ساختمان داده‌ها و الگوریتم‌ها

فصل ششم

صمد نجارقابل

S.Najjar.G@Gmail.com

S.Najjar.G@Gmail.com

فهرست مطالب

S.Najjar.c@Gmail.com

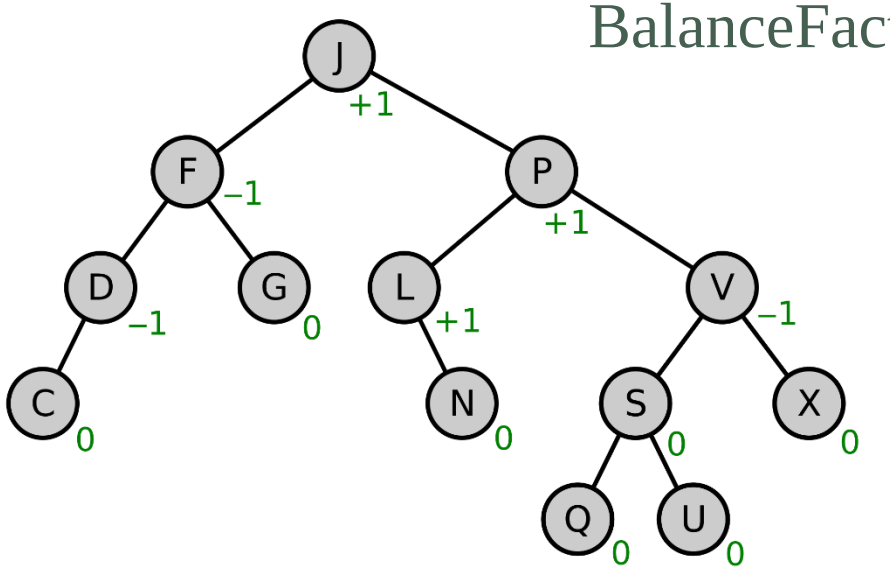
مباحث تکمیلی در ساختمان داده‌ها

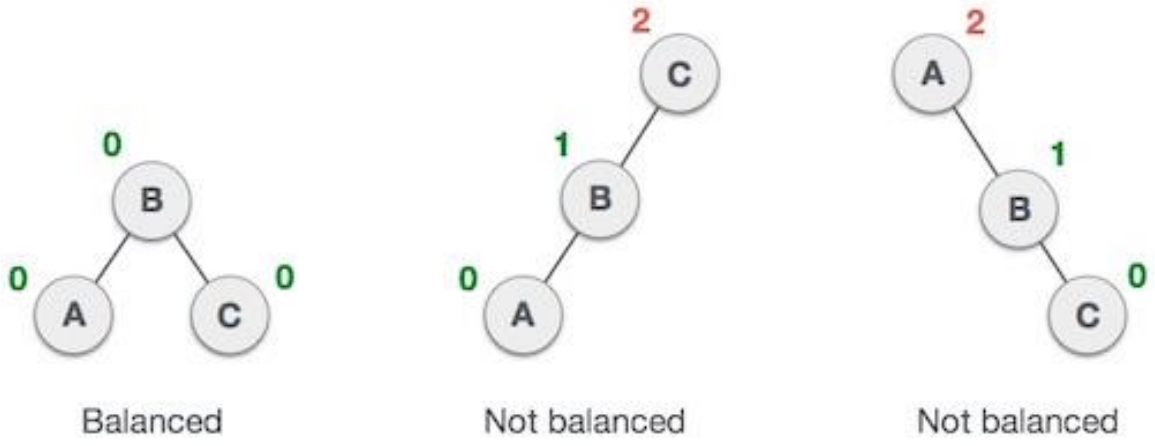
- **درخت AVL**
- درخت Heap

S.Najjar.G@Gmail.com

درخت AVL

- عنوان AVL TREE از اول نام‌های دو مخترع آن به نام‌های G.M. Adelson-Velsky و E.M. Landis، که مقاله خود را در سال ۱۹۶۲ با موضوع «یک الگوریتم برای سازماندهی اطلاعات» منتشر کردند گرفته شده‌است.
- درخت AVL: یک نوع درخت جستجوی دودویی خود متوازن‌کننده‌است.
- درختAVL سمت راست و چپ را بررسی کرده و تضمین می‌کند که زیردرخت راست و چپ اختلافی بیش‌تر از ۱ واحد ندارند. این اختلاف به نام عامل تعادل (Balance Factor) نامیده می‌شود.
- BalanceFactor(n)=Heght(RightSubtree(n))-Heght(LeftSubtree(n))



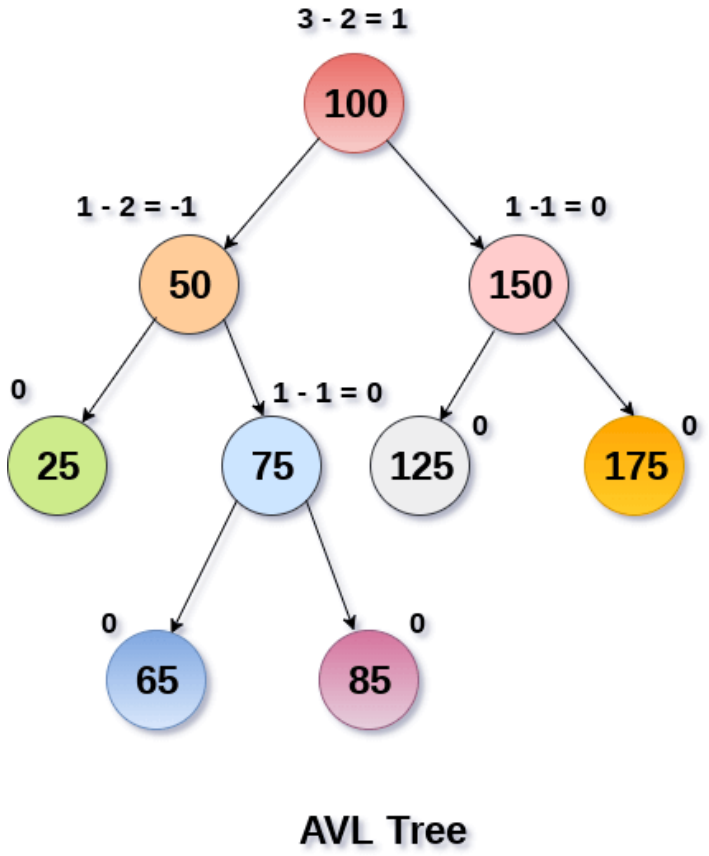


S.Najjar.G@Gmail.com

درخت AVL

- هر گره ای که ضریب توازن آن برابر ۱، ۰ یا ۱- باشد به عنوان گره متوازن در نظر گرفته می‌شود. هر درخت جست‌وجوی دودویی که تمام راس‌های آن متوازن باشند درخت ای‌وی‌ال است.

Balance Factor (k) = height (left(k)) - height (right(k))



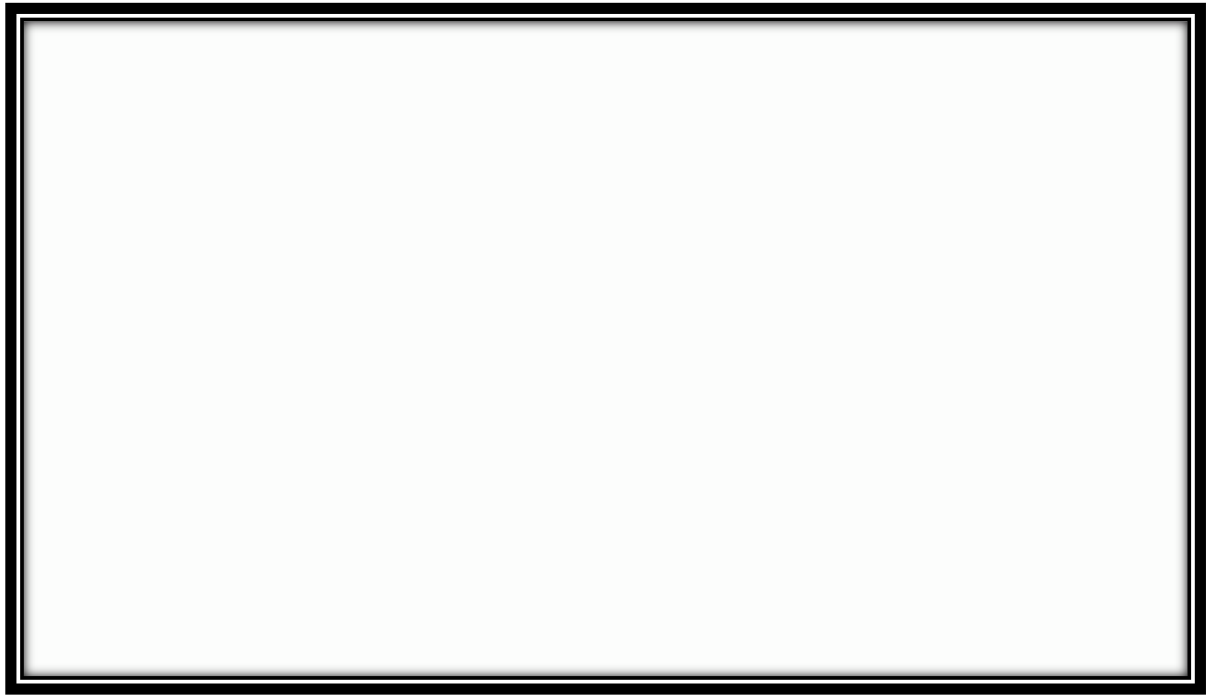
Complexity

Algorithm	Average case	Worst case
Space	o(n)	o(n)
Search	o(log n)	o(log n)
Insert	o(log n)	o(log n)
Delete	o(log n)	o(log n)

S.Najjar.G@Gmail.com

درخت AVL

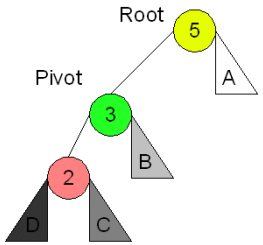
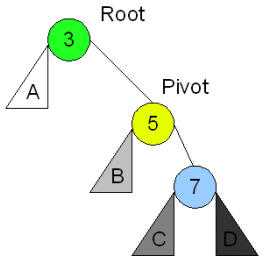
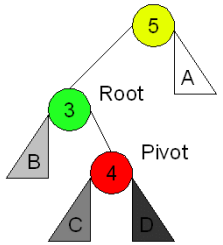
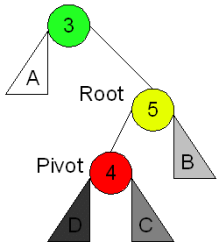
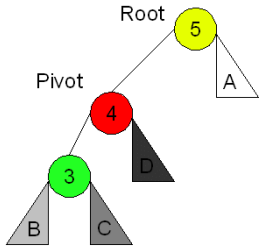
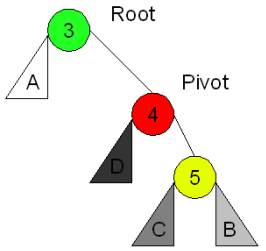
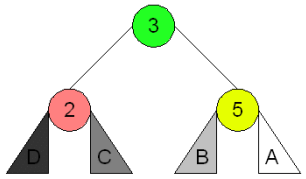
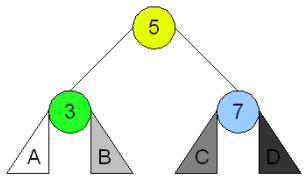
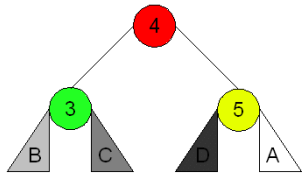
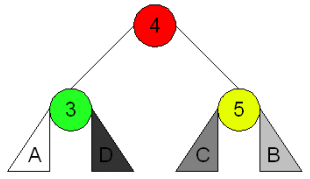
- به‌طور کلی عملیات اساسی در درخت‌های ای‌وی‌ال شامل همان عملیاتی که در درختهای جستجوی دودویی انجام می‌شود می‌باشد، اما اصلاحات وتغییرات به صورت فراخوانی یک یا چند باره چرخش درخت خواهد بود که به باز ذخیره کردن ارتفاع زیر درخت کمک می‌کند.



S.Najjar.G@Gmail.com

درخت AVL

- عملیات متوازن سازی

Left Left Case	Right Right Case	Left Right Case	Right Left Case
			
Right Rotation	**Left Rotation**	**Left Rotation**	**Right Rotation**
			
		Right Rotation	**Left Rotation**
			

S.Najjar.G@Gmail.com

S.Najjar.G@Gmail.com

درخت AVL

- https://cmps-people.ok.ubc.ca/ylucet/DS/AVLtree.html

S.Najjar.c@Gmail.com

مباحث تکمیلی در ساختمان داده‌ها

- درخت AVL
- **درخت Heap**

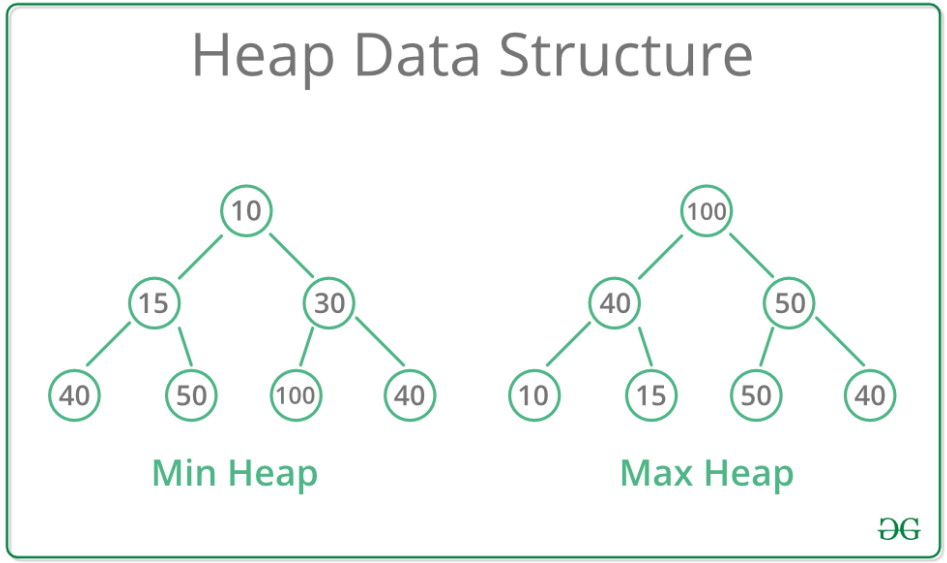
S.Najjar.G@Gmail.com

درخت Heap

- هرم ماکزیمم (ماکس هیپ / max-heap): درخت دودویی کاملی است که مقدار هر گره بیشتر یا مساوی فرزندان خود است.

- هرم مینیمم (مین هیپ / min-heap): درخت دودویی کاملی است که مقدار هر گره کمتر یا مساوی فرزندان خود است.

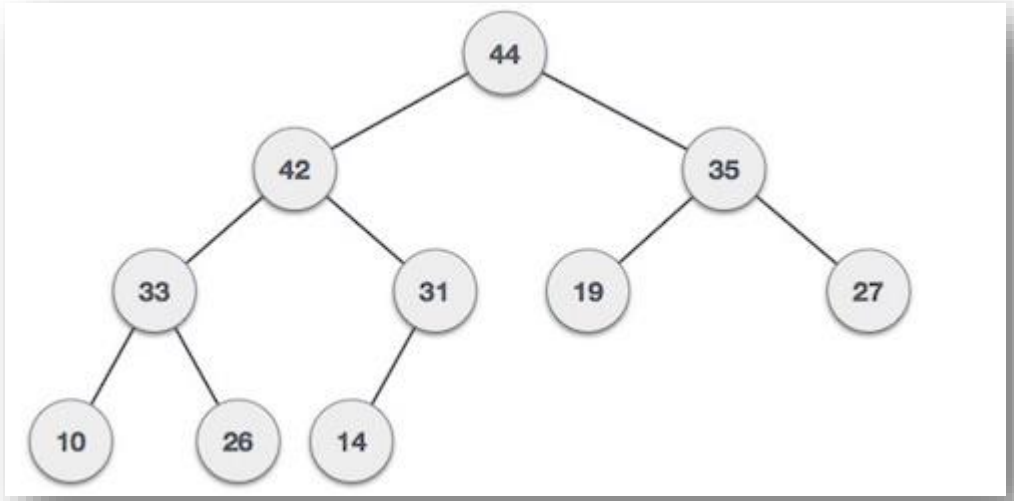
یادآوری: یک درخت دودویی کامل است، هرگاه تمامی سطوح درخت به غیر از احتمالا آخرین سطح پر بوده و برگ‌های سطح آخر از سمت چپ قرار گرفته باشند.



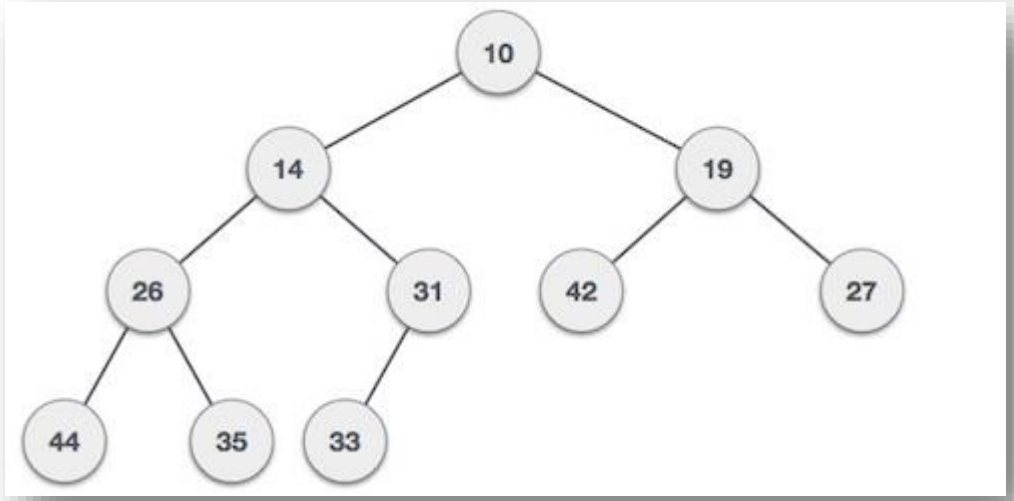
S.Najjar.G@Gmail.com

درخت Heap

- هرم ماکزیمم (ماکس هیپ / max-heap):



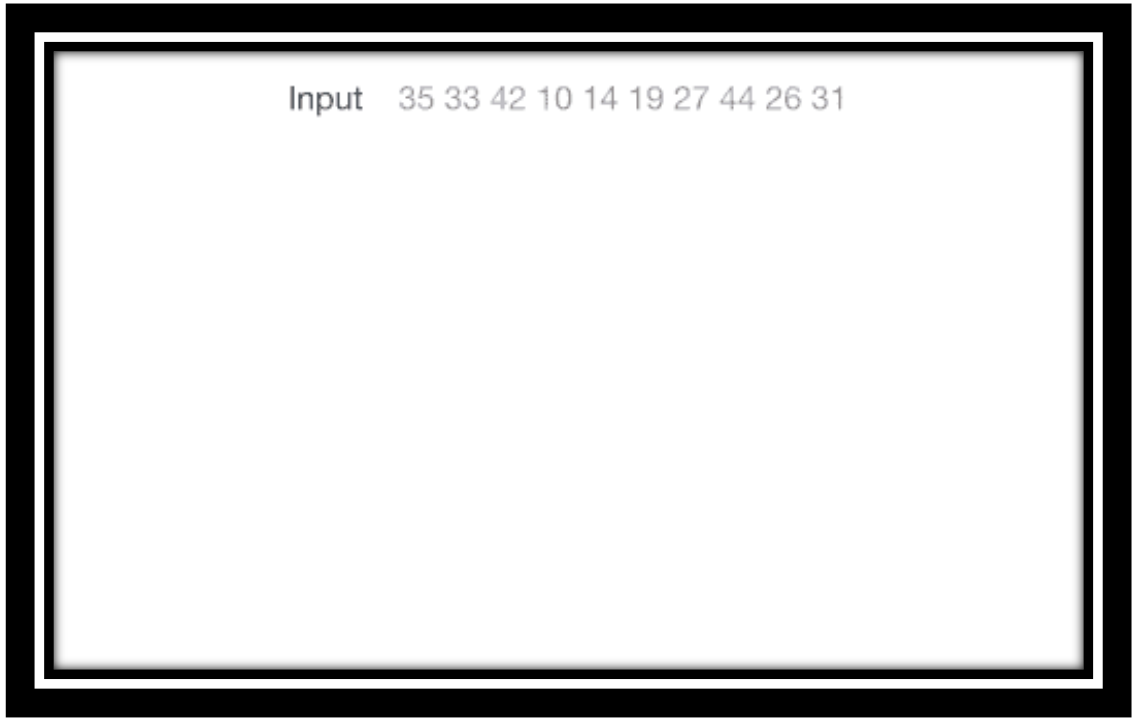
- هرم مینیمم (مین هیپ / min-heap):



S.Najjar@Gmail.com

درج در Max-Heap

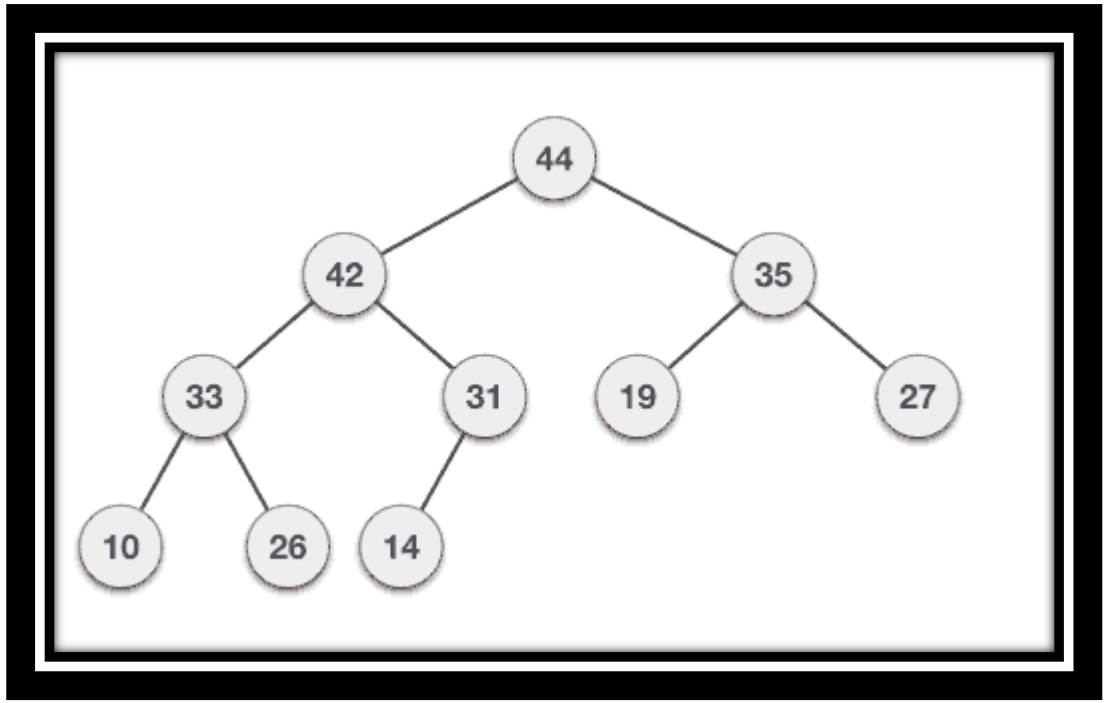
- **گام ۱** – ایجاد یک گره جدید در انتهای هیپ
- **گام ۲** – انتساب مقدار جدید به گره
- **گام ۳** – مقایسه مقدار این گره فرزند با والدهایش
- **گام ۴** – اگر مقدار والد کمتر از فرزند باشد، در این صورت تعویض می‌شوند.
- **گام ۵** – گام‌های ۳ و ۴ تا زمانی که همه مشخصات هیپ برقرار شوند تداوم می‌یابند.



S.Najjar.G@Gmail.com

حذف Max Heap

- **گام ۱** – گره ریشه را حذف کن
- **گام ۲** – آخرین عنصر آخرین سطح را به ریشه جابجا کن.
- **گام ۳** – مقدار این گره فرزند را با والدهایش مقایسه کن.
- **گام ۴** – اگر مقدار والد کمتر از فرزند بود، جای آن‌ها را عوض کن.
- **گام ۵** – گام‌های ۳ و ۴ را تا زمانی که مشخصات هیپ برقرار است تکرار کن.



S.Najjar.G@Gmail.com

درخت Heap

- لینک انیمیشنی از درخت Heap:

- https://www.cs.usfca.edu/~galles/visualization/Heap.html
- https://visualgo.net/en/heap

S.Najjar.G@Gmail.com

تمرین

- پیاده سازی درخت AVL
- پیاده سازی درخت Heap
- بررسی درخت ۲-۳
- بررسی درخت قرمز-سیاه
- بررسی درخت ۲-۳-۴