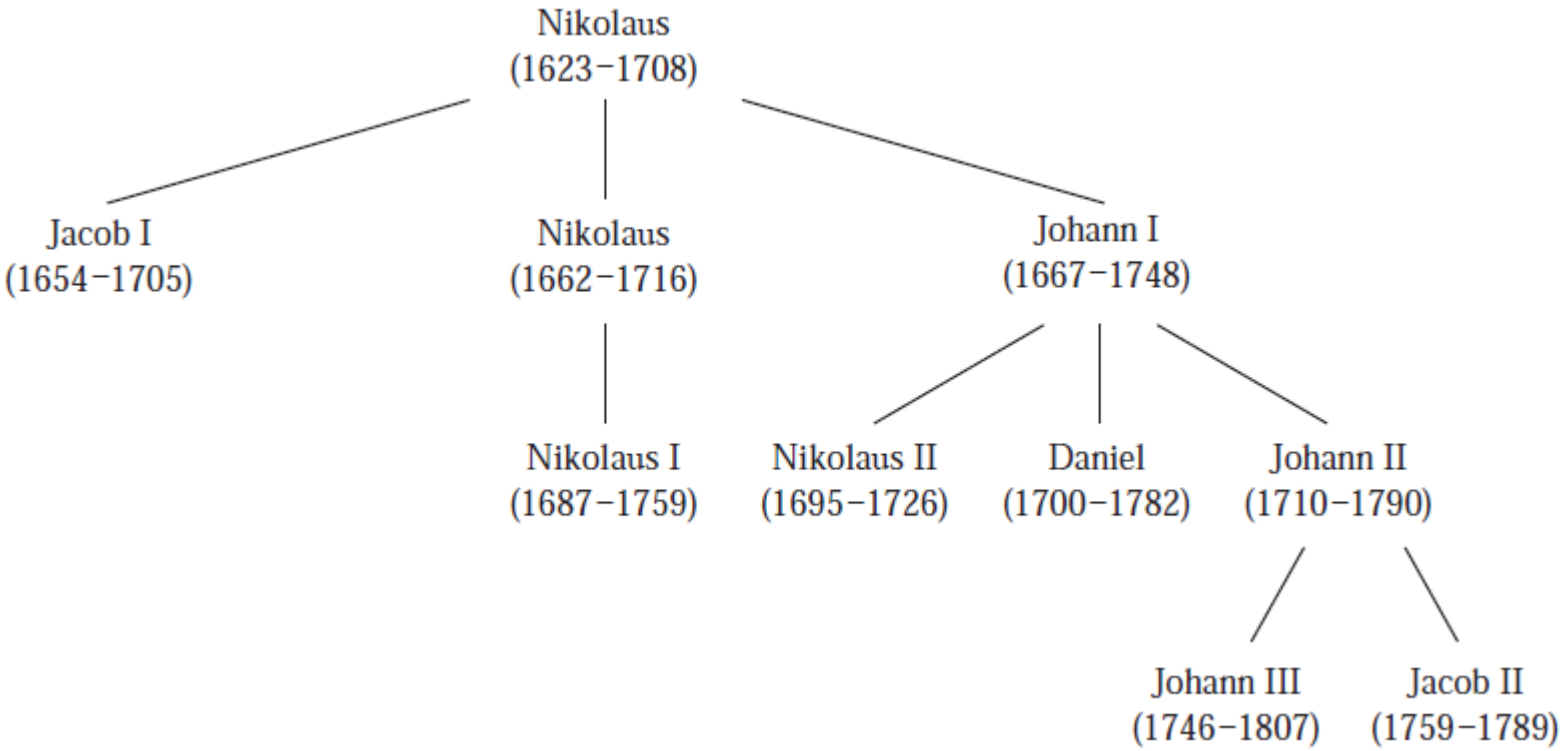
به نام خدا

ساختمان‌های گسسته

درخت

Dr. Aref Karimiafshar
A.karimiafshar@ec.iut.ac.ir

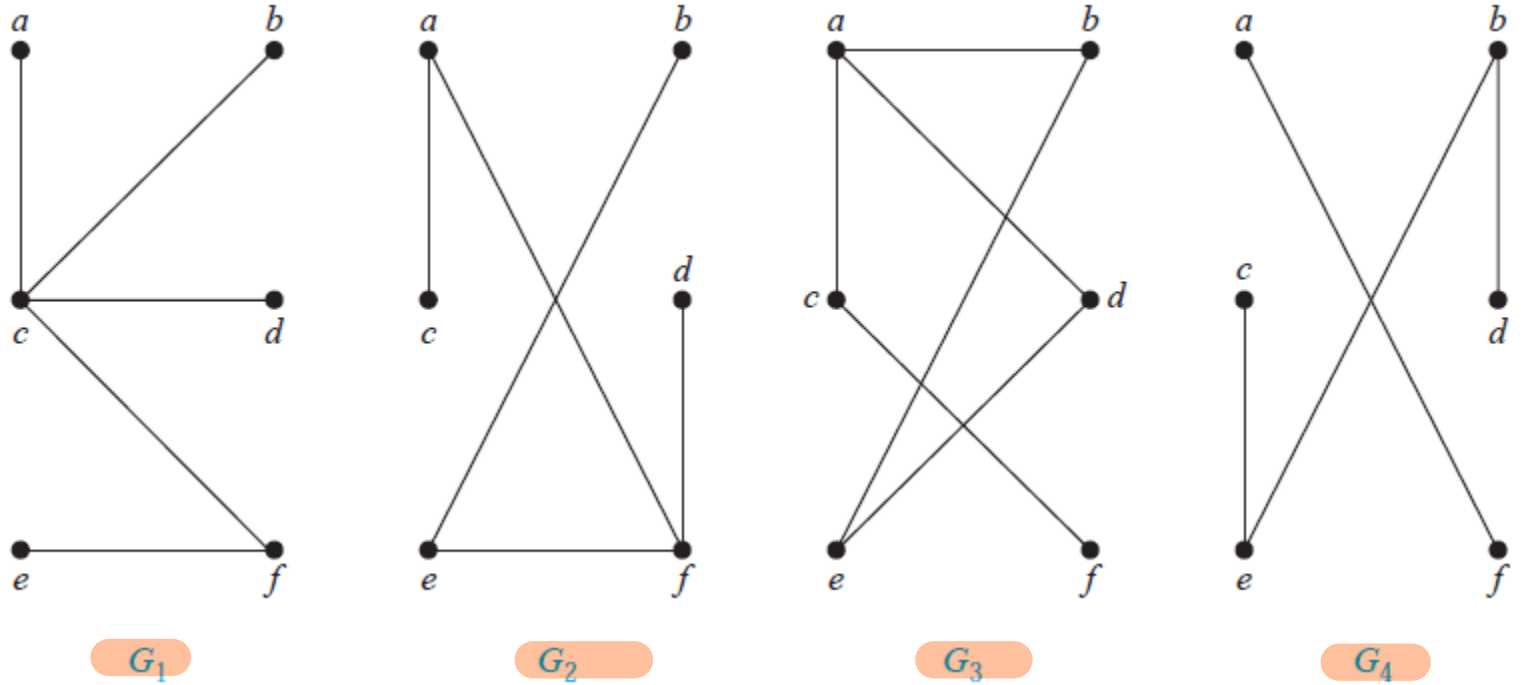
درخت


Nikolaus
(1623–1708)
Jacob I
(1654–1705)
Nikolaus
(1662–1716)
Johann I
(1667–1748)
Nikolaus I
(1687–1759)
Nikolaus II
(1695–1726)
Daniel
(1700–1782)
Johann II
(1710–1790)
Johann III
(1746–1807)
Jacob II
(1759–1789)

درخت

- تعریف:

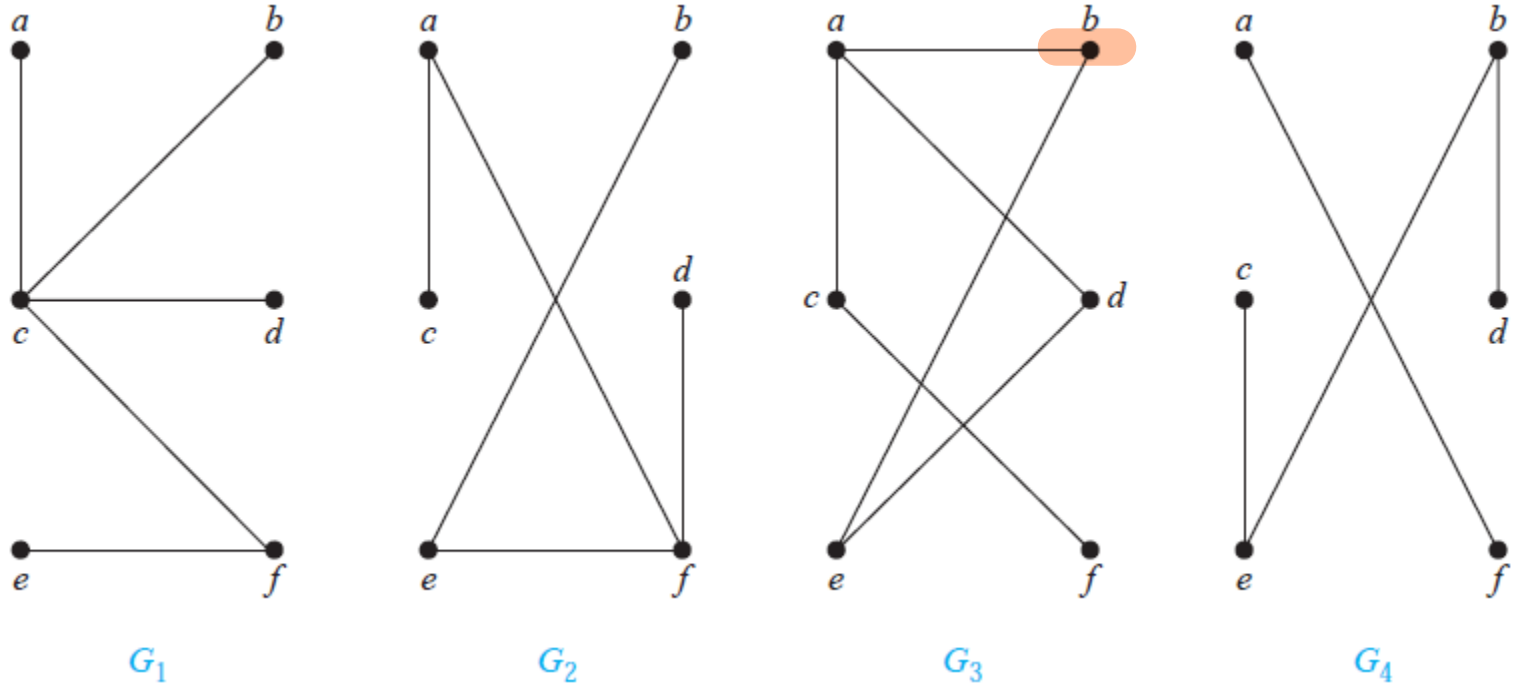
درخت گراف بدون جهت همبندی است که دارای دور نباشد.



درخت

- قضیه:

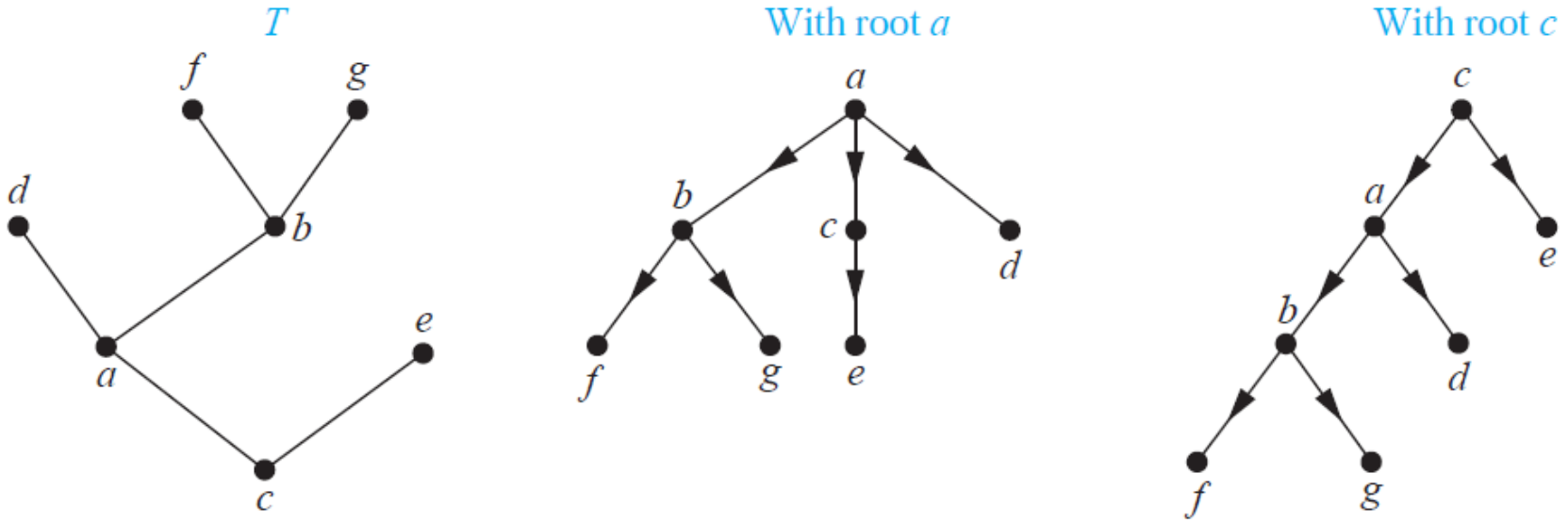
یک گراف بدون جهت درخت است اگر و فقط اگر یک مسیر ساده منحصر به فرد بین هر دو راس آن وجود داشته باشد.



درختهای ریشه‌دار

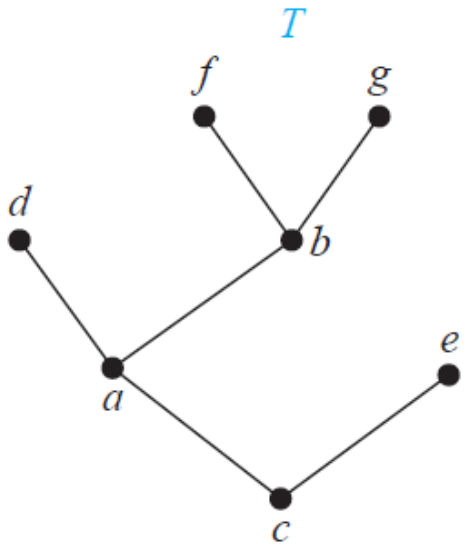
- درخت ریشه‌دار

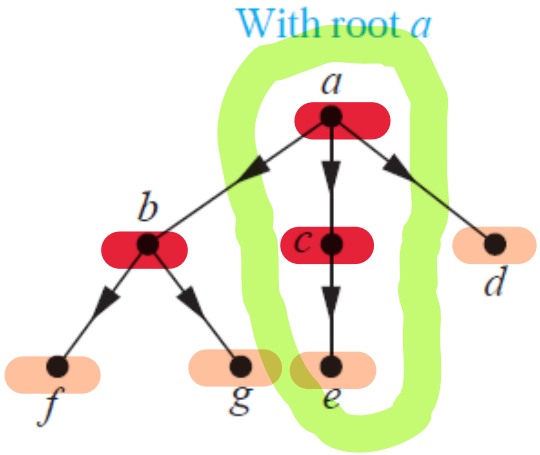
  درختی که در آن یک راس به عنوان ریشه انتخاب میشود و یالها (به سمت سایر رئوس) از آن انشعاب پیدا می کنند.

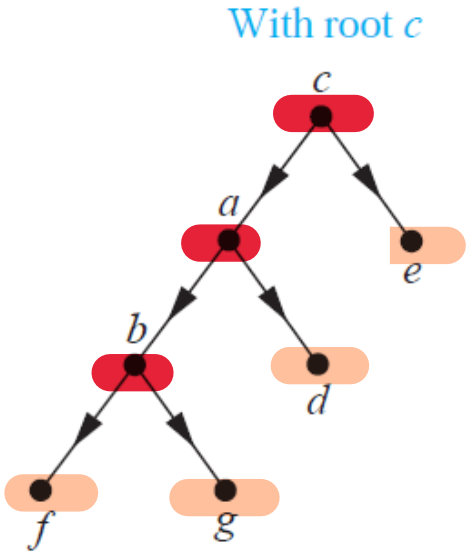


درختهای ریشه‌دار

- برخی اصطلاحات:
  - برگ
  - راس داخلی
  - فرزند
  - والد

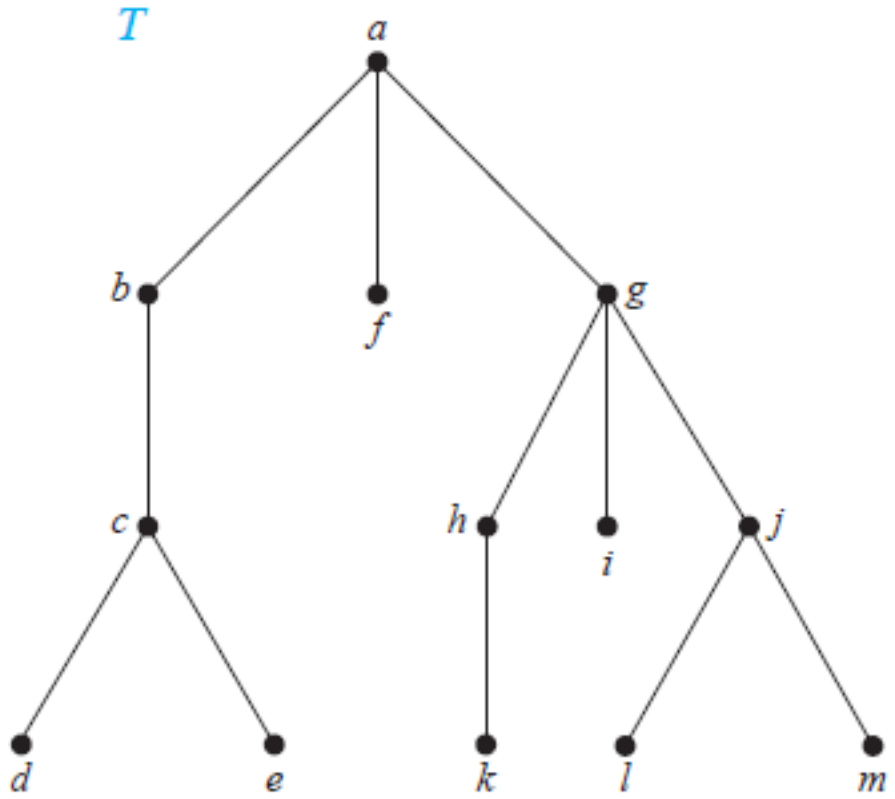


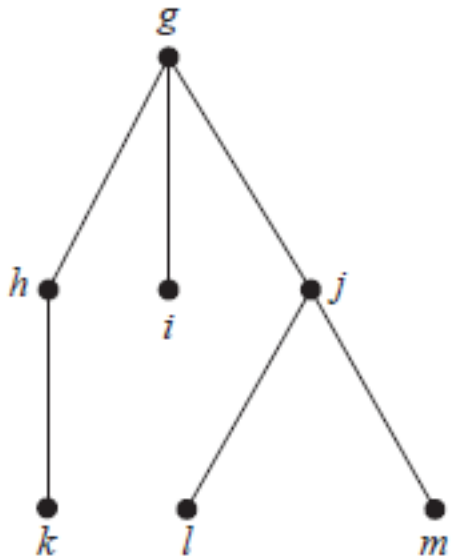




زیر درخت

- زیر درخت با یک ریشه مشخص



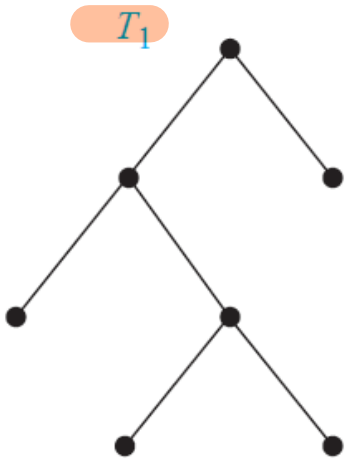


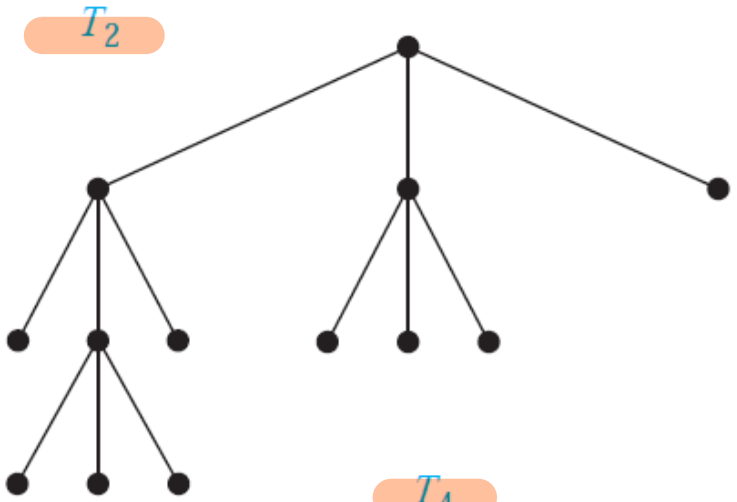
درخت m-تایی

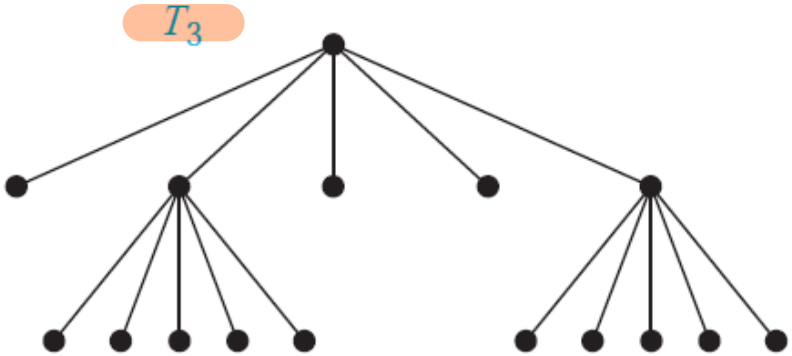
- درخت m-تایی
  - اگر هر راس داخلی آن بیشتر از m فرزند نداشته باشد
- درخت m-تایی کامل
  - اگر هر راس داخلی آن دقیقا m فرزند داشته باشد
- درخت باینری

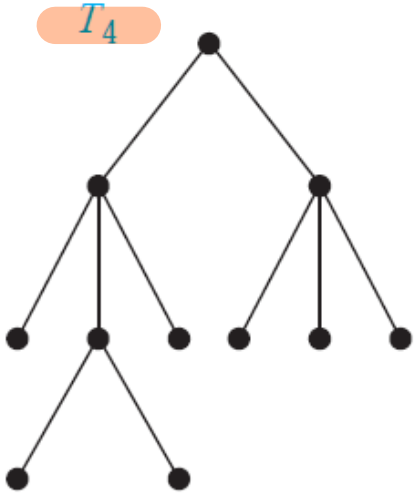
(درخت m-تایی و m=2) اگر هر راس داخلی آن دقیقا دو فرزند داشته باشد

درخت m-تایی

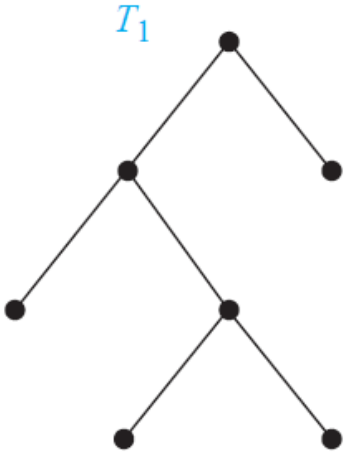






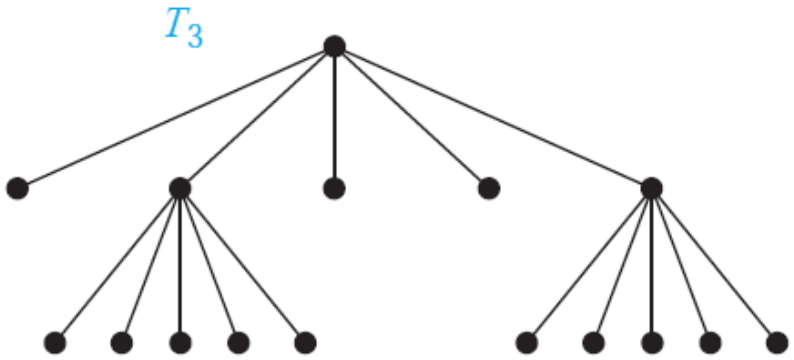


ویژگیهای درختها



- درختی با n راس دارای n-1 یال است

- یک درخت کامل m-تایی با i راس داخلی دارای n=mi+1 راس است



ویژگیهای درختها

- اگر T یک درخت کامل m-تایی باشد
  - اگر i تعداد رئوس داخلی
  - l تعداد برگها
  - n تعداد رئوس

با داشتن هر کدام از این پارامترها میتوان مقدار بقیه را تعیین کرد.

- قضیه: در یک درخت کامل m-تایی با
  - n راس، $i = (n-1)/m$ راس داخلی و $l = [(m-1)n+1]/m$ برگ داریم
  - i راس داخلی، $n = mi + 1$ راس و $l = (m-1)i + 1$ برگ داریم
  - l برگ، $n = (ml-1)/(m-1)$ راس و $i = (l-1)/(m-1)$ راس داخلی

ویژگیهای درختها

- قضیه: در یک درخت کامل m-تایی با
  - n راس، $i = (n-1)/m$ راس داخلی و $l = [(m-1)n+1]/m$ برگ داریم

- اثبات

$$n = mi + 1$$
$$n = l + i$$

$$n = mi + 1 \longrightarrow i = (n-1)/m$$

$$n = l + i \longrightarrow l = n - i = n - (n-1)/m = [(m-1)n+1]/m$$

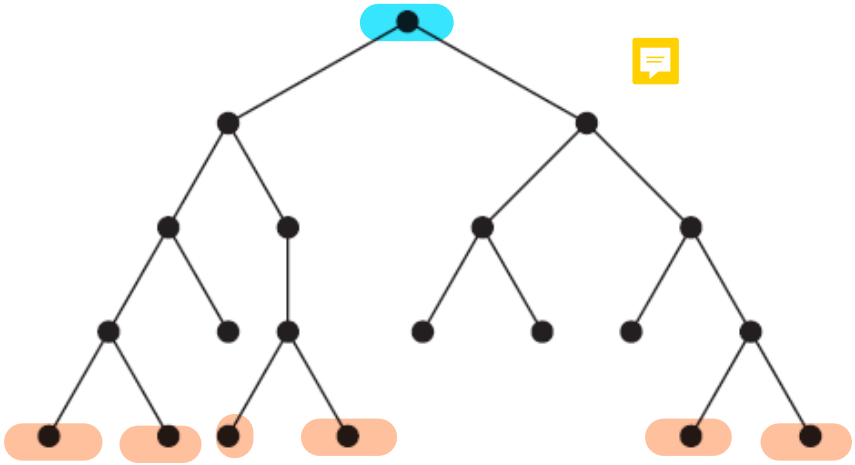
تعاریف

- سطح (در یک درخت ریشه دار):
  - طول مسیر یکتا از ریشه به راس مورد نظر
  - سطح ریشه، صفر تعریف می شود

- ارتفاع (در یک درخت ریشه دار):
  - بزرگترین طول مسیر یکتا از ریشه به تمام رئوس
  - بزرگترین سطح قابل تعریف برای رئوس

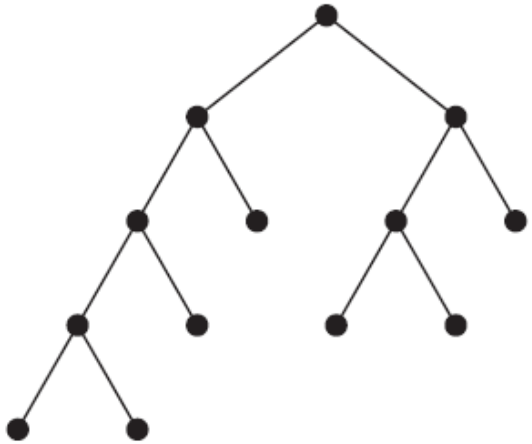
- درخت balanced:
  - یک درخت m-تایی با ارتفاع h یک درخت balanced گفته می شود اگر هر برگ آن در ارتفاع h یا h-1 باشد

مثال



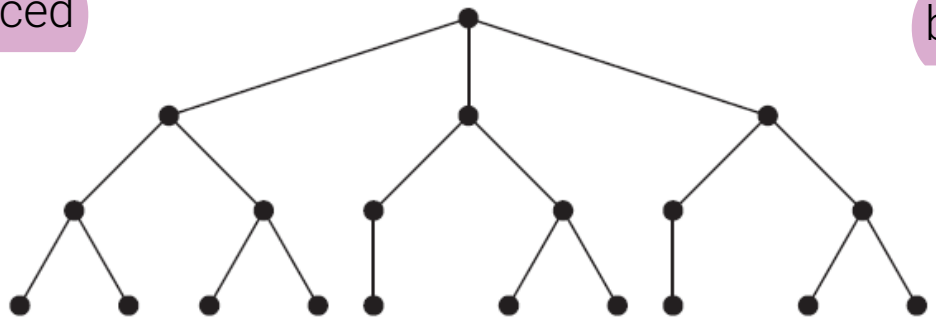
T_1

balanced



T_2

balanced



T_3

balanced

ویژگیهای درختها

- قضیه: در یک درخت m-تایی با ارتفاع h حداکثر m^h برگ داریم

- قضیه: اگر یک درخت m-تایی با ارتفاع h دارای L برگ باشد
  - آنگاه $h \geq \lceil \log_m l \rceil$
  - و اگر درخت m-تایی کامل و بالانس باشد داریم $h = \lceil \log_m l \rceil$

- اثبات

$$l \leq m^h \longrightarrow \log_m l \leq h \longrightarrow h \geq \lceil \log_m l \rceil$$

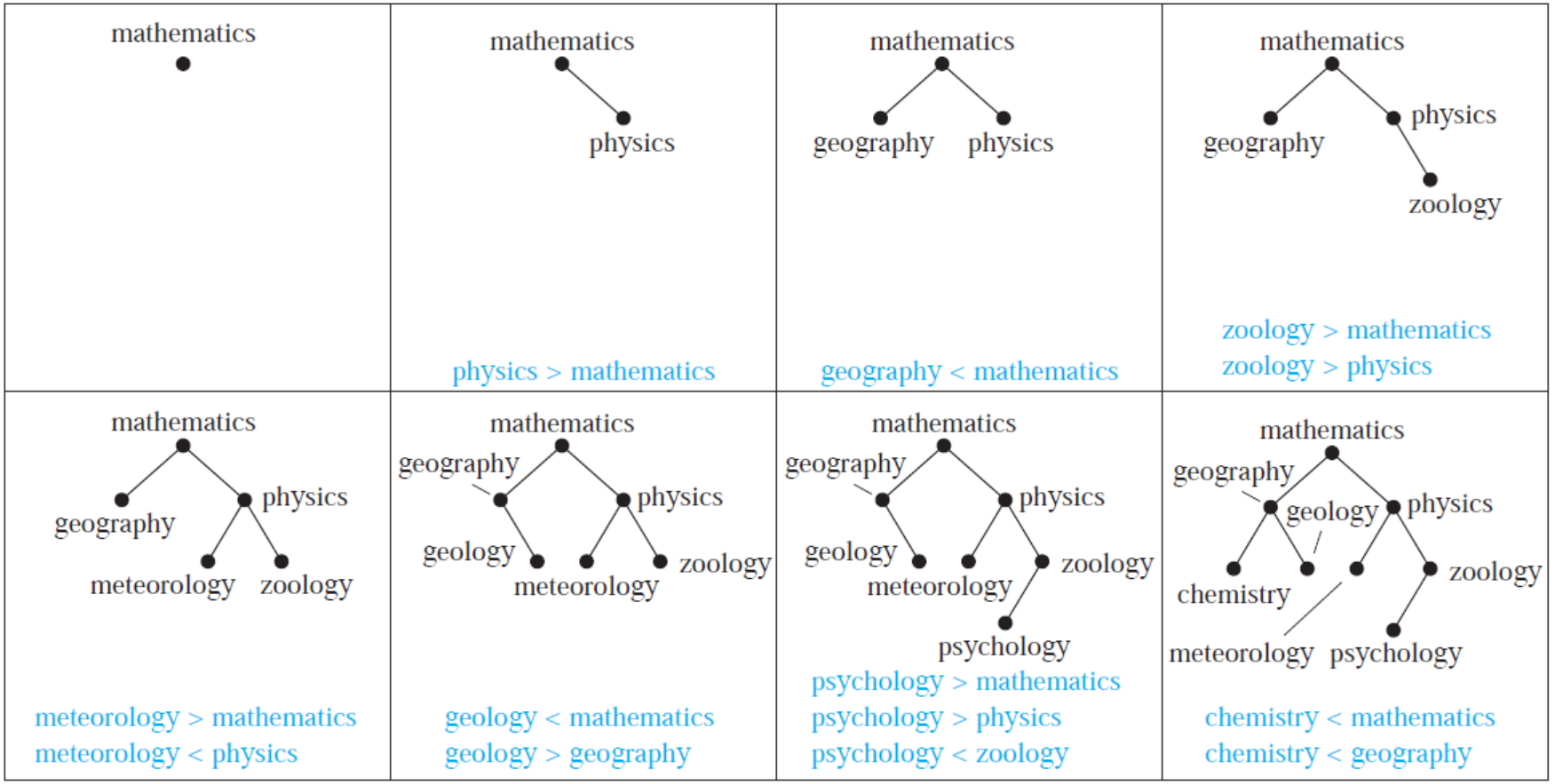
کاربردهای درختها

- درخت جست و جوی دودویی
  - یک درخت دودویی
  - هر فرزند یک راس می تواند فرزند چپ یا فرزند راست باشد
  - هر راس نمی تواند بیشتر از یک فرزند چپ یا راست داشته باشد
  - هر راس با یک کلید برچسب گذاری می شود
  - برچسب یک راس از تمامی رئوس موجود در زیر درخت سمت چپ بزرگتر و از تمام رئوس زیر درخت سمت راست کوچکتر است

- پیدا کردن یک آیتم مشخص در یک لیست مرتب

درخت جست وجوی دودویی

mathematics, physics, geography, zoology, meteorology, geology, psychology and chemistry



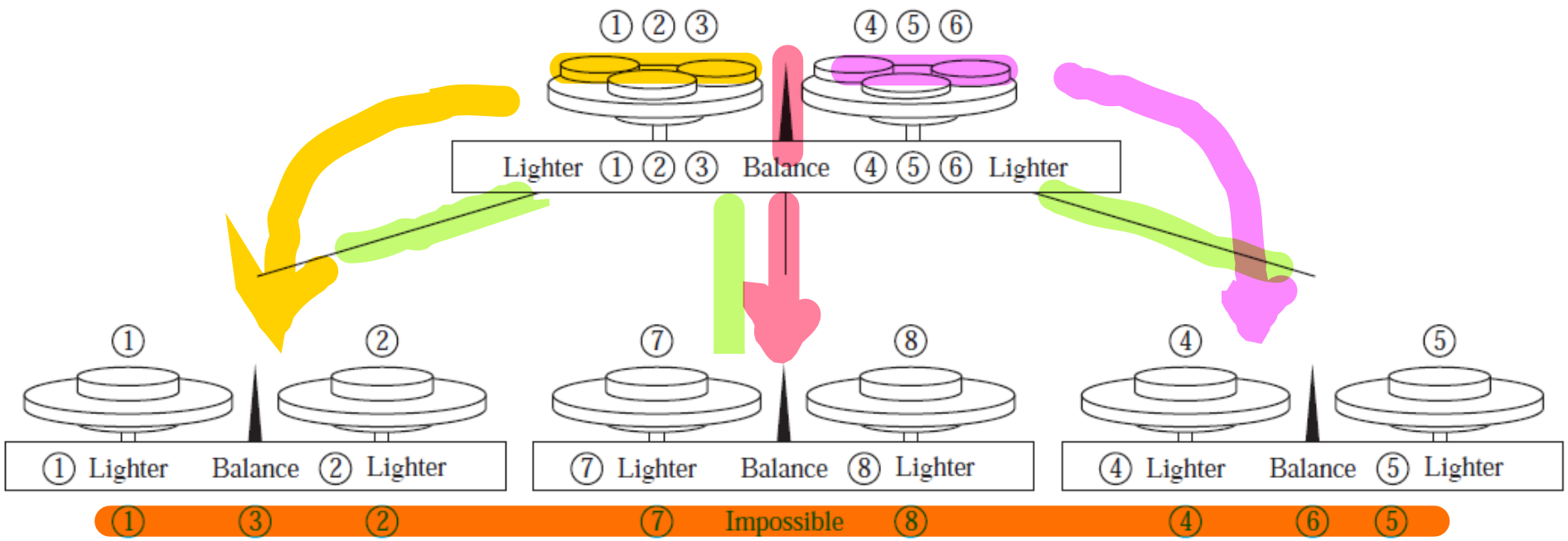
کاربردهای درختها

- درخت تصمیم
  - یک درخت ریشه دار
  - هر راس داخلی بیانگر یک تصمیم و زیر درخت آن بیان کننده خروجی های آن موقعیت تصمیم
  - مجموعه راه حلها = مسیرهای موجود به برگهای این درخت ریشه دار

- نمایش فضای تصمیم و تحلیل موقعیتهای مختلف

درخت تصمیم

- 8 سکه داریم که یکی از آنها تقلبی است. وزن سکه تقلبی کمتر از سایر سکه هاست. برای تشخیص سکه تقلبی چندبار باید وزن کنیم؟

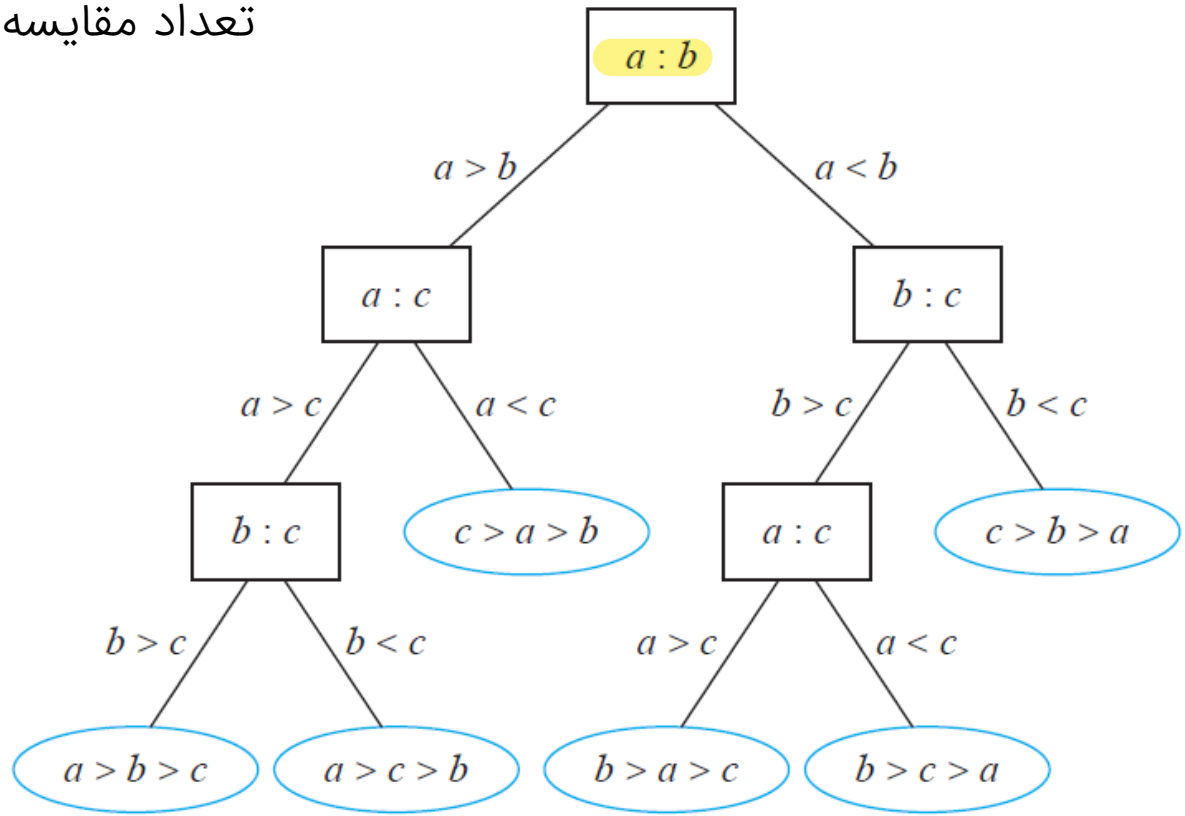


درخت تصمیم

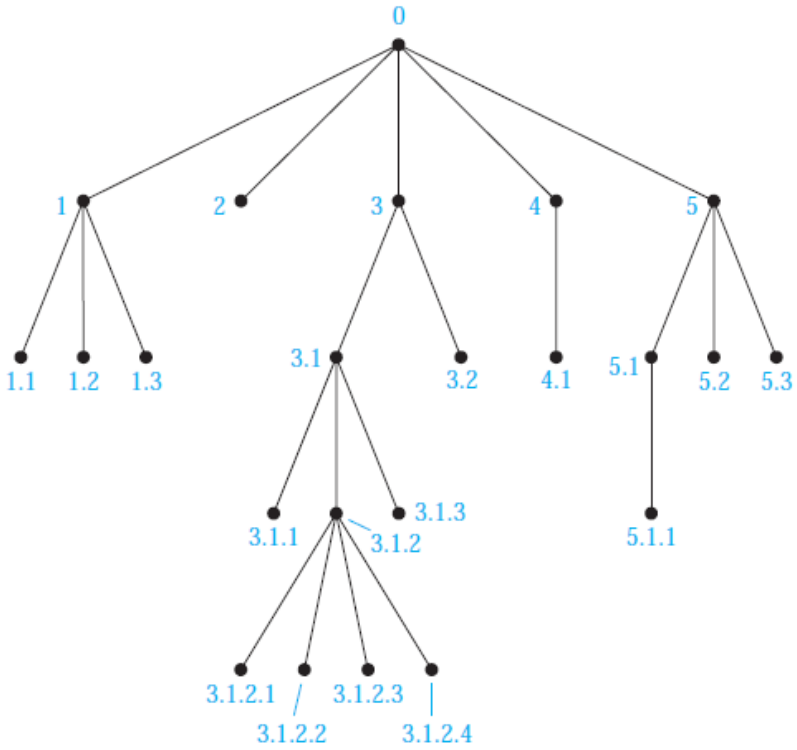
- درخت تصمیم برای مرتب کردن سه مولفه a, b, c.

تعداد مقایسه های مورد نیاز

$\lceil \log n! \rceil$



پیمایش درخت



- درخت مرتب
  - فرزندان رئوس داخلی دارای ترتیب هستند

- درختان مرتب برای ذخیره سازی اطلاعات مورد استفاده قرار می گیرند!
  - برای دستیابی به اطلاعات یک درخت مرتب به یک روش پیمایش رئوس نیاز داریم

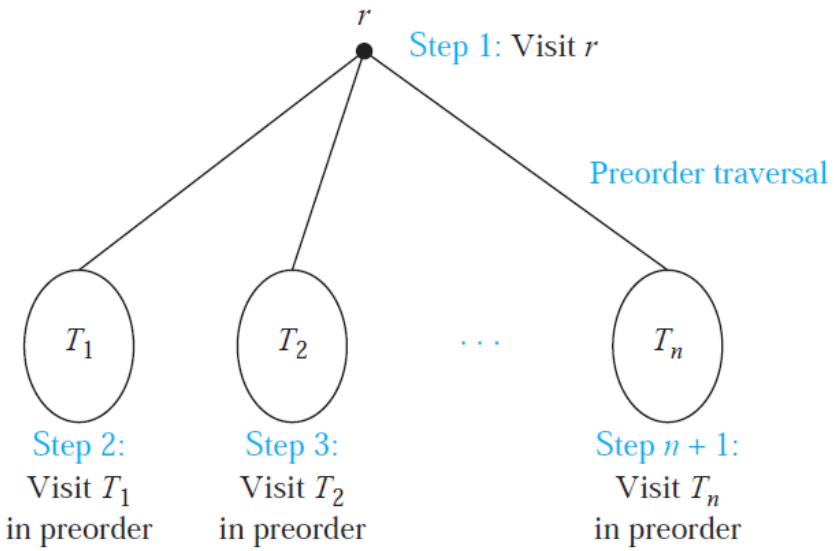
الگوریتم‌های پیمایش

- الگوریتم‌های پیمایش
  - رویه‌های مشاهده سیستماتیک هر راس در یک درخت مرتب!

- انواع الگوریتم پیمایش
  - پیش ترتیب (preorder)
  - میان ترتیب (inorder)
  - پس ترتیب (postorder)

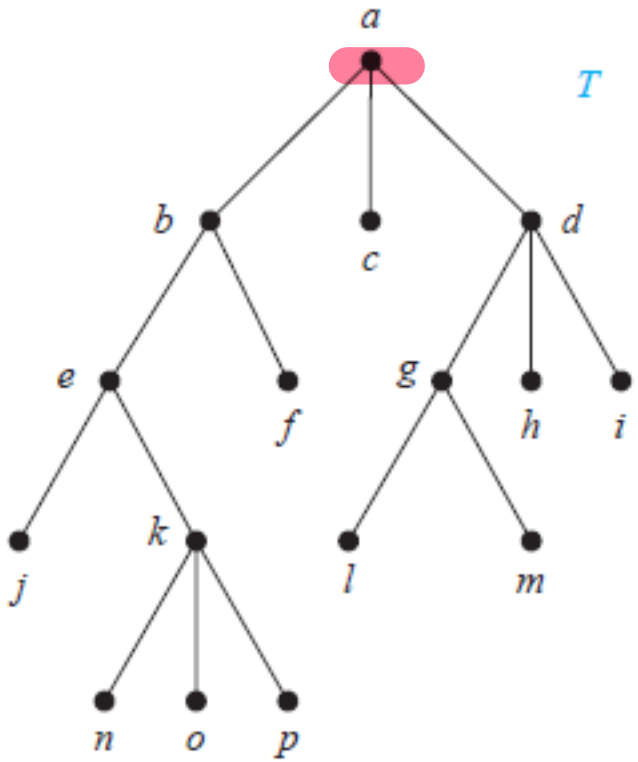
پیمایش پیش ترتیب

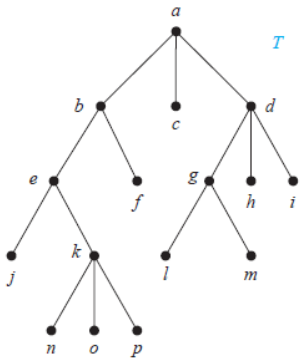
- اگر T یک درخت ریشه‌دار مرتب با ریشه r باشد
  - اگر T فقط از r تشکیل شده باشد
    - آنگاه r پیمایش پیش ترتیب T خواهد بود
  - در غیر این صورت، اگر $T_1, T_2, \ldots, T_n$ زیر درختهای T در راس r از چپ به راست باشند
    - آنگاه پیمایش پیش ترتیب T با r شروع می شود و بعد با پیمایش پیش ترتیب درخت T_1 و بعد T_2 تا T_n ادامه پیدا می کند.



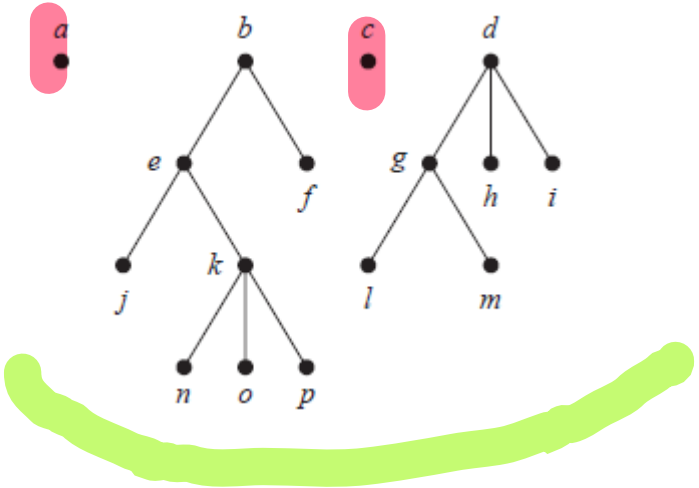
مثال

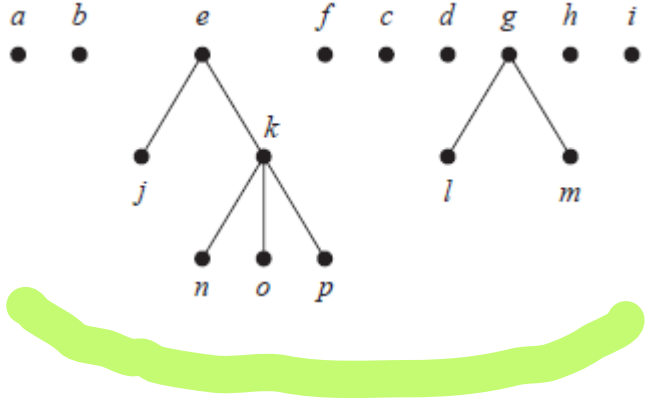
- پیمایش پیش ترتیب درخت T را بیاید.

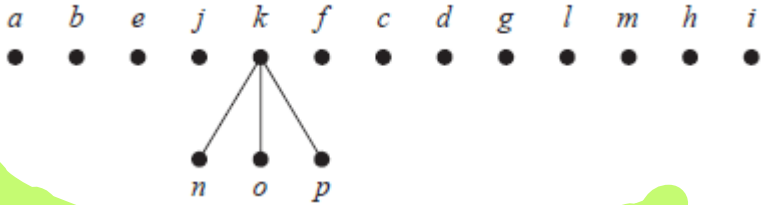


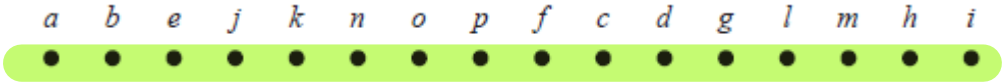

T
a
b
c
d
e
f
g
h
i
j
k
l
m
n
o
p

مثال (ادامه ...)


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

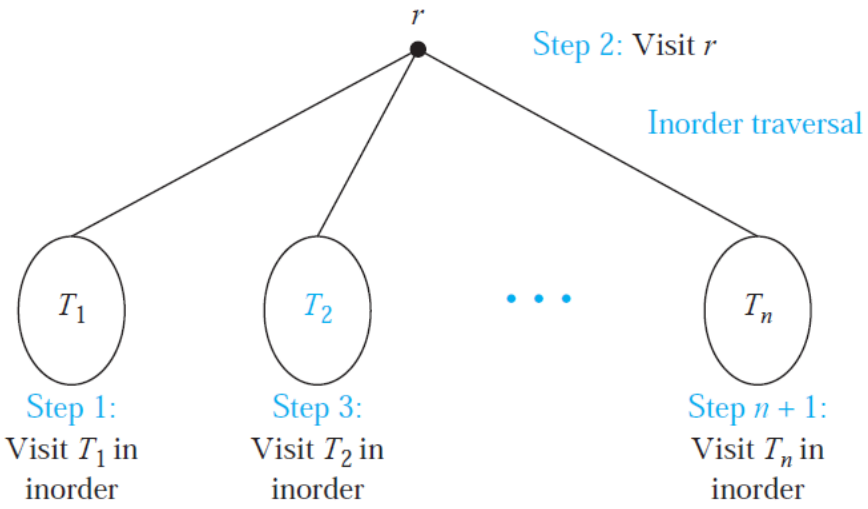

a
b
e
f
c
d
g
h
i
j
k
l
m
n
o
p


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


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

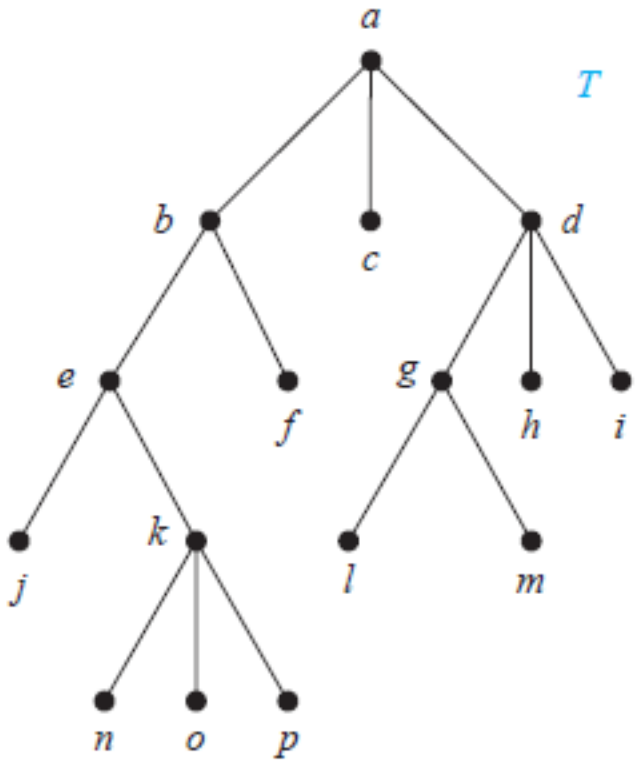
پیمایش میان ترتیب

- اگر T یک درخت ریشه‌دار مرتب با ریشه r باشد
  - اگر T فقط از r تشکیل شده باشد
    - آنگاه r پیمایش میان ترتیب T خواهد بود
  - در غیر این صورت، اگر $T_1, T_2, \ldots, T_n$ زیر درختهای T در راس r از چپ به راست باشند
    - آنگاه پیمایش میان ترتیب T با پیمایش میان ترتیب درخت T_1 و بعد مشاهده ریشه شروع می شود و سپس با پیمایش میان ترتیب درخت T_2 و بعد T_3 تا T_n ادامه پیدا می کند.

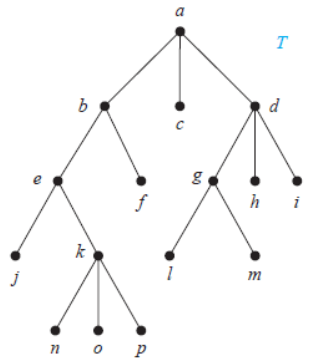


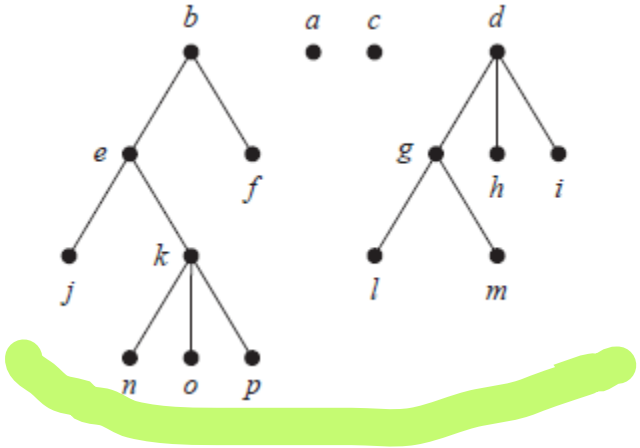
مثال

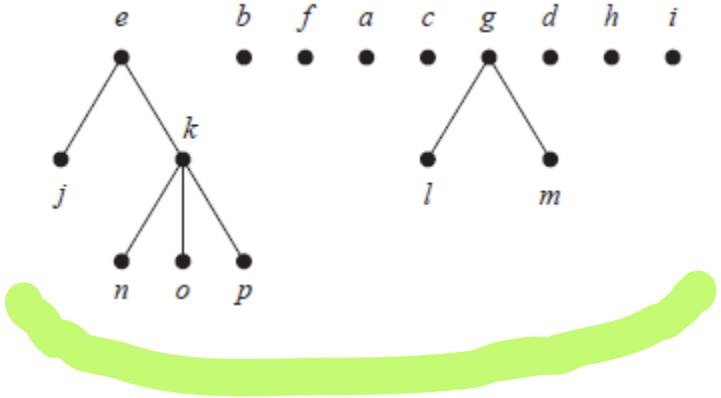
- پیمایش میان ترتیب درخت T را بیاید.

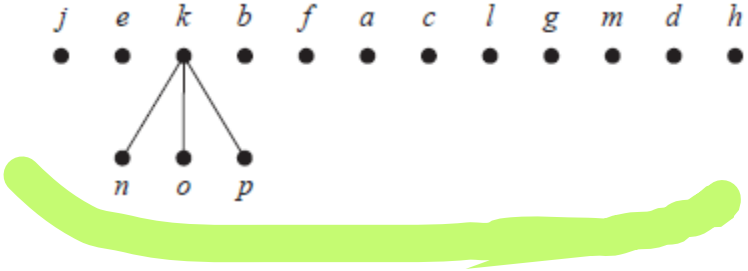


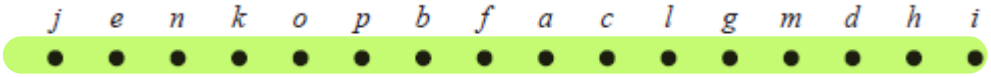
مثال (ادامه ...)


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


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

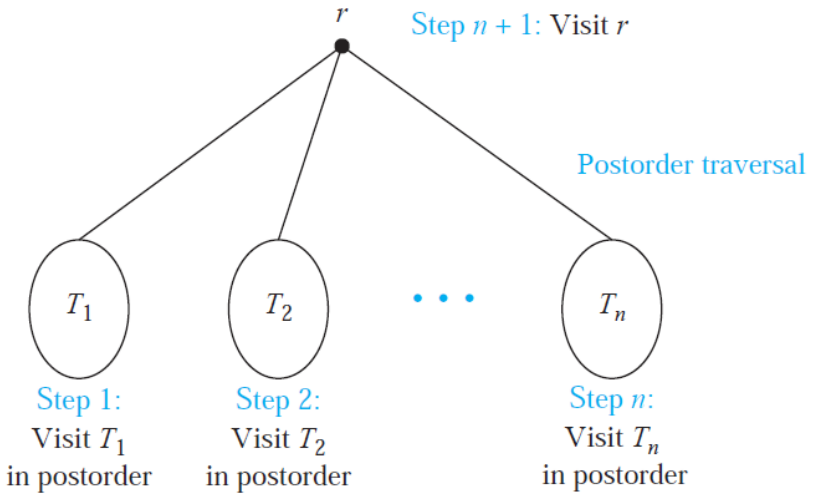

e
b
f
a
c
g
d
h
i
k
j
l
m
n
o
p


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


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

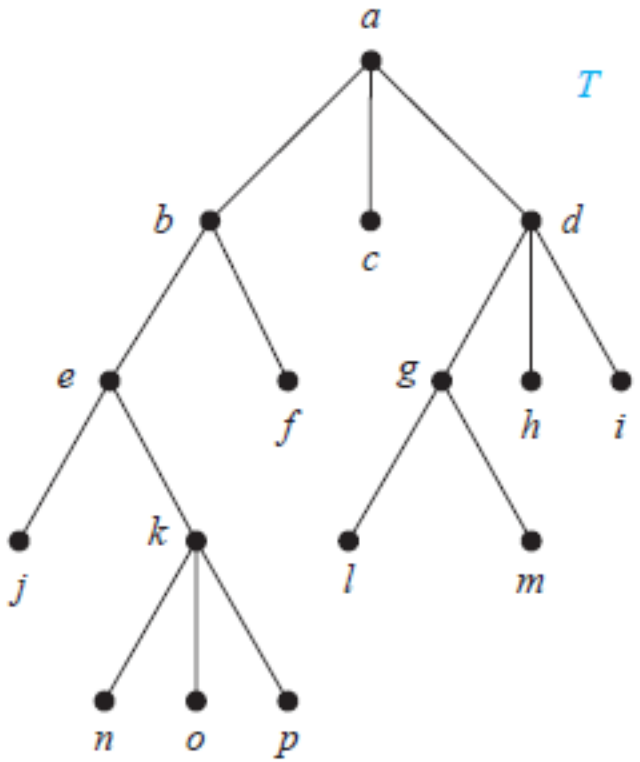
پیمایش پس ترتیب

- اگر T یک درخت ریشه‌دار مرتب با ریشه r باشد
  - اگر T فقط از r تشکیل شده باشد
    - آنگاه r پیمایش پس ترتیب T خواهد بود
  - در غیر این صورت، اگر $T_1, T_2, \ldots, T_n$ زیر درختهای T در راس r از چپ به راست باشند
    - آنگاه پیمایش پس ترتیب T با پیمایش پس ترتیب درخت T_1 و بعد T_2 و بعد T_3 تا T_n ادامه پیدا می کند و با مشاهده r خاتمه می یابد.

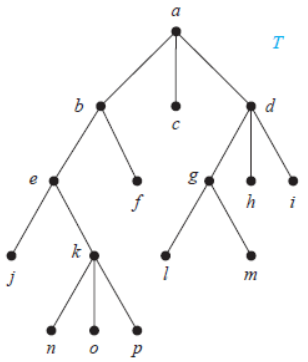


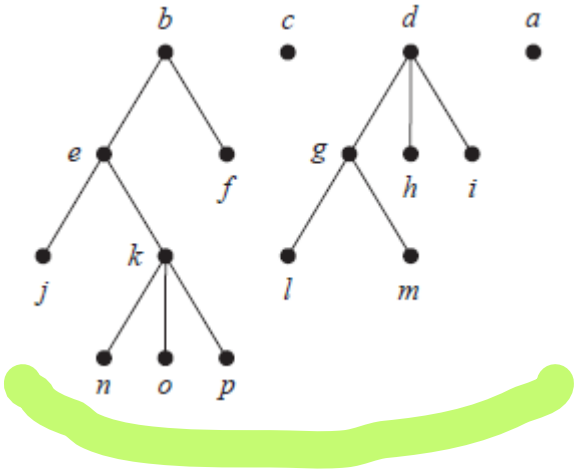
مثال

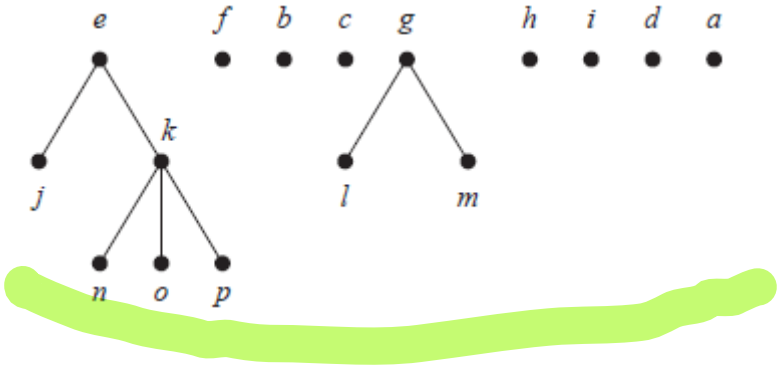
- پیمایش پس ترتیب درخت T را بیاید.

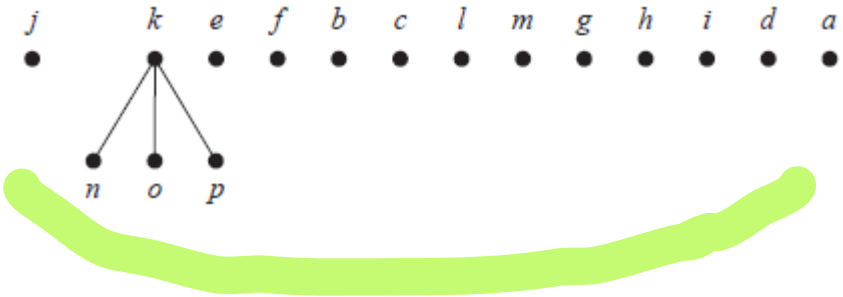


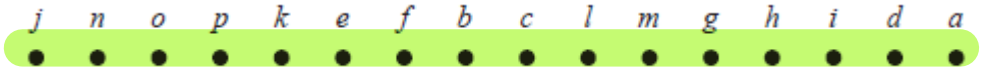
مثال (ادامه ...)


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


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

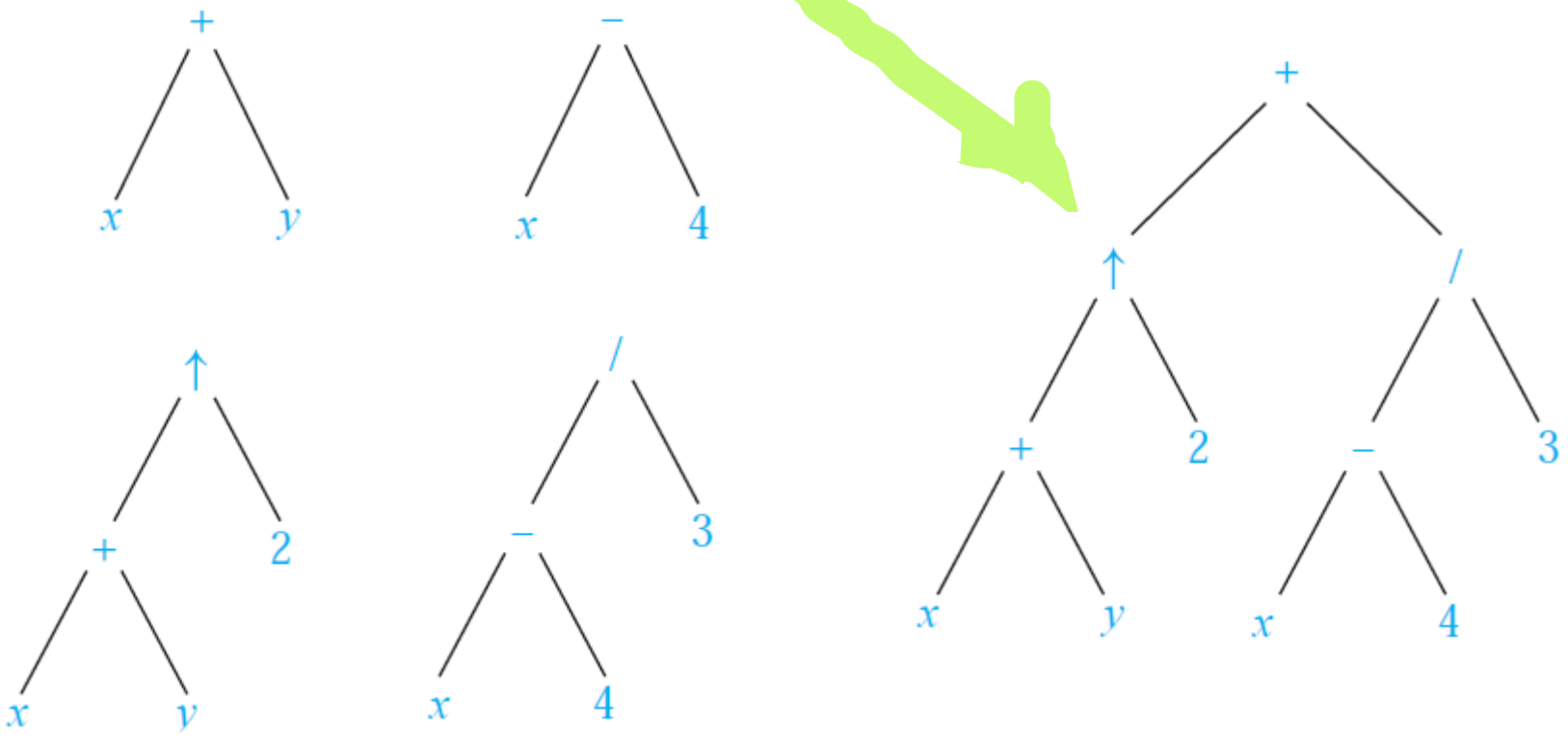

e
f
b
c
g
h
i
d
a
k
j
l
m
n
o
p


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


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

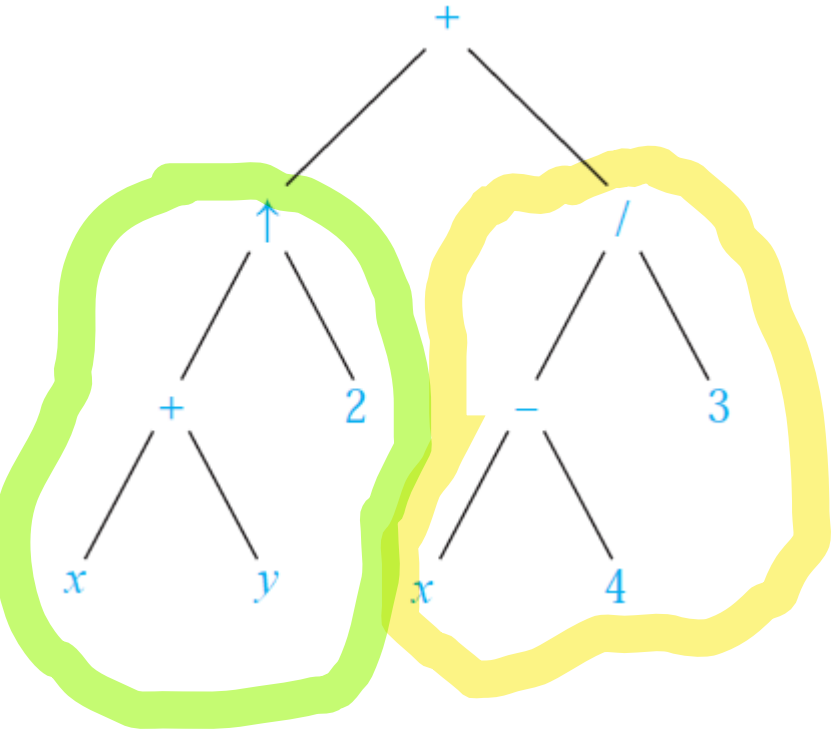
نمایش درختی عبارات

- درخت ریشه‌دار مرتب مرتبط با عبارت $((x + y) \uparrow 2) + ((x - 4)/3)$



نمایش پیشوندی

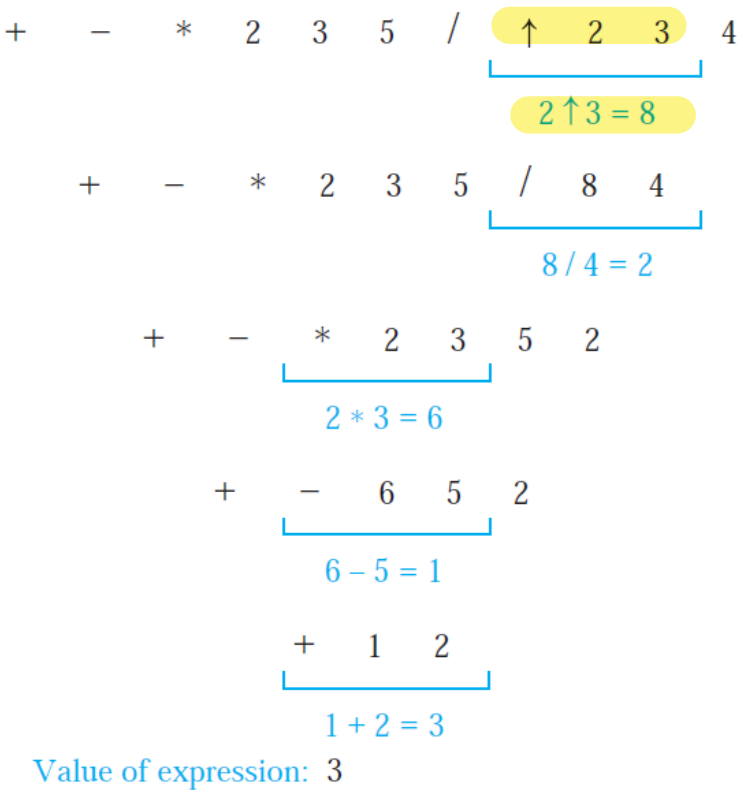
- نمایش پیشوندی عبارت $((x + y) \uparrow 2) + ((x - 4)/3)$



$+ \uparrow + x\ y\ 2\ / - x\ 4\ 3$

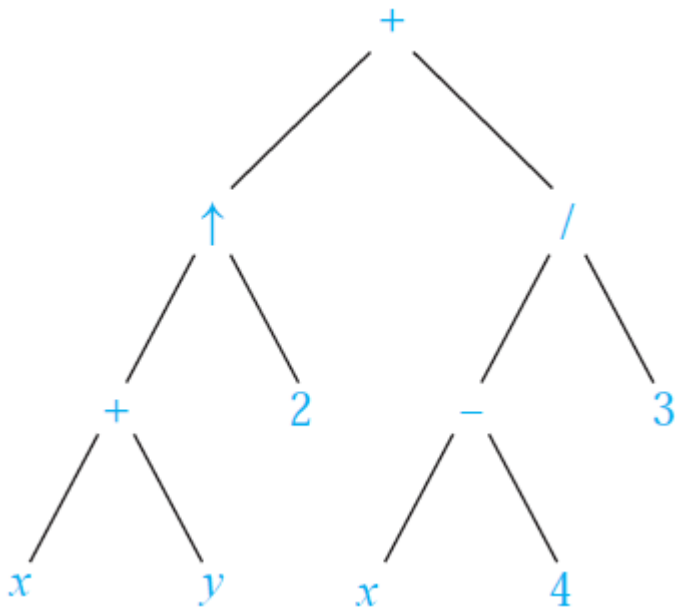
نمایش پیشوندی

- مقدار عبارت پیشوندی $+\ -\ *\ 2\ 3\ 5\ /\ \uparrow\ 2\ 3\ 4$ را حساب کنید



نمایش پسوندی

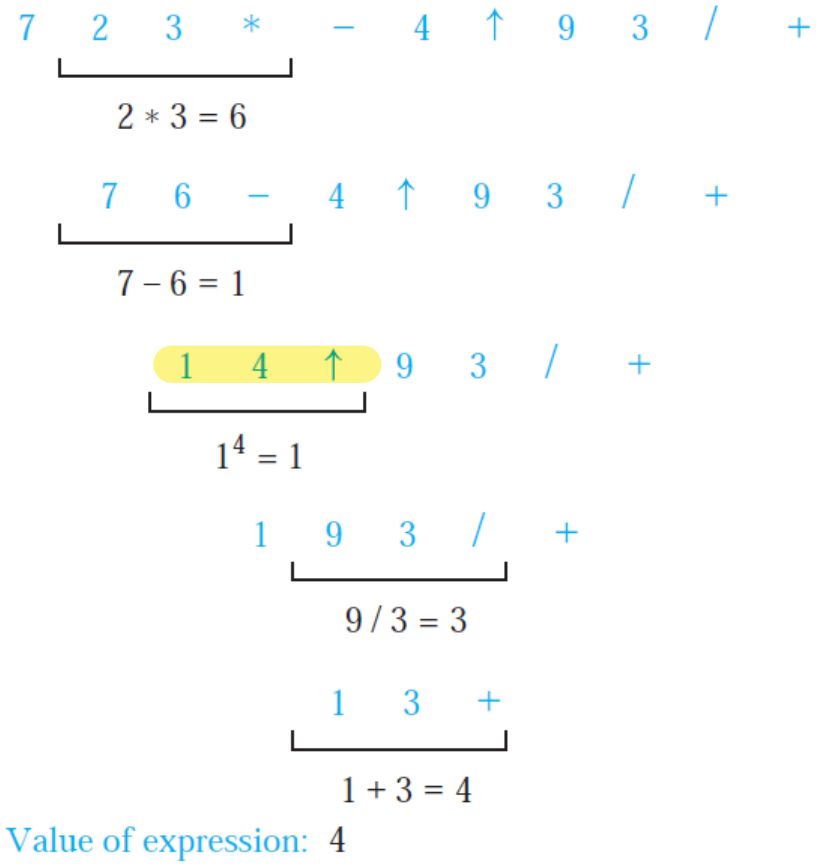
- نمایش پسوندی عبارت $((x + y) \uparrow 2) + ((x - 4)/3)$



$x\,y + 2 \uparrow x\,4 - 3 / +$

نمایش پسوندی

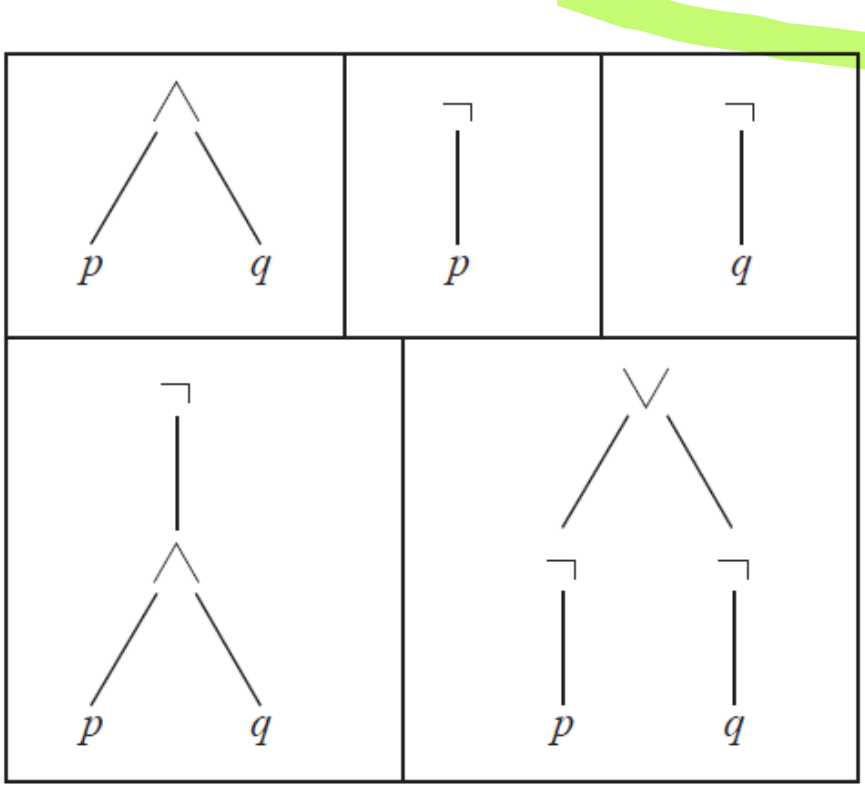
- مقدار عبارت پسوندی $7\ 2\ 3\ *\ -\ 4\ \uparrow\ 9\ 3\ /\ +$ را حساب کنید

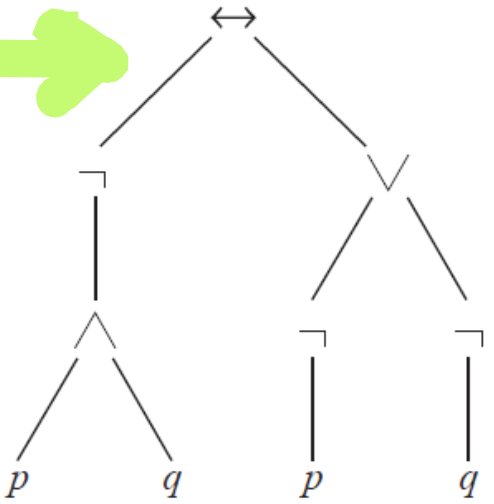


نمایش پسوندی

- درخت ریشه دار مرتب نمایش دهنده گزاره مرکب زیر را بیابید.

$(\neg(p \wedge q)) \leftrightarrow (\neg p \vee \neg q)$



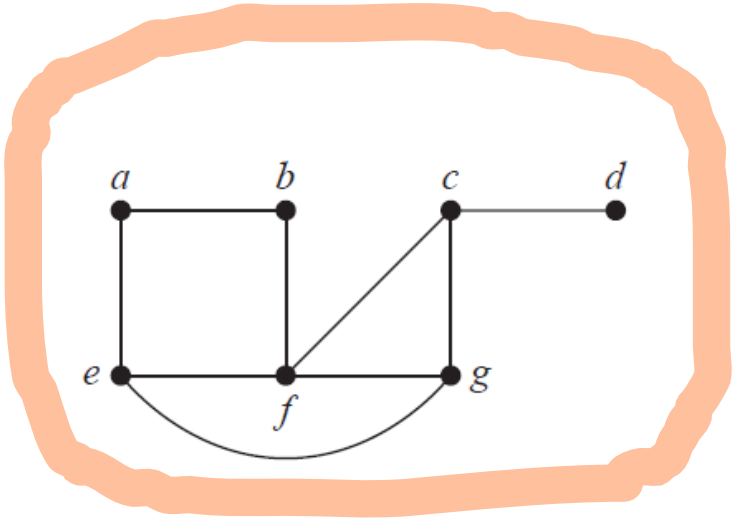


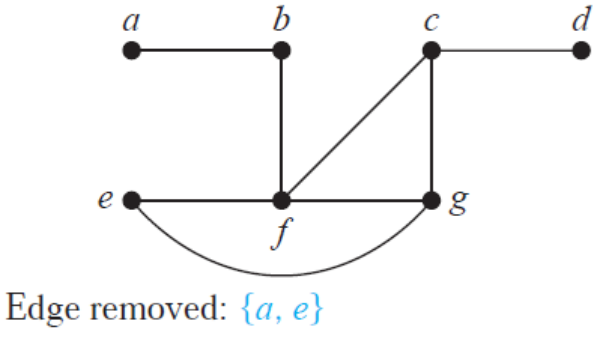
پیش ترتیب $\leftrightarrow \neg \wedge pq \vee \neg p \neg q$

پس ترتیب $pq \wedge \neg p \neg q \neg \vee \leftrightarrow$

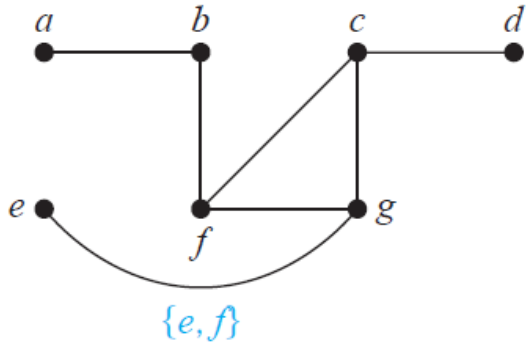
درخت فراگیر

- درخت فراگیر (Spanning Tree)
  - درخت باشد! (همبند و فاقد دور)
  - همه رئوس را شامل شود
- مثال

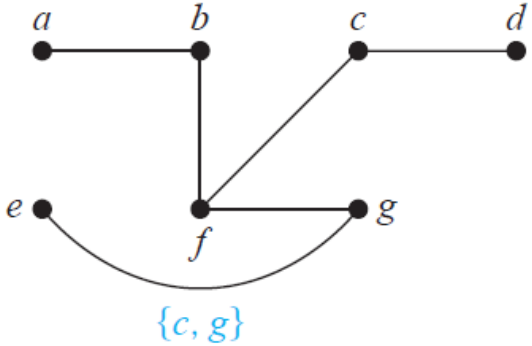




(a)

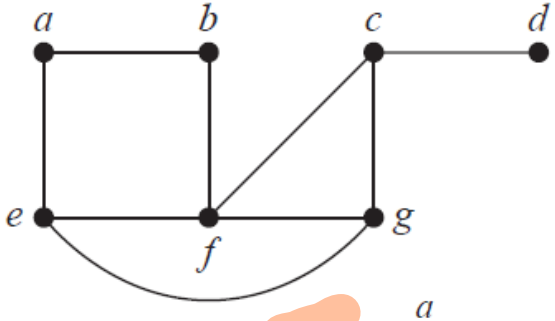


(b)

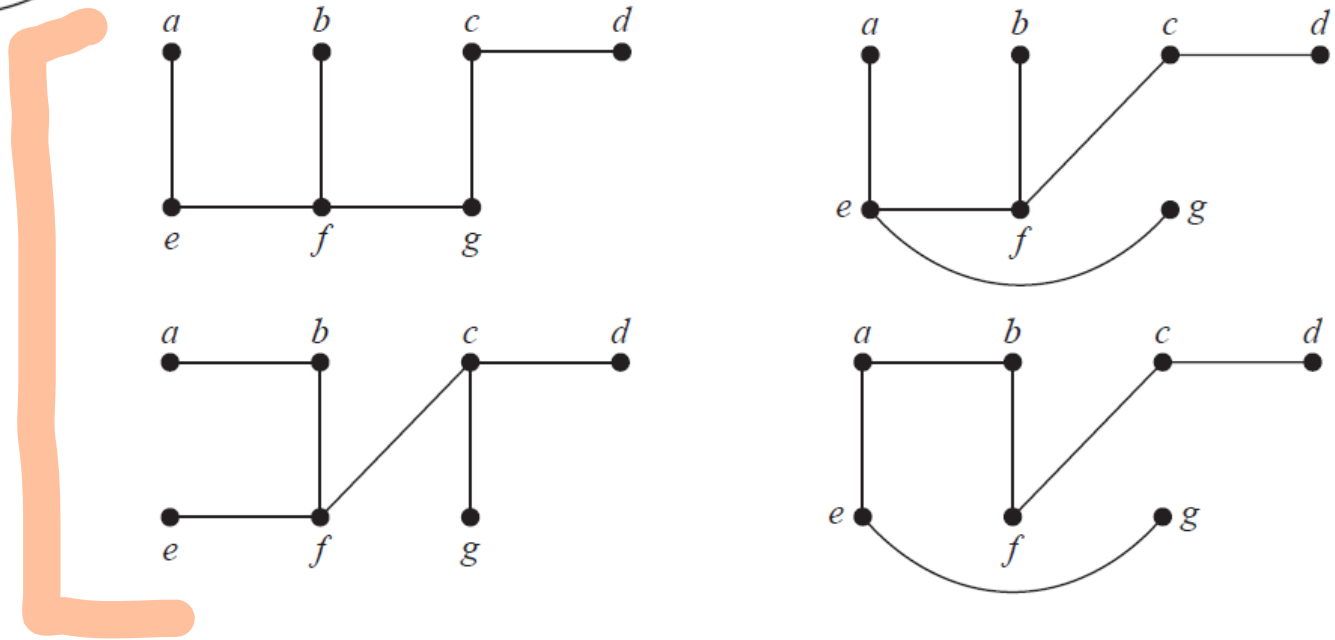


(c)

درخت فراگیر



- مثال



- قضیه: یک گراف درخت فراگیر دارد اگر و تنها اگر همبند باشد.

الگوریتم‌های پیدا کردن درخت فراگیر

- الگوریتم پیدا کردن درخت فراگیر
  - ورودی: یک گراف همبند
  - خروجی: درخت فراگیر

- الگوریتم‌های پیدا کردن درخت فراگیر
  - Depth-First Search (DFS)
  - Breadth-First Search (BFS)

الگوریتم DFS

1. به صورت دلخواه یک راس از گراف را به عنوان ریشه انتخاب کنید
2. یک راس جدید انتخاب کنید و یال واقع بر آن و راس قبلی را پیمایش کنید، این مسیر را تا زمانی که راس جدیدی به مسیر اضافه می شود ادامه دهید. اگر همه رئوس پوشش داده شد الگوریتم خاتمه می یابد.
3. در غیر این صورت به راس قبلی بر گردید و یک راس جدید را انتخاب کنید. اگر چنین راسی پیدا نکردید به راس قبل از آن بر گردید و این کار را ادامه دهید تا راس جدیدی به مسیر اضافه شود.
4. دو گام فوق را تا مشاهده همه رئوس ادامه دهید.

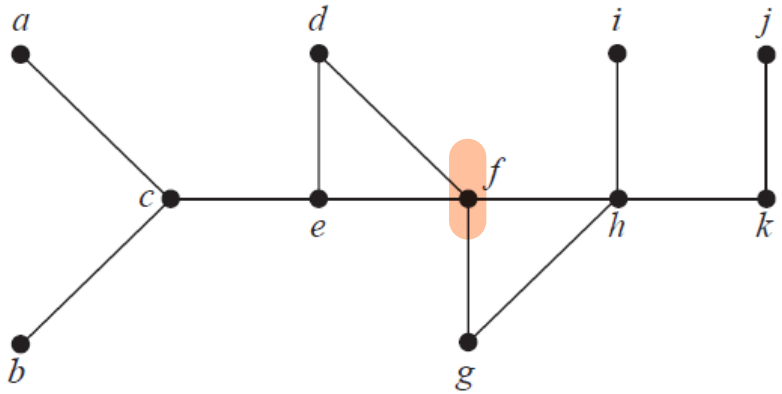
ALGORITHM 1 Depth-First Search.

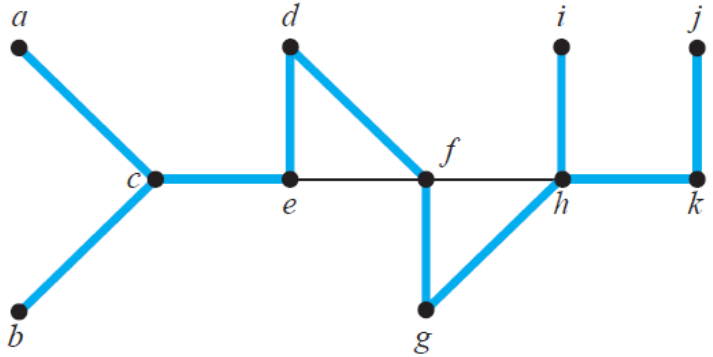
procedure *DFS*(G: connected graph with vertices $v_1, v_2, \ldots, v_n$)
$T :=$ tree consisting only of the vertex v_1
visit(v_1)

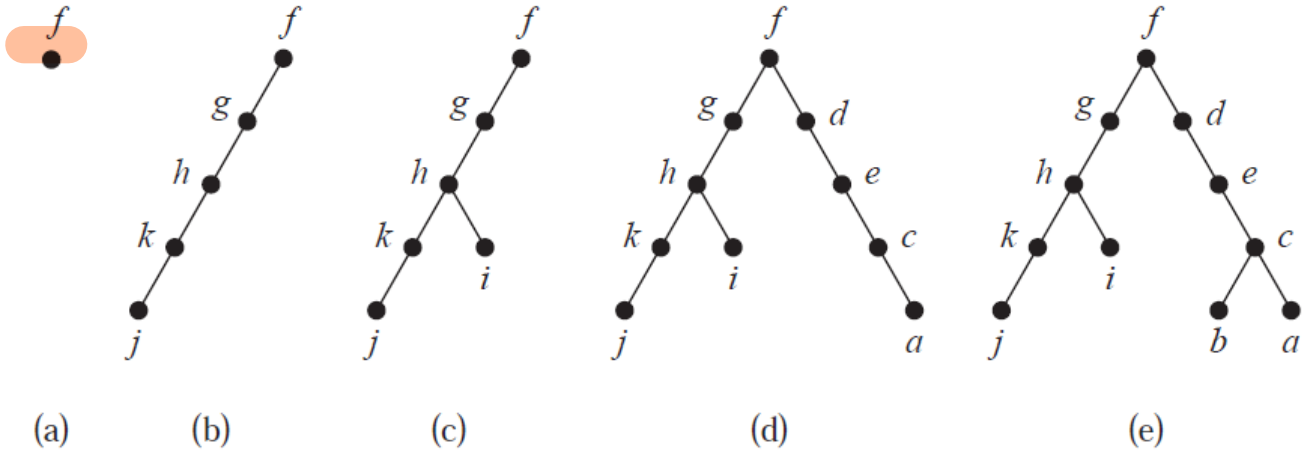
procedure *visit*(v: vertex of G)
for each vertex w adjacent to v and not yet in T
  add vertex w and edge $\{v, w\}$ to T
  visit(w)

الگوریتم DFS

مثال: درخت فراگیر گراف زیر را رسم کنید







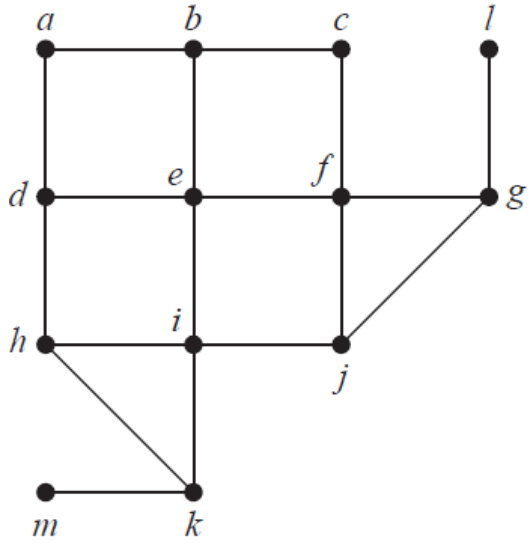
الگوریتم BFS

1. به صورت دلخواه یک راس از گراف را به عنوان ریشه انتخاب کنید
2. همه یالهای واقع بر آن را به گراف اضافه کنید
3. رئوسی که در این مرحله به درخت اضافه می شوند رئوس سطح یک را تشکیل می دهند.
4. یالهای واقع بر رئوس جدید را به درخت اضافه کنید. رئوس جدید را به درخت اضافه کنید، اینها رئوس سطوح بعدی هستند. این کار تا مشاهده همه رئوس ادامه دهید.

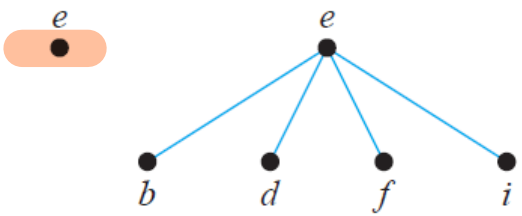
ALGORITHM 2 Breadth-First Search.

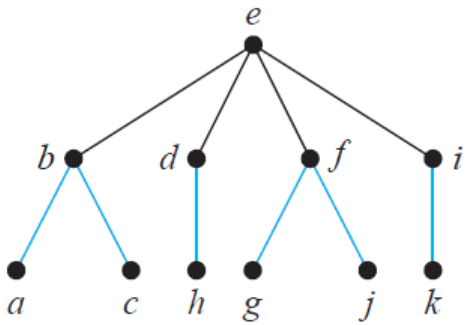
procedure *BFS* (G: connected graph with vertices $v_1, v_2, \ldots, v_n$)
T := tree consisting only of vertex v_1
L := empty list
put v_1 in the list L of unprocessed vertices
while L is not empty
  remove the first vertex, v, from L
  for each neighbor w of v
    if w is not in L and not in T **then**
      add w to the end of the list L
      add w and edge $\{v, w\}$ to T

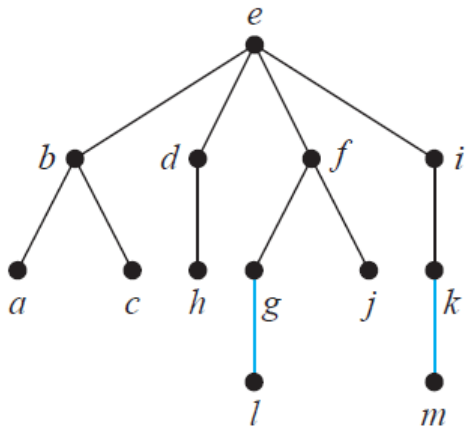
الگوریتم BFS



مثال: درخت فراگیر گراف زیر را رسم کنید







درخت فراگیر مینیمم

- درخت فراگیر مینیمم (Minimum Spanning Tree) در یک گراف وزن‌دار همبند، درخت فراگیری است که کمترین مجموع وزن یالها را داشته باشد.

- الگوریتمهای پیدا کردن درخت فراگیر مینیمم
  - الگوریتم پرایم (Prim)
  - الگوریتم کراسکال (Kruskal)

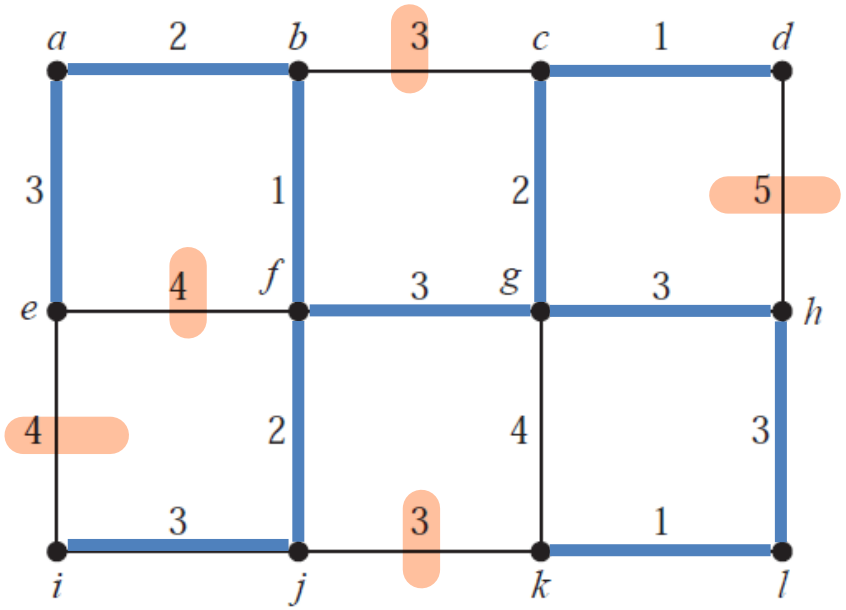
الگوریتم پرایم

ALGORITHM 1 Prim's Algorithm.

procedure *Prim*(G: weighted connected undirected graph with n vertices)
T := a minimum-weight edge
for $i := 1$ **to** $n - 2$
    e := an edge of minimum weight incident to a vertex in T and not forming a simple circuit in T if added to T
    $T := T$ with e added
return T {T is a minimum spanning tree of G}

الگوریتم پرایم

- مثال: درخت فراگیر مینیمم برای گراف زیر را بیابید.



Choice	Edge	Weight
1	$\{b, f\}$	1
2	$\{a, b\}$	2
3	$\{f, j\}$	2
4	$\{a, e\}$	3
5	$\{i, j\}$	3
6	$\{f, g\}$	3
7	$\{c, g\}$	2
8	$\{c, d\}$	1
9	$\{g, h\}$	3
10	$\{h, l\}$	3
11	$\{k, l\}$	1
	Total:	24

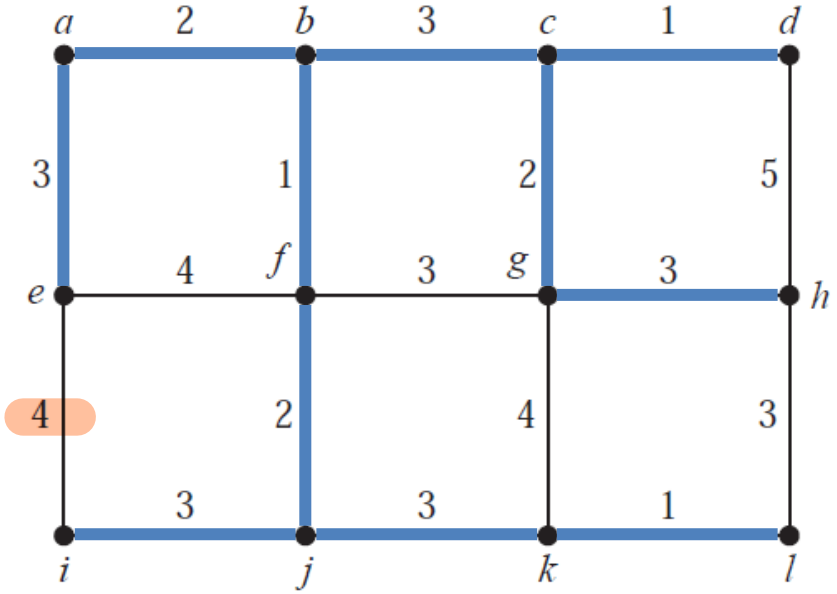
الگوریتم کراسکال

ALGORITHM 2 Kruskal's Algorithm.

procedure *Kruskal*(G: weighted connected undirected graph with n vertices)
T := empty graph
for $i := 1$ **to** $n - 1$
    e := any edge in G with smallest weight that does not form a simple circuit when added to T
    $T := T$ with e added
return T {T is a minimum spanning tree of G}

الگوریتم کراسکال

- مثال: درخت فراگیر مینیمم برای گراف زیر را بیابید.



Choice	Edge	Weight
1	$\{c, d\}$	1
2	$\{k, l\}$	1
3	$\{b, f\}$	1
4	$\{c, g\}$	2
5	$\{a, b\}$	2
6	$\{f, j\}$	2
7	$\{b, c\}$	3
8	$\{j, k\}$	3
9	$\{g, h\}$	3
10	$\{i, j\}$	3
11	$\{a, e\}$	3
	Total:	24

پایان

موفق و پیروز باشید