CS 3100, Fall 2018, Assignment 2
150 pts total (will be scaled to 30)

Note: Note: We provide TA names so that you know whom to approach for grading-related matters for each question. As for seeking help, *any* TA will help you with *any* of the questions.

In doing all these questions, you may want to keep a copy of our book's Appendix B in front of you (in hardcopy or PDF format). This lists *all* of Jove's functions that you can search down and employ in your work.

NOTE: In general, whenever you finish an exercise using Jove, please do not merely have a code-cell run and produce an answer. *Always write a few sentences (ideally in bulletted form) of comments before/after each code cell that contains a result that you produced.*

1. **25 points:** (Maryam) By Pumping Lemma, prove that the language

$$L = \{1^{i^2} \ : \ i \geq 0\}$$

is not regular (this has strings of the form $\varepsilon, 1, 1111, 111111111, \ldots$). Write your answer again as given on Page 10 of the report `https://www.overleaf.com/read/fcsdttjhztyb`.

Solution:

1. Pick word $w = \{1^N : N \geq 0\}$.
2. Given $w = xyz$, then x and z must be zero or more 1's, and y one or more 1's.
3. No matter how many 1's are in x, y or z, y can be pumped up or down to create a word 1^N where $N \neq i^2$ for some $i \geq 0$.
4. Therefore $\neg Reg(L)$.

2. **25 points:** (Harshitha) By Pumping Lemma, prove that Language number 9 listed in Section 7 of the report `https://www.overleaf.com/read/fcsdttjhztyb` is not regular. Present your answer as described on Page 10 of this report.

Solution:

$L_9 = \{b^5 w : w \in \{a, b\}^*, 2\#_a(w) = 3\#_b(w)\}$

1. Pick a word $w = b^5(ab)^N b^N$
2. $w = xyz$ gives a few cases based around y:
   - y is $(ab)^k$ for some k
   - y is $(ab)^k$ for some k, minus the last b or the first a
   - y is $(ab)^k$ with some b's on the end and/or beginning
   - y is a some number of b's from the beginning or end
3. For all of these cases y can be pumped to zero and w will no longer be in L_9. Either there won't be 5 beginning b's, or the ratio in w will be incorrect.

3. **40 points:** (Paridhi, 10 points each part)

   (a) **By hand** (in your PDF), apply the DFA minimization procedure to the DFA shown below. Describe all the steps in filling the table of pairs. The level of detail of your work and the clarity of your presentation of the table and the distinguishing steps must match that of the book Section 6.4 (especially Figure 6.8).

```
Tz_bloated = md2mc('''DFA
IF : 0 -> A
A  : 0 -> B1
B  : 0 -> IF
IF : 1 -> IF
A  : 1 -> A1
B  : 1 -> B1
A1 : 0 -> B
A1 : 1 -> A
B1 : 0 -> IF
B1 : 1 -> B
''')
```

Solution outline:

```
Frame-0                    Frame-1
(Initial)                  (0-distinguishable)
--------------------       -------------------
A   -1                     A    0
B   -1  -1                 B    0  -1
A1  -1  -1  -1             A1   0  -1  -1
B1  -1  -1  -1  -1         B1   0  -1  -1  -1
    IF   A   B   A1            IF   A   B   A1

Frame-2 = Frame-3
(1-distinguishable)
--------------------
A    0
B    0   1
A1   0  -1   1
B1   0   1  -1   1
    IF   A   B   A1
```

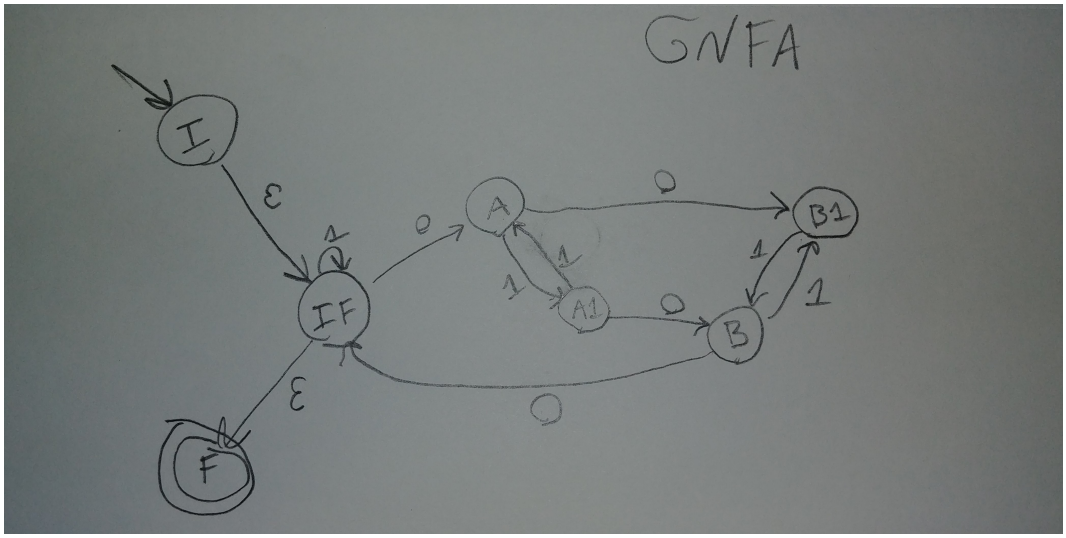
Equivalence classes: {A1, A}, {B1, B}, and {IF, IF}

```
Which gives:
Tz_min = md2mc('''DFA
IF : 0 -> AA
IF : 1 -> IF
AA : 0 -> BB
AA : 1 -> AA
BB : 0 -> IF
BB : 1 -> BB
''')
```

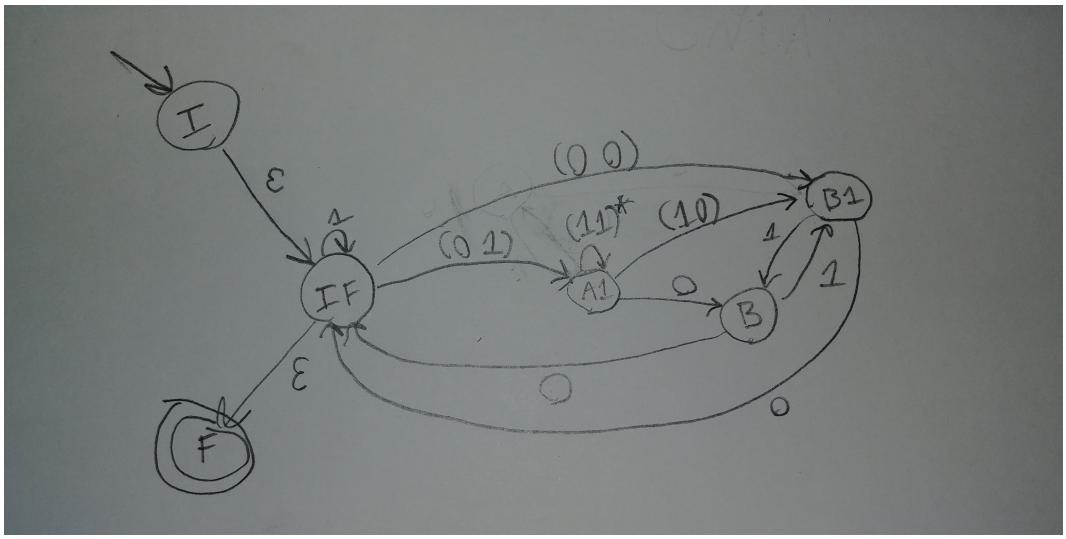
(b) **Using Jove**, using suitable Jove commands, minimize the above DFA. Then, underneath those checking steps in the Jove file, open a markdown cell, and explain there whether the minimized DFA obtained using Jove matches the one obtained by hand in Part 3a of this question.

(c) **By hand** (in your PDF), demonstrate the NFA to RE conversion procedure on this DFA. Build a GNFA first, and then eliminate states A and A1 (in that order) by hand.

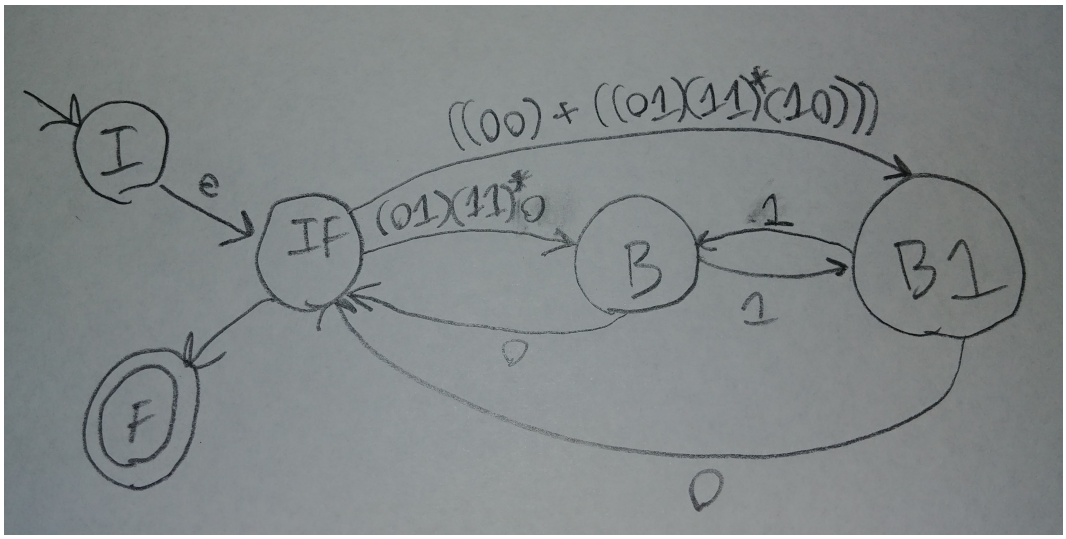
GNFA:



Deletion of A:



Deletion of A1:



Then do the same elimination of states **by using Jove**. You'll have to specify all the states to be eliminated as a list, but make A1 and A the first two entries in this list. Jove will carry the elimination all the way. Grab hold of the first two GNFAs generated by Jove. Manually check the correctness of your first two GNFAs generated against the Jove-generated GNFA, thus ensuring that your work is correct. Write the results of the check in briefly in your Jove submission (where I provide bullets).

(d) Once assured, finish the conversion to RE **using Jove**, grab-hold of the RE it generates, and then send this RE through the Jove pipeline to re-obtain a minimal DFA. Check that the minimal DFA agrees once again. This agreement can be argued through visual inspection. Write down a few bullets in the Jove file, arguing that the same DFA is produced yet again from this RE.

The RE generated finally is shown in the Jove cell. (It is very long, so don't bother copying into your PDF answer.)

4. **35 points:** (Arnab, 10,10, 5, and 10 points for the parts) This whole assignment is done **using Jove** except for Part 4d which expects some PDF bullets. You may study `DFAUnit2` to do the following questions. There is an embedded Youtube video describing its contents (plus more helpful tips). **Note:** File `DFAUnit2` was created a while ago, and hence the examples and book sections mentioned there may be different. You are perhaps better off studying DFA minimization from our book.

   (a) We want a DFA for the language with an **odd number of** 0**'s or** an **even number of** 1**'s**. The alphabet is $\{0, 1\}$. **By Jove**, obtain this DFA by designing it in the markdown syntax of Jove, minimize it, and set it aside, calling it `DFAOdd0OREven1`. This result should appear in one Jove code cell.

   (b) Now obtain the *same DFA* through an alternative procedure, as follows:

      i. Obtain a DFA for an **even number of** 0**'s**. The result must appear in a Jove code cell.

      ii. Obtain a DFA for an **odd number of** 1**'s**. The result must appear in a Jove code cell.

      iii. **Intersect** these DFA and minimize the result. The result must appear in a Jove code cell.

      iv. Then complement the result, calling it `DFAOdd0OREven1Alt`. The result must appear in a Jove code cell.

   (c) Show the isomorphism of `DFAOdd0OREven1Alt` and `DFAOdd0OREven1` after minimizations. The result must appear in a Jove code cell. Provide a markdown cell at this juncture and write down 2-3 bulletted comments showing that the resulting DFA are isomorphic.

   (d) In Quiz3, you had to design a DFA for the language of strings that began XOR ended with 01. We called this DFA `Db01XORe01`. There were then many checking steps presented leading to Presto-1 and Presto-2 (two convincing checks). **By hand**, write down in your PDF answer what these checks mean and why an empty DFA is expected:

      i. (5 pts) Argue in about six neat bullets, why Presto-1 and Presto-2 should result in an empty DFA. Write down the language operations involved, and argue that it means that if your solution were correct, then Presto-1 would be an empty DFA (its language is φ).

> - Presto 1 creates an intersection of a DFA for "begins with 01" and a DFA for "neither begins with 01 nor ends with 01"
> - An input that both begins with 01 and does not begin with 01 is impossible.
> - Therefore this DFA will never move past its initial, non-final, state, meaning it is empty.
> - Presto 2 creates a DFA that is an intersection of "end with 01" and "neither begins with 01 nor ends with 01".

- As with presto 1, an input that both does and does not end in 01 is impossible.
- Presto 2 will also never move past it's initial, non-final, state, meaning it is empty.

ii. (5 pts) Is this a sufficient check? That is, if there were to be any mistake in `Db01XORe01`, will it have emerged in either Presto-1 or Presto-2 (or both) being non-empty? Answer in 3 short bullets.

- If there were some mistake with your DFA then it would accept some input other than "begins with 01 XOR ends with 01".
- If your DFA accepted unintended inputs then so would the subsequent complementation and intersects.
- Therefore this is a sufficient check, and any mistakes would be evident.

5. **25 points:** (Paul, 15,5 points for the parts) Given the following NFA, do these tasks (write your answers in PDF when indicated, or in the Jove notes when asked).

```
 N5 = md2mc('''NFA
I : 0|1 -> I
I : ''  -> B,C
B : 0   -> D
D : 0|1 -> F
D : ''  -> C
C : 1   -> F
''')
```

(a) **By hand** (in your PDF) perform the NFA to DFA conversion algorithm using subset construction. **Using Jove** write down the markdown for this DFA, calling it `HD5.`

In your solution, you must write a series of bullets and sub-bullets of this form:[1]

- Epsilon-close the initial state, thus attaining a set of states of the form $\{A, B, C\}$
- Beginning at set of states $\{A, B, C\}$, consider the move of A upon $x \in \Sigma$ (for each such state "A"; list the cases one by one in your solution)
- After such a move, say where you land initially
- Then Epsilon-close from the state landed into, and say where all you are now
- Do that for *all* of $\{A, B, C\}$ moved wrt x
- Now from the new set of Epsilon-closed states, repeat the process.

[1] Yes, this will be many bullets, but then the NFA is not very large, and this is one of the few occasions where you'll be writing solutions at this level of detail.

- Declare when no more states need to be added (we have seen all those state combinations that will be seen). This needn't go as high as the full powerset in most cases.

- Epsilon-closure of I: {I, B, C}
- Fire 0 from {I, B, C}
  - State I yields {I}, which epsilon-closes into {I, B, C}
  - State B yields {D}, which epsilon-closes into {C, D}
  - State C yields the empty set
  - The union of these three results gives {I, B, C, D}
- Fire 1 from {I, B, C}
  - State I yields {I}, which epsilon-closes into {I, B, C}
  - State B yields the empty set
  - State C yields {F}, which epsilon-closes into {F}
  - The union of these three results gives {I, B, C, F}
- Fire 0 from {I, B, C, D}
  - State I yields {I}, which epsilon-closes into {I, B, C}
  - State B yields {D}, which epsilon-closes into {C, D}
  - State C yields the empty set
  - State D yields {F}, which epsilon-closes into {F}
  - The union of these three results gives {I, B, C, D, F}
- Fire 1 from {I, B, C, D}
  - State I yields {I}, which epsilon-closes into {I, B, C}
  - State B yields the empty set
  - State C yields {F}, which epsilon-closes into {F}
  - State D yields {F}, which epsilon-closes into {F}
  - The union of these three results gives {I, B, C, F}
- Fire 0 from {I, B, C, F}
  - State I yields {I}, which epsilon-closes into {I, B, C}
  - State B yields {D}, which epsilon-closes into {C, D}
  - State C yields the empty set
  - State F yields the empty set
  - The union of these three results gives {I, B, C, D}
- Fire 1 from {I, B, C, F}
  - State I yields {I}, which epsilon-closes into {I, B, C}
  - State B yields the empty set
  - State C yields {F}, which epsilon-closes into {F}
  - State F yields the empty set
  - The union of these three results gives {I, B, C, F}
- Fire 0 from {I, B, C, D, F}
  - State I yields {I}, which epsilon-closes into {I, B, C}
  - State B yields {D}, which epsilon-closes into {C, D}
  - State C yields the empty set
  - State D yields {F}, which epsilon-closes into {F}

  - State F yields the empty set
  - The union of these three results gives {I, B, C, D, F}
- Fire 1 from {I, B, C, D, F}
  - State I yields {I}, which epsilon-closes into {I, B, C}
  - State B yields the empty set
  - State C yields {F}, which epsilon-closes into {F}
  - State D yields {F}, which epsilon-closes into {F}
  - State F yields the empty set
  - The union of these three results gives {I, B, C, F}
- There are no new states

(b) **Using Jove**, obtain the DFA for `N5`, calling it `JD5`. Then in your PDF, **by hand**, write down two bullets pointing out whether `HD5` and `JD5` are language-equivalent or isomorphic.

- HD5 and JD5 have the same number of states, as well as identical transitions between equivalent states.
- Therefore HD5 and JD5 are language-equivalent as well as isomorphic.