Important

There are general submission guidelines you must always follow. If you fail to follow any of the following guidelines you risk receiving a **0** for the entire assignment.

1. All submitted code must compile under **JDK 8**. This includes unused code, so don't submit extra files that don't compile.
2. Do not include any package declarations in your classes.
3. Do not change any existing class headers, constructors, or method signatures.
4. Do not add additional public methods when implementing an interface.
5. Do not use anything that would trivialize the assignment. (e.g. don't import/use `java.util.LinkedList` for a Linked List assignment. Ask if you are unsure.)
6. You must submit your source code, the `.java` files, not the compiled `.class` files.
7. All methods must be efficient, even if an expected runtime is not specified.
8. After you submit your files redownload them and run them to make sure they are what you intended to submit. You are responsible if you submit the wrong files.
9. Do not add any new instance variables.
10. The submission on T-Square **MUST** contain only files you want us to grade. **Remove any old versions of files, and submit only the latest files.**

Skip List

You are to code a skip list. The bottom layer of the skip list is an ordinary skip list with every element. The very first level in your skip list is level 1. See the interface for more information regarding implementation.

Phantom Nodes

You are to implement your skip list with a layer of phantom nodes at the beginning. The method `getHead()` should return the topmost phantom node which will act as the head.

Coin Flipper

This will be the source of randomness for your skip list. You MUST use the `CoinFlipper` passed into the given constructor and no other source of randomness (eg a `CoinFlipper` you instantiate, a new `Random` object, etc) or you will lose a lot of points. When adding an item to your skip list, it will always be on the first level. After that, you will use the coin flipper to determine if it will be promoted to the next level. Heads means that you promote the item to the next level. Example: You are adding the String "i <3 skiplists". You flip HHT. That means it will be put on the bottom level and then promoted two levels. It will be on the bottom 3 levels.

printSkipList()

This is a method provided to help with debugging. This is only meant to help give you a glimpse of what is in your skip list. It is not meant to replace testing.

Methods

See the interface for details. Note that some methods are worth more than others. If `put()` is incorrect, then you are likely to fail most tests, as putting things into the skip list is crucial to the usability of a data structure.

Exceptions

When throwing exceptions, you must include a message by passing in a String as a parameter. For example:

```
throw new PDFReadException("Did not read PDF, will lose points.");
```

Style and Formatting

It is important that your code is not only functional but is also written clearly and with good style. We will be checking your code against a style checker that we are providing. It is located in resources along with instructions on how to use it. We will take off a point for every style error that occurs. If you feel like what you wrote is in accordance with good style but still sets off the style checker please email Jonathan Jemson (jonathanjemson@gatech.edu) with the subject header of "CheckStyle XML".

Javadocs

Javadoc any helper methods you create in a style similar to the Javadocs for the methods in the interface.

Forbidden Statements

You may not use these in your code at any time in CS 1332.

- `break` may only be used in switch-case statements
- `return` may not be used in void methods
- `continue`
- `package`
- `System.arrayCopy()`
- `clone()`
- `assert()`
- `Arrays` class
- `Array` class
- `Collections` class
- Reflection APIs

If you use these, we will take off points.

Debug print statements are fine, but should not print anything when we run them. We expect clean runs - printing to the console when we're grading will result in a penalty.

Provided

The following file(s) have been provided to you. There are several, but you will only edit one of them.

1. `SkipListInterface.java` This is the interface you will implement when designing your skip list. All instructions for what the methods should do are in the javadocs. **Do not alter this file.**
2. `SkipList.java` This class is where you will be implementing your skip list. **Do not add any public methods to this file.**
3. `Node.java` This is the `Node` class that will represent a node in your skip list. **Do not alter this file.**
4. `CoinFlipper.java` This is the `CoinFlipper` class that will determine whether or not a node gets promoted. **Do not alter this file.**
5. `SkipListStudentTest.java` These are some sample JUnits, similar to those that will be used to grade your assignment. **Passing this does not guarantee any sort of grade**

Deliverables

You must submit all of the following file(s). Please make sure the filename matches the filename(s) below. Be sure you receive the confirmation email from T-Square, and then download your uploaded files to a new folder, copy over the interfaces, recompile, and run. It is your responsibility to re-test your submission and discover editing oddities, upload issues, etc. **Submit only the files you want us to grade, and remove any old versions of files. There should be exactly ONE file for each of the files listed below.**

1. `SkipList.java`

You may attach each file individually or submit them in a zip archive.