COP 3503
Spring 2025

Optional Assignment to Drop your Lowest Assignment Grade

Implementing a Student Directory with Skip List

In this assignment, you will implement a **skip list** data structure to manage student records. Each student record consists of a **first name** and an **ID**. Your skip list should support:

- Insertion of new student records
- Searching for a student by first name

Requirements:

1) Data Stored in Each Node:
   a) Student first name (as `String`)
   b) Student ID (as `int`)
   c) A `forward[]` array for multiple levels

2) Skip List Behavior:
   a) The maximum number of levels in your skip list should be based on:

$$\text{MaxLevel} = \lfloor \log_2(n) \rfloor$$

where n is the number of inserted student records.

   b) You must simulate a fair coin flip (50% chance) to decide whether to promote a node to the next level.
   c) This helps keep the skip list balanced in a probabilistic way.

3) Operations to Implement:

   a) `insert(String name, int id)`

- Insert a new student using the coin-flip strategy to determine the number of levels.
- After insertion, print:

```
Inserted: <name> <id>, Levels: <number of levels>
```

b) `search(String name)`

- Search the skip list for a student by name.
- If found, print:

```
Found: <name> <id>
```

- If not found, print:

```
Student <name> not found.
```

Testing Instructions:

For this assignment, you will create your own testing files. You will create **two input files**, each containing a set of students to insert:

Test Case 1:

- 10 unique students (first name and ID)

Test Case 2:

- 20 unique students

Each file should list the students in the format:

```
Alice 1023
Bob 1045
Charlie 1088
...
```

- Your program should read these inputs from a file.
- After inserting the students, your program must:
  - Print all **insertion results** to the **terminal**
  - Perform **at least two searches** per test case:
    - One for an existing student
    - One for a student not in the list

Output Example:

```
Inserted: Alice 1023, Levels: 3
Inserted: Bob 1045, Levels: 1
Inserted: Charlie 1088, Levels: 2

Found: Alice 1023
Student David not found.
```

Implementation Notes:

- The insertion and search procedures must follow the skip list strategy discussed in class (top-down traversal using forward pointers).
- You may use `Random` in Java to simulate the coin flips (e.g., `rand.nextBoolean()`).
- Assume all names and IDs are unique.
- You should modularize your code into clear classes and methods.
- Your program should prompt the user to enter the input file name at runtime. **Do not hard-code** the file name into your code.

What to Submit:

- Submit:
  - Your Java code: `StudentSkipList.java`
  - Two input files: `test1.txt` and `test2.txt`
- The output should be printed to the **terminal** only
- Your code should be properly commented and formatted to receive full credit.

Grading:

Your input file will be used for grading as long as it meets all the requirements outlined in the assignment.