- 20% will be allocated based on viva on your code and concepts regarding cellular automaton.
- 30% will be based on the generated diagrams

2 Games

2. [15 points] **Rock-Paper-Scissors**
Pre-Final Evaluation Due Date: **6th September 2022, 11:59 PM**
Final Evaluation Due Date: **9th September 2022, 11:59PM**

In this assignment, you will write a program which, when given as input, a finite state automaton (FSA) which plays Rock-Paper-Scissors (RPS) with an unknown initial state, outputs a FSA that beats it in as many rounds as possible.

2.1 Game Description - For the uncultured

In each round, two players simultaneously choose one of rock, paper and scissors. Scissors beats paper, rock beats scissors, and paper beats rock. It's a draw if they choose the same. The game is repeated for multiple such rounds.

2.2 Finite State Automaton

Each state of a RPS FSA is described by c: what the FSA will play in the current round, and $r\ p\ s$: the new state if the opponent plays rock, paper and scissors respectively.

2.3 Task

You are given the opponent's FSA with $n \leq 30$ states but not its initial state. You have to submit the following files: 1. A program that outputs an FSA with $m \leq 1000$ states and a fixed initial state such that it beats the opponent FSA as many times as possible. We will simulate 10,000 rounds for each possible initial state of the opponent FSA. 2. A file containing a single FSA which your batchmates programs will be evaluated against. The better your test case, i.e. the harder it is to win against this FSA, the greater your score.

2.4 Input Format

Applies to both the input to the program you submit and the FSA test case you submit.

$$
\begin{array}{l}
n \\
c_1\ r_1\ p_1\ s_1 \\
\vdots \\
c_n\ r_n\ p_n\ s_n
\end{array}
$$

2.5 Output Format

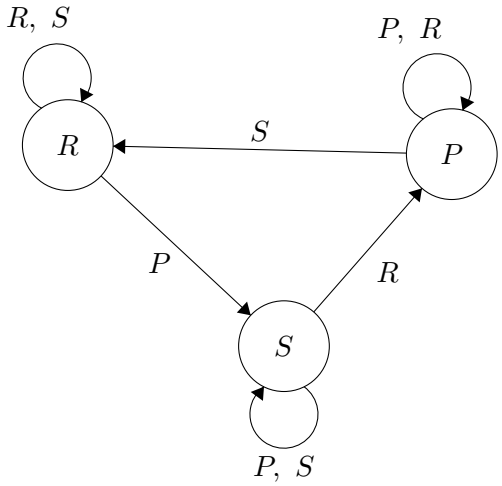
The output format for your program.

m
$c_1\ r_1\ p_1\ s_1$
$\vdots$
$c_m\ r_m\ p_m\ s_m$

2.6 Example

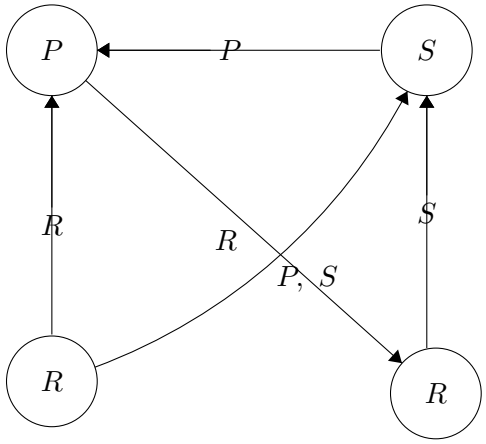
2.6.1 Sample Input

3
R 1 2 1
S 3 2 2
P 3 3 1



2.6.2 Sample Output

4
R 2 3 3
P 4 1 1
S 1 2 1
R 1 1 3



2.6.3 Sample Explanation

We can see that the input FSA does not take a particularly smart approach, only changing its move when it loses in a way that beats the opponents previous move. Since we know this, we can design the output FSA to be 1 step ahead of this process. Initially, it outputs rock (state 1 bottom left), and then based on the opponents response it transitions to a state such that it wins every future round (states 2-4). Thus, if the opponent input state is Rock or Paper (states 1 or 3), it wins 9,999 rounds and if it is Scissors (state 2) it wins all 10,000 rounds. We cannot do better than this against this input FSA.

2.7 Scoring

- There will be a pre-final evaluation a few days before the deadline. You will have to submit your progress on both the program and the testcase. We will use this to provide feedback on common mistakes, as well as an opportunity for you to see how your approach is performing.
- The weightages for the different evaluations are the following: Pre-final evaluation (20%), Final evaluation (60%), Viva (20%). Within the 2 evaluations, there is $75\% - 25\%$ split between the performance of your program and the testcase. Thus, for example, your program in the pre-final evaluation holds an overall weightage of $0.20 * 0.75 = 15\%$.
- If your program wins (draws not counted) $x\%$ of rounds tested against, you will receive $min(0, 2x - 100)\%$ points for this component. This means a program that wins only 50% rounds gets $2 * 50 - 100 = 0\%$ points. In the final evaluation which consists of $0.60 * 0.75 = 45$ points, a program that wins 90% rounds gets $(2 * 90 - 100)\% * 45 = 80\% * 45 = 36$ points for this component. Note that your program will be tested against other cases as well, not just those submitted by your batchmates.
- For scoring the testcases, we will use a relative scale. Among the the top $(10 * k)\%$ cases (in terms of numbers of rounds lost or drawn by the opponent program) will receive $100 - 10(k-1)\%$ marks for the smallest value of k that applies. So in the final evaluation where the total weightage of the test case is $0.60 * 0.25 = 15\%$, the top 10% cases ($k = 1$) will receive full 15 points, the top $10 - 20\%$ cases ($k = 2$) receive $100 - 10(2 - 1)\% * 15 = 90\% * 15 = 13.5$ points, top $20 - 30\%$ 12 points and so on until the bottom 10% cases receive 1.5 points.

Please ensure you strictly follow the I/O format mentioned above and the submission format mentioned below as the evaluation is automated. We will be forced to give you a 0 otherwise. Please