نمونه سوالات تمرین شماره ۴

دانشکده مهندسی برق و کامپیوتر

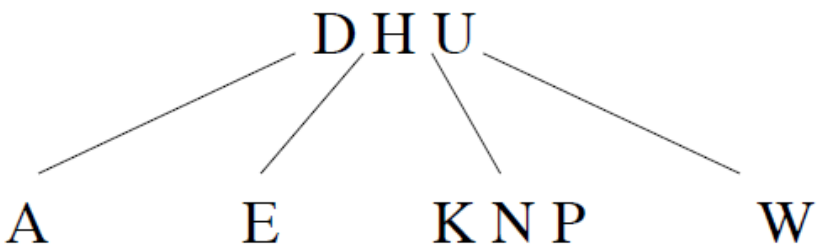
ساختمان داده – بهار ۱۳۹۹

استاد : دکتر فقیه

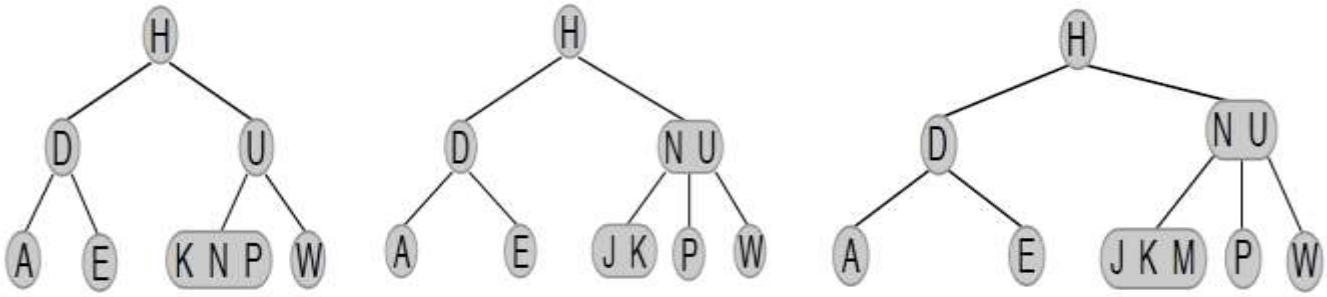
مسئول تمرین : حمید تراشیون
htarashion@gmail.com

---

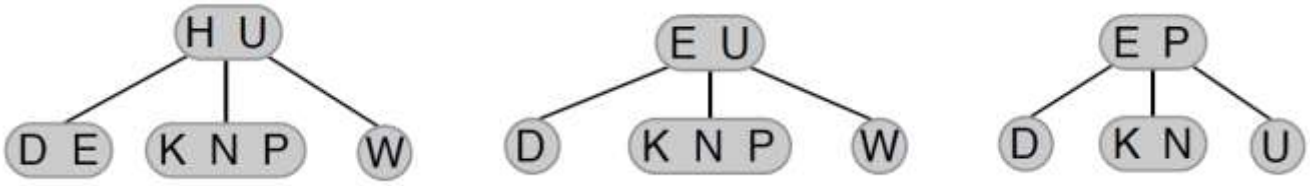
۱. B-tree ای با کلید $t = 2$ در نظر بگیرید مطابق شکل زیر در نظر بگیرید: (مقایسه ی حروف بر مبنای جایگاه آنها در حروف الفبا مشخص می شود یعنی اگر حرفی پیش از حرف دیگر باشد از نظر value مقدار کمتری برای آن متصور خواهیم بود.)



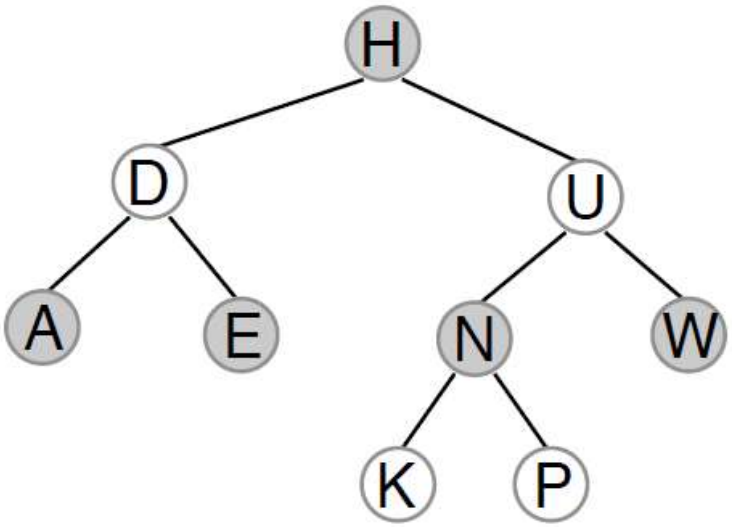
الف) می خواهیم حروف J و سپس M را در این درخت درج نماییم. مراحل را نشان دهید.



ب) درخت ابتدای مساله را دوباره در نظر بگیرید. اینبار هر یک از مراحل برای حذف حروف A، H و W را به ترتیب نشان دهید.



پ) آیا می توانید درخت قرمز – سیاه معادل درخت اصلی را نشان دهید؟



۲. دو درخت جستجوی دودویی T_1 و T_2 را که هر کدام به ترتیب n و m گره دارند، در نظر بگیرید. روشی بهینه با زمان $O(m + n)$ ارایه دهید تا عناصر تکراری این دو درخت جستجوی دودویی را پیدا کنیم.

اگر درخت ها را به صورت inorder پیمایش کنیم می توانیم عناصر هر کدام از آنها را به صورت مرتب شده در ۲ لیست در $O(m +n)$ بدست آوریم (بخاطر پیمایش inorder لیست ها مرتب شده نگه داشته شده اند). سپس دو لیست بدست آمده را با هم merge می کنیم (دو نشانه گر برای هر کدام از لیست ها نگه می داریم که در ابتدای کار به ابتدای

لیست ها اشاره می کنند و با مقایسه ی آن دو عنصر از دو لیست عدد کوچکتر را در لیست جدید می گذاریم و نشانه گر لیستی که از آن عنصر برداشته شده است را جلو می بریم) و به کمک آنها در $O(m+n)$ یک درخت جدید می سازیم.

روش ساختن درخت جدید:

عنصر میانه لیست را بدست می آوریم و به عنوان ریشه درخت جدید قرار می دهیم و زیر درخت چپ و راست را به همین شکل به صورت بازگشتی بدست می آوریم:

$T(n+m) = O(1) + 2T((n+m)/2) \rightarrow \text{overall: } O(n+m)$

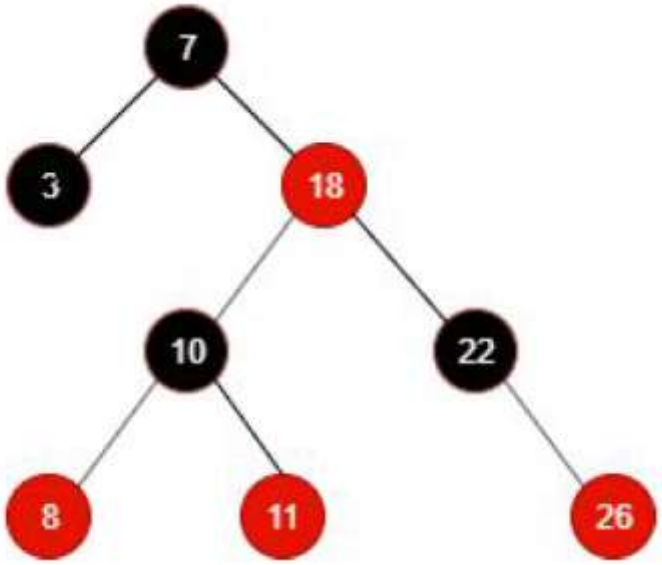
۳. الگوریتمی به صورت شبه کد ارایه دهید که k امین عدد بزرگ در یک درخت جستجوی دودویی را بیابد. این الگوریتم را با پیچیدگی زمانی از مرتبه $O(k+h)$ بیابید که در آن h ارتفاع درخت می باشد.

برای هر گره ابتدا باید تعداد گره های زیر درخت چپ به اضافه یک را نگه داریم (برای این کار معکوس روش inorder پیمایش می کنیم تا گره ها به صورت نزولی چیده شوند). سپس برای پیدا کردن k امین عنصر کافیست از راست ترین گره شروع کنیم (ابتدای لیست) و k بار در پیمایش ای که انجام داده ایم، عقب بیاییم.

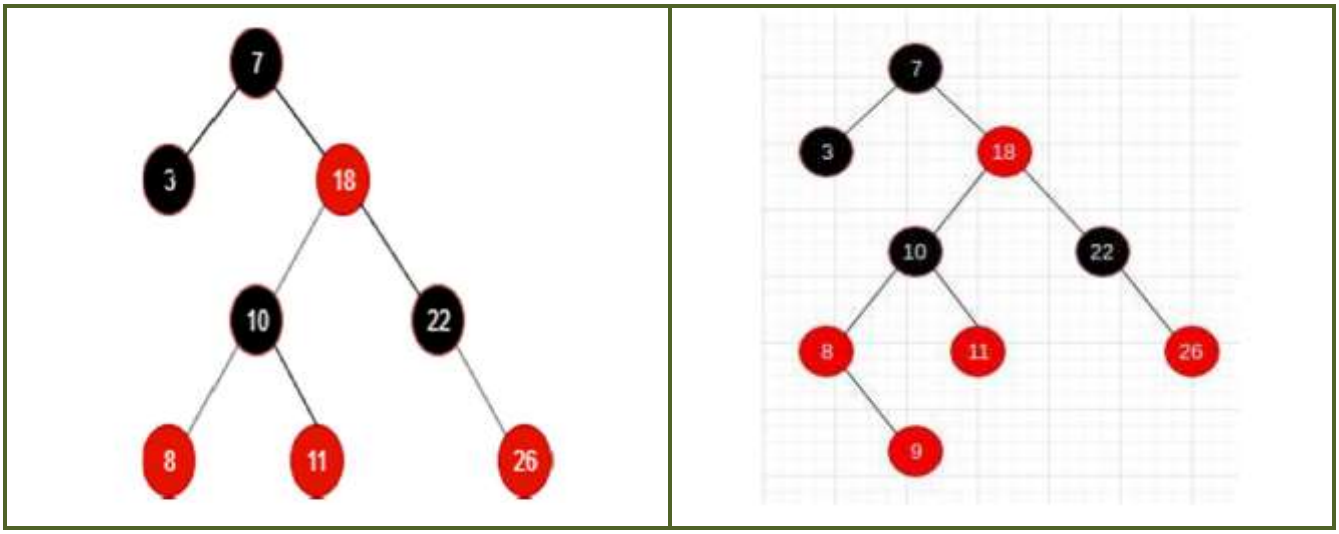
برای بررسی مرتبه زمانی این الگوریتم دو مورد را باید در نظر بگیریم:

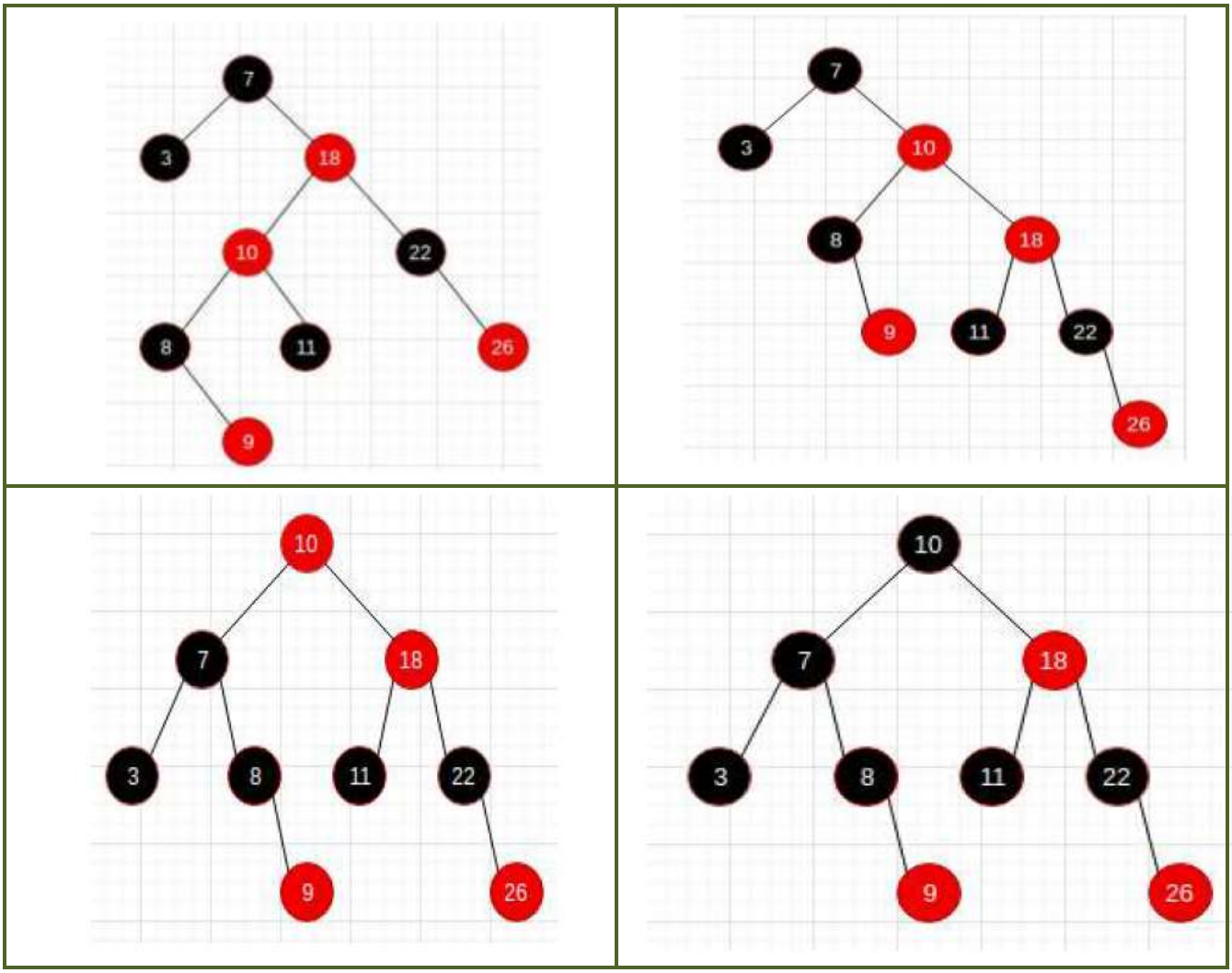
- پیمایش اولیه درخت که از مرتبه ی زمانی $O(h)$ می باشد.
- پیدا کردن k امین عنصر که به k عملیات از $O(1)$ نیاز دارد و کلا $O(k)$ می باشد.

۴. درخت قرمز – سیاه زیر را در نظر بگیرید:

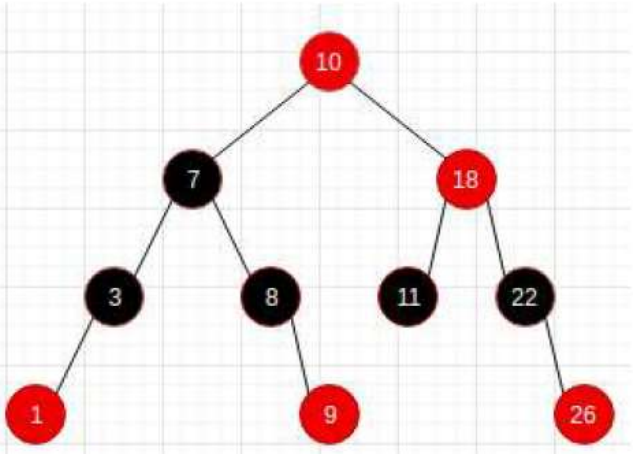


الف) عضو ۹ را با رسم هر مرحله از اصلاح، در درخت اضافه کنید.

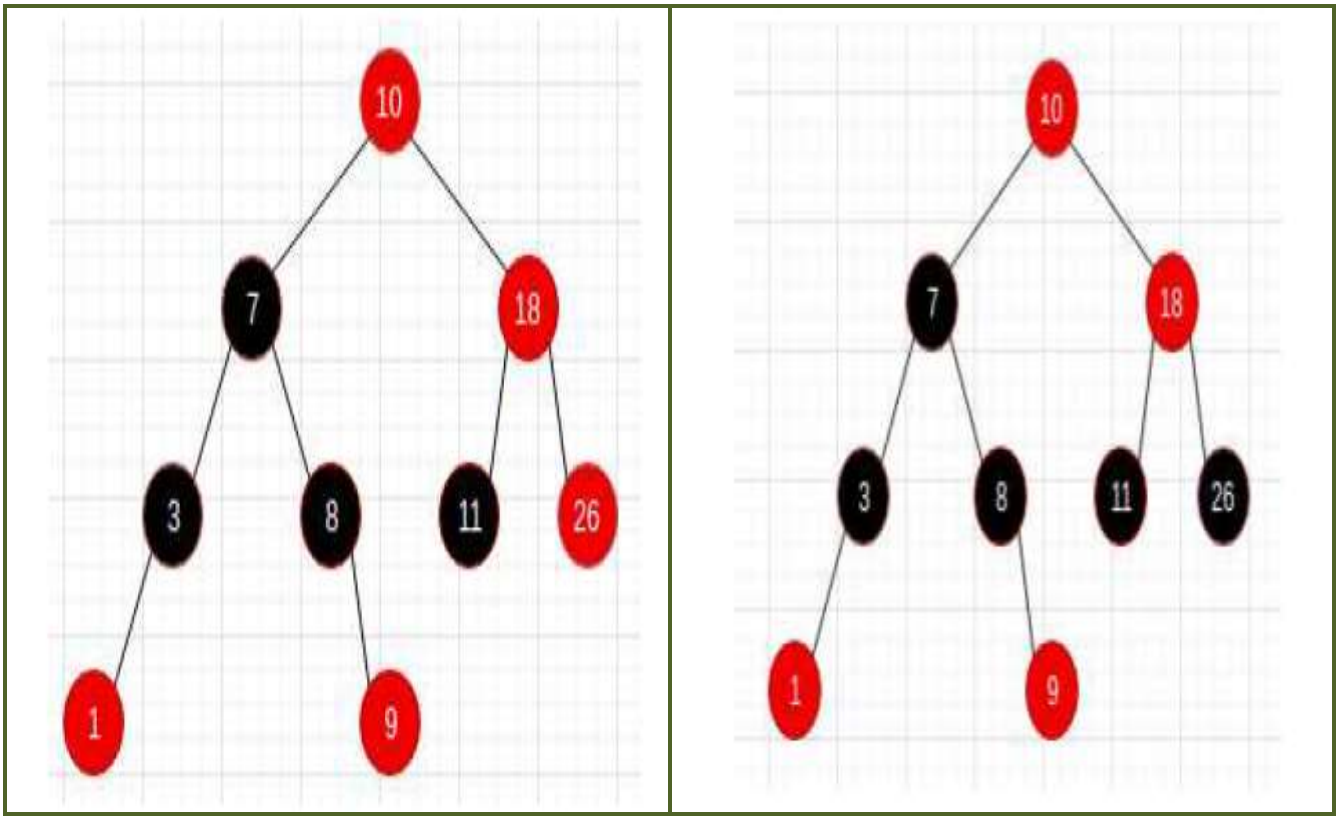




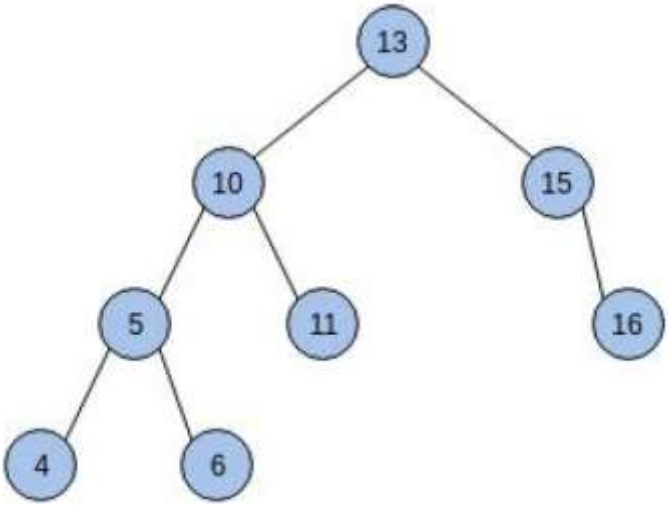
ب) عضو ۱ را با رسم هر مرحله از اصلاح، در درخت اضافه کنید.



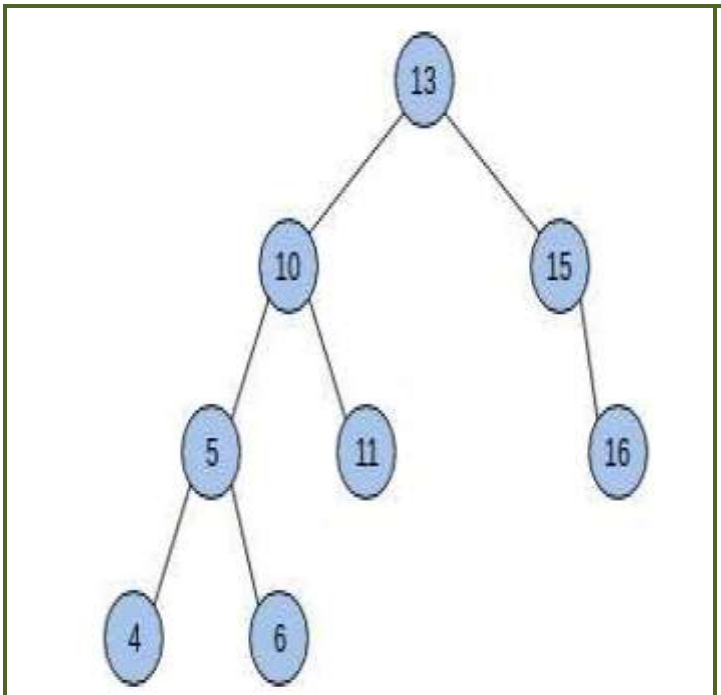
پ) عضو ۲۲ را با رسم هر مرحله از اصلاح، از درخت حذف کنید.

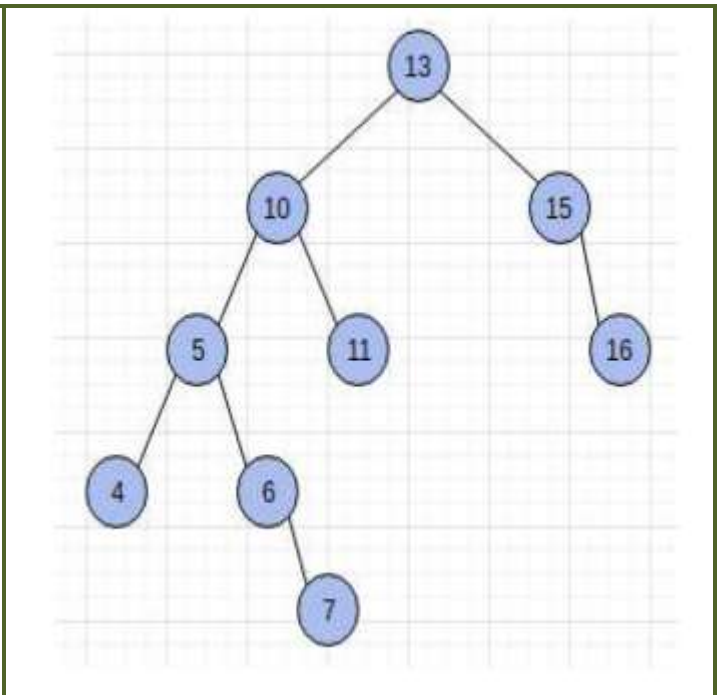


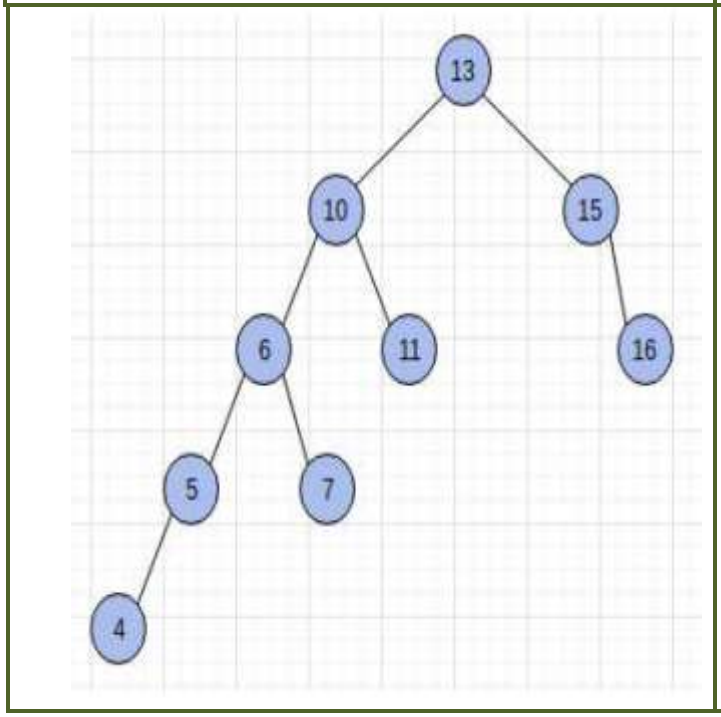
۵. درخت AVL زیر را در نظر بگیرید.

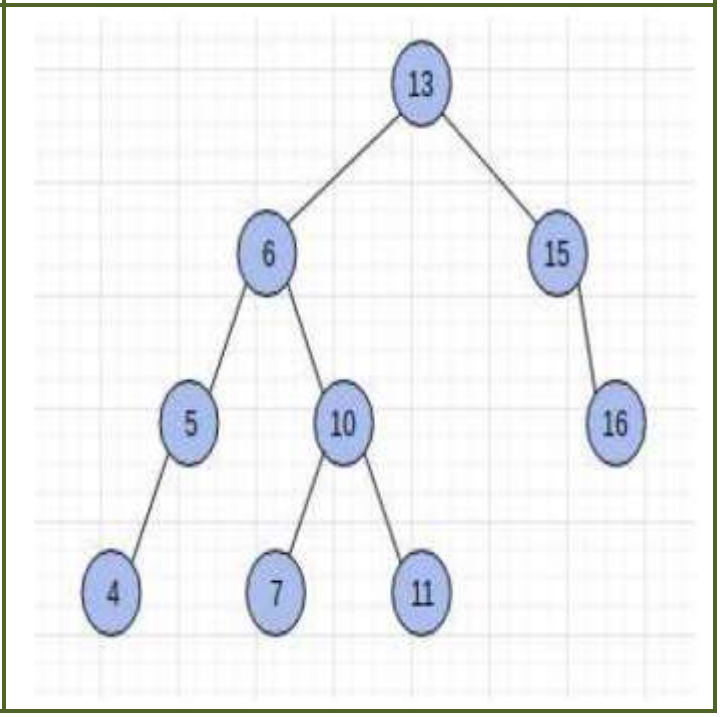


عضو ۷ را در درخت اضافه کنید. نام تمامی مراحل اصلاح را به ترتیب ذکر کنید.









پینوشت: عکس های جدول های سوالات ۵ و ۶ از بالا سمت چپ آغاز می شود و به ترتیب عکس بعدی در سطر از چپ به راست و سپس سراغ ستون پایینی می رویم.