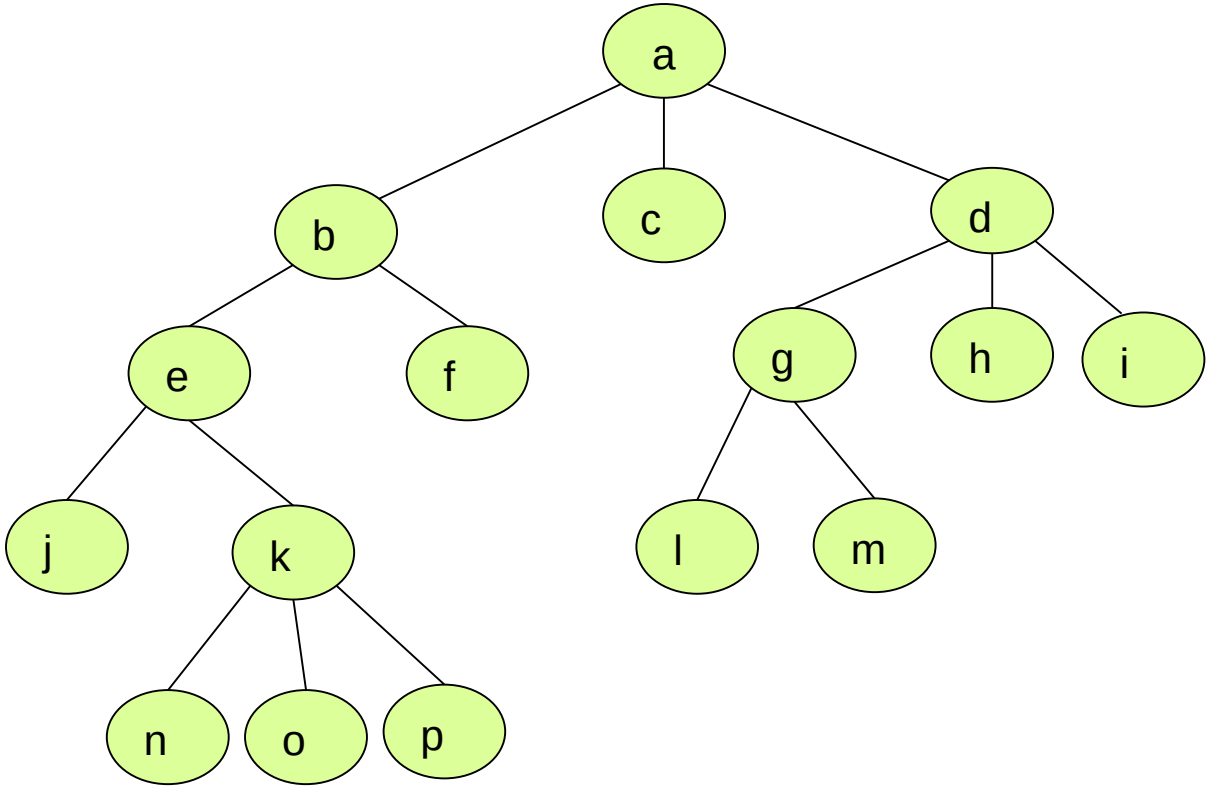
Pre-order r T_1 T_2 … T_n



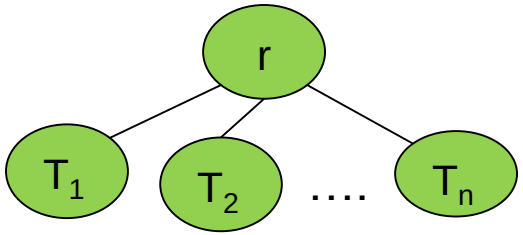
a b e j k n o p f c d g l m h i



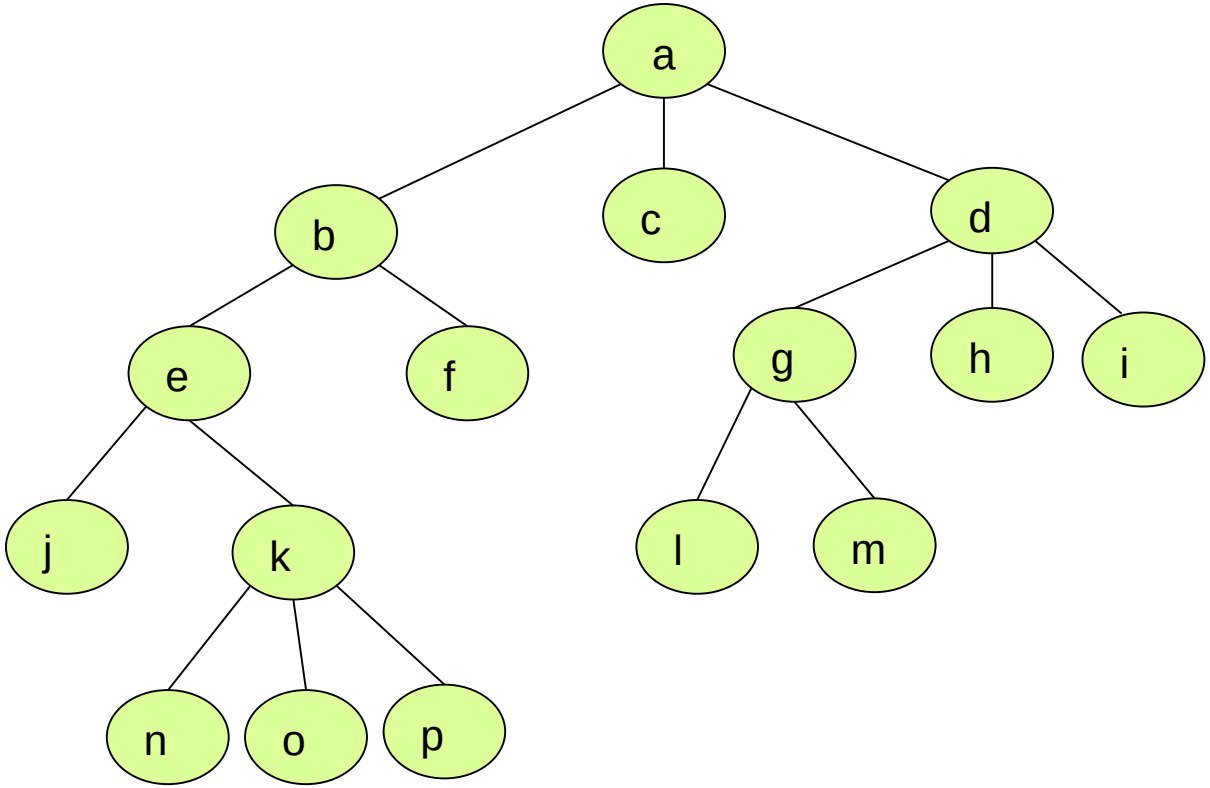
In-order T_1 r T_2 … T_n



j e n k o p b f a c l g m d h i



Post-order $T_1\ T_2 \ldots T_n\ r$



j n o p k e f b c l m g h i d a

Aritmetik İfadeler

- Standart: *infix* formu

  (A+B) ∗ C – D/ E

- Tüm parentezli form (in-order & parenthesis):

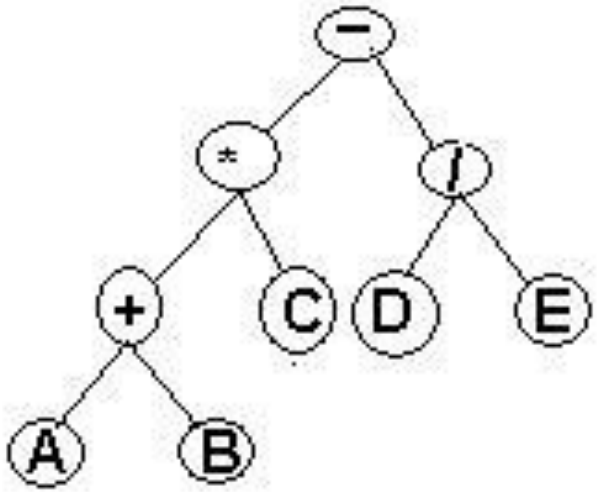
  (((A + B) ∗ C) – (D / E))

- *Postfix* formu (Polish notasyonunun tersi):

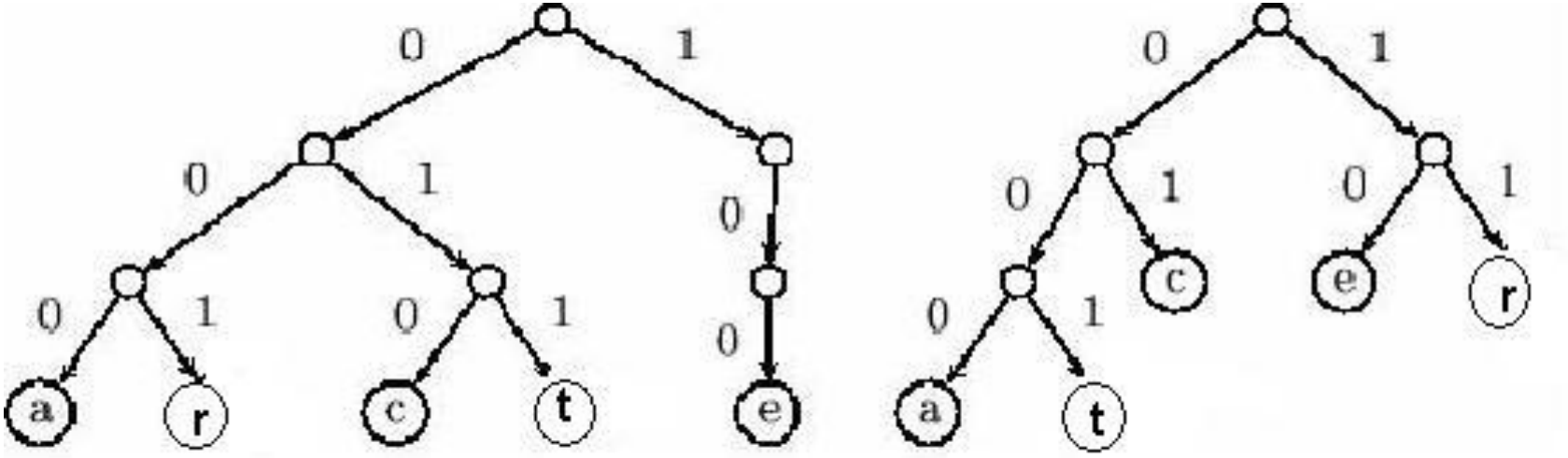
  A B + C ∗ D E / -

- *Prefix* formu (Polish notasyonu):

  - ∗ + A B C / D E



Huffman Kodları (Huffman Codes)



- Sol taraftaki ağacı kullanarak ***rate*** kelimesini kodlarsak

  001 000 011 100

- Sağ taraftaki ağacı kullanarak ***rate*** kelimesini kodlarsak

  11 000 001 10

HUFFMAN KODLAMA

Elimizde a, e, k, l, m ve z sembollerinden sırası ile 25, 14, 8, 7, 4 ve 2 adet olsun. Her bir sembol için Huffman kod değerini bulalım.

Adım 1: Sembollerin kullanım sıklıkları küçükten büyüğe doğru sıralanır.

2 4 7 8 14 25

Adım 2: Sıklık değerleri, soldan başlayarak sıradaki iki sıklık değerinin toplamlarının sıralamayı bozmayacak şekilde diziye yerleştirilmesi ile devam edilir.

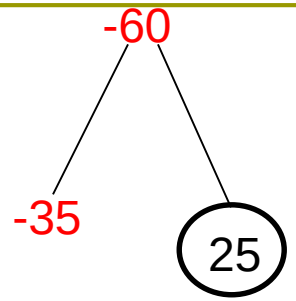
2 4 -6 7 8 -13 14 -21 25 -35 -60

Toplam değerleri kırmızı ile gösterilmiştir. Eksi değer bir ara toplam olduğunu söyler.

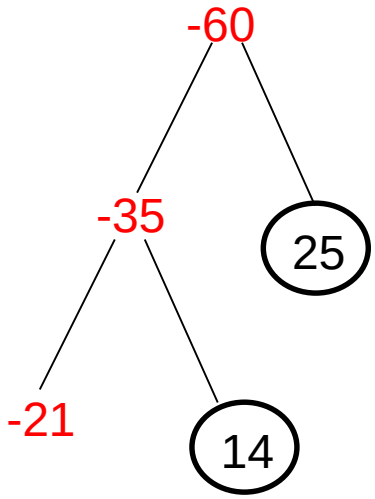
n tane sembolümüz varsa, genişletilmiş dizimiz **2n-1** adet olacaktır.

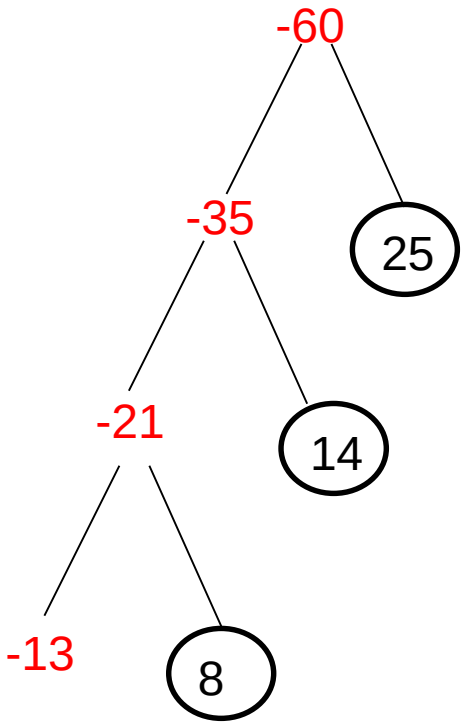
Adım 3: En sondaki eleman Huffman kodlama ağıcının kök değeri olacak şekilde ikili ağaç şeklinde diğer elemanlarda ağaca belli bir kural ile yerleştirilir.

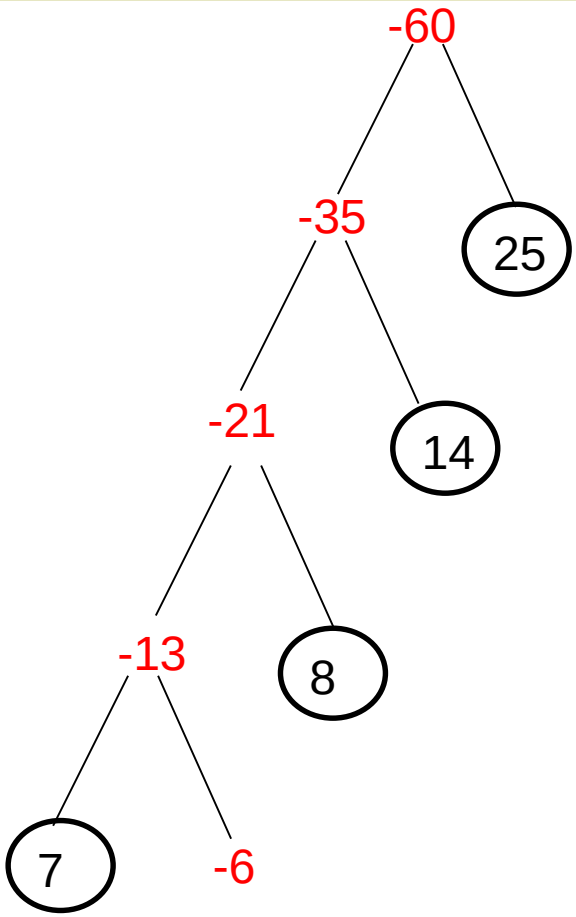
Bir sonraki (2n-2) değeri ağacın sol dalına, (2n-3) değeri de sağ dalına yerleşir.



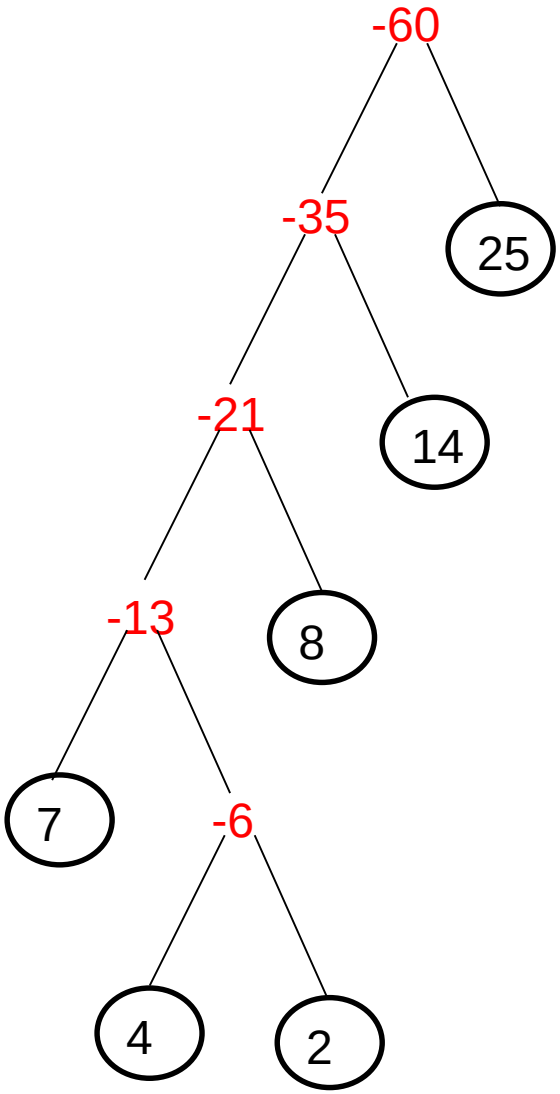
25 değeri "**a**" sembolün sıklık değeri olup, bir yapraktır.
-35 ise bir ara düğümdür. Ağaç bu düğüm üzerinden oluşturulmaya devam edilir.



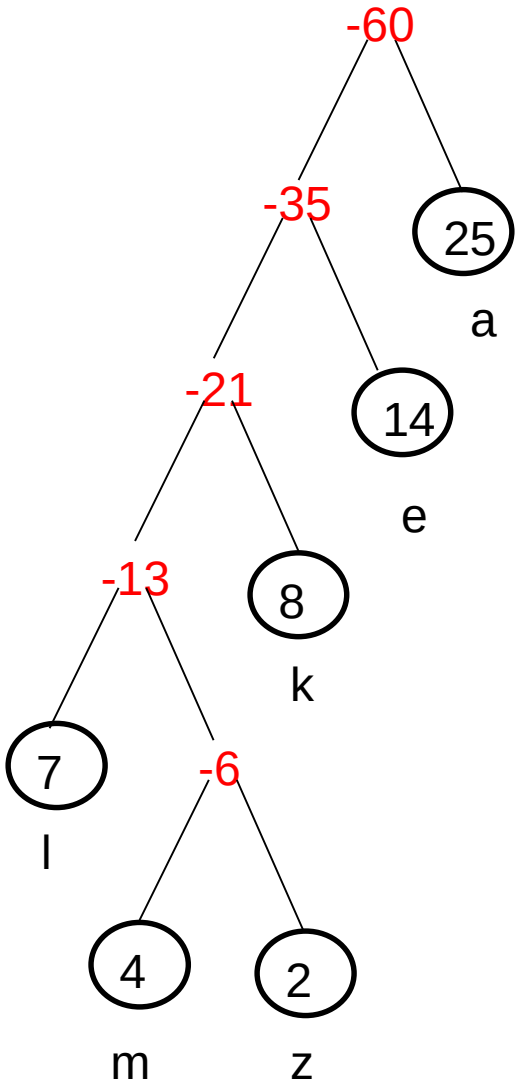

-60
-35
25
-21
14
-13
8


-60
-35
25
-21
14
-13
8
7
-6

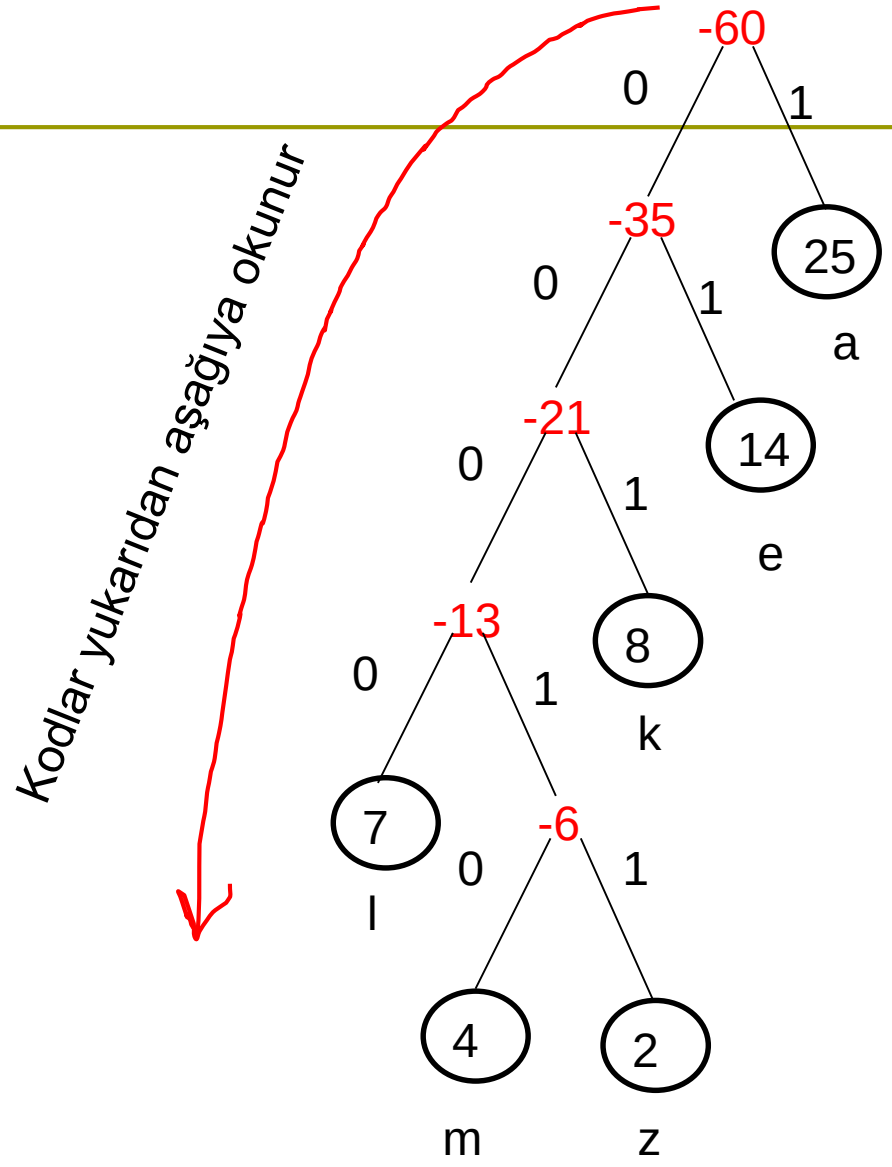
Huffman Kodlama Ağacının son durumu



Huffman Kodlama Ağacının yapraklarına sembolleri yerleştirelim.



Adım 4: Ağacın dallarına Huffman kodlarını oluşturacak olan 1 ve 0 değerlerini yerleştirelim. Sol dal için **0**, sağ dal için **1** değerini yazalım.



a→ 1
e→ 01
k→001
l→0000
m→00010
z→00011

Spanning Trees

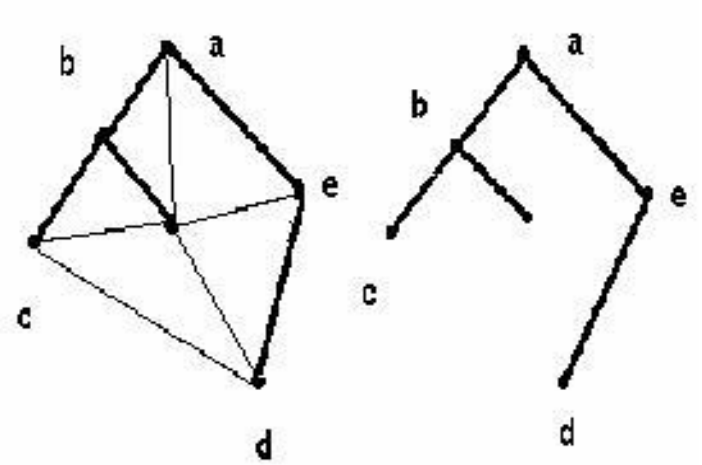
Verilen **G** grafında,***T*** bir *spanning tree* ise;

T,

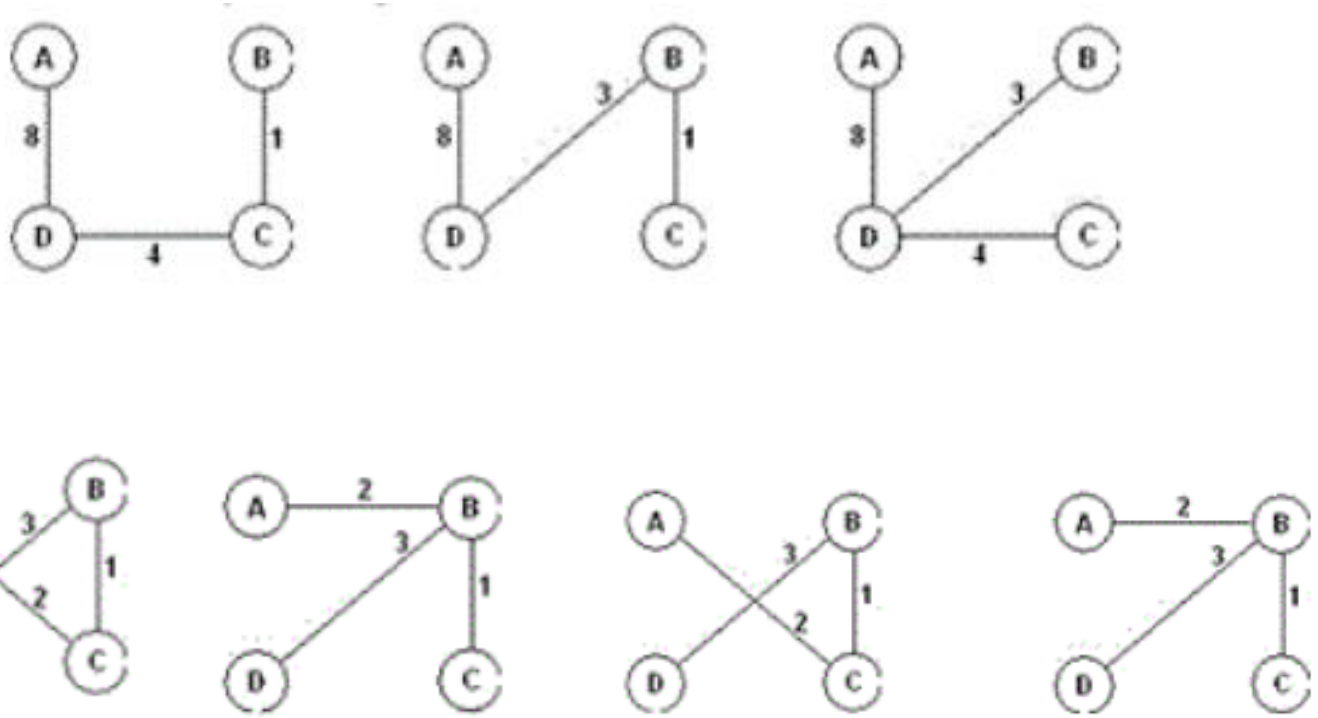
- **G** grafının bir *subtree*'sidir

ve

- **G** grafının bütün düğümlerini içerir

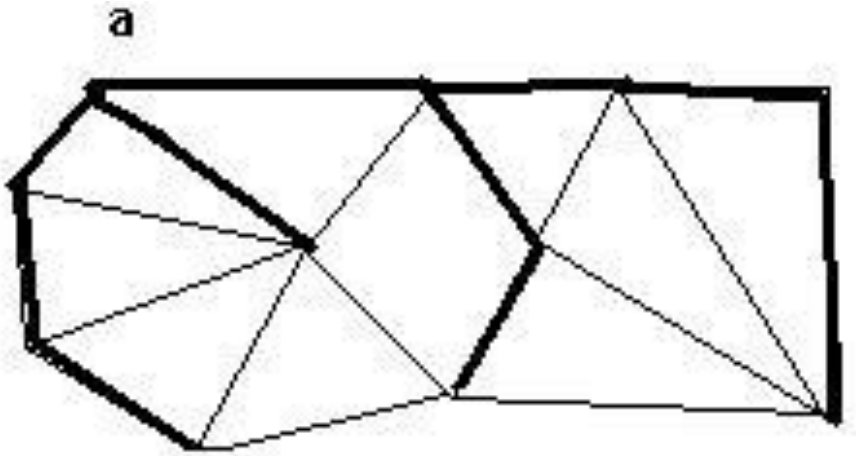


Farklı Spanning Trees

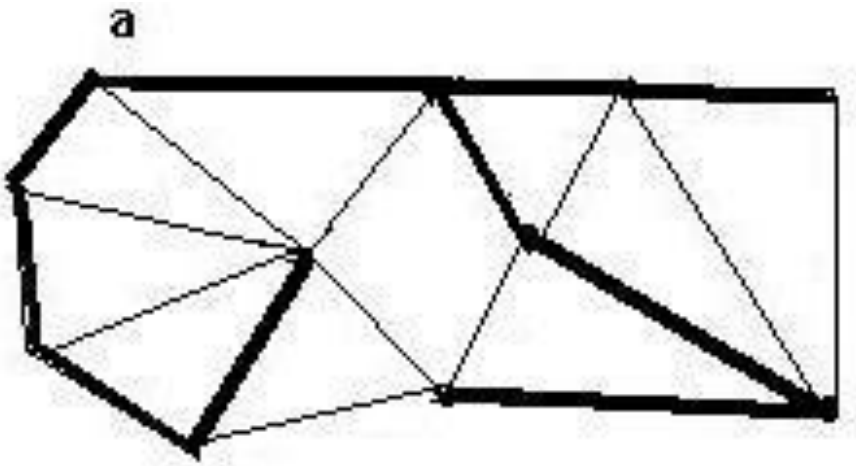


Spanning tree oluşturma yöntemleri

- Breadth-first search method



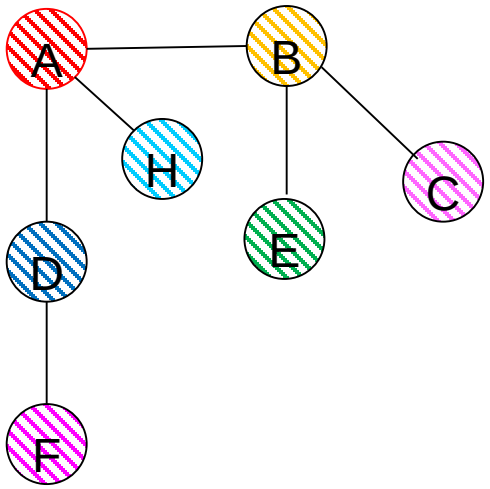
- Depth-first search method (backtracking)



DFS (Depth First Search – Derin Öncelikli Arama)

- Ziyareti kök düğümden başlatır ve gidebildiği en uzak düğüme kadar gider.
- Tüm düğümler ziyaret edilene kadar işlem tekrar edilir.
- Stack-yığın veri yapısını kullanır.

Start A

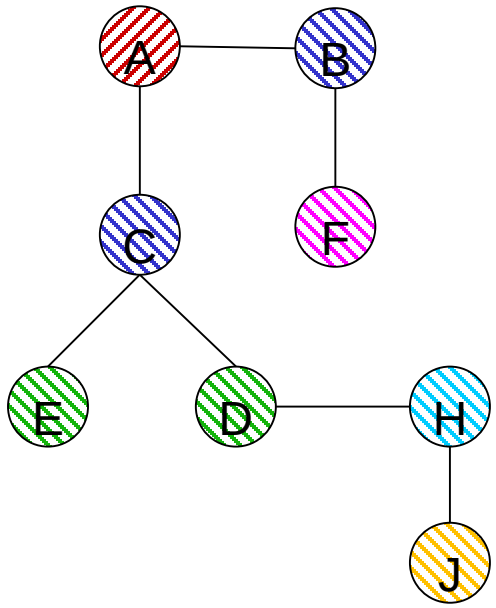


Output : A-D-F-H-B-E-C

Adım	İşlem	Stack	Çıkan	Output
1	**Push(A)**	**A**	-	**A**
2	**Push(D)**	**D-A**	-	**D**
3	**Push(F)**	**F-D-A**	-	**F**
4	Pop( )	D-A	F	-
5	Pop( )	A	D	-
6	**Push(H)**	**H-A**	-	**H**
7	**Pop( )**	**A**	**H**	-
8	**Push(B)**	**B-A**	-	**B**
9	**Push(E)**	**E-B-A**	-	**E**
10	Pop( )	B-A	E	-
11	**Push(C)**	**C-B-A**	-	**C**
12	Pop( )	B-A	C	-
13	**Pop( )**	**A**	**B**	-
14	Pop( )	-	A	-

BFS (Breadth First Search – Geniş Öncelikli Arama)

- Düğümleri seviye bazında ziyaret eder.
- Kök düğümden başlar ve düğümleri düzey düzey ziyaret eder.
- Bir sonraki seviyeye geçmeden önce bulunduğu seviyedeki tüm düğümleri ziyaret eder.
- Queue-kuyruk veri yapısını kullanır.



Start A

Adım	Eklenen	Çıkan	Kuyruk	Output
1	A	-	A	A
2	B-C	A	B-C	B-C
3	F	B	C-F	F
4	E-D	C	F-E-D	E-D
5	-	F	E-D	-
6	-	E	D	-
7	H	D	H	H
8	J	H	J	J
9	-	J	-	-

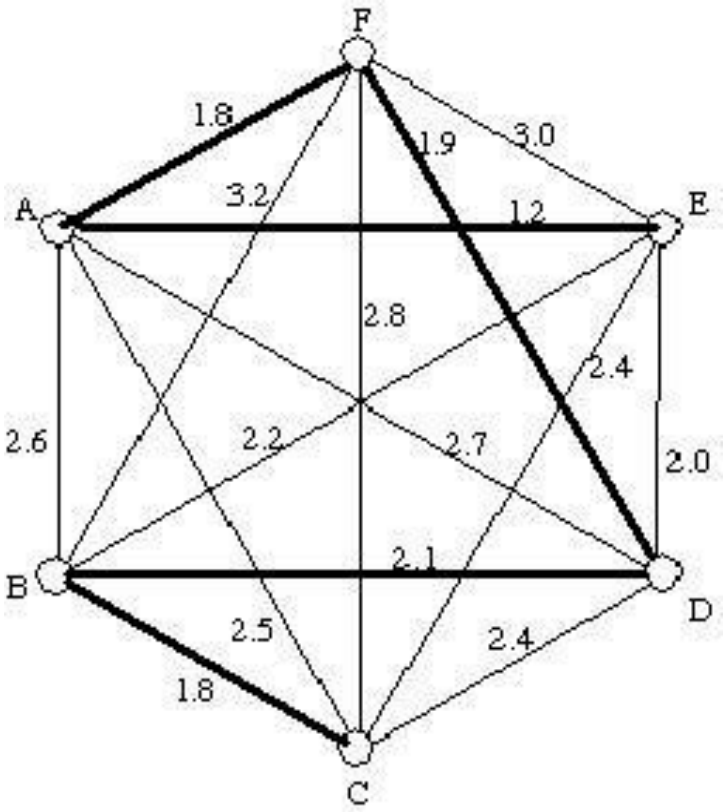
Output : A-B-C-F-E-D-H-J

Minimal spanning trees

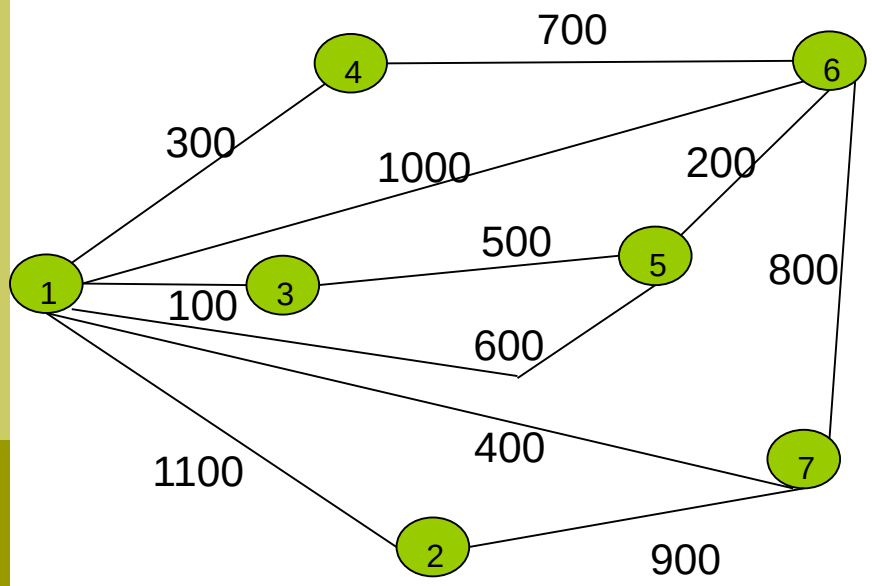
Verilen ağırlıklı bir **G** grafının

Minimal spanning tree'si

- **G** grafının spanning tree 'si olup (tüm düğümlerden geçilmiş)
- Ağırlıklar toplamıda <u>minimumdur</u>



Örnek : PRIM Algoritması



	1	2	3	4	5	6	7
1	0	1100	100	300	600	1000	400
2	1100	0	M	M	M	M	900
3	100	M	0	M	500	M	M
4	300	M	M	0	M	700	M
5	600	M	500	M	0	200	M
6	1000	M	M	700	200	0	800
7	400	900	M	M	M	800	0

	1	2	3	4	5	6	7
1	0	1100	100	300	600	1000	400
2	1100	0	M	M	M	M	900
3	100	M	0	M	500	M	M
4	300	M	M	0	M	700	M
5	600	M	500	M	0	200	M
6	1000	M	M	700	200	0	800
7	400	900	M	M	M	800	0

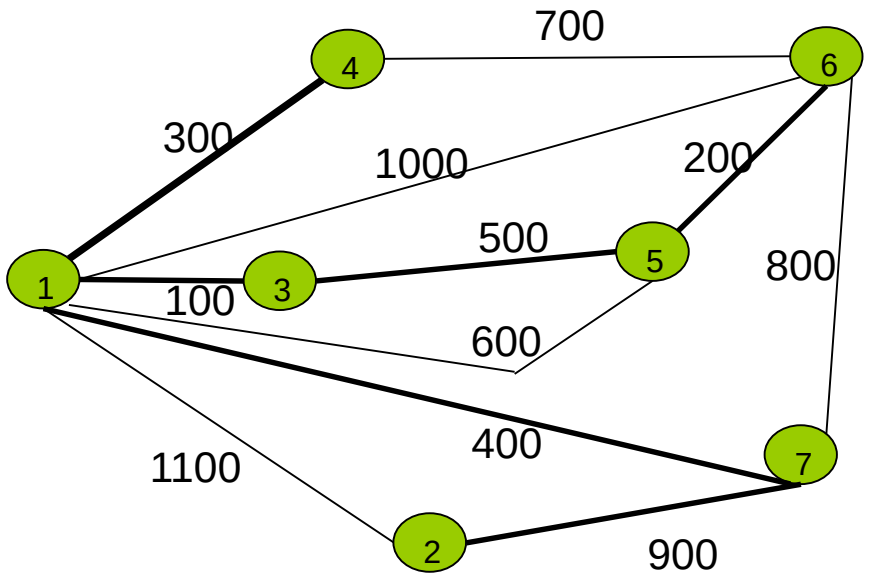
	1	2	3	4	5	6	7
1	0	1100	100	300	600	1000	400
2	1100	0	M	M	M	M	900
3	100	M	0	M	500	M	M
4	300	M	M	0	M	700	M
5	600	M	500	M	0	200	M
6	1000	M	M	700	200	0	800
7	400	900	M	M	M	800	0

	1	2	3	4	5	6	7
1	0	1100	100	300	600	1000	400
2	1100	0	M	M	M	M	900
3	100	M	0	M	500	M	M
4	300	M	M	0	M	700	M
5	600	M	500	M	0	200	M
6	1000	M	M	700	200	0	800
7	400	900	M	M	M	800	0

	1	2	3	4	5	6	7
1	0	1100	100	300	600	1000	400
2	1100	0	M	M	M	M	900
3	100	M	0	M	500	M	M
4	300	M	M	0	M	700	M
5	600	M	500	M	0	200	M
6	1000	M	M	700	200	0	800
7	400	900	M	M	M	800	0

	1	2	3	4	5	6	7
1	0	1100	100	300	600	1000	400
2	1100	0	M	M	M	M	900
3	100	M	0	M	500	M	M
4	300	M	M	0	M	700	M
5	600	M	500	M	0	200	M
6	1000	M	M	700	200	0	800
7	400	900	M	M	M	800	0

	1	2	3	4	5	6	7
1	0	1100	100	300	600	1000	400
2	1100	0	M	M	M	M	900
3	100	M	0	M	500	M	M
4	300	M	M	0	M	700	M
5	600	M	500	M	0	200	M
6	1000	M	M	700	200	0	800
7	400	900	M	M	M	800	0



Prim's Algorithm Demo - Microsoft Internet Explorer

File Edit View Favorites Tools Help

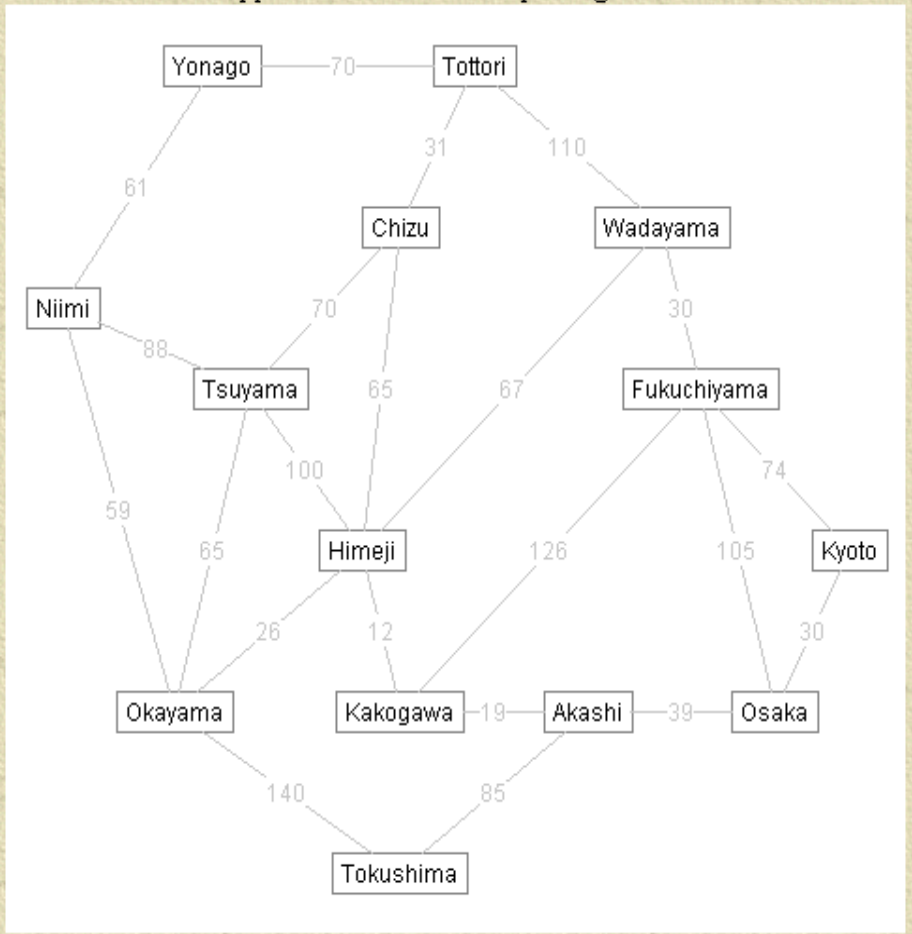
Back Search Favorites Media

Address http://www.unf.edu/~wkloster/foundations/PrimApplet/PrimApplet.htm Go Links

Google prim algorithm demo Search 26 blocked Check AutoLink AutoFill Options prim algorithm demo

Java Applet Demo of Prim's Algorithm

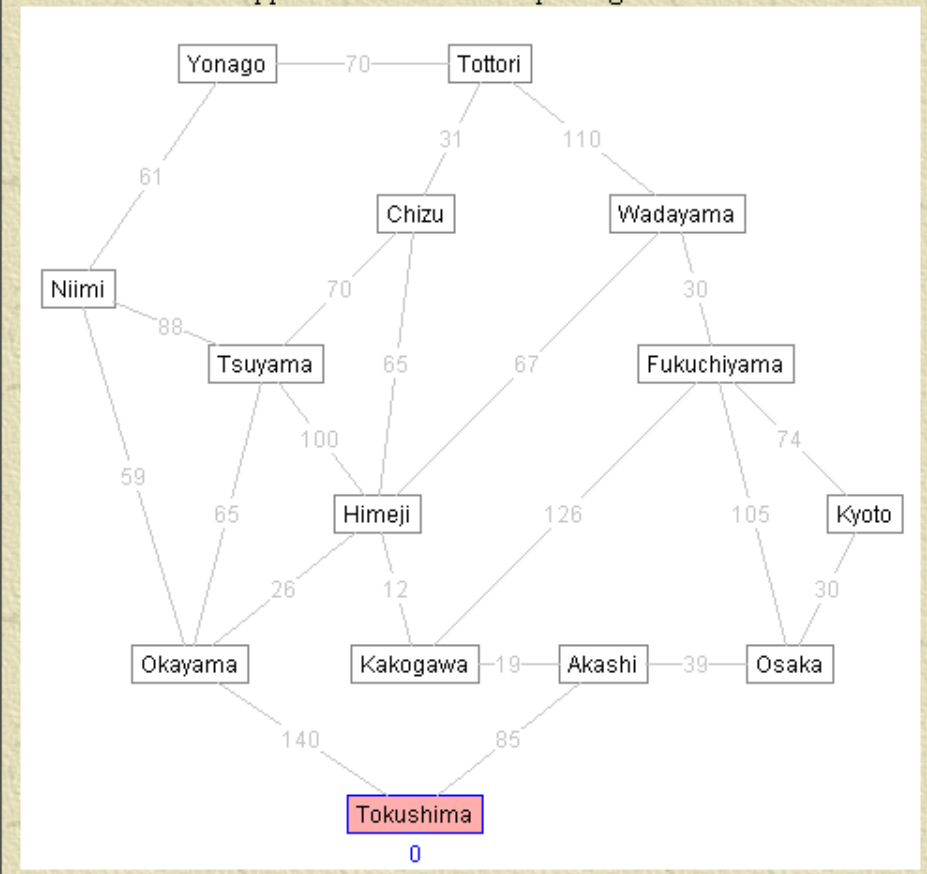
Click on the below applet to find a minimum spanning tree.



Applet written by: Kenji Ikeda

Java Applet Demo of Prim's Algorithm

Click on the below applet to find a minimum spanning tree.



Applet written by: Kenji Ikeda

Prim's Algorithm Demo - Microsoft Internet Explorer

File Edit View Favorites Tools Help

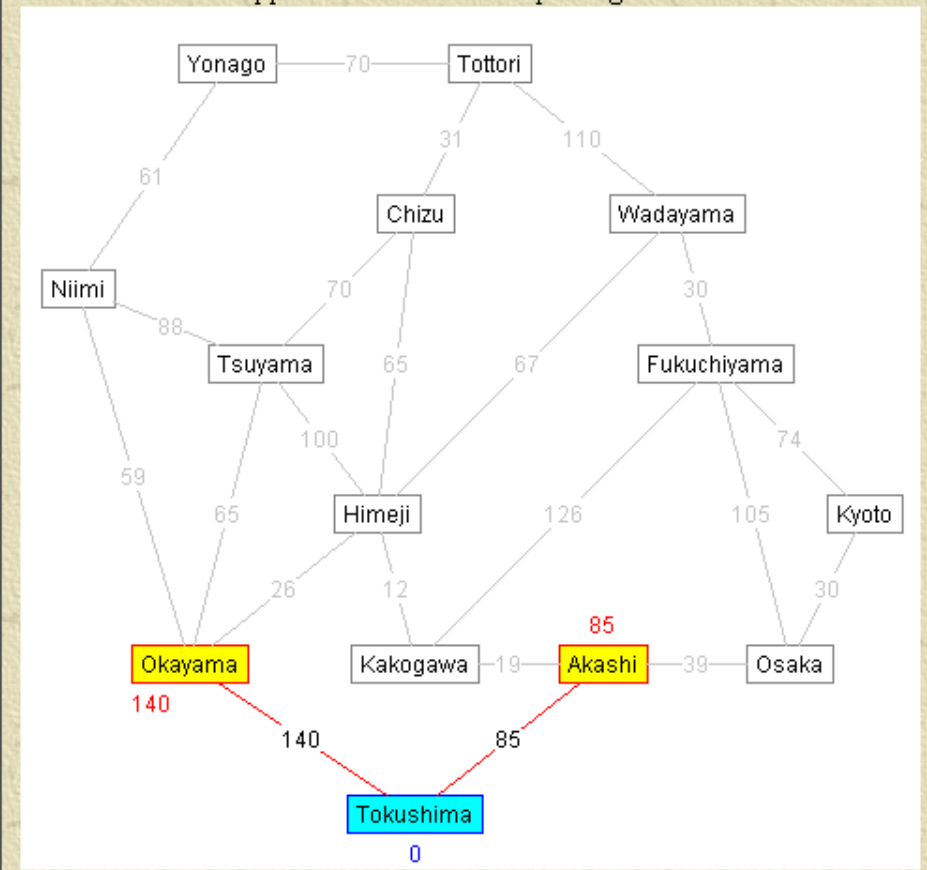
Back Search Favorites Media

Address http://www.unf.edu/~wkloster/foundations/PrimApplet/PrimApplet.htm Go Links

Google prim algorithm demo Search 26 blocked Check AutoLink AutoFill Options prim algorithm demo

Java Applet Demo of Prim's Algorithm

Click on the below applet to find a minimum spanning tree.



Applet written by: Kenji Ikeda

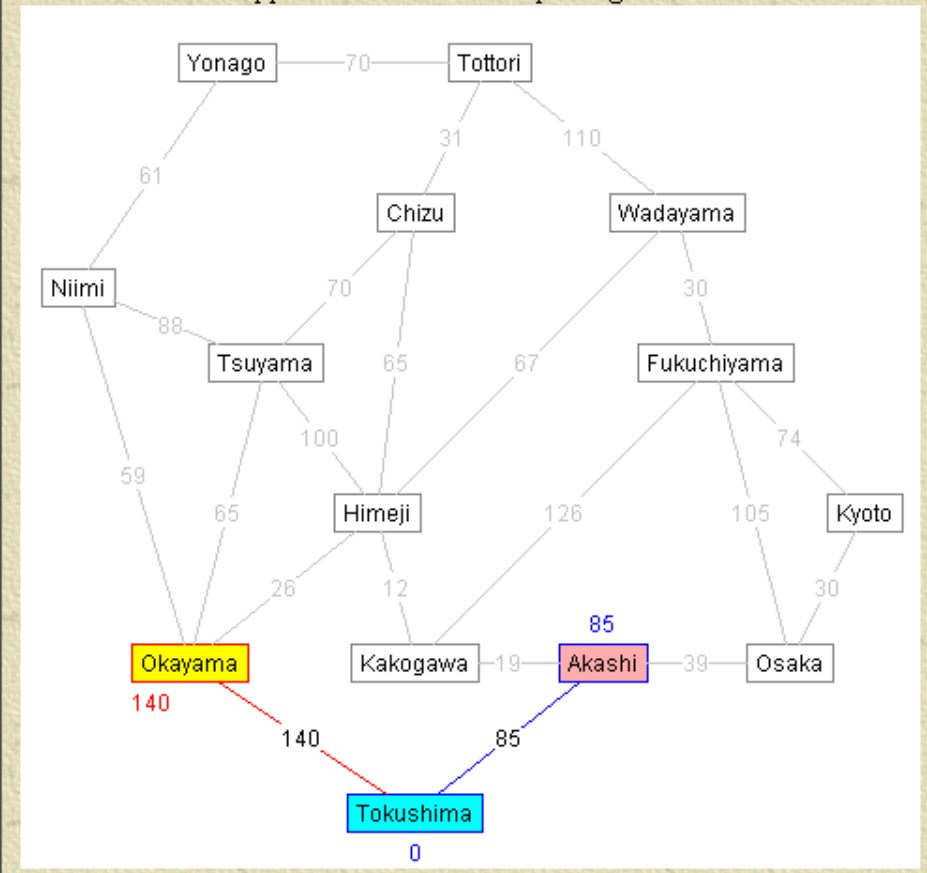
Applet Prim started

Internet

Start the shortest ... Prim's Algor... HELP JUNG Demo: ... Microsoft Pow... 20:34

Java Applet Demo of Prim's Algorithm

Click on the below applet to find a minimum spanning tree.



Applet written by: Kenji Ikeda

Prim's Algorithm Demo - Microsoft Internet Explorer

File Edit View Favorites Tools Help

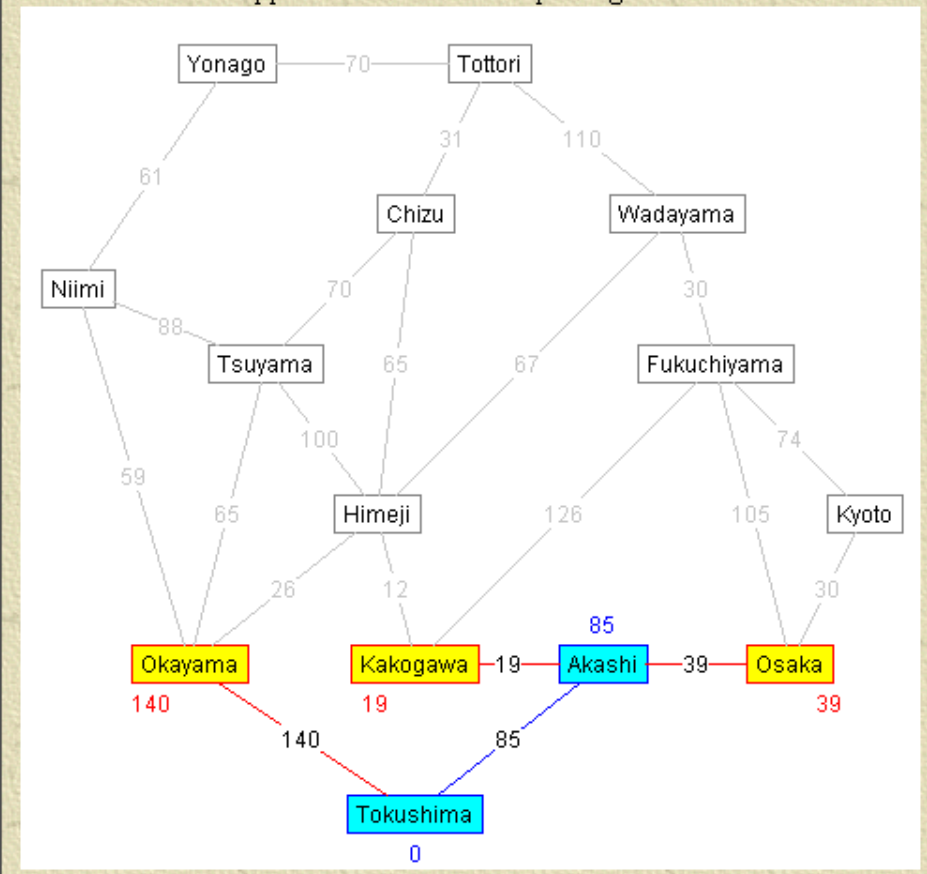
Back Search Favorites Media

Address http://www.unf.edu/~wkloster/foundations/PrimApplet/PrimApplet.htm Go Links

Google prim algorithm demo Search 26 blocked Check AutoLink AutoFill Options prim algorithm demo

Java Applet Demo of Prim's Algorithm

Click on the below applet to find a minimum spanning tree.



Applet written by: Kenji Ikeda

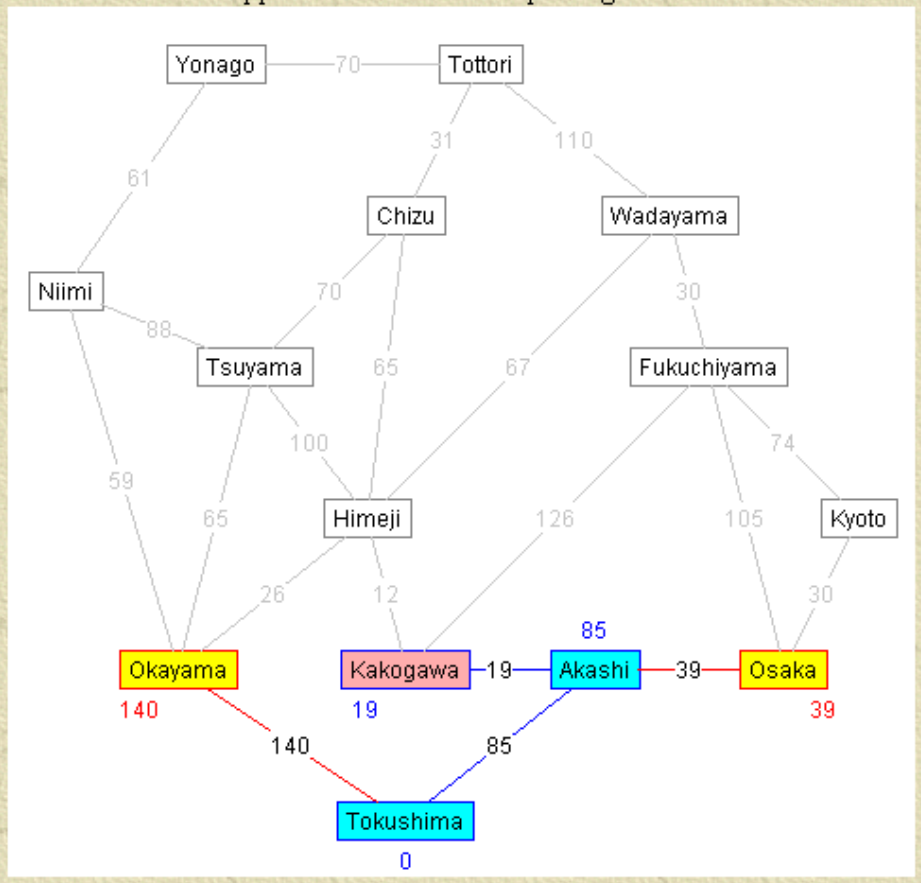
Applet Prim started

Internet

Start the shortest ... Prim's Algor... HELP JUNG Demo: ... Microsoft Pow... 20:35

Java Applet Demo of Prim's Algorithm

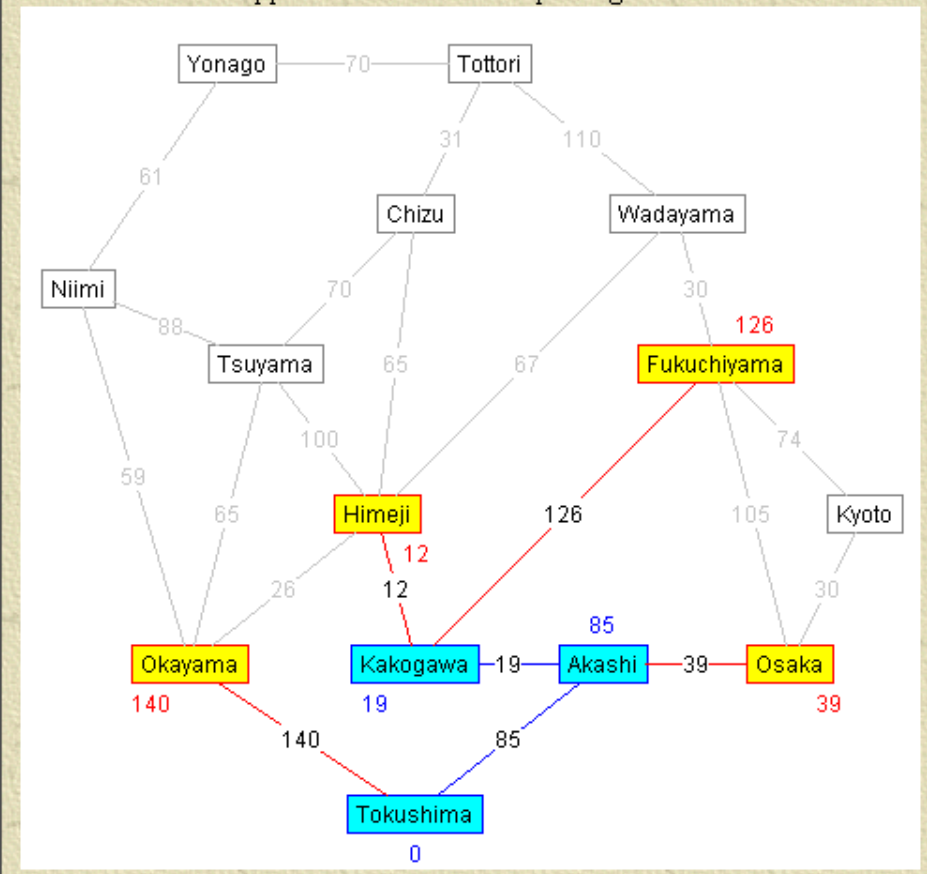
Click on the below applet to find a minimum spanning tree.



Applet written by: Kenji Ikeda

Java Applet Demo of Prim's Algorithm

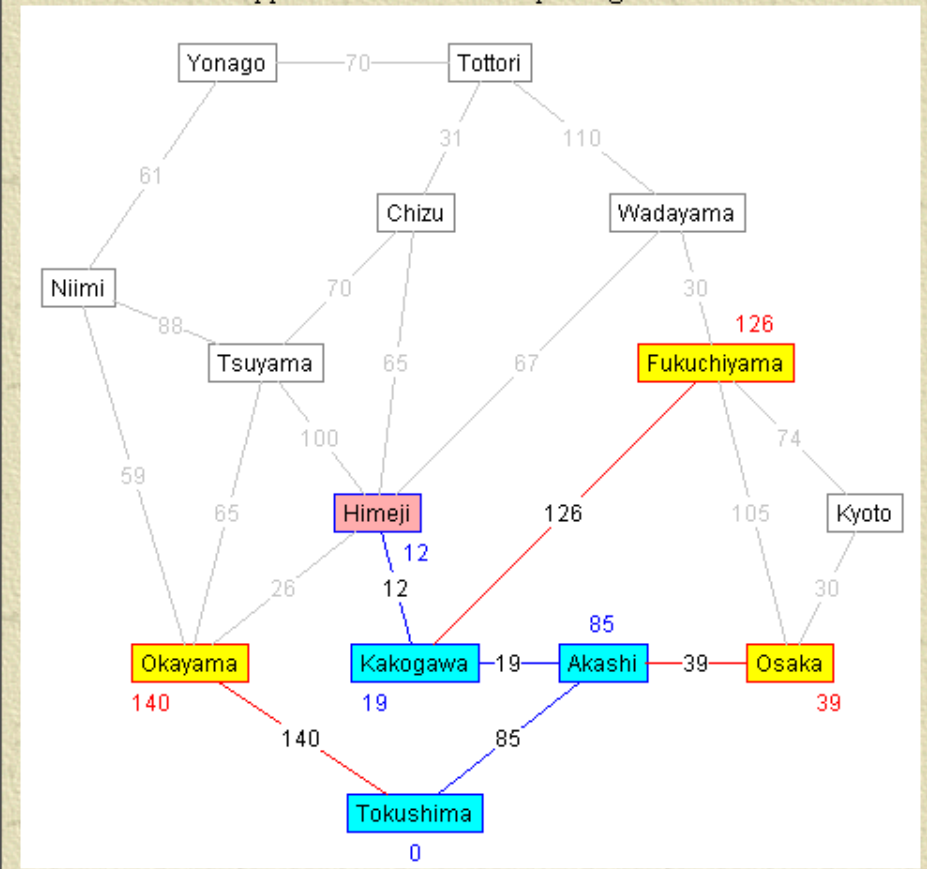
Click on the below applet to find a minimum spanning tree.



Applet written by: Kenji Ikeda

Java Applet Demo of Prim's Algorithm

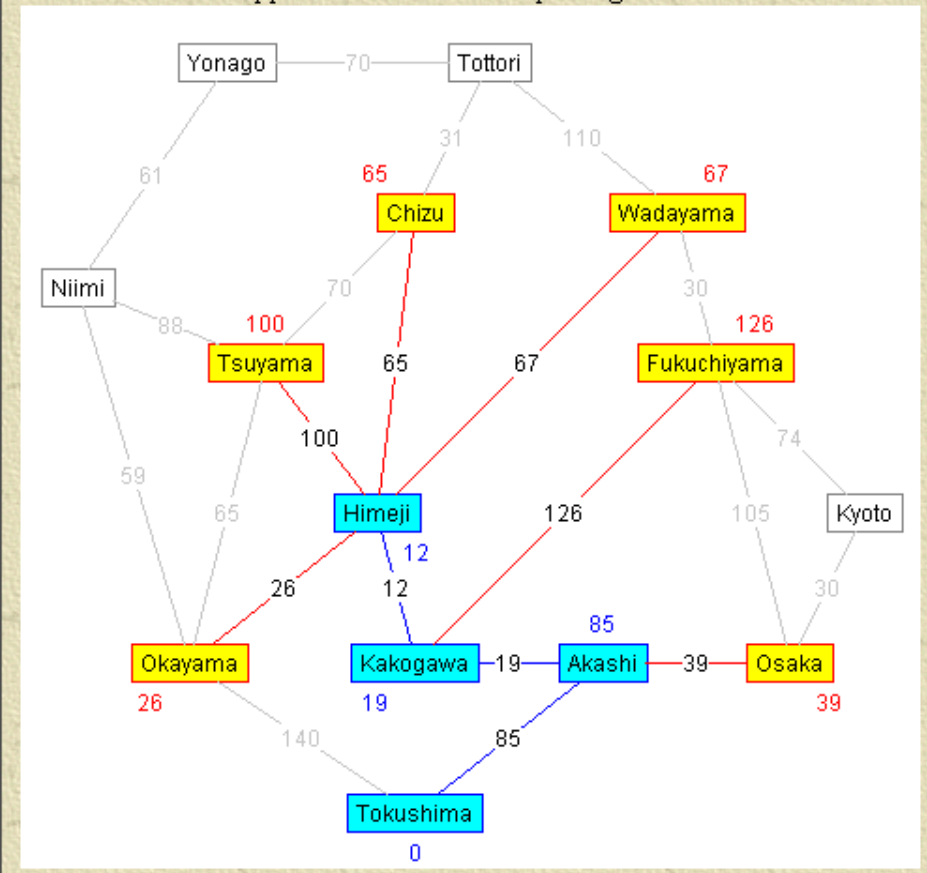
Click on the below applet to find a minimum spanning tree.



Applet written by: Kenji Ikeda

Java Applet Demo of Prim's Algorithm

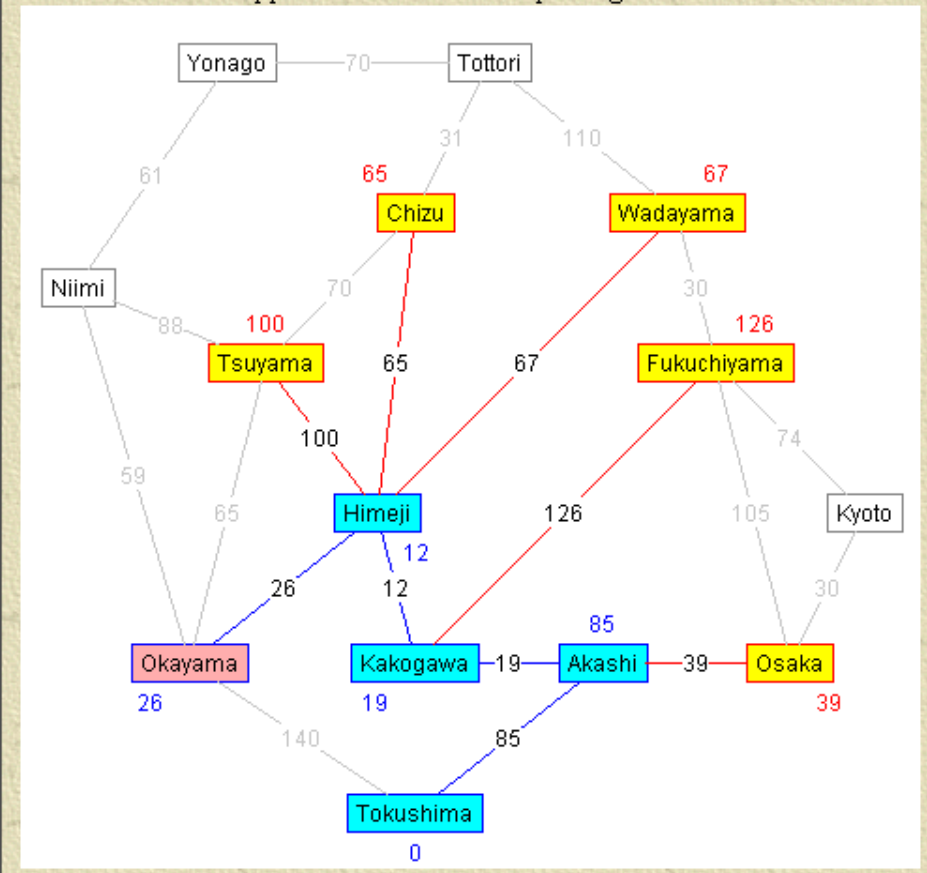
Click on the below applet to find a minimum spanning tree.



Applet written by: Kenji Ikeda

Java Applet Demo of Prim's Algorithm

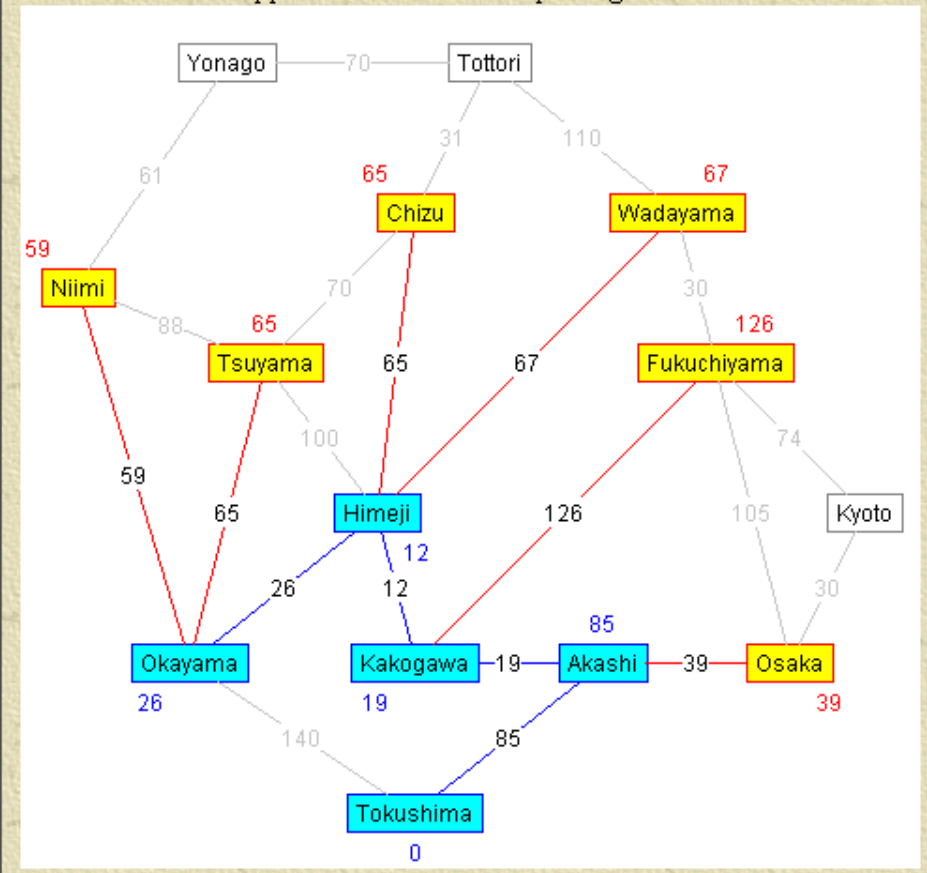
Click on the below applet to find a minimum spanning tree.



Applet written by: Kenji Ikeda

Java Applet Demo of Prim's Algorithm

Click on the below applet to find a minimum spanning tree.



Applet written by: Kenji Ikeda

Prim's Algorithm Demo - Microsoft Internet Explorer

File Edit View Favorites Tools Help

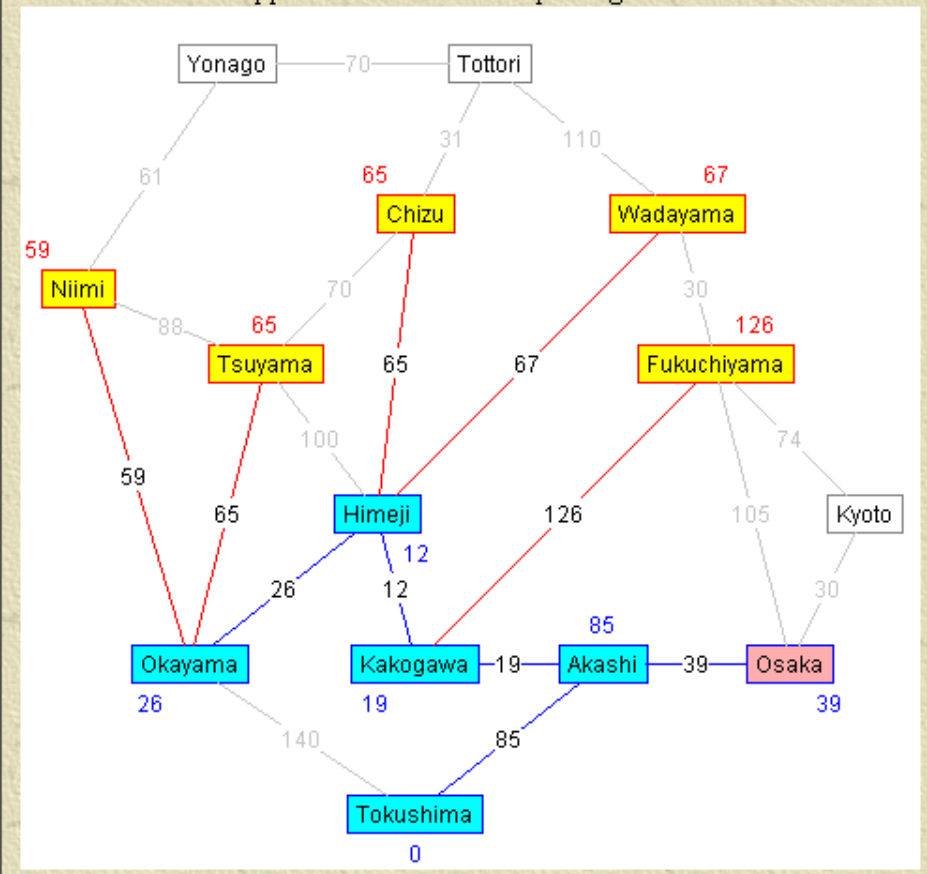
Back Search Favorites Media

Address http://www.unf.edu/~wkloster/foundations/PrimApplet/PrimApplet.htm Go Links

Google prim algorithm demo Search 26 blocked Check AutoLink AutoFill Options prim algorithm demo

Java Applet Demo of Prim's Algorithm

Click on the below applet to find a minimum spanning tree.



Applet written by: Kenji Ikeda

Applet Prim started

Internet

Start the shortest ... Prim's Algor... HELP JUNG Demo: ... Microsoft Pow... 20:38

Prim's Algorithm Demo - Microsoft Internet Explorer

File Edit View Favorites Tools Help

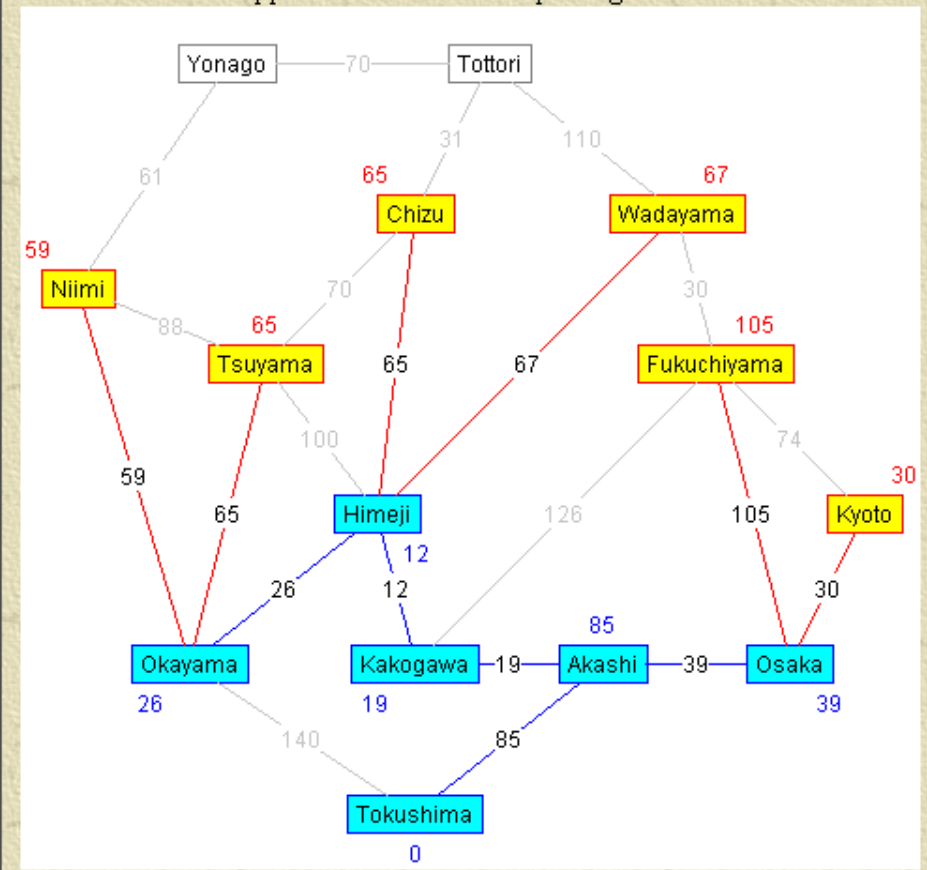
Back Search Favorites Media

Address http://www.unf.edu/~wkloster/foundations/PrimApplet/PrimApplet.htm Go Links

Google prim algorithm demo Search 26 blocked Check AutoLink AutoFill Options prim algorithm demo

Java Applet Demo of Prim's Algorithm

Click on the below applet to find a minimum spanning tree.



Applet written by: Kenji Ikeda

Applet Prim started

Internet

Start the shortest ... Prim's Algor... HELP JUNG Demo: ... Microsoft Pow... 20:38

Prim's Algorithm Demo - Microsoft Internet Explorer

File Edit View Favorites Tools Help

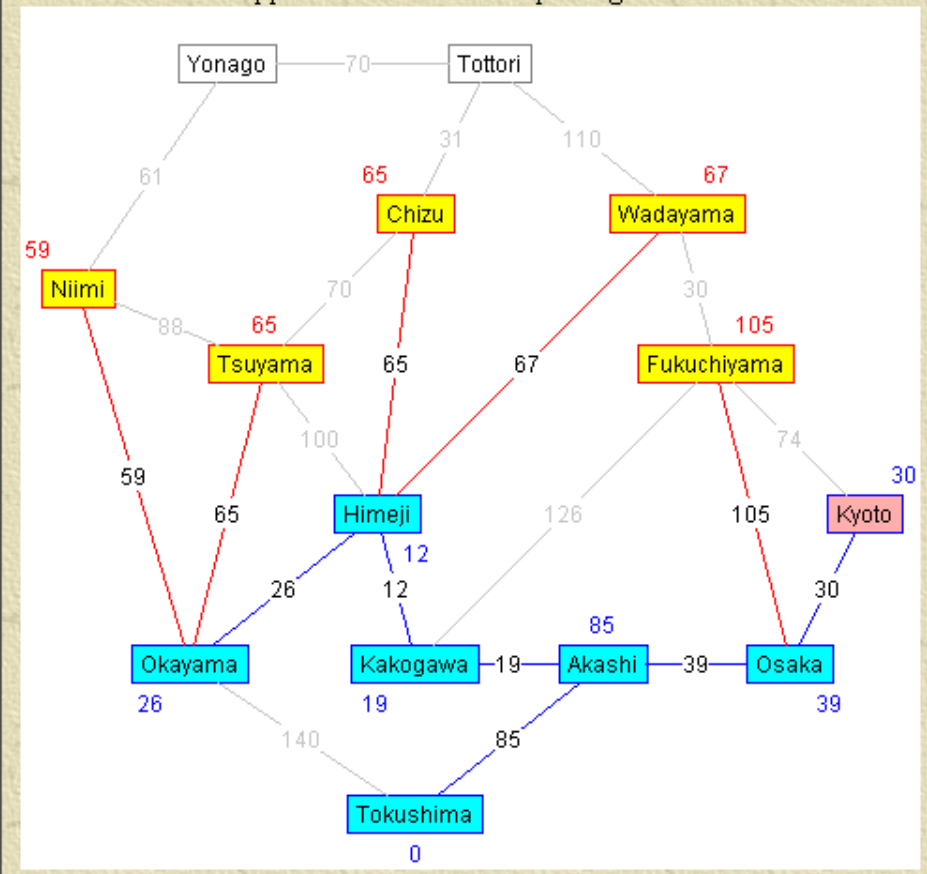
Back Search Favorites Media

Address http://www.unf.edu/~wkloster/foundations/PrimApplet/PrimApplet.htm Go Links

Google prim algorithm demo Search 26 blocked Check AutoLink AutoFill Options prim algorithm demo

Java Applet Demo of Prim's Algorithm

Click on the below applet to find a minimum spanning tree.



Applet written by: Kenji Ikeda

Applet Prim started

Internet

Start the shortest ... Prim's Algor... HELP JUNG Demo: ... Microsoft Pow... 20:38

Prim's Algorithm Demo - Microsoft Internet Explorer

File Edit View Favorites Tools Help

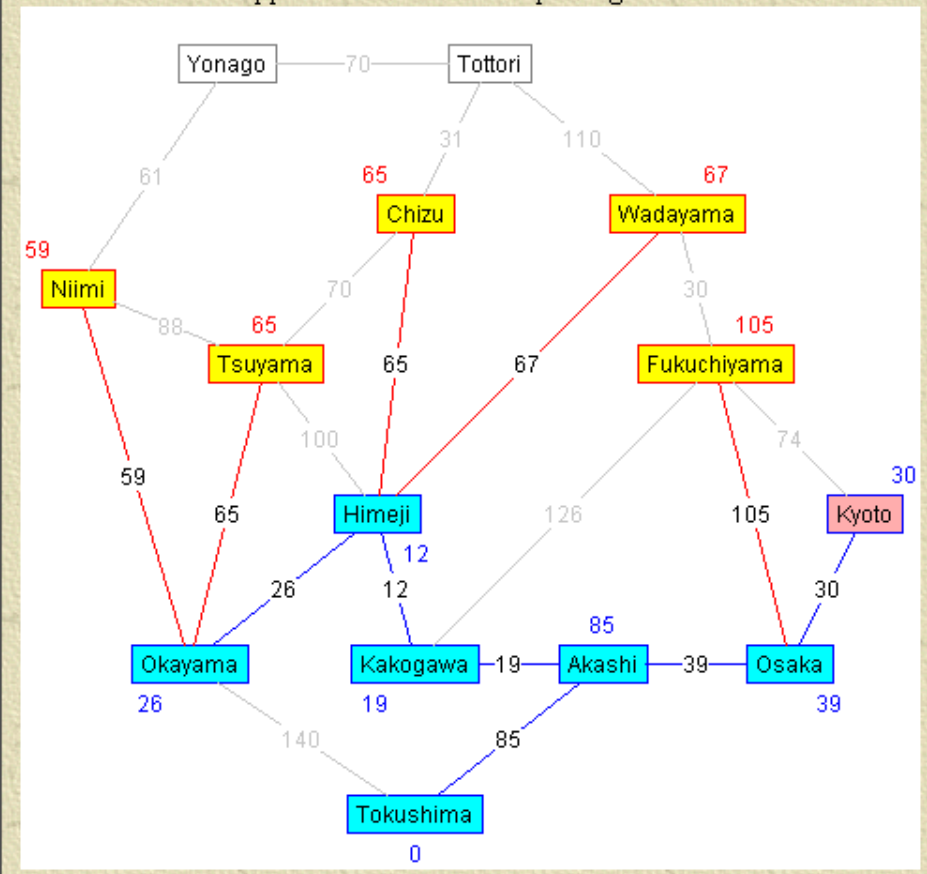
Back Search Favorites Media

Address http://www.unf.edu/~wkloster/foundations/PrimApplet/PrimApplet.htm Go Links

Google prim algorithm demo Search 26 blocked Check AutoLink AutoFill Options prim algorithm demo

Java Applet Demo of Prim's Algorithm

Click on the below applet to find a minimum spanning tree.



Applet written by: Kenji Ikeda

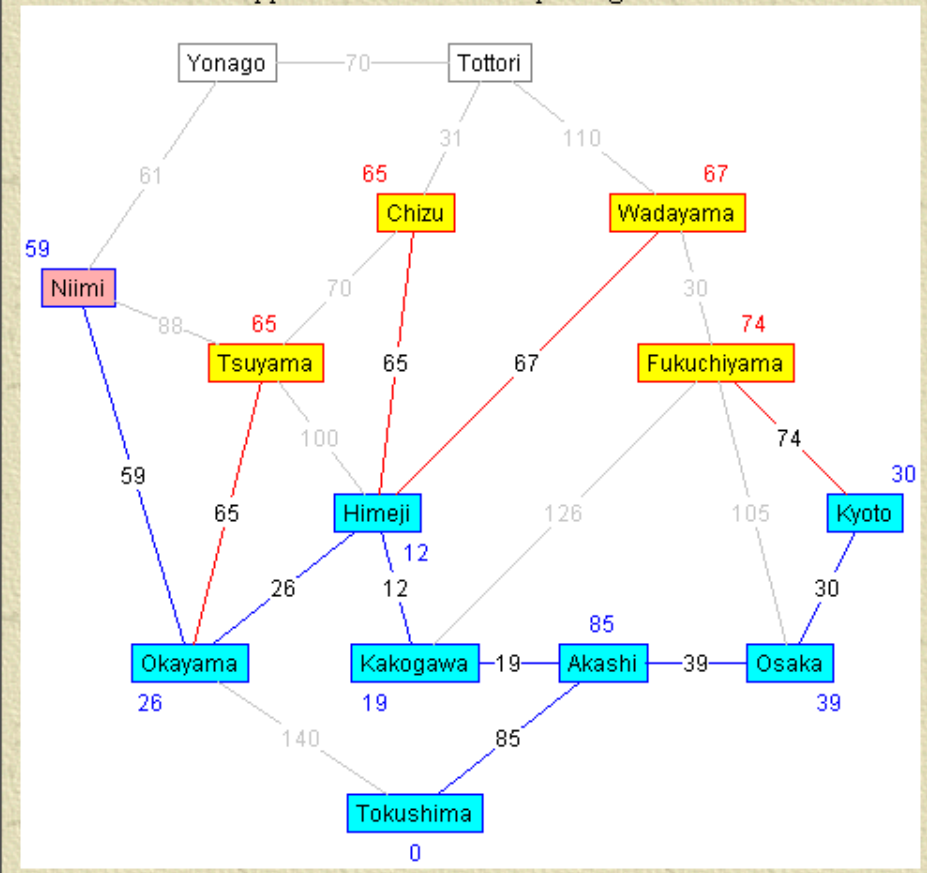
Applet Prim started

Internet

Start the shortest ... Prim's Algor... HELP JUNG Demo: ... Microsoft Pow... 20:38

Java Applet Demo of Prim's Algorithm

Click on the below applet to find a minimum spanning tree.



Applet written by: Kenji Ikeda

Prim's Algorithm Demo - Microsoft Internet Explorer

File Edit View Favorites Tools Help

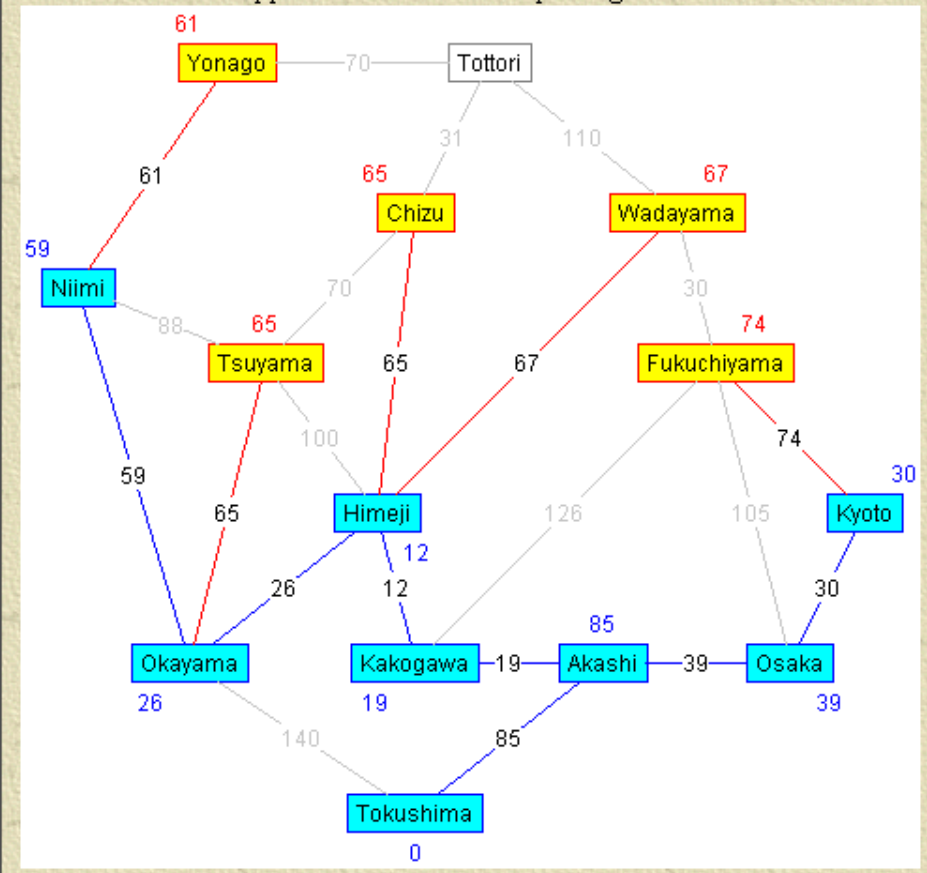
Back Search Favorites Media

Address http://www.unf.edu/~wkloster/foundations/PrimApplet/PrimApplet.htm Go Links

Google prim algorithm demo Search 26 blocked Check AutoLink AutoFill Options prim algorithm demo

Java Applet Demo of Prim's Algorithm

Click on the below applet to find a minimum spanning tree.



Applet written by: Kenji Ikeda

Applet Prim started

Internet

Start the shortest ... Prim's Algor... HELP JUNG Demo: ... Microsoft Pow... 20:39

Prim's Algorithm Demo - Microsoft Internet Explorer

File Edit View Favorites Tools Help

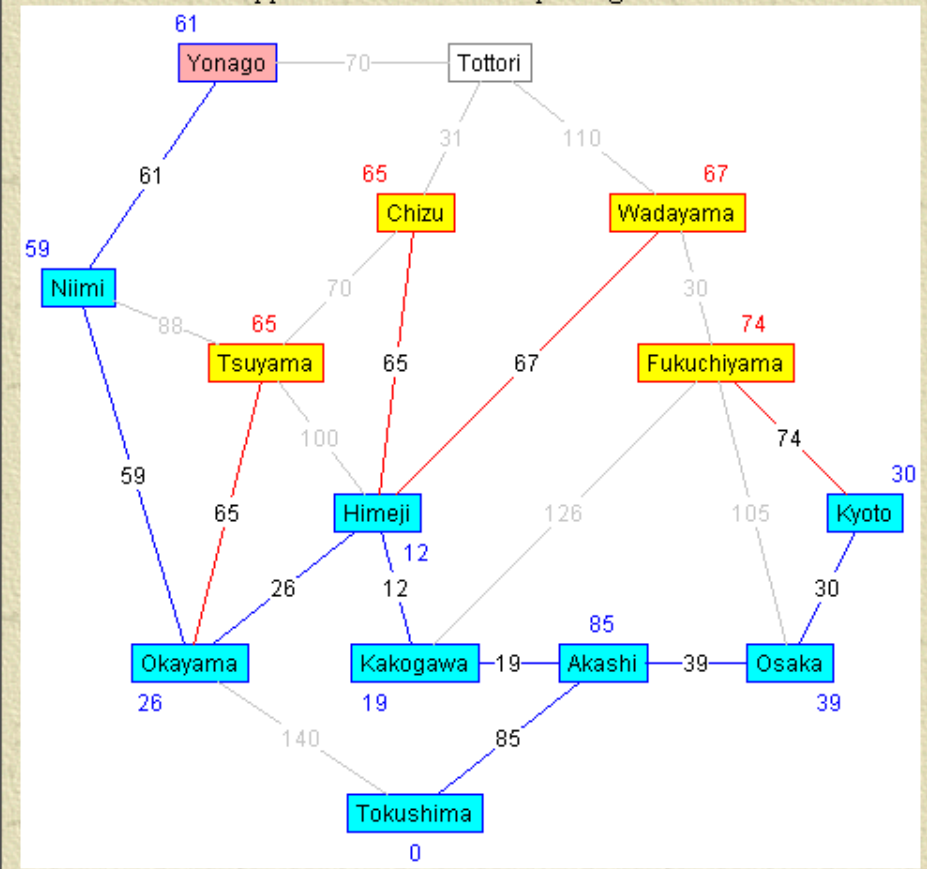
Back Search Favorites Media

Address http://www.unf.edu/~wkloster/foundations/PrimApplet/PrimApplet.htm Go Links

Google prim algorithm demo Search 26 blocked Check AutoLink AutoFill Options prim algorithm demo

Java Applet Demo of Prim's Algorithm

Click on the below applet to find a minimum spanning tree.



Applet written by: Kenji Ikeda

Applet Prim started

Internet

Start the shortest ... Prim's Algor... HELP JUNG Demo: ... Microsoft Pow... 20:40

Prim's Algorithm Demo - Microsoft Internet Explorer

File Edit View Favorites Tools Help

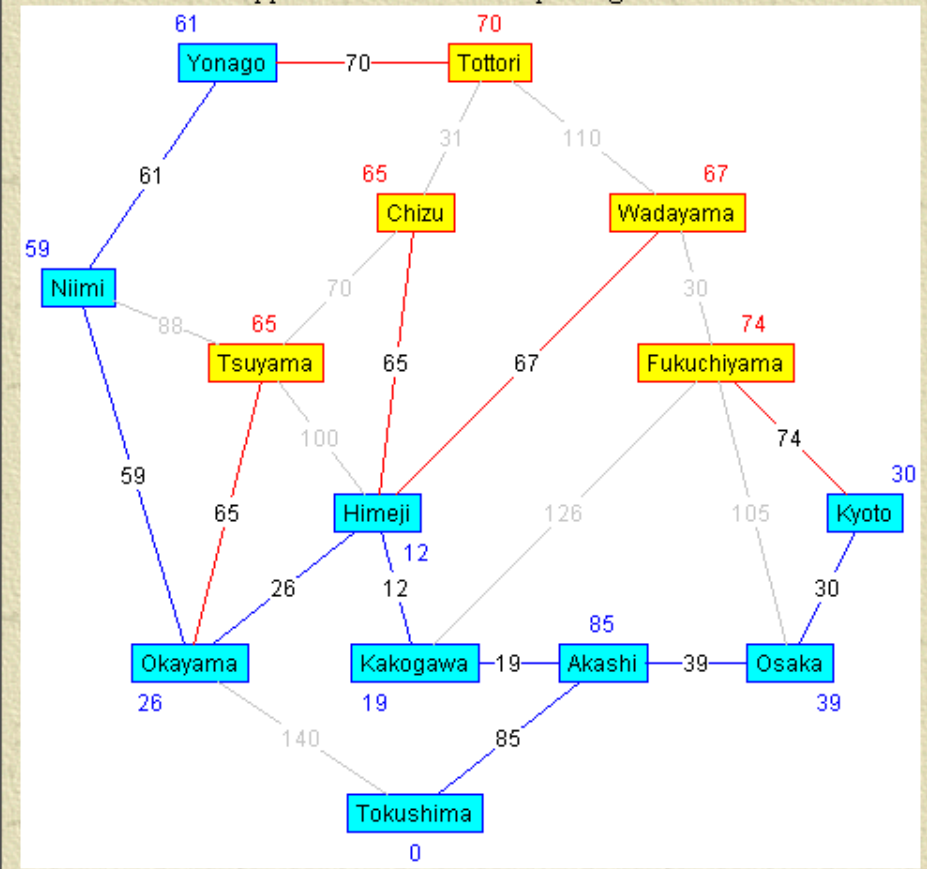
Back Search Favorites Media

Address http://www.unf.edu/~wkloster/foundations/PrimApplet/PrimApplet.htm Go Links

Google prim algorithm demo Search 26 blocked Check AutoLink AutoFill Options prim algorithm demo

Java Applet Demo of Prim's Algorithm

Click on the below applet to find a minimum spanning tree.



Applet written by: Kenji Ikeda

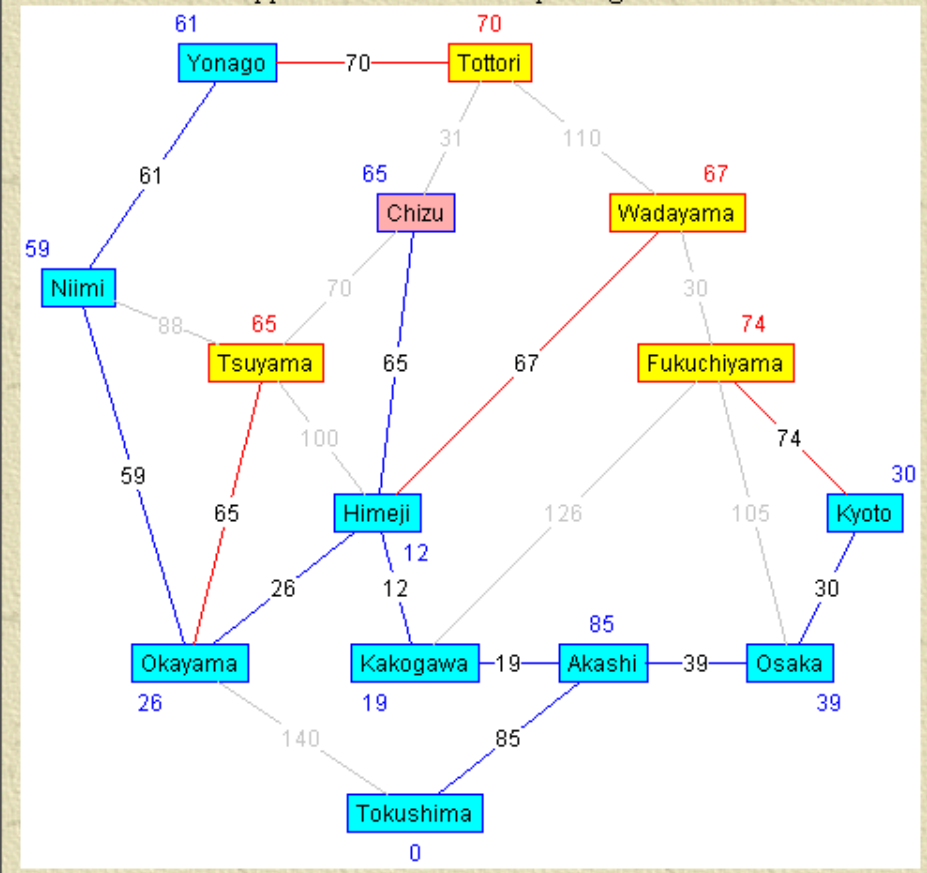
Applet Prim started

Internet

Start the shortest ... Prim's Algor... HELP JUNG Demo: ... Microsoft Pow... 20:40

Java Applet Demo of Prim's Algorithm

Click on the below applet to find a minimum spanning tree.



Applet written by: Kenji Ikeda

Prim's Algorithm Demo - Microsoft Internet Explorer

File Edit View Favorites Tools Help

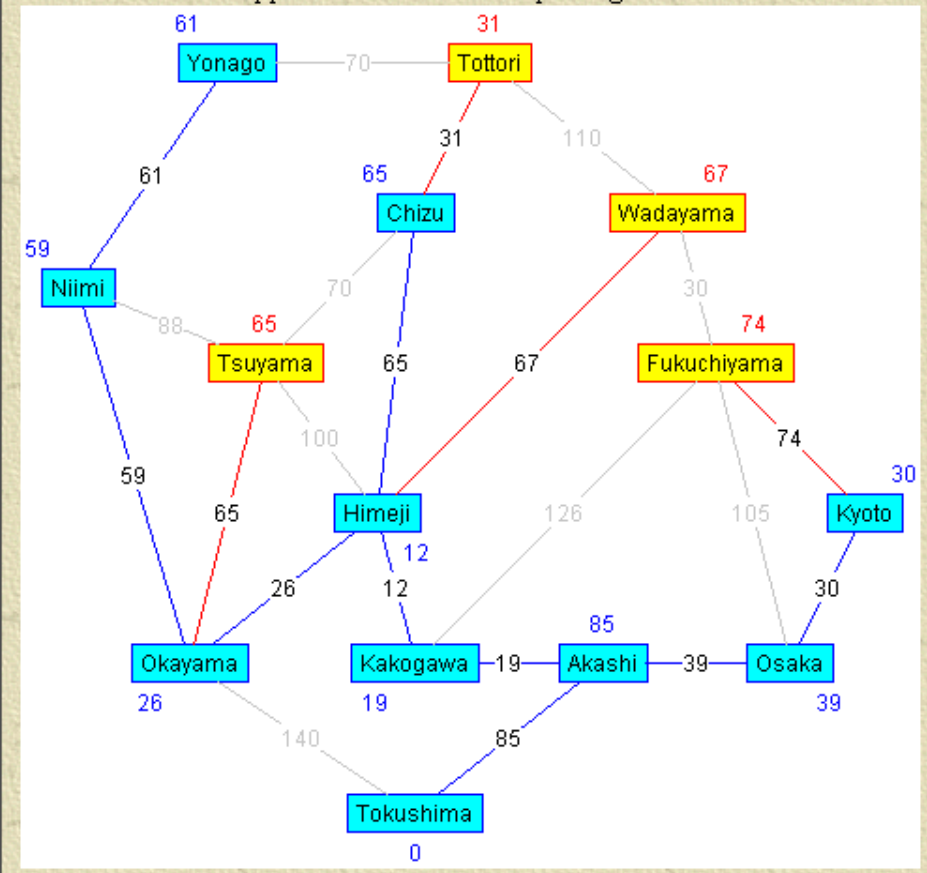
Back Search Favorites Media

Address http://www.unf.edu/~wkloster/foundations/PrimApplet/PrimApplet.htm Go Links

Google prim algorithm demo Search 26 blocked Check AutoLink AutoFill Options prim algorithm demo

Java Applet Demo of Prim's Algorithm

Click on the below applet to find a minimum spanning tree.



Applet written by: Kenji Ikeda

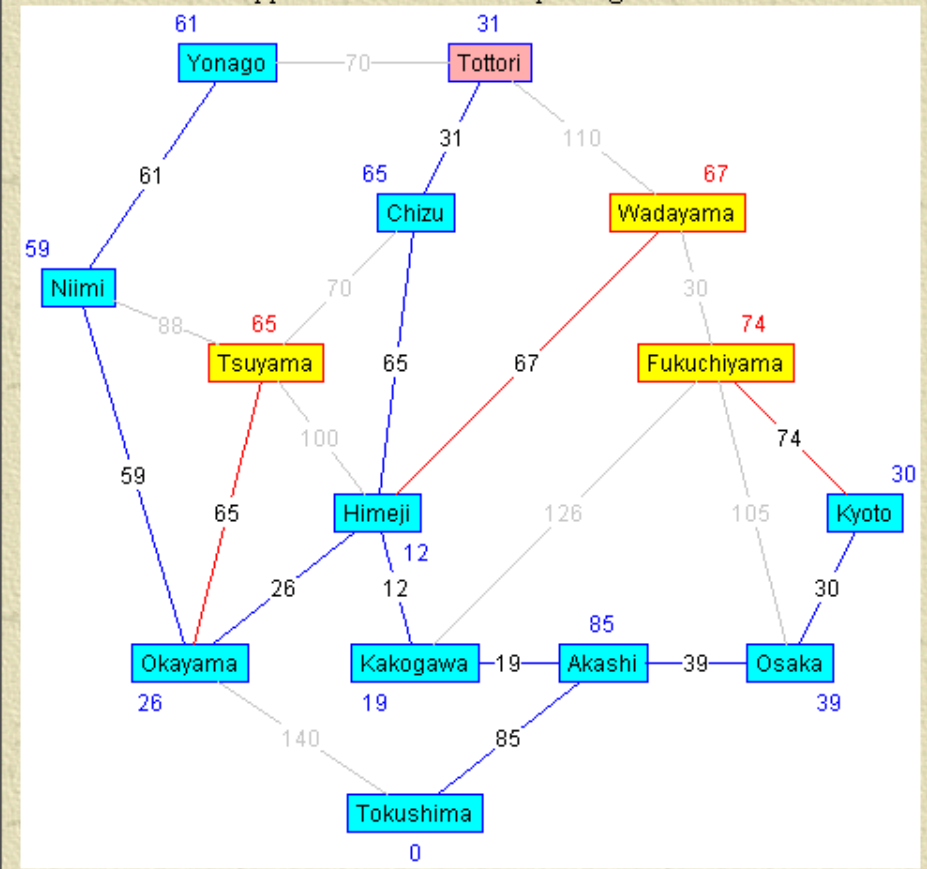
Applet Prim started

Internet

Start the shortest ... Prim's Algor... HELP JUNG Demo: ... Microsoft Pow... 20:41

Java Applet Demo of Prim's Algorithm

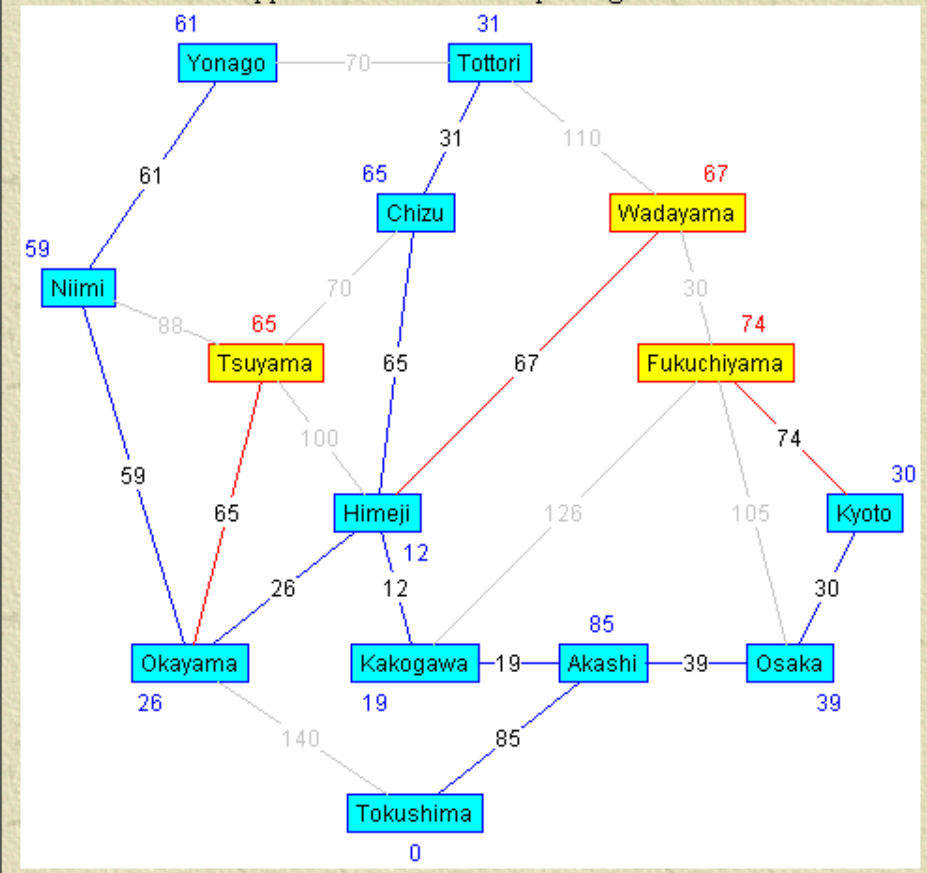
Click on the below applet to find a minimum spanning tree.



Applet written by: Kenji Ikeda

Java Applet Demo of Prim's Algorithm

Click on the below applet to find a minimum spanning tree.



Applet written by: Kenji Ikeda

Prim's Algorithm Demo - Microsoft Internet Explorer

File Edit View Favorites Tools Help

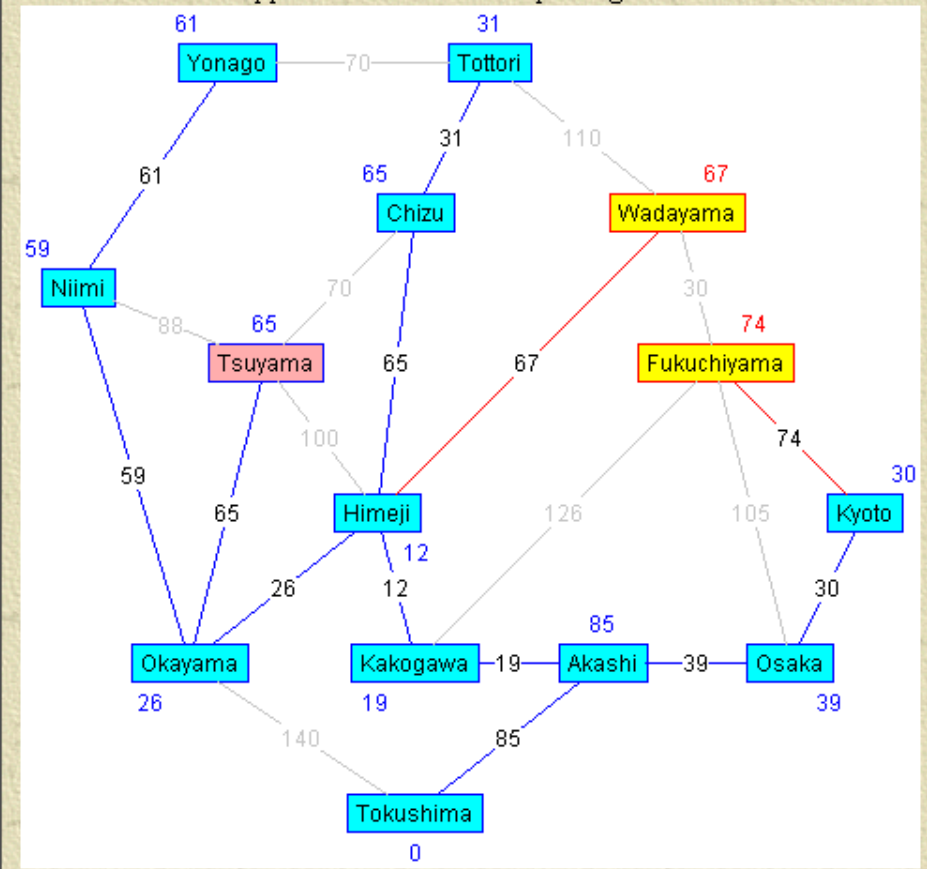
Back Search Favorites Media

Address http://www.unf.edu/~wkloster/foundations/PrimApplet/PrimApplet.htm Go Links

Google prim algorithm demo Search 26 blocked Check AutoLink AutoFill Options prim algorithm demo

Java Applet Demo of Prim's Algorithm

Click on the below applet to find a minimum spanning tree.



Applet written by: Kenji Ikeda

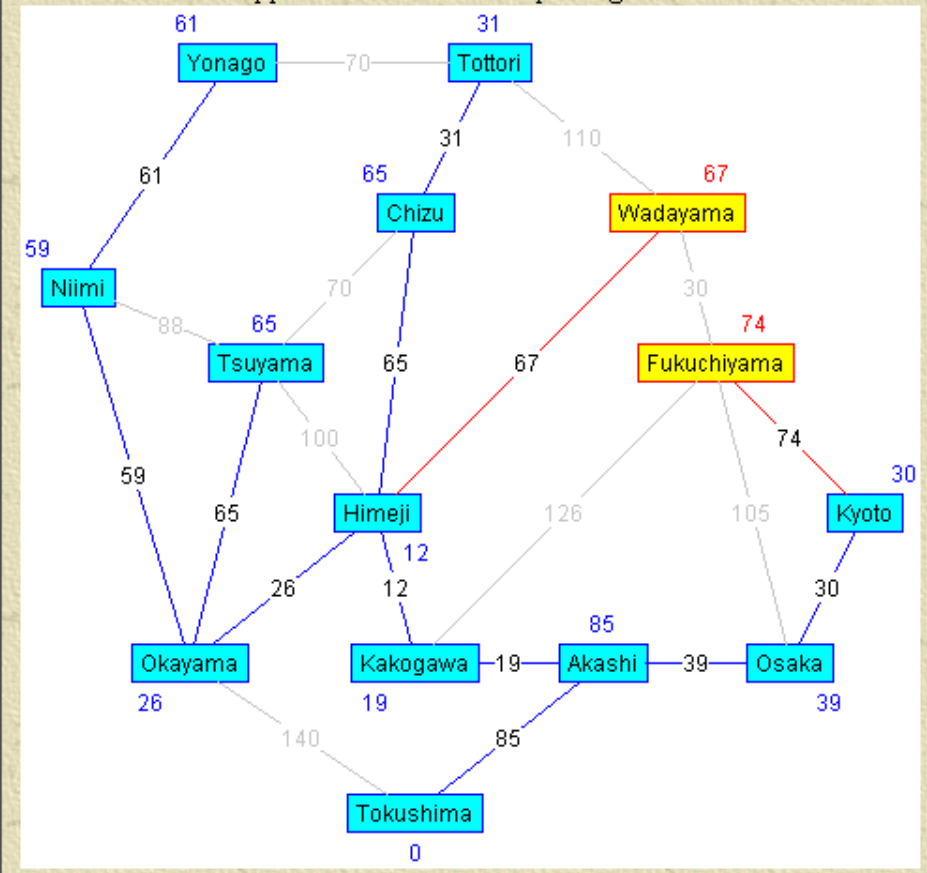
Applet Prim started

Internet

Start the shortest ... Prim's Algor... HELP JUNG Demo: ... Microsoft Pow... 20:42

Java Applet Demo of Prim's Algorithm

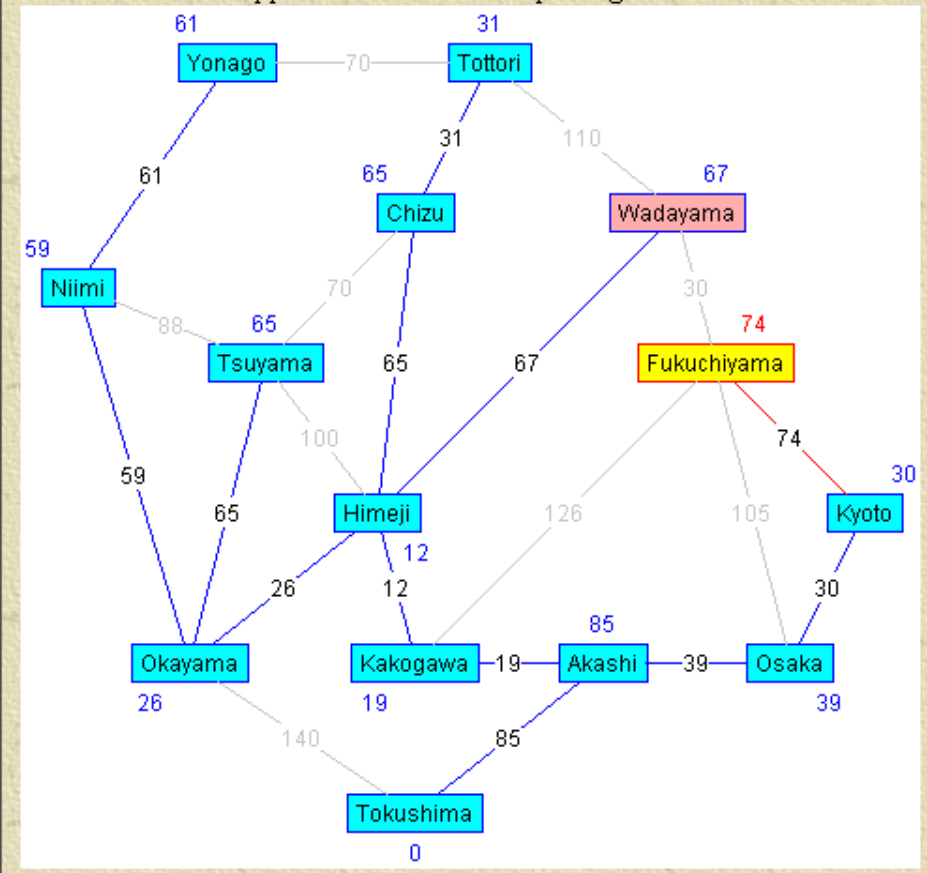
Click on the below applet to find a minimum spanning tree.



Applet written by: Kenji Ikeda

Java Applet Demo of Prim's Algorithm

Click on the below applet to find a minimum spanning tree.



Applet written by: Kenji Ikeda

Prim's Algorithm Demo - Microsoft Internet Explorer

File Edit View Favorites Tools Help

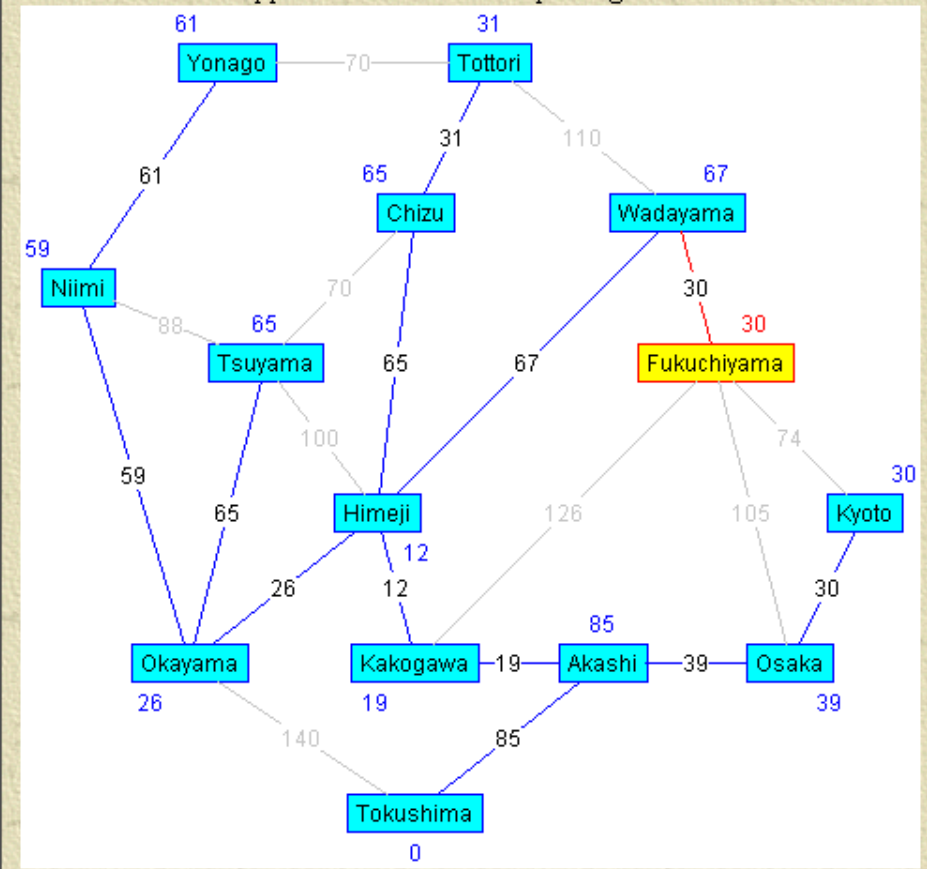
Back Search Favorites Media

Address http://www.unf.edu/~wkloster/foundations/PrimApplet/PrimApplet.htm Go Links

Google prim algorithm demo Search 26 blocked Check AutoLink AutoFill Options prim algorithm demo

Java Applet Demo of Prim's Algorithm

Click on the below applet to find a minimum spanning tree.



Applet written by: Kenji Ikeda

Applet Prim started

Internet

Start the shortest ... Prim's Algor... HELP JUNG Demo: ... Microsoft Pow... 20:43

Prim's Algorithm Demo - Microsoft Internet Explorer

File Edit View Favorites Tools Help

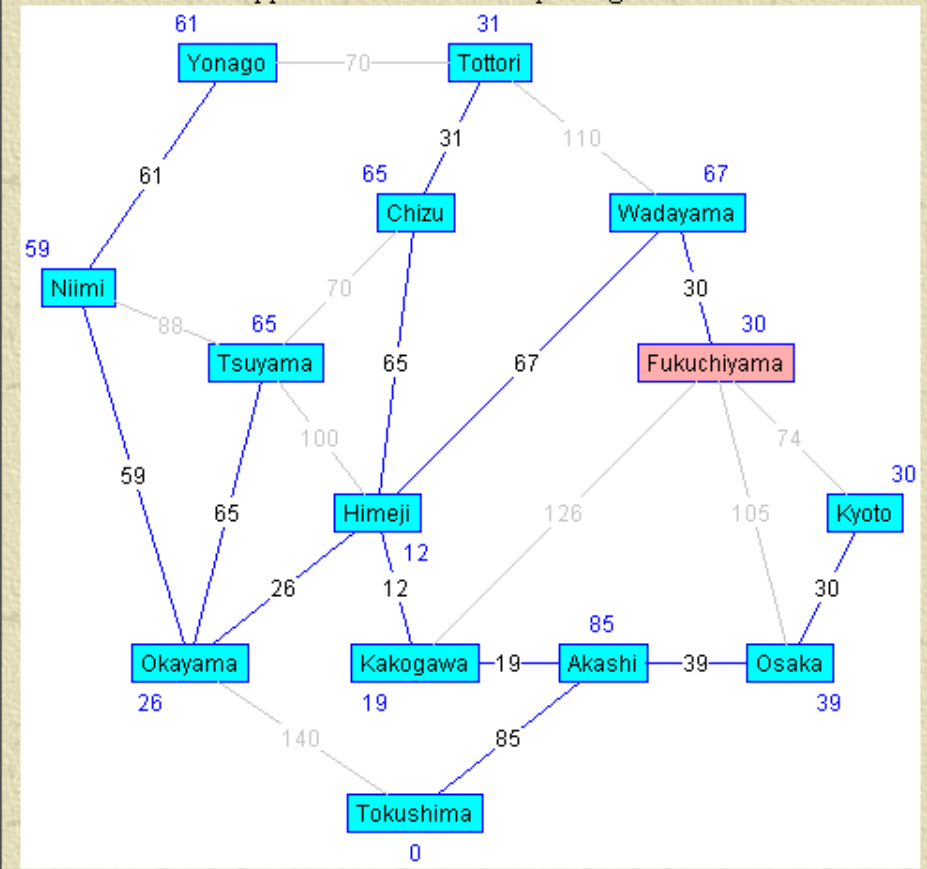
Back Search Favorites Media

Address http://www.unf.edu/~wkloster/foundations/PrimApplet/PrimApplet.htm Go Links

Google prim algorithm demo Search 26 blocked Check AutoLink AutoFill Options prim algorithm demo

Java Applet Demo of Prim's Algorithm

Click on the below applet to find a minimum spanning tree.



Applet written by: Kenji Ikeda

Applet Prim started

Internet

Start the shortest ... Prim's Algor... HELP JUNG Demo: ... Microsoft Pow... 20:43

Prim's Algorithm Demo - Microsoft Internet Explorer

File Edit View Favorites Tools Help

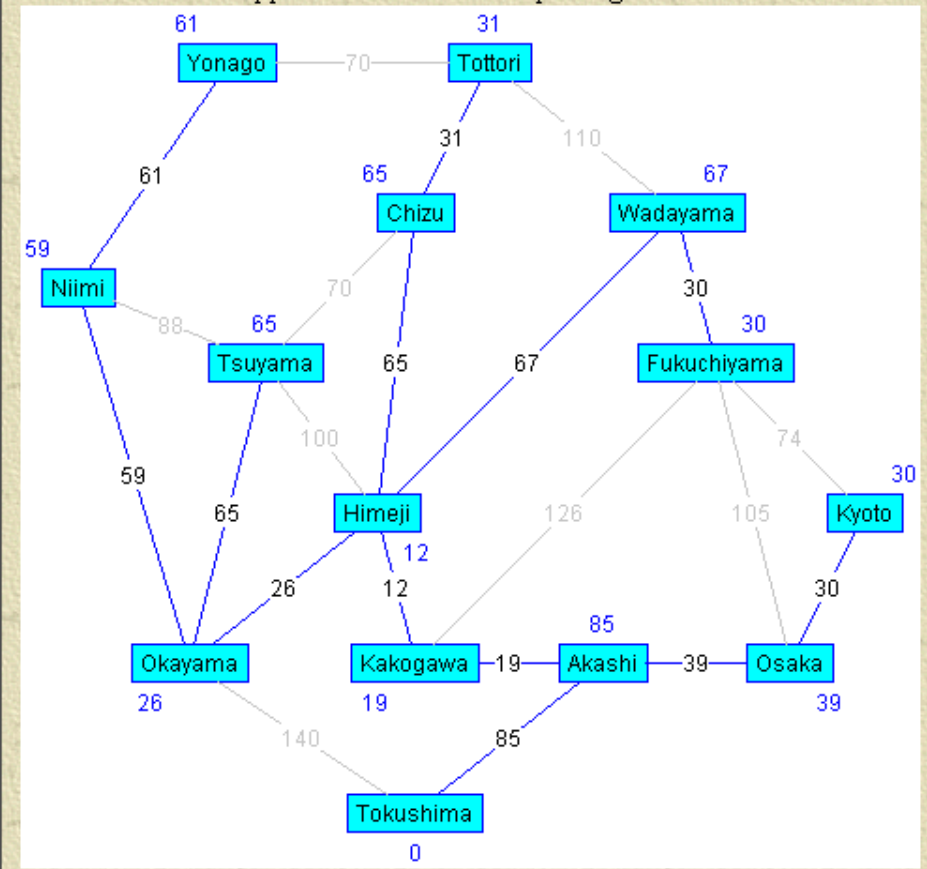
Back Search Favorites Media

Address http://www.unf.edu/~wkloster/foundations/PrimApplet/PrimApplet.htm Go Links

Google prim algorithm demo Search 26 blocked Check AutoLink AutoFill Options prim algorithm demo

Java Applet Demo of Prim's Algorithm

Click on the below applet to find a minimum spanning tree.



Applet written by: Kenji Ikeda

Applet Prim started

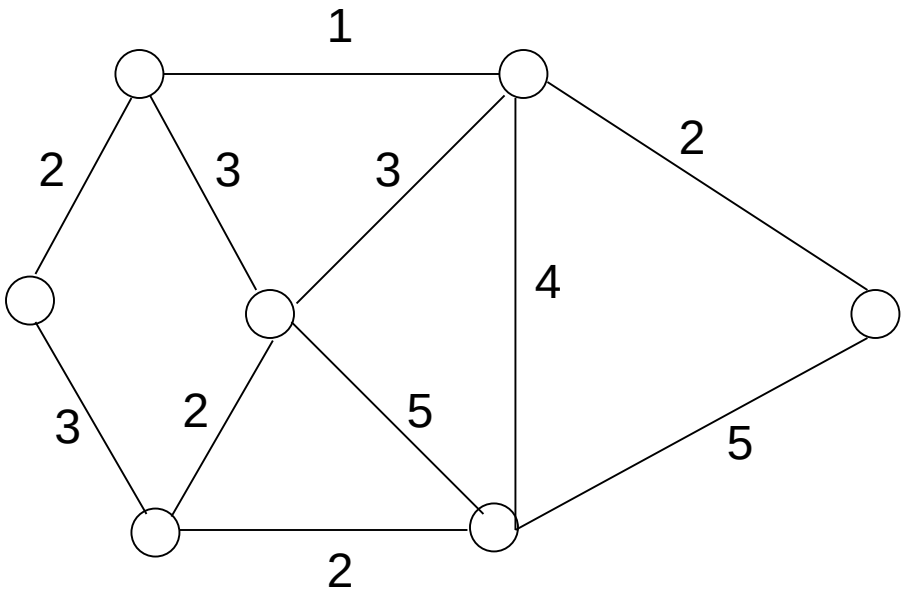
Internet

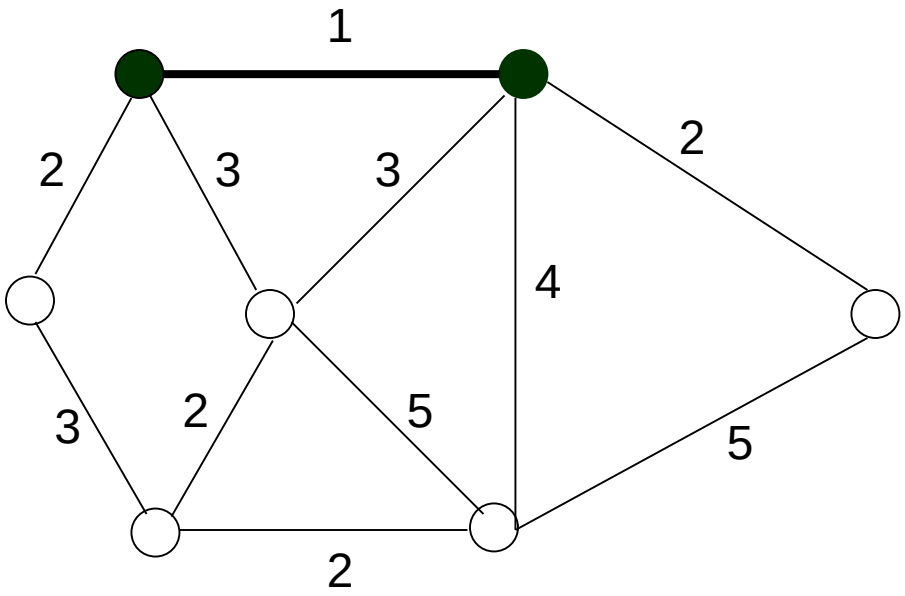
Start the shortest ... Prim's Algor... HELP JUNG Demo: ... Microsoft Pow... 20:44

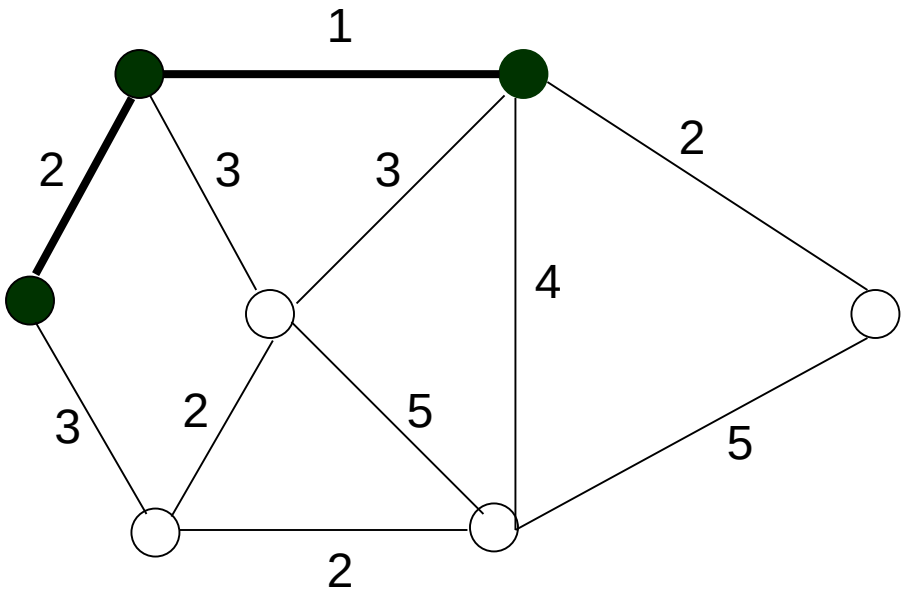
Kruskal's Algorithm

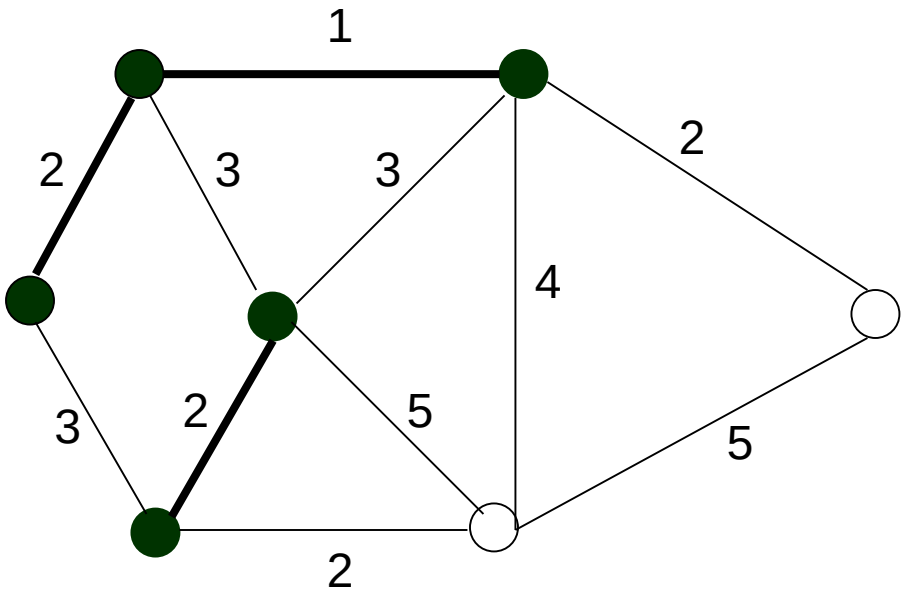
- Adım 1: Graftaki en küçük ağırlık değerine sahip kenarı bul (eğer birden fazla varsa rastgele birini seç)
  - Adım 2: En küçük değere sahip bir sonraki kenarı bul ancak kapalı bir döngü oluşturmasın
    - Adım 3: Graftaki her bir düğümü dolaşana kadar adım-2 yi yap

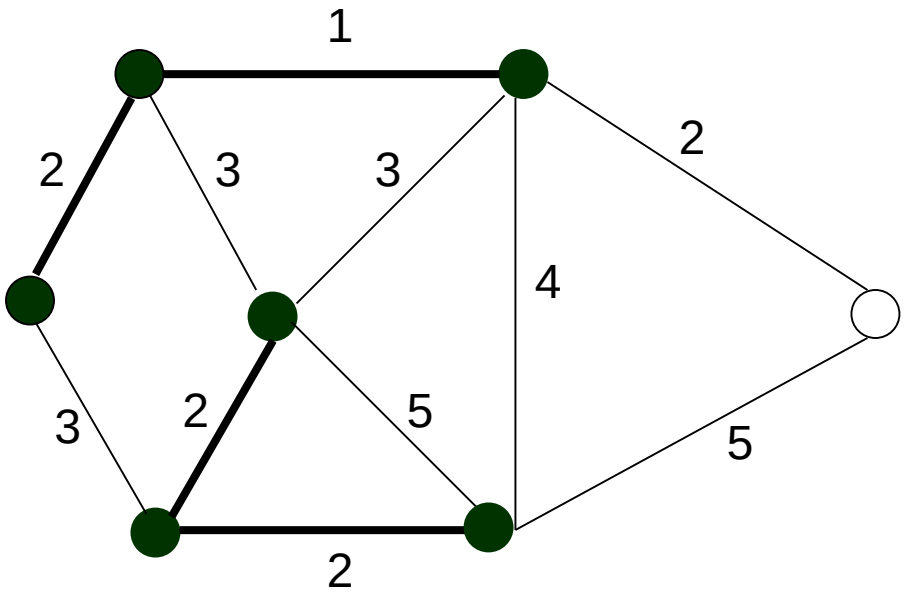
n düğüm için n-1 kenar seçilmelidir

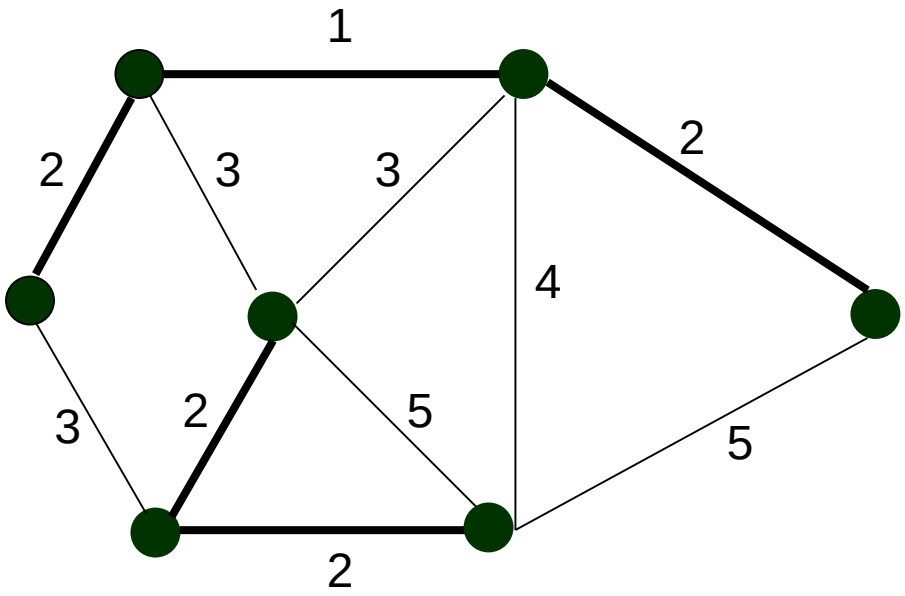

1
2
3
3
2
4
3
2
5
5
2

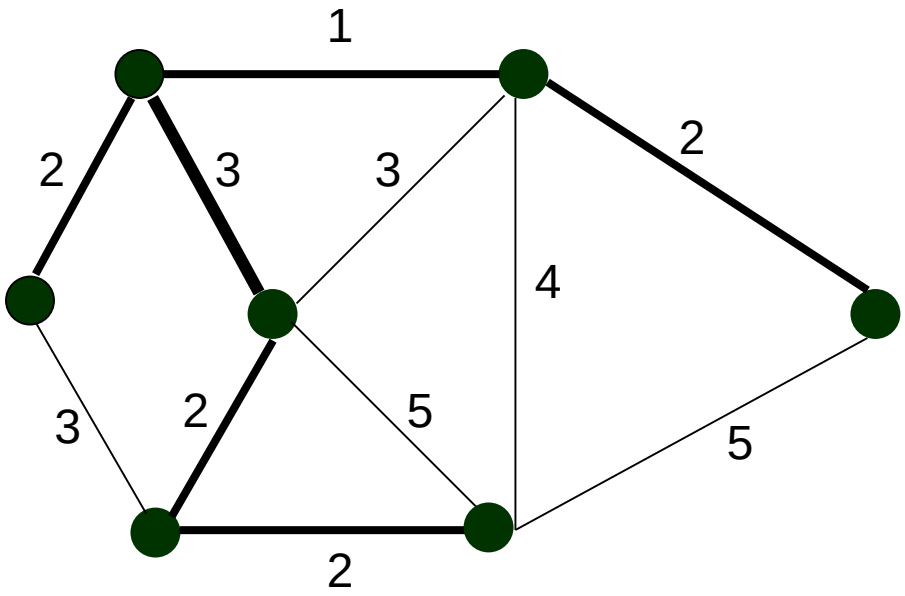

1
2
3
3
2
4
3
2
5
5
2


1
2
3
3
2
4
3
2
5
5
2

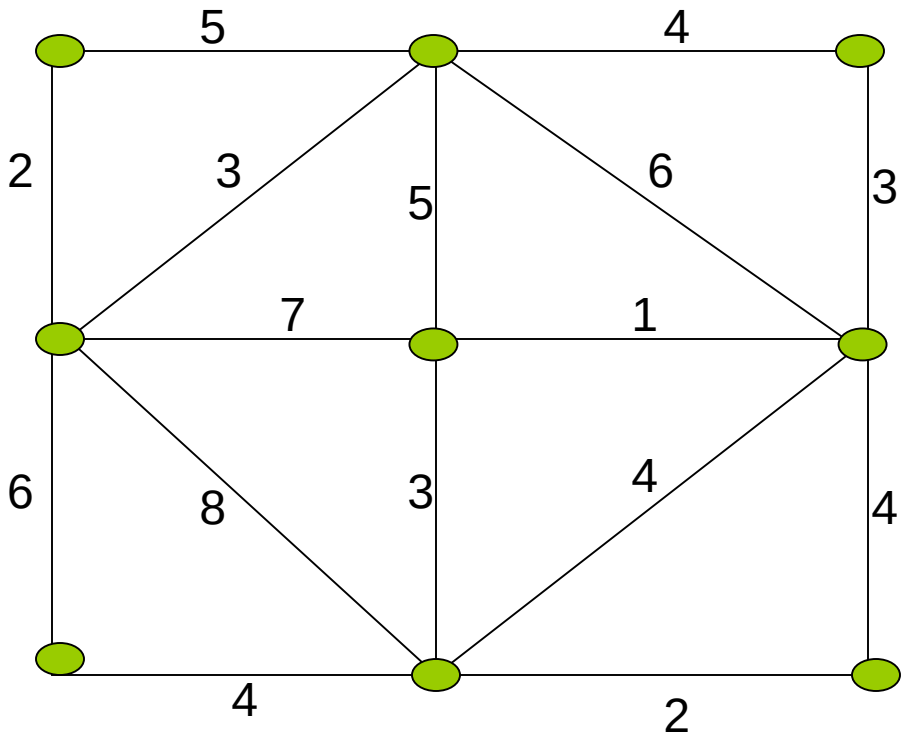

1
2
3
3
2
4
3
2
5
5
2


1
2
3
3
2
4
3
2
5
5
2

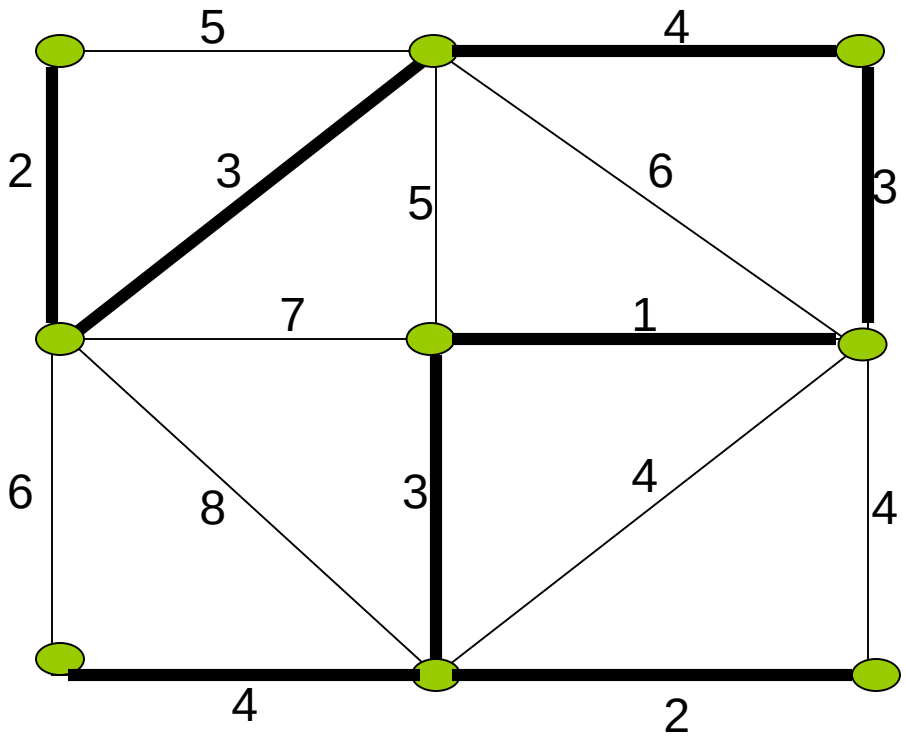

1
2
3
3
2
4
3
2
5
5
2


1
2
3
3
2
4
3
2
5
5
2

Örnek



çözüm



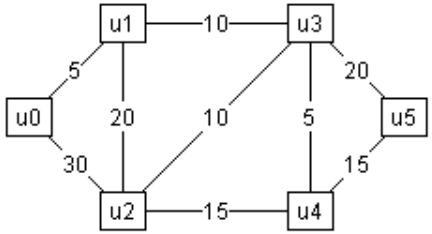
Kruskal
Java applet demos:

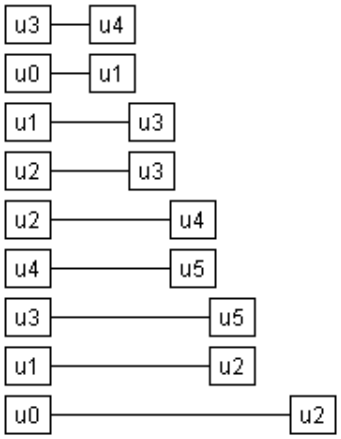
- demo1
- demo2
- demo3
- demo4
- demo5

Kenji Ikeda's Home Page

Java Applet Demos of Kruskal's Algorithm

Click on the applet below to find a minimum spanning tree. (Not on the right one.)

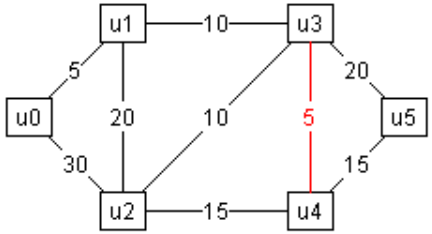


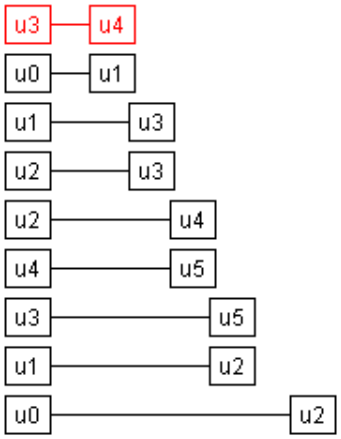


Last Modified: Friday, 24-May-2002 12:17:20 JST

Java Applet Demos of Kruskal's Algorithm

Click on the applet below to find a minimum spanning tree. (Not on the right one.)





Kruskal
Java applet demos:

- demo1
- demo2
- demo3
- demo4
- demo5

FAQ

Kenji Ikeda's Home Page

Last Modified: Friday, 24-May-2002 12:17:20 JST

Mathematical Programming

Simplex

Twophase

Dijkstra

Prim

Kruskal

Ford-Fulkerson

Kruskal
Java applet demos:

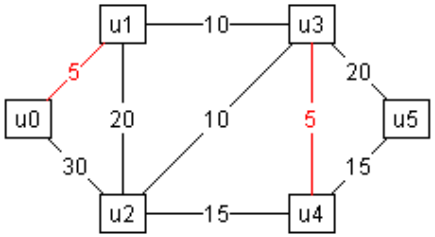
- demo1
- demo2
- demo3
- demo4
- demo5

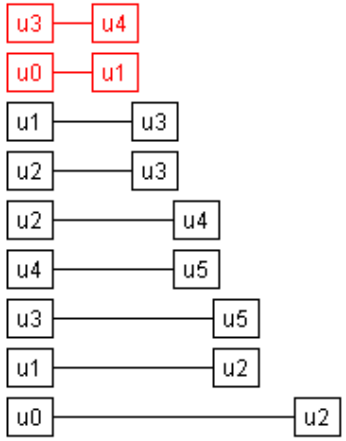
FAQ

Kenji Ikeda's Home Page

Java Applet Demos of Kruskal's Algorithm

Click on the applet below to find a minimum spanning tree. (Not on the right one.)





Last Modified: Friday, 24-May-2002 12:17:20 JST

Mathematical Programming

Simplex

Twophase

Dijkstra

Prim

Kruskal

Ford-Fulkerson

Kruskal
Java applet demos:

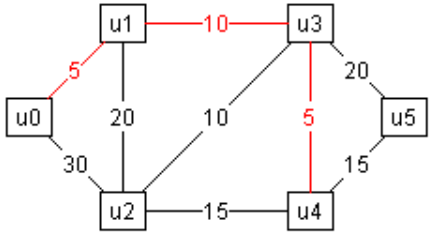
- demo1
- demo2
- demo3
- demo4
- demo5

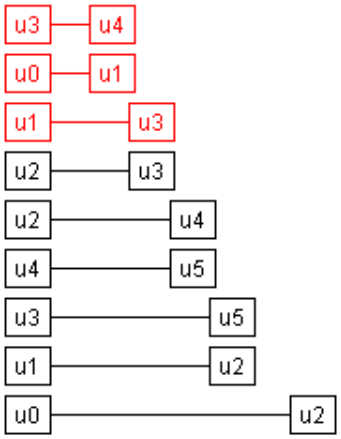
FAQ

Kenji Ikeda's Home Page

Java Applet Demos of Kruskal's Algorithm

Click on the applet below to find a minimum spanning tree. (Not on the right one.)





Last Modified: Friday, 24-May-2002 12:17:20 JST

Mathematical Programming

Simplex

Twophase

Dijkstra

Prim

Kruskal

Ford-Fulkerson

Kruskal
Java applet demos:

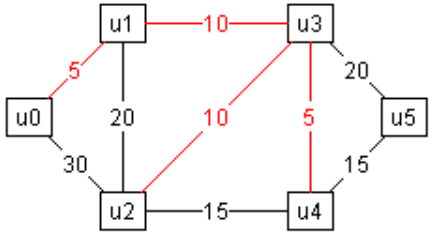
- demo1
- demo2
- demo3
- demo4
- demo5

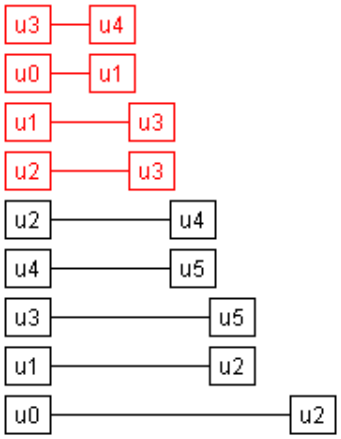
FAQ

Kenji Ikeda's Home Page

Java Applet Demos of Kruskal's Algorithm

Click on the applet below to find a minimum spanning tree. (Not on the right one.)





Last Modified: Friday, 24-May-2002 12:17:20 JST

Mathematical Programming

Simplex

Twophase

Dijkstra

Prim

Kruskal

Ford-Fulkerson

Kruskal
Java applet demos:

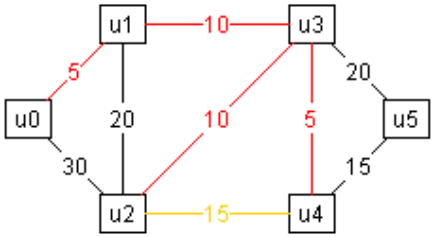
- demo1
- demo2
- demo3
- demo4
- demo5

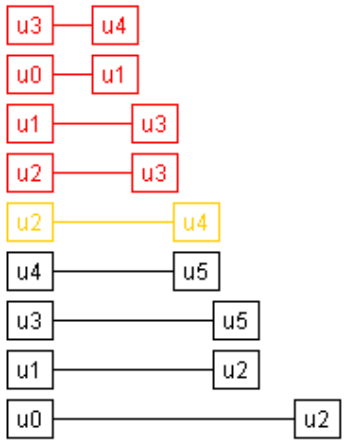
FAQ

Kenji Ikeda's Home Page

Java Applet Demos of Kruskal's Algorithm

Click on the applet below to find a minimum spanning tree. (Not on the right one.)





Last Modified: Friday, 24-May-2002 12:17:20 JST

Mathematical Programming

Simplex

Twophase

Dijkstra

Prim

Kruskal

Ford-Fulkerson

Kruskal
Java applet demos:

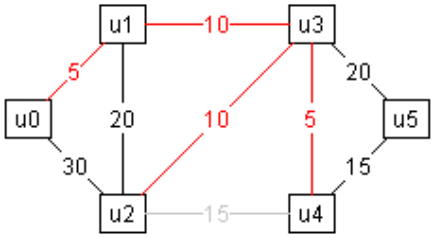
- demo1
- demo2
- demo3
- demo4
- demo5

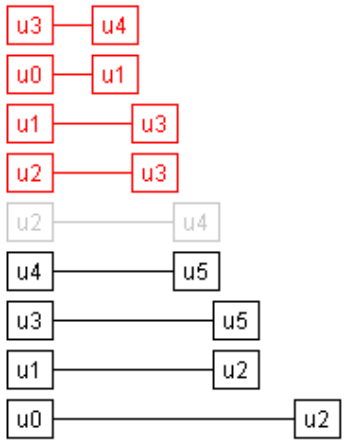
FAQ

Kenji Ikeda's Home Page

Java Applet Demos of Kruskal's Algorithm

Click on the applet below to find a minimum spanning tree. (Not on the right one.)





Last Modified: Friday, 24-May-2002 12:17:20 JST

Mathematical Programming

Simplex

Twophase

Dijkstra

Prim

Kruskal

Ford-Fulkerson

Kruskal
Java applet demos:

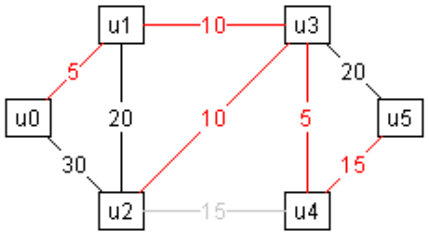
- demo1
- demo2
- demo3
- demo4
- demo5

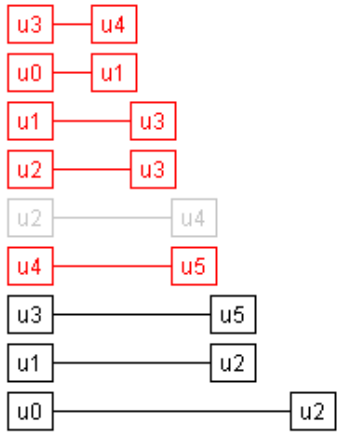
FAQ

Kenji Ikeda's Home Page

Java Applet Demos of Kruskal's Algorithm

Click on the applet below to find a minimum spanning tree. (Not on the right one.)





Last Modified: Friday, 24-May-2002 12:17:20 JST

Mathematical Programming

Simplex

Twophase

Dijkstra

Prim

Kruskal

Ford-Fulkerson

Kruskal
Java applet demos:

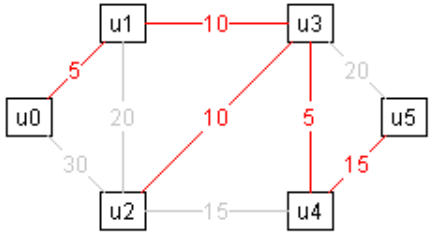
- demo1
- demo2
- demo3
- demo4
- demo5

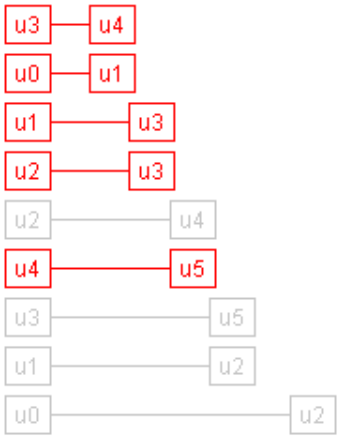
FAQ

Kenji Ikeda's Home Page

Java Applet Demos of Kruskal's Algorithm

Click on the applet below to find a minimum spanning tree. (Not on the right one.)



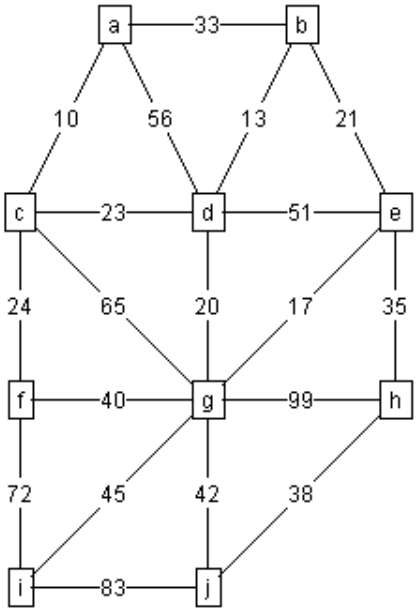


Last Modified: Friday, 24-May-2002 12:17:20 JST

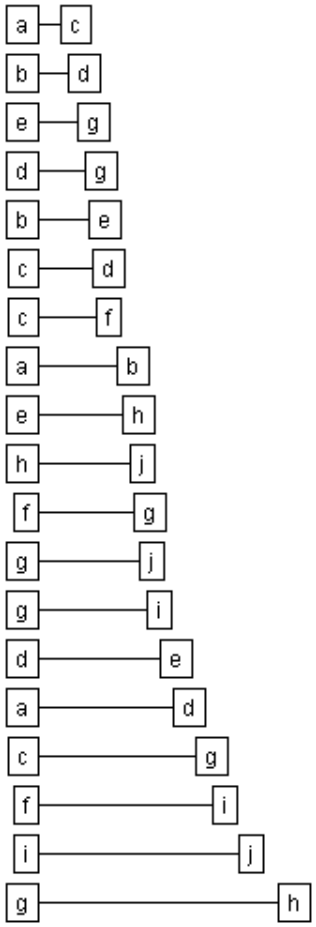
Kruskal
Java applet demos:

- demo1
- demo2
- demo3
- demo4
- demo5

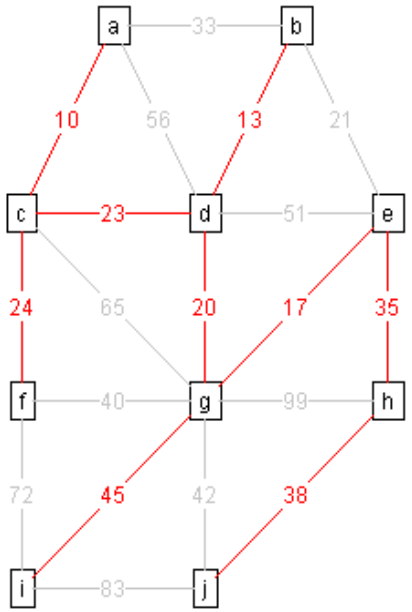
Java Applet Demos of Kruskal's Algorithm

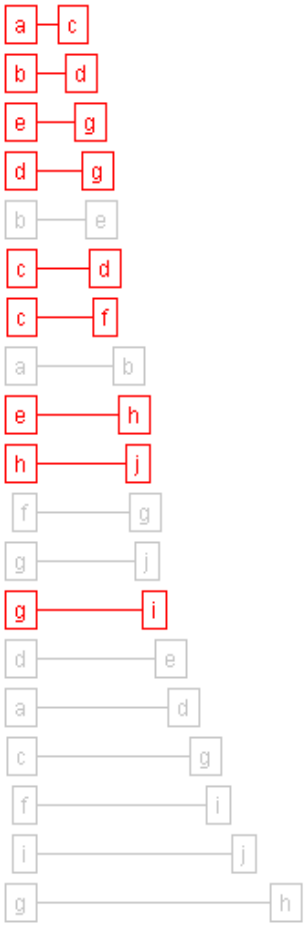


Click on the applet above to find a minimum spanning tree. (Not on the right one.)



Java Applet Demos of Kruskal's Algorithm





Kruskal
Java applet demos:

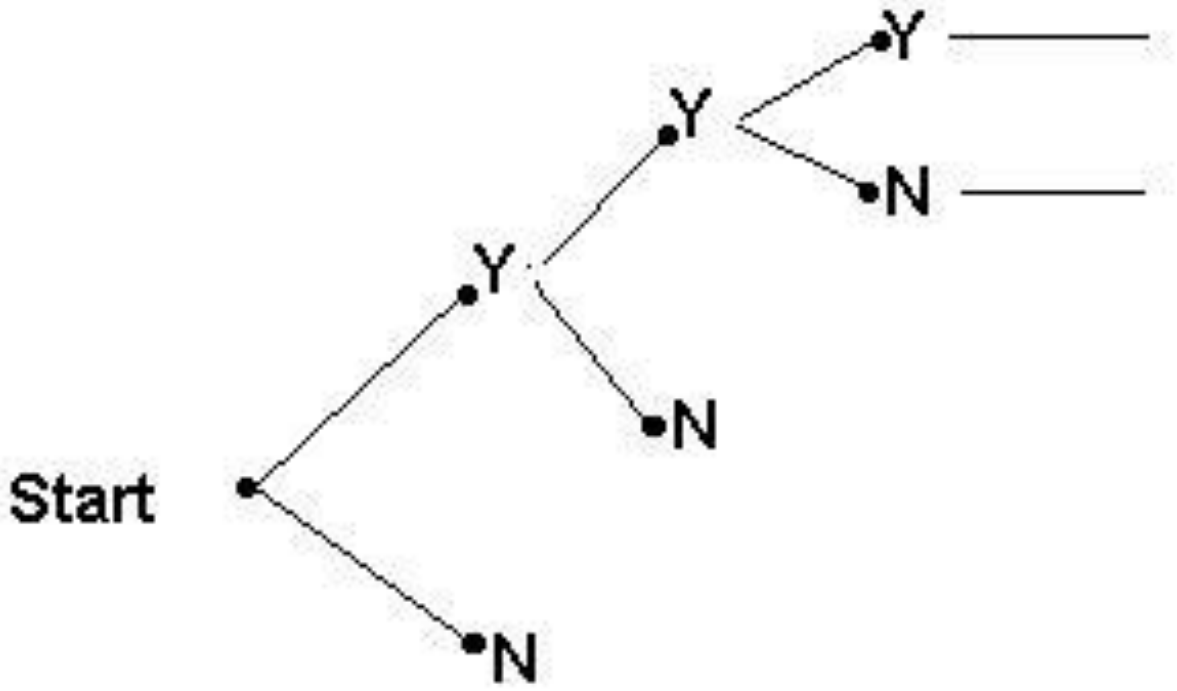
- demo1
- demo2
- demo3
- demo4
- demo5

Click on the applet above to find a minimum spanning tree. (Not on the right one.)

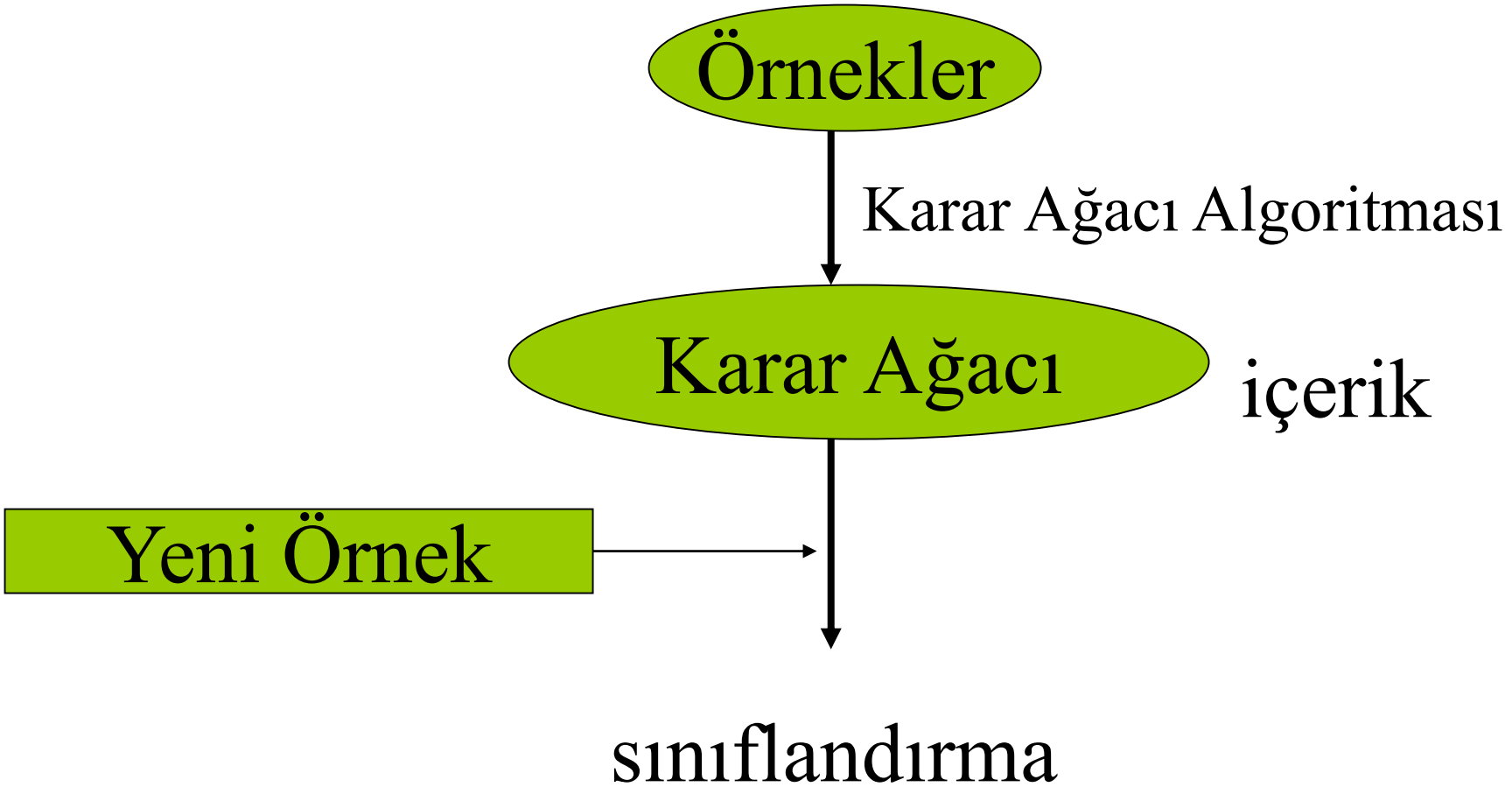
FAQ

Karar Ağaçları (Decision trees)

Karar Ağaçları, bir Binary Tree olup bir olayın sonuçlandırılmasında sorunun cevabına göre hareket ederler. Binary Search, Hileli paranın bulunması örnek olarak verilebilir



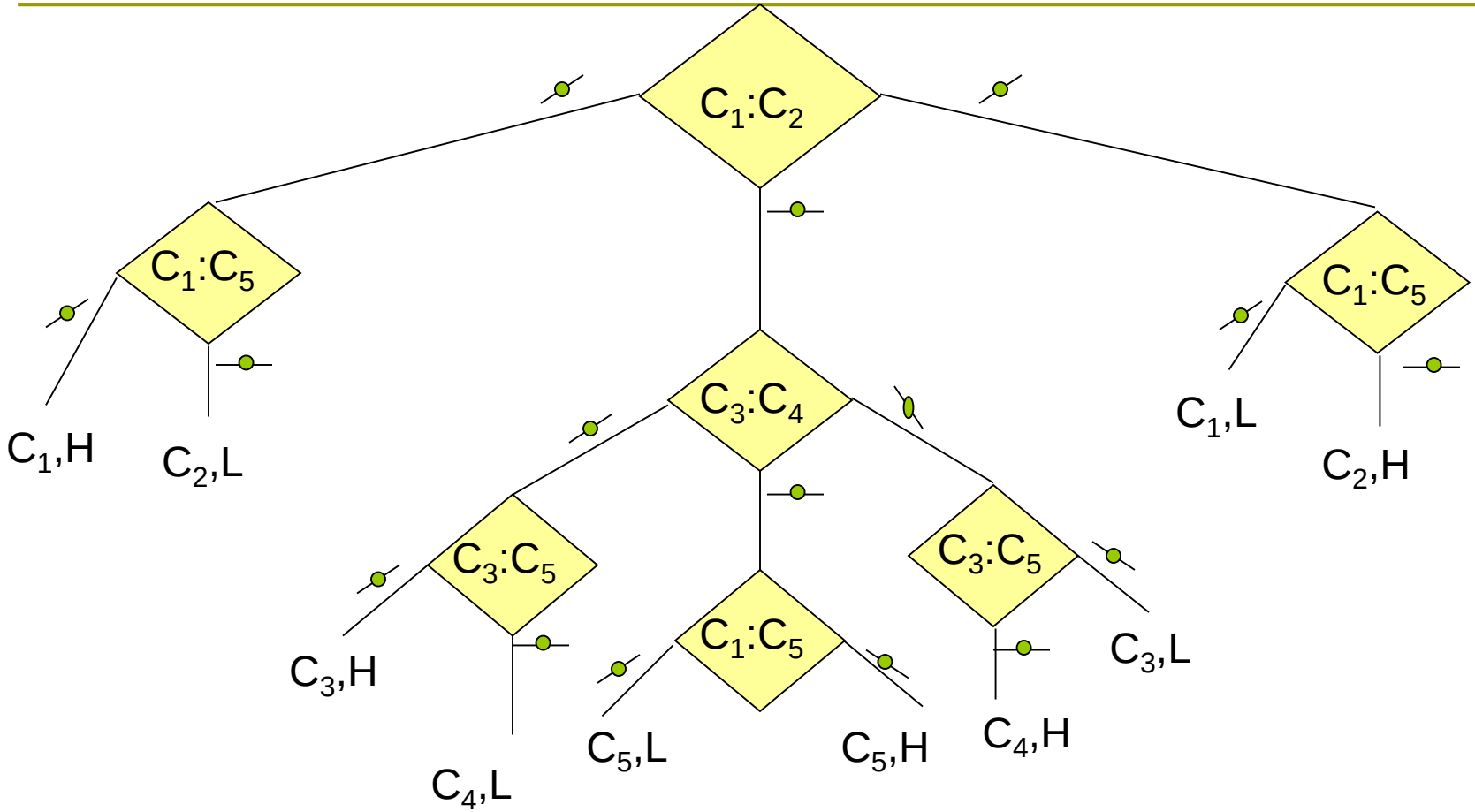
Tümevarımsal Sonuç Çıkarım (inductive inference) için kullanılan en popüler yöntemlerden birisi Karar Ağaç'larıdır (Decision Tree)


Örnekler
Karar Ağacı Algoritması
Karar Ağacı
içerik
Yeni Örnek
sınıflandırma

FIVE-COINS PUZZLE

- Görünüşleri birbirinin aynı olan 5 paradan biri sahtedir.
- Sahte olan para diğerlerinden daha ağır veya daha hafif olabilir.
- Karar ağacını kullanarak hangi paranın sahte olduğuna karar veriniz
- Sahte olan para diğerlerine göre hafif midir yoksa ağır mıdır?

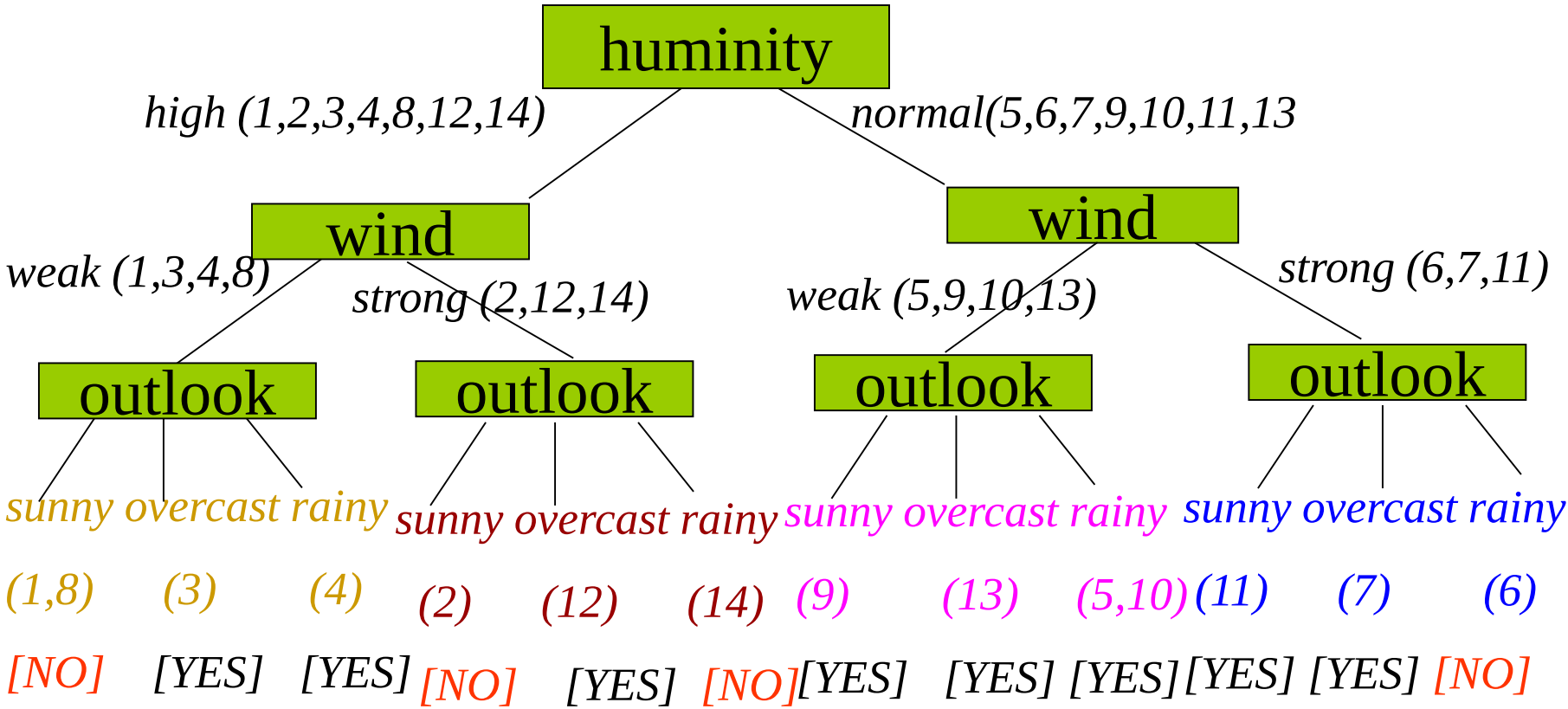
K A R A R ????


C1:C2
C1:C5
C1:C5
C3:C4
C1,H
C2,L
C1,L
C2,H
C3:C5
C3:C5
C1:C5
C3,H
C3,L
C4,L
C5,L
C5,H
C4,H

Play Tennis ?????

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
6	rain	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rain	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rain	mild	high	strong	no

Özelliklerden biri kök seçilir (outlook, temp, huminity, wind)



Karar Ağacı iyi bir çözümdür

ancak

optimum değildir

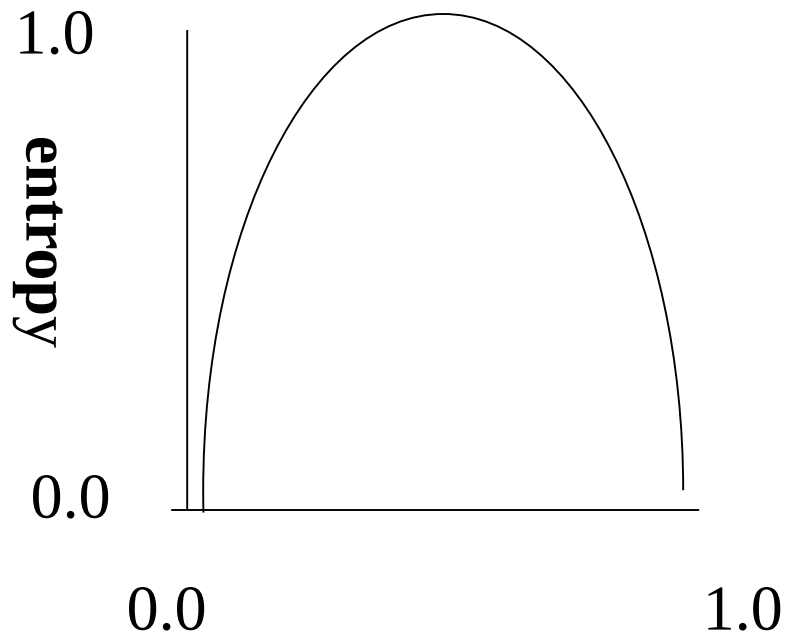
optimum bir karar ağacının oluşturulması için bir kuralın olması gerekir

Information Gain (maksimum kazanç)

$$\text{Entropy}(S) \equiv -p_{\oplus} \log_2 p_{\oplus} - p_{\ominus} \log_2 p_{\ominus}$$

Bütün örnekler aynı sınıfa ait ise E(S)=0 (homojen)

Bütün örnekler sınıflara eşit dağılmış ise E(S)=1 (heterojen)



Entropy([9+,5-] = 0.940

Entropy([7+,7-] = 1

A özelliğinin, S örneği için kazancı (information gain)

$$Gain(S,A) \equiv Entropy(S) - \Sigma\ P(v)\ Entropy(S(v))$$

v: Values of A

$$P(v) \equiv |S(v)| / |S|$$

S:[9+,5-]

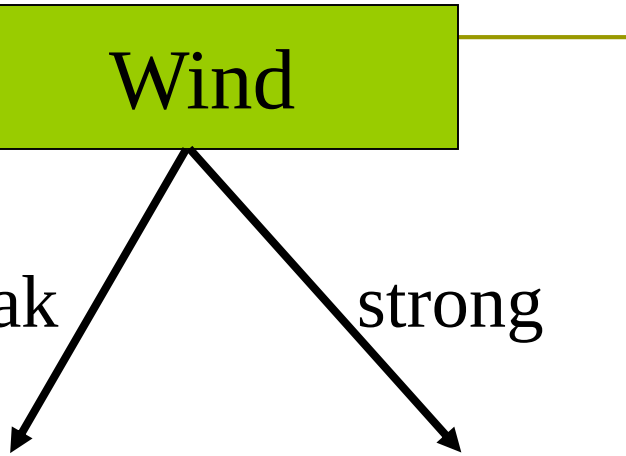
E = 0.940

Gain(S,wind) = ?

Gain(S,huminity) = ?

Gain(S,temperature) = ?

Gain(S,outlook) = ?



[6+,2-]

E = 0.811

[3+,3-]

E = 1.0

Gain(S,wind)

$= 0.940 - (8/14)[-(6/8)\log_2(6/8) - (2/8)\log_2(2/8)]$

$-(6/14)[-(3/6)\log_2(3/6) - (3/6)\log_2(3/6)]$

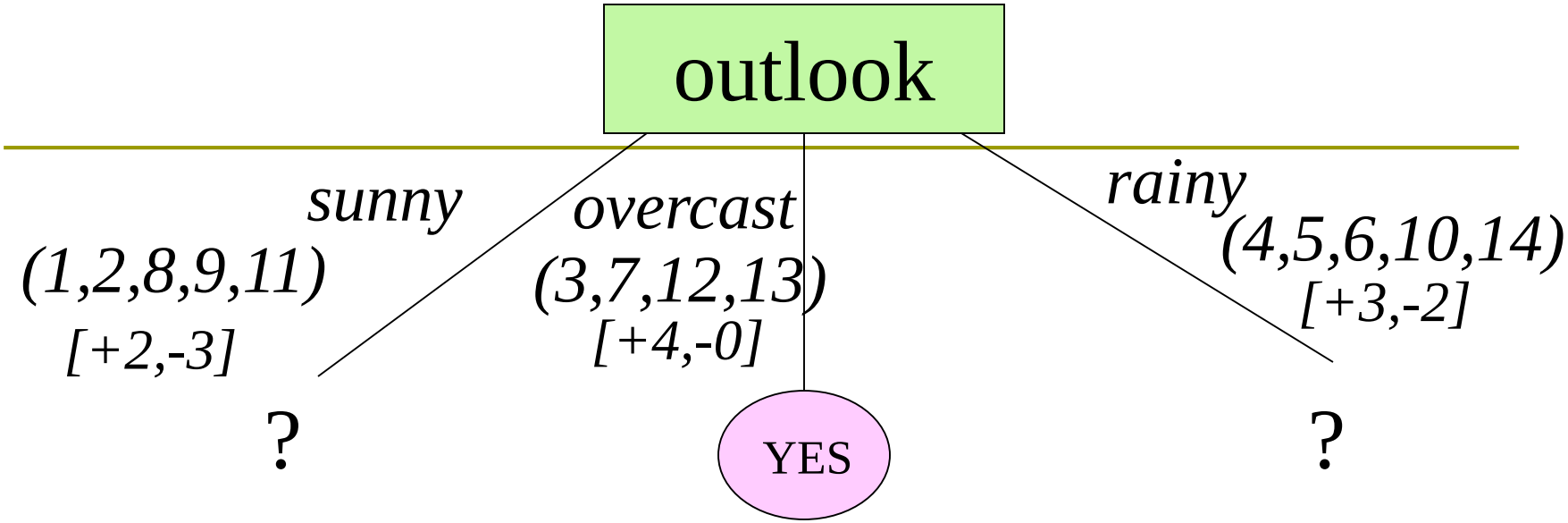
=0.048

Gain(S,wind) = 0.048

Gain(S,huminity) = 0.15

Gain(S,temperature) = 0.029

Gain(S,outlook) = 0.246



S_{sunny}=[+2,-3]

$E(sunny)=-(2/5)\log_2(2/5)-(3/5)\log_2(3/5)=0.97$

$Gain(S_{sunny,huminity}) = ?$

$Gain(S_{sunny,temp}) = ?$

$Gain(S_{sunny,wind}) = ?$

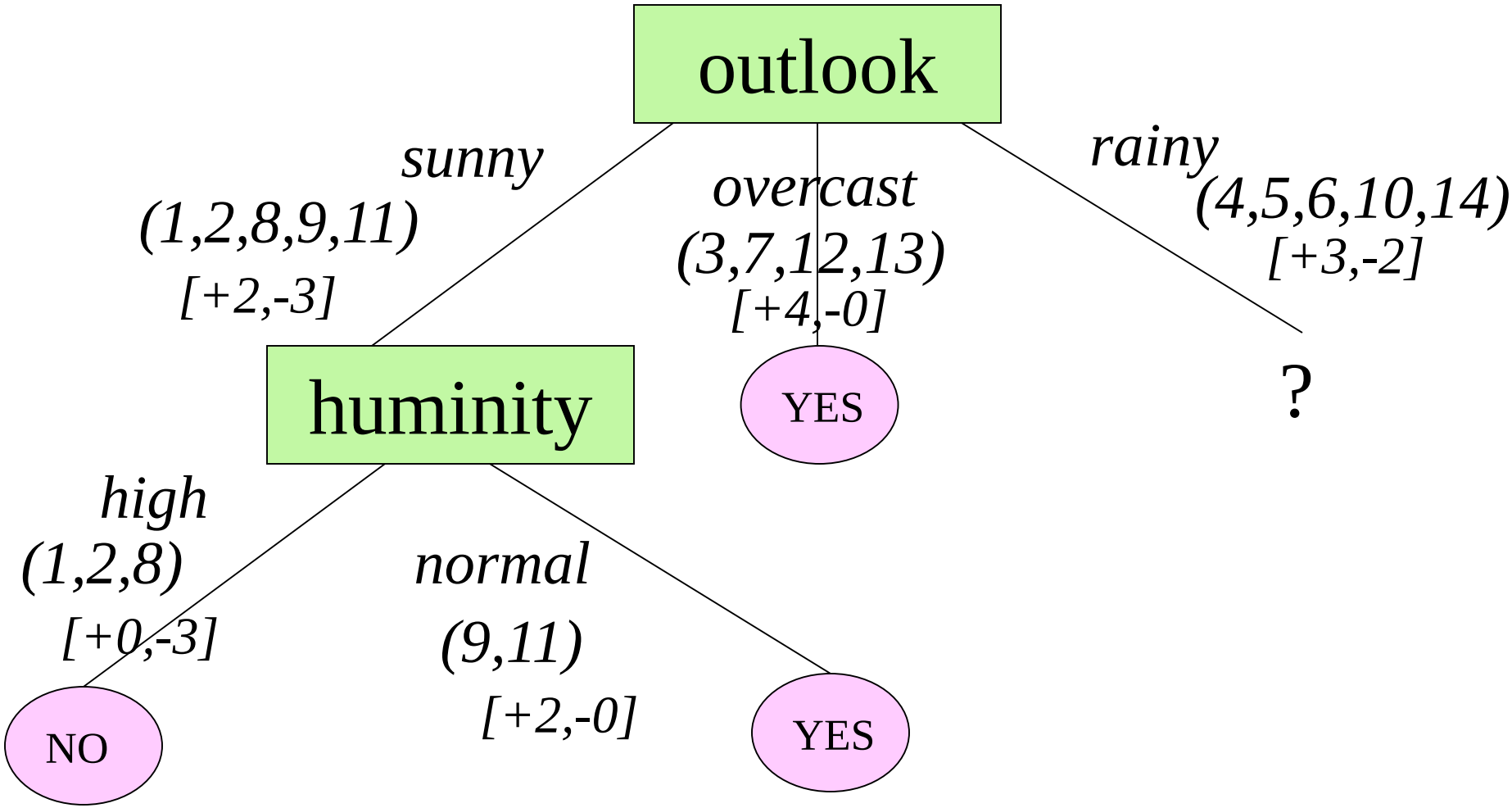
$$Gain(S_{sunny,huminity}) = 0.97-(2/5)[-(2/2)\log_2(2/2)-(0/2)\log_2(0/2)]$$

$$-(3/5)[-(3/3)\log_2(3/3)-(0/3)\log_2(0/3)]$$

$$= 0.97$$

$$Gain(S_{sunny,wind}) = 0.019$$

$$Gain(S_{sunny,temp}) = 0.57$$


outlook
sunny
(1,2,8,9,11)
[+2,-3]
overcast
(3,7,12,13)
[+4,-0]
rainy
(4,5,6,10,14)
[+3,-2]
huminity
YES
?
high
(1,2,8)
[+0,-3]
normal
(9,11)
[+2,-0]
NO
YES

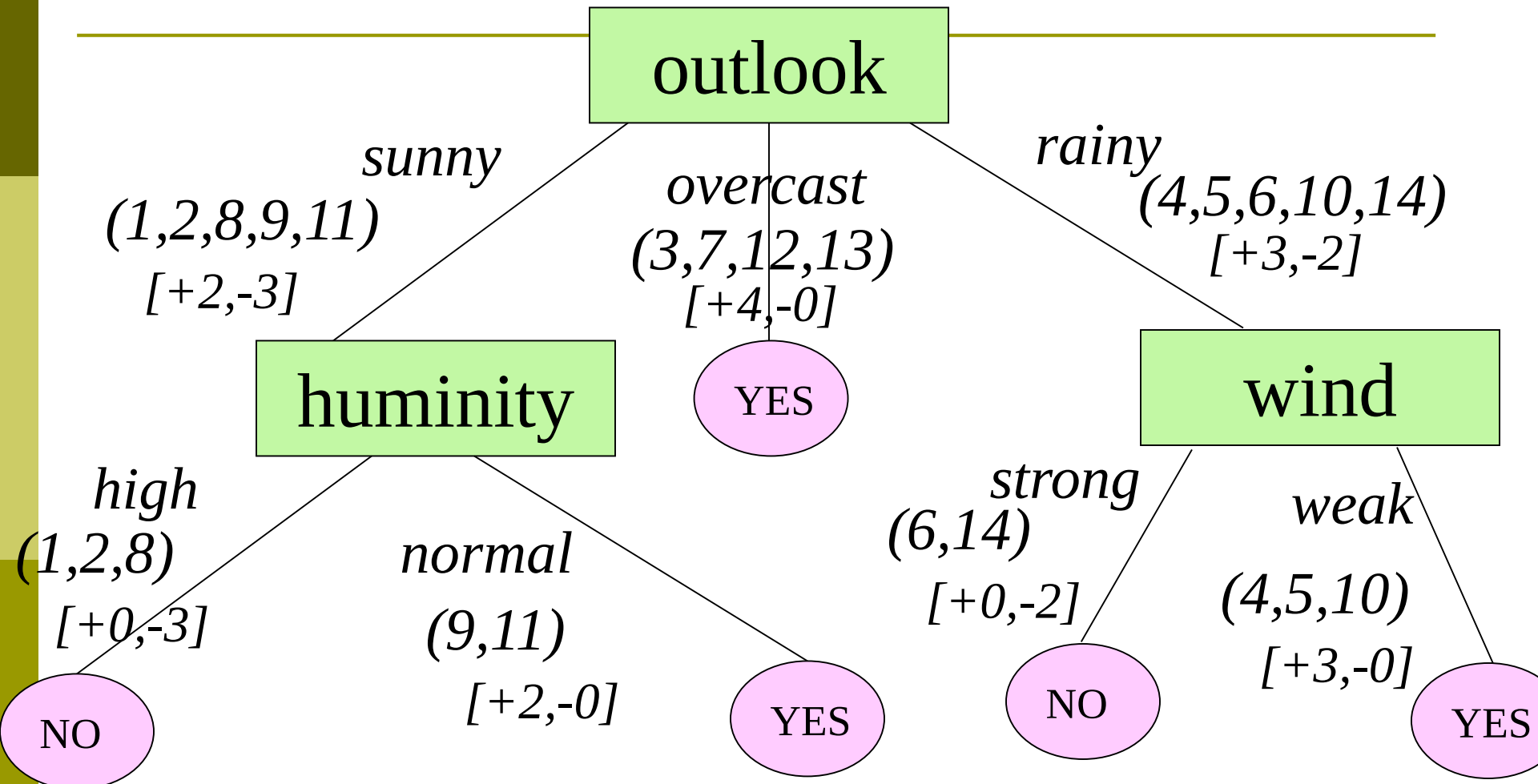
Aynı işlem

$$Gain(S_{rainy,huminity}) = ?$$

$$Gain(S_{rainy,temp}) = ?$$

$$Gain(S_{rainy,wind}) = ?$$

bulmak için yapılır.


outlook
sunny
(1,2,8,9,11)
[+2,-3]
overcast
(3,7,12,13)
[+4,-0]
rainy
(4,5,6,10,14)
[+3,-2]
huminity
YES
wind
high
(1,2,8)
[+0,-3]
normal
(9,11)
[+2,-0]
strong
(6,14)
[+0,-2]
weak
(4,5,10)
[+3,-0]
NO
YES
NO
YES

Ağaçlarda Benzerlik (Isomorphism

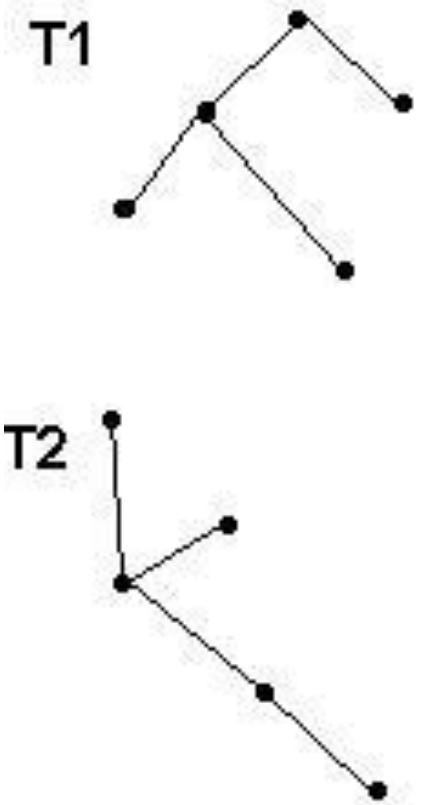
Verilen iki ağaç T_1 ve T_2 olsun

- T_1 ve T_2 *isomorphic* dir

- Bire-Bir ve örten fonksiyon özelliklerini görmemiz gerekir

  $f : T_1 \rightarrow T_2$

- T_1 ve T_2 birbirlerine çok yakındır



Köklü Ağaçlarda (Isomorphism)

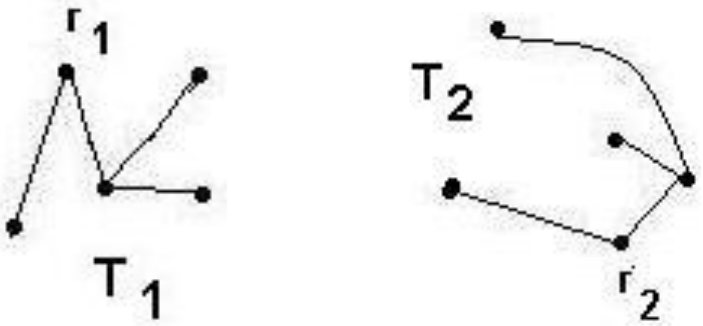
T_1 ve T_2 , r_1 ve r_2 kök değerine sahip köklü iki ağaç olsun. Eğer

- Bire-Bir fonksiyon özelliği mevcut ise

  f: $V(T_1) \rightarrow V(T_2)$

  Örnek:

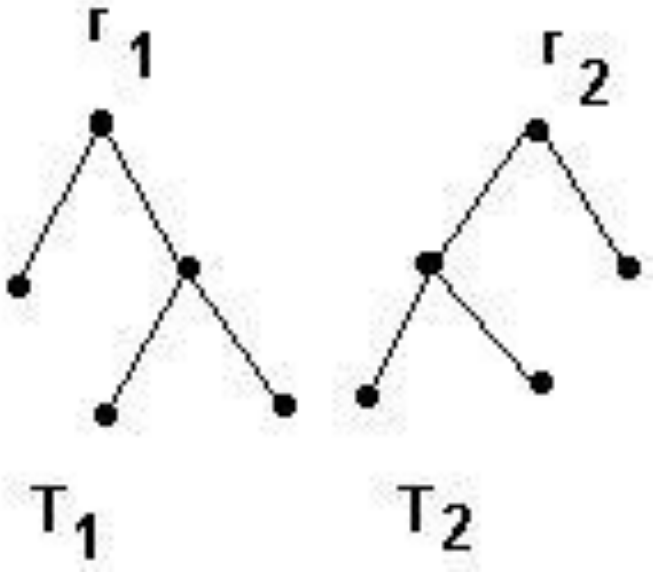
  - T_1 ve $T_{2,}$ köklü bir ağaç olarak isomorphic



Devam...

Örnek: aşağıdaki iki ağaç

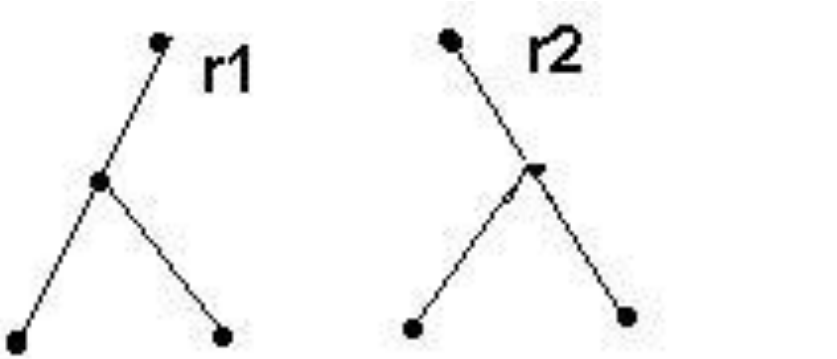
- isomorphic köklü bir ağaç olarak
- not isomorphic binary tree olarak



Özet...

Ağaçlarda 3 çeşit isomorphism vardır

- Ağaçlarda Isomorphism
- Köklü ağaçlarda Isomorphism
- Binary Tree'de Isomorphism



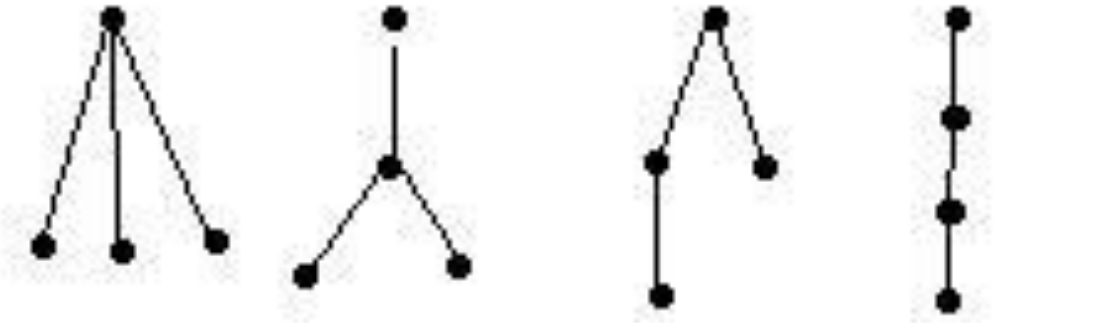
Two binary trees
isomorphic as rooted trees,
not as binary trees

Ağaçlarda Non-isomorphism

- Çoğu zaman iki ağaçda **not isomorphism** olduğunu göstermek ağaçlarda **isomorphism** olduğunu göstermekten daha kolaydır
- İki ağaç arasında isomorphism aramak için aşağıdaki şartların oluşması beklenir
  - Düğüm sayıları aynı olmalı
  - Kenar sayıları aynı olmalı
  - Karşılıklı düğümlerin dereceleri aynı olmalı
  - Kökten köke eşleme yapılmalı (rooted tree)
  - Çocukların(child-leave) pozisyonları aynı olmalı (binary tree)

Köklü Ağaçlarda Non-isomorphism

- En tepedeki düğümü kök olarak kabul edersek düğüm ve kenar sayıları eşit olsa bile bu köklü ağaçlar non-isomorphism dir
- Köklü ağaç özelliği verilmeseydi, 1 ve 2. ağaç, 3 ve 4.ağaç kendi aralarında isomorphism olacaktı



Binary Tree'lerde Non-isomorphic

Theorem

- n düğümlü, C(2n,n) / (n+1) adet non-isomorphic ikili ağaç(binary tree) vardır, n ≥ 0

5 nonisomorphic binary trees
with 3 vertices