بسمه تعالی

درس: ساختمان های داده ها **دانشگاه فردوسی مشهد** **مدرس: شیرازی**

تمرین BinaryTreeIterationWithoutParentNodes:

در این تمرین قرار است درخت دودویی پیاده سازی شود تا امکان پیمایش به انواع گوناگونی که در درس آمده است را بر روی درخت داشته باشد. کلاس های BinaryTreeNode و InternalBinaryTreeNode کلاس هایی هستند که گره های مربوط به درخت را می‌سازد. پس ابتدا لازم است تا این کلاس ها را مطالعه نموده و از نحوه ذخیره‌سازی اطلاعات گره‌های درخت مطلع شوید. نکته بسیار مهم در این تمرین این است که گره‌های داخلی درخت تنها شامل فرزند چپ و راست (و البته داده ذخیره شده در گره) هستند و اشاره‌گری به گره پدر وجود ندارد. پس از بررسی این موضوع و مطالعه گره‌های داخلی، به کمک کلاس مربوط به گره داخلی، کلاس های مربوط به درخت و iterator را پیاده‌سازی نمایید و تمرین را تکمیل کنید. برای پیاده‌سازی پیمایشگر ها، درون هر کدام از آنان یک پشته از گره‌های پدر است که لازم است دانشجویان از آن برای نگه‌داری اجداد گره فعلی استفاده کنند و هر زمان که نیاز به بازگشت به گره پدر بود از آن کمک بگیرند. کلاس هایی که شما باید پیاده‌سازی کنید به این شرح هست:

۱. در کلاس BinaryTree، ابتدا لازم است dummy root ها را تعریف کنید تا در قسمت های بعدی از آن ها استفاده کنید. پس در این قسمت ویژگی‌های مربوط به آنان تعریف می‌شود.

```
private int mSize;

//Write your code here (Add appropriate members to the BinaryTree class.
```

همچنین در این قسمت شی مربوط به گره ها را بسازید و اتصالات بین گره های dummy root را برقرار نمایید.

```
public BinaryTree () {
    //Write your code here
}
```

*** توصیه اکید می‌شود قبل از پیاده‌سازی این قسمت مطالب مربوط به dummy root ها را از درس مطالعه نمایید تا این قسمت را به درستی متوجه شوید و به یک نتیجه صحیح برسید.

۲. در این متد ها از کلاس BinaryTree، اطلاعاتی به عنوان پارامتر دریافت می‌شود و در گره ریشه‌ای که ساخته می‌شود، قرار می‌گیرد و همین طور در صورت نیاز آن گره دریافت می‌شود. همان طور که اطلاع دارید در دنیای کامپیوتر وقتی با درخت ها سر و کار داریم، داشتن گره ریشه به معنا داشتن تمام درخت است و به کمک آن می‌توانید تمام درخت را پیمایش کنید و به تمامی اطلاعات آن دسترسی داشته باشید. پس این تابع به نوعی سازنده درخت است و به کمک این تابع است که اولین گره را به درخت اضافه می‌کنیم و شروع به ساختن آن می‌نماییم. پارامتر های این تابع به صورت زیر است:

data: که شامل اطلاعات مربوط به گره ریشه است.

```
public void insertRootNode(T data) {
    //Write your code here
}

public BinaryTreeNode<T, IBTN> getRootNode(){
    //Write your code here
}
```

۳. در این دو متد از BinaryTree، گره و اطلاعاتی را به عنوان ورودی دریافت می‌کنید و بر اساس نوع متد، آن اطلاعات را در فرزند چپ یا راست گره دریافت شده، قرار می‌دهید. پارامتر های این متد به صورت زیر است:

parentNode: این پارامتر گره مورد نظر را می‌دهد که بر اساس نوع تابع اطلاعات در گره جدید به عنوان فرزند آن قرار می‌گیرد.
data: این پارامتر اطلاعات مورد نظر است که باید در گرهی جدید به عنوان فرزند گره پدر ذکر شده اضافه شود.

```
public void insertLeftChild(BinaryTreeNode<T, IBTN> parentNode, T data) {
    //Write your code here
}

public void insertRightChild(BinaryTreeNode<T, IBTN> parentNode, T data) {
    //Write your code here
}
```

* در این دو تابع، در صورتی که گره ارسال شده به عنوان پدر از قبل فرزندی داشت و امکان اضافه شدن فرزند جدید در آن قسمت را نداشت، exception ای ارسال کنید تا از ادامه برنامه جلوگیری نماید.

۴. در این متد از BinaryTree، باید گرهی را از درخت حذف نمایید. پس برای این کار، متد به همراه گره مورد نظر و گره پدر آن صدا زده می‌شود تا به کمک آن ها گره را حذف کند. این متد در دو تابع deleteLeftChild و deleteRightChild فراخوانی می‌شود که می‌توانید آن‌ها را در کد بررسی کنید. پارامتر های این تابع به این صورت است:

theNode: گره مورد نظر که قصد حذف کردن آن را از درخت داریم.
parentNode: پدر گره مورد حذف که برای حذف درست به آن نیاز داریم.

```
// Only leaf nodes and nodes with degree 1 can be deleted.
// If a degree 1 node is deleted, it is replaced by its subtree.
protected void deleteNode(IBTN parentNode, IBTN theNode) {
    // Write your code here
}
```

* توجه کنید که در یک درخت دودویی تنها حق حذف کردن گره های برگ را داریم و برنامه نباید اجازه حذف گرهی از

وسط درخط را بدهد. پس با توجه به این نکته در صورتی که گره مورد نظر درجه یک یا بالاتر از آن را داشت (فرزند داشت) باید با پرتاب کردن exception ای از ادامه برنامه جلوگیری نمایید.

۵. حال باید متد های مربوط به iterator ها را تکمیل نمایید. کلاس های مربوط به پیمایشگر ها، اینترفیسی به نام iterator را در جاوا پیاده‌سازی می‌کنند که قالب استاندارد iterator ها در جاواست. برای هر پیمایشگر لازم است سازنده آن همراه با دو متد next و hasNext پیاده‌سازی گردد. در constructor باید فیلد ها را مقدار دهی کنید و حالت اولیه iterator را آماده نمایید تا در متد ها از آنان استفاده گردند. از موارد مهم آن، گره اولی که iterator به آن اشاره می‌کند است که در mCurrentNode قرار می‌گیرد و وضعیت اولیه پشته است که باید به درستی ساخته و آماده گردد. متد next وظیفه برگرداندن اطلاعات گره فعلی و رفتن به گره بعدی را دارد و متد hasNext وظیفه این را دارد که بررسی کند آیا همچنان گره قابل پیمایشی از درخت باقی‌مانده است یا mCurrentNode آخرین گره از درخت را نیز پیمایش کرده است. به عنوان مثال برای کلاس BinaryTreeBackwardInorderIterator لازم است موارد زیر پیاده‌سازی گردند:

```
    // The binaryTree parameter is used to initialize the mCurrentNode to
    // the appropriate dummy root. In addition, it is used to initialize mParents
Stack
    public BinaryTreeBackwardInorderIterator(BinaryTree<T, IBTN> binaryTree)
    {
        // Write your code here
    }

    //overloading operators:
    @Override
    public T next() {
        // Write your code here
    }

    @Override
    public boolean hasNext() {
        // Write your code here
    }
```

** توجه داشته باشید که iterator در ابتدا به یک گره قبل از اولین گرهی که باید از درخت پیمایش کند، اشاره می‌کند.

*** برای این قسمت نیز توصیه اکید می‌شود تا dummy root ها را از درس مطالعه فرمایید تا به مبحث مسلط گردید.

۶. در نهایت باید متد های مربوط به iterator ها را در BinaryTree پیاده کنید. در این قسمت تنها لازم است شئ درستی از iterator مورد نظر بسازید و آن را پس از آماده سازی return کنید.
به عنوان مثال متد زیر مربوط به forwardInorderIterator است که باید پیاده‌سازی گردد:

```
public Iterator<T> forwardInorderIterator() {
    // Write your code here
}
```

توجه: بهتر است جهت راحتی در پیاده‌سازی این تمرین، به ترتیب گفته شده شروع به تکمیل موارد کنید.

نحوه نمره دهی به این سوال بدین ترتیب است:

۱- TestInsertion: تست اضافه کردن گره به درخت که ۱۲٪ از نمره را به خود اختصاص داده‌اند. (موارد ۲ و ۳)

۲- TestDeletion: تست حذف کردن گره که شامل ۱۲٪ نمره است. (مورد ۴)

۳- تست های مربوط به پیمایشگر های زیر که هر کدام ۱۲٪ از نمره را دارند. (مورد ۵ و ۶)

TestBinaryTreeBackwardPostorderIterator,
TestBinaryTreeBackwardInorderIterator,
TestBinaryTreeForwardPreorderIterator,
TestBinaryTreeForwardInorderIterator

۴- تست های مربوط به پیمایشگر های زیر که هر کدام ۱۴٪ از نمره را دارند. (مورد ۵ و ۶)

TestBinaryTreeForwardPostorderIterator,
TestBinaryTreeBackwardPreorderIterator

* دقت داشته باشید که مورد ۱ پیشنیاز بقیه موارد، به خصوص مورد ۶، است و بدون پیاده‌سازی آن نمی‌توانید بقیه موارد را به درستی و با روش صحیح آماده کنید. برای همین تست مخصوص به خودش را ندارد ولی در تست های دیگر تاثیر مستقیم دارد و بررسی می‌گردد.

برای انجام این تمرین کارهای زیر را انجام دهید:

۱- ابتدا در این پوشه فایل info.txt را با مشخصات خود پر کنید.
۲- کد برنامه را تکمیل کنید.
۳- برنامه را بر روی تست(های) داده شده آزمایش نمایید.
۴- در صورت تمایل تست های بیشتری بنویسید تا از عملکرد برنامه خود اطمینان بیشتری حاصل نمایید.
۵- برنامه را اجرا کنید و پس از اطمینان از صحت عملکرد آن، با استفاده از کلید PrtScr از خروجی برنامه عکس بگیرید.
۶- عکس را با استفاده از mspaint در پوشه img ذخیره نمایید.

۷- همه فایل های اضافی که به دلیل کامپایل برنامه بوجود آمده اند را پاک نمایید.

۸- پوشه های src, img و test و همچنین فایل info.txt را به صورت یک فایل zip در آورید.

۹- مطمئن شوید که وقتی فایل zip را باز می کنید پوشه های src, img و test و همچنین فایل info.txt را می بینید.

۱۰- نام این فایل zip را به «نام تمرین-شماره دانشجویی» تغییر دهید (مثلا 1234567890-BinaryTreeIteration).

۱۱- ابتدا این فایل را به سیستم «سپهر» ایمیل کنید تا از نحوه عملکر برنامه خود بر روی تست های تکمیلی آگاه شوید.

۱۲- اشکالاتی را که سیستم «سپهر» مشخص کرده است برطرف نمایید و مجددا تمرین را به سیستم «سپهر» تحویل دهید.

۱۳- مرحله قبل را آن قدر ادامه دهید که از صحت عملکرد برنامه خود اطمینان حاصل نمایید.

۱٤- نسخه نهایی فایل zip خود را تهیه نمایید.

۱۵- این فایل را از طریق سیستم vu تحویل دهید.

با آرزوی موفقیت