# CSCI 650 Assignment #4 – TM

Solutions to the written questions on this assignment should be submitted via PDF to Canvas before the due date. Make sure to justify your answers.

- **No handwritten version of the answers will be accepted**

You are encouraged to collaborate with one another. However, you must write up your own solutions independently.

## Submission

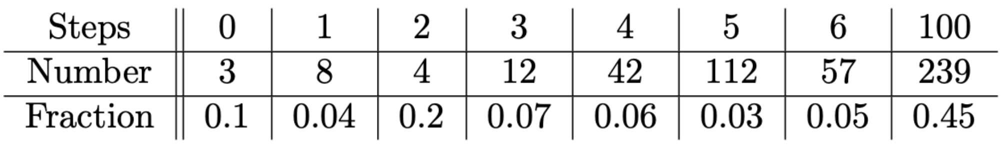
- Due: as shown in Canvas
- PDF submission: in Canvas
- Code submission: INIGinious https://inginious.csuchico.edu. Look for CSCI 650 Section 1. Note that you need a connection through **VPN** to access the INGInious site.

## Problems

1. In this question we will write a simple function to simulate a Turing machine and use it to investigate BB(2), the second busy beaver number.

(a) (30 pts) Write a function **simulateTM(delta, A, w)** which takes a transition function, a set of accepting states, and a string as input. This function should return a string associated with the final configuration of the Turing machine along with a boolean indicating whether or not **w** is accepted. We will assume that states are labeled A, B, . . . and that **A** is the start state. Submit your code on INGInious (use filename: turingMachin.py) and include a code snippet of this function in your PDF submission. turingMachineTests.py, a script meant to help with local testing, is available on Canvas.

(b) (40 pts) Consider the busy beaver problem as discussed in class. Write code to consider all Turing machines with two states, a binary input alphabet, and a tape initially filled with 0s. Augment your code from part (a) to count the number of steps each machine takes before halting up to a maximum of 100 steps. Create a table similar to the one below indicating the number and fraction of machines that take ***i*** steps to finish.



| Steps | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 100 |
|---|---|---|---|---|---|---|---|---|
| Number | 3 | 8 | 4 | 12 | 42 | 112 | 57 | 239 |
| Fraction | 0.1 | 0.04 | 0.2 | 0.07 | 0.06 | 0.03 | 0.05 | 0.45 |

2. (30 pts) Context-free grammars play an important role in many areas of computer science including in programming languages and compilers, natural language processing, and parsing among others. Write a function **cyk(G, w)** that takes a context-free grammar and a string as

input and returns a boolean indicating whether or not the string is in the language associated with the context-free grammar along with a table filled in according to the CYK algorithm as discussed in class. Submit your code on INGInious (using filename: CYK.py) and include a code snippet of this function in your PDF submission. cykTests.py, a script meant to help with local testing, is available on Canvas.