

۱۴۰۰/۸/۴

خوبه زدم

ADT : داده ساختار جهت ذخیره درخت

$root()$: ریشه درخت را برمی گرداند

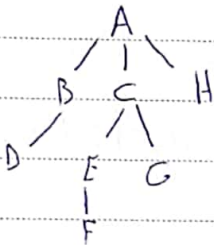
$parent(x)$: پدر x را برمی گرداند

$children(x)$: فرزندان x را برمی گرداند (پیاپی می کند)

$size()$: تعداد راس های درخت را برمی گرداند

$delete(x)$: راس x را حذف می کند

$insert(y)$: راس y را درج می کند



* یا ده ساختار های درخت

۱-۱. انتخاب از آرایه :

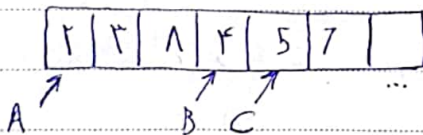
	۱	۲	۳	۴	۵	۶	۷	۸
راس ها ←	A	B	C	D	E	F	G	H
پدر راس ها ←	-	۱	۱	۲	۳	۵	۶	۱

$parent(x)$:

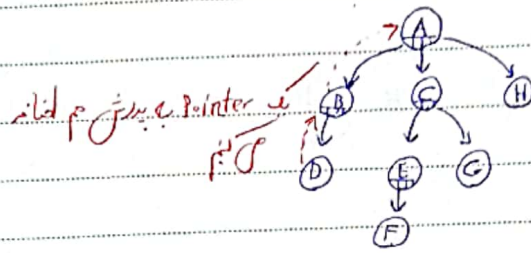
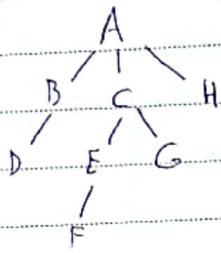
$C : parent(G) \leftarrow O(1)$

$children(x) : درختان \leftarrow O(n)$ (تعداد فرزندان) $?$

حافظه : $O(n)$



۵-۱. انتخاب پیاپی preorder و postorder را ذخیره کنیم




$O(1)$: Parent(x)

$O(K)$: children(x)

کے فرزند K کے x کے فرزند

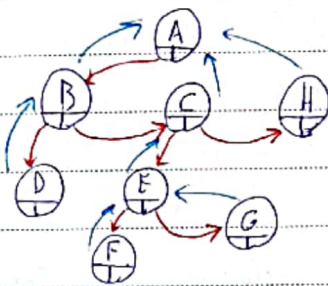
$O(1)$: size

* مسئلہ : node کا مشابہ  جس کا pointer ہے

مراہیں ممکن است n فرزند داشته باشد $O(n^2)$

رفع مسئلہ : * داخل ہر node آرایہ جویا استفادہ کنیم
 * لیکن لیست :

درخت را با استفاده از node های پیاده سازی کنیم که فقط ردنا pointer داشته باشد



۲۔ استفادہ از لیست پیوندی فرزند برادر

$O(1)$: Parent(x)

$O(K)$: children(x)

کے فرزند K کے x کے فرزند

حافظہ : $O(n)$: چون n node داریم

Subject:

کلاس ۱۰م

Year. ۱۴۰۰ Month. ۸ Date. ۴ ()

درخت k تایی: درختی که تعداد فرزندان هر راس k است.

درخت k تایی کامل: درختی که تعداد فرزندان هر راس k است.



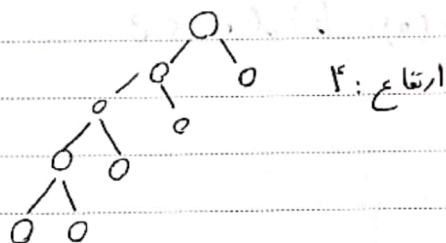
درخت کاملاً متوازن: درختی که عمق تمام برگها برابر است.



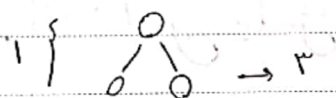
درخت متوازن: درختی که عمق تمام برگها کمتر از حداکثر ارتفاع باشد.

سوال: تعداد راسهای درخت 2 تایی کامل با ارتفاع h :

پاسخ: $2^{h+1} - 1$



ارتفاع: ۴

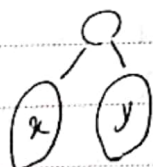


پسینه: $2^{h+1} - 1$

سوال: تعداد برگهای یک درخت 2 تایی کامل؟ (تعداد راسها n)

$\frac{n+1}{2}$

اثبات: با استقرا از ابتدا برای $n=1$ برگ $\frac{1+1}{2} = 1$ ✓



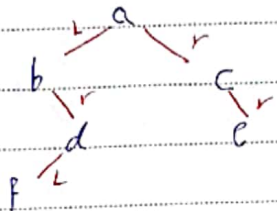
برای کمتر از n راس درست است. برای n راس ثابت کنیم

$$\text{تعداد برگ} : \frac{n+1}{2} + \frac{1+1}{2} = \frac{n+1+2}{2} \rightarrow \frac{n+1}{2}$$

$$n = n+1+1$$

تحریف: درخت دودویی

درخت ۲ تا ۲ برابری از یال های یک راس بر حسب چپ یا راست یا ۲ دارد. هر گره حداکثر یک فرزند چپ و یک فرزند راست دارد.



هر راس حداکثر ۲ تا فرزند دارد.

b فرزند چپ a است
 c فرزند راست a

```

inorder(n) {
    inorder(L of (n))
    visit(n)
    inorder(right(n))
}
    
```

$inorder(a): b f d a c e$

سوال: آیا با دانستن پیش از این $inorder$ می توان درخت را تشخیص داد؟ خیر

اما اگر $in + Pre$ یا $in + Post$ می توانیم درخت را تشخیص دهیم

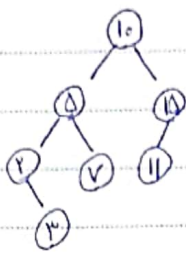
درخت دودویی جست و جوی: (BST) Binary search tree

فرمان یکسان قرار است در راس های درخت عدد ذخیره شود.

درخت دودویی جست و جوی درختی است که:

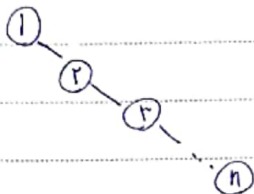
۱- دودویی است

۲- به ازای هر راس n مقدار n بزرگتر از تمام اولاد چپ و مقدار n کوچکتر از تمام اولاد راست است.



درخت دودویی جست و یابی

inorder: ۳ ۲ ۵ ۷ ۱۰ ۱۱ ۱۵

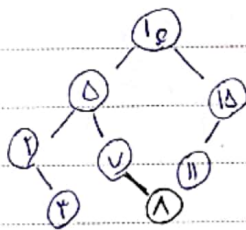


درخت جست و یابی دودویی

* بیاییم inorder یک درخت جست و یابی دودویی را برآورد کرده اند

* اگر بیاییم preorder یک درخت جست و یابی دودویی را داشته باشیم آیا می توانیم درخت را بازسازی کنیم؟

چون inorder را داریم (مرتبه مرتب شده است) اگر Postorder هم داشته باشیم می توانیم درخت را بازسازی کنیم.



insert داخل BST

: find(x)

: find(N) : از گره ۱ تا ۵ و از ۵ به ۷

insert(x) : x را داخل درخت قرار دهیم طوری که حداقل درج BST باشد. از لحاظ زمانی درج: ارتفاع درخت $\rightarrow O(n)$

insert(N) : ابتدا ۸ را find می کنیم