Lab 12 (October 18 or October 19)

Instructions: Complete the steps below. **Be sure to show your code to one of the lab TAs before you leave, so that you can receive credit for this lab.** You must also upload a copy of all your source code (.java) files to the link on Blackboard by 11:59 PM on Thursday, October 19.

Suppose 9 coins are placed into a 3 x 3 matrix, with some face-up and some face-down. Use a 0 to represent a face-up coin (heads) and 1 to represent a face-down coin (tails). For example:

```
0  0  0              1  1  0
0  1  0              1  0  0
0  0  0              0  0  1
```

Each possible configuration can also be represented as a binary number. For example, the two matrices above could be represented as 000010000 and 110100001, respectively. There are 512 possible configurations in all, so we can assign them the decimal numbers 0-511, where each decimal value can be converted into a 9-bit binary value representing one 3 x 3 arrangement of coins. The matrices above would be assigned the decimal values 16 and 417.

Write a program that asks the user to enter an integer between 0 and 511 and displays the corresponding 3 x 3 matrix, using 'H' for heads and 'T' for tails.

Sample run:

Enter a number between 0 and 511: **7**

H H H
H H H
T T T

Hint: This can be done using a two-dimensional array, but you can also do it easily using just loops and String variables. Start by converting the integer input into a 9-character `String` of 0s and 1s (with extra 0s added on the left to extend the length to exactly 9 digits). Then just use a loop to print three characters of the resulting “binary” string on each line.

Grading Guidelines: This lab is graded on a scale of 0-3 points, assigned as follows:

0 points: Student is absent or does not appear to have completed any work for the lab

1 point: Student has written the program, but the code does not compile or run due to errors.

2 points: Student has written the program and it compiles, but it produces incorrect output.

3 points: The student’s program compiles and runs correctly, without any apparent errors.