CS 3100, Fall 2017, Assignment 2 – 160 pts total
Given 9/5 – Due 9/19 by 11pm on Canvas

Name: **UNID:**

How to answer and turn in this assignment

- Submit **three files** named as follows. In the following, `UDDDDDDD` stands for your UNID (each "D" is a digit). For instance a unid of the form `U1234567` is an example.
  1. `A2W_UDDDDDDD.pdf`: Your written solutions.
  2. `A2J_UDDDDDDD.ipynb` : Your Jupyter notebook solutions obtained by adding to `A2J.ipynb` that we provide. When I say *your file* `A2J.ipynb`, it refers to this renamed entity.
  3. `DFAUnit2_UDDDDDDD.ipynb` : An additional Jupyter notebook solution obtained by modifying a file with name `DFAUnit2.ipynb` that we will be providing you.
- The PDF submission must contain your "written" solutions. The PDF may be obtained by writing neatly and scanning, or by using any word-processing system that can produce mathematical symbols exactly as we define. Non-standard mathematical symbols cannot be present in lieu of the ones we define and use in our book.
- We will provide an overleaf template for those who like to edit an answer template and turn in a neat PDF. Overleaf is a latex processor (see www.overleaf.com).
- Your code must be well commented. Code comments must pertain to your code and it is no substitute for an "English answer" to a question.
- All English explanations in Jove are to be provided using Markdown—a simple facility to insert comments into a Jupyter file. Your Jupyter notebook file must run without any hitches. No points for notebooks that crash.
- You must not remove any cells from the file A2J.ipynb or DFAUnit2.ipynb that we provide. You must only add cells. This will allow our auto-grading script to work.
- The graders of questions are named against each question, for follow-up you may need to discuss issues or questions with grading. Questions on clarification can be asked to any TA or the instructor—not just the TA named.
- Each question will suggest where all an answer is expected. It will be "**by hand**," "**by Jove**," or "**by both**" (show by hand in PDF, and also have an entry in an answer-cell in Jove).

1 Asg-2 Questions

1. **20 points:** (Sergio)

   (a) Study our book's Section 5.3.2 and Figure 5.3. Now, for the "MSB-first" scenario, **by hand** (in your PDF), design a DFA that accepts all strings that represent binary numbers where the number modulo 5 equals 0 (in the equations in Figure 5.3, you must use $N = 5$). Calculate the state names resulting from S1 upon b=1 and S3 upon b=1 using this equation in full detail (i.e., state with name sN goes to state with name sM where M is calculated from N using the said equation; detail this calculation for S1 and S3). For the remaining state names, you can do the calculation "in your mind."

   Notice that the book clearly specifies what to make of ε; it even shows the sequence of Boolean truths emitted. Pay attention to this in your solution.

      i. Detail for S1's next state upon an arrival of b=1 here.
      ii. Detail for S3's next state upon an arrival of b=1 here.

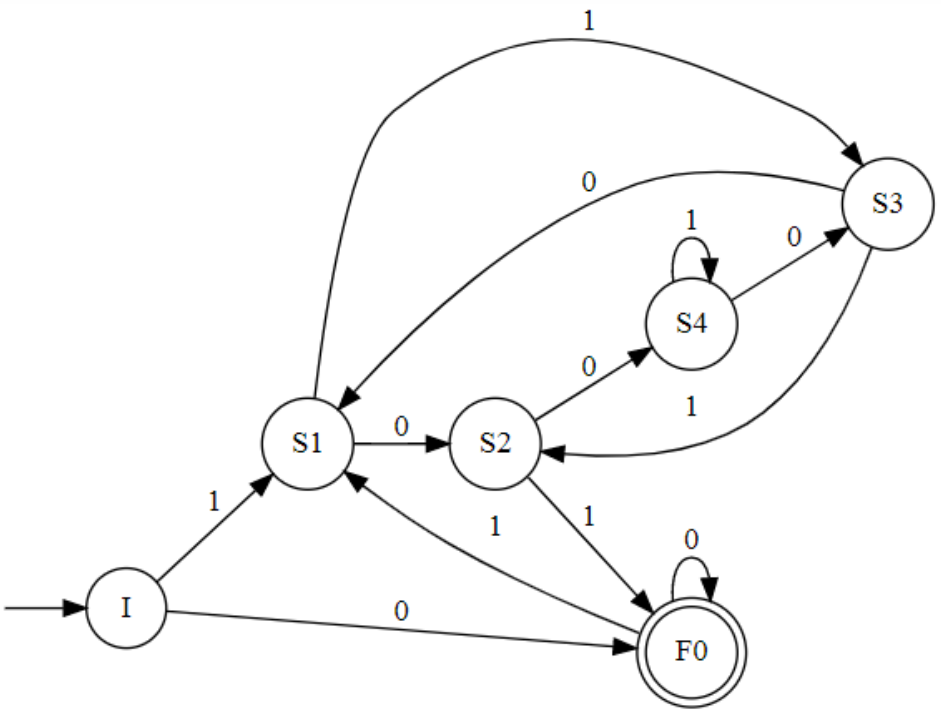
   (b) Now, **by Jove**, enter your DFA into Jove's cells of *your* file `A2J.ipynb`, and demonstrate that the DFA is working, using a test that you create using the `nthnumeric` function, as illustrated in `Drive_DFA_Unit1.ipynb`. Do the testing for `range(64)`. *Clearly point out* in a Jove markdown cell that things are working till this point of testing, at least.

   (c) **By Jove**, in designated markdown cells of Jove, describe the following functions in a few lines each. Describe how their code is organized. I don't mind if you reuse my descriptions (plz make an attempt to really understand the functions, though):

      i. Describe `totalize_dfa` in a few lines ("few lines" is about six).
      ii. Describe `accepts_dfa` in a few lines
      iii. Describe `union_dfa` in a few lines
      iv. Describe `pruneUnreach` in a few lines
      v. Describe `langeq_dfa` in a few lines
      vi. Describe `fixptDist` in a few lines

   Solution:

   (a) **by hand** (in your PDF), modulo 5 is 0
      i. With the arrival of $b = 1$, the number represented is left shifted once and 1 is added to that result. Therefore, the next state for $S1$ upon the arrival of $b = 1$ is $(1 * 2 + 1) \mod 5 \Rightarrow S3$
      ii. The same formula as in $i.$ applies. Therefore, the next state for $S3$ upon an arrival of $b = 1$ is $(3 * 2 + 1) \mod 5 \Rightarrow S2$



iii.

(b) Insert answer in *your* `A2J.ipynb` file, **by Jove**, showing how mod 5 is 0 demonstrated and tested.

(c) **By Jove**,

i. Describe `totalize_dfa` in a few lines

ii. Describe `accepts_dfa` in a few lines

iii. Describe `union_dfa` in a few lines

iv. Describe `pruneUnreach` in a few lines

v. Describe `langeq_dfa` in a few lines

vi. Describe `fixptDist` in a few lines

——— END OF QUESTION 1 ———

2. **40 points:** (Khalid)

It is quite likely that after reading this question you still have doubts. Just go through the Jove template and you'll know exactly what to answer where!

Read and understand Figure 5.4 which teaches you how to design a DFA by carrying input information in the state names.

(a) Using a similar approach to name DFA states, **by Jove**, design a DFA for the language of strings over $\{0, 1\}$ that begin with 110 and end with 011. Call this DFA `DQ1`. Your solution must be entered in a Jove cell of file A2J.ipynb.

Hint: There are a few subtle differences. You don't need to exactly store all the 0s and 1s seen so far exactly. Thus if you have seen an input 110110, then you can *pretend* that you've in effect seen 110. Thus, state S11011, upon seeing a 0, can go to state S110. Got the rest of the story?

Hint: Best way to design this DFA:

- Draw the transitions you care about. Thus, the first transition from state "I" can be on a 1. Don't worry what happens when a 0 comes (you know it should go into the black-hole state). We will fix it momentarily.
- Thus, you end up specifying a partial DFA (be building `pDQ1`, which is a partial DFA).
- Call `totalize_dfa` to totalize the `pDQ1` you are building to get the "real" `DQ1` DFA.
- Draw it using `dotObj_dfa_w_bh(DQ1)` to see that you have indeed all the edges!
- All this will be clear from the Jove template. We will also work out similar questions in class.

You must demonstrate that your solution is working, as follows:

(b) **By Jove**, design a DFA for strings that begin with 110 (call this DFA `DQ1b`). Again, design `pDQ1b` and totalize it; that will be more effortless.

(c) **By Jove**, design a DFA for strings that end with 011 (call this DFA `DQ1e`). Again, design `pDQ1e` and totalize it; that will be more effortless.

(d) **By Jove**, take the intersection of `DQ1b` and `DQ1e` and call it `DQ1be`.

(e) **By Jove**, show that `DQ1be` is language-equivalent to `DQ1`.

(f) Are these machines also isomorphic? If so, how come? If not why not? Answer as a text answer field a markdown cell **by Jove** (not by hand in your PDF).

(g) **By Jove**, minimize `DQ1be` and `DQ1` and call the results of the minimization `DQ1bem` and `DQ1m`.

(h) Show (**by Jove**) that these minimal machines are isomorphic.

Solution:

Jove solution – code and markdown.

——— END OF QUESTION 2 ———

3. **25 points:** (Paridhi)

In this question, whenever we ask you to do something "by hand", you may be tempted to take the easy path and do it "merely by Jove." But then, if you are asked to do a similar problem in the exam by hand, you'll have no Jove (nor joy).

So please do it by hand and then your **must check your work "by Jove".** Show the Jove work in your `A2J.ipynb`.

The alphabet involved is $\{0, 1\}$

(a) Design a DFA **by both** (in your PDF, and then checked by Jove) for the set of **all** strings that do not begin with a 1. Call this `nB1`. *Note: ε does not begin with* 1.

(b) Design a DFA **by both** for the set of **all** strings that do not end with a 0. Call this `nE0`. *Note: ε does not end with* 0.

(c) Obtain the union of `nB1` and `nE0` **by both**. Call this `nB1ORnE0`.

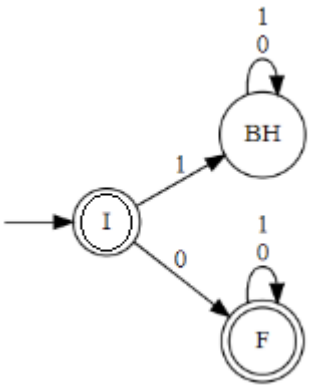
(d) Complement `nB1ORnE0` **by both**. Call it `CnB1ORnE0`.

(e) Now, **by both**, design a DFA for the set of strings that begin with a 1 *and* end with a 0. Call this `B1ANDE0`.

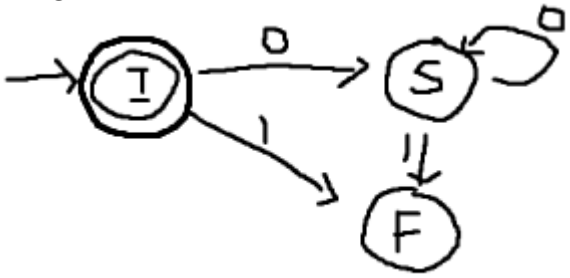
(f) **By both**, compare `B1ANDE0` against `CnB1ORnE0`. Are they language-equivalent? Isomorphic? Explain the answers you obtain if one is true and the other is false.

Solution:

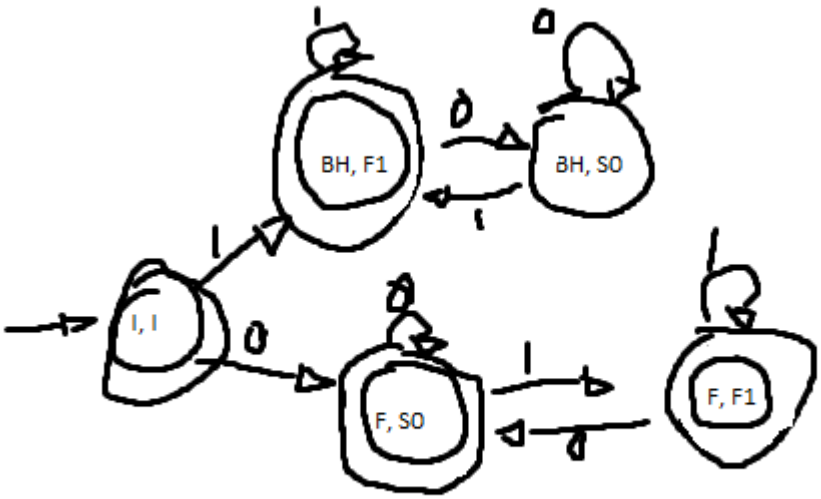
(a)



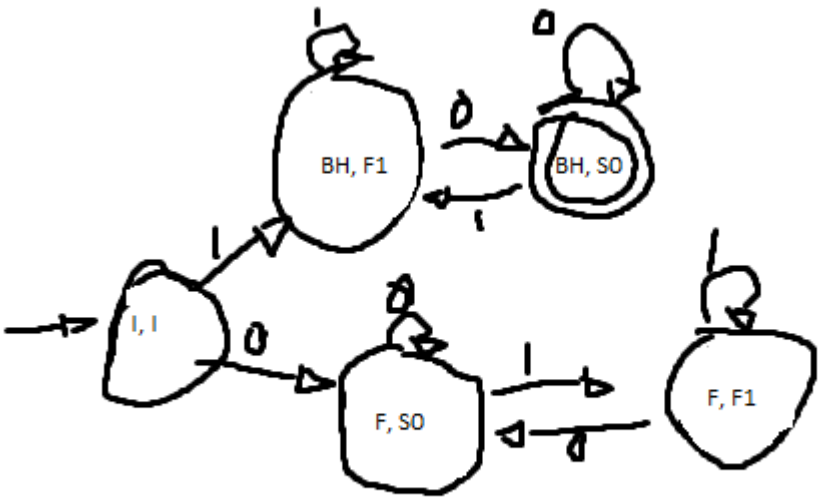
(b) Note that I do not reuse the initial state. This DFA could be made simpler if I did. Also forgot to double circle the final state. The final state is the one



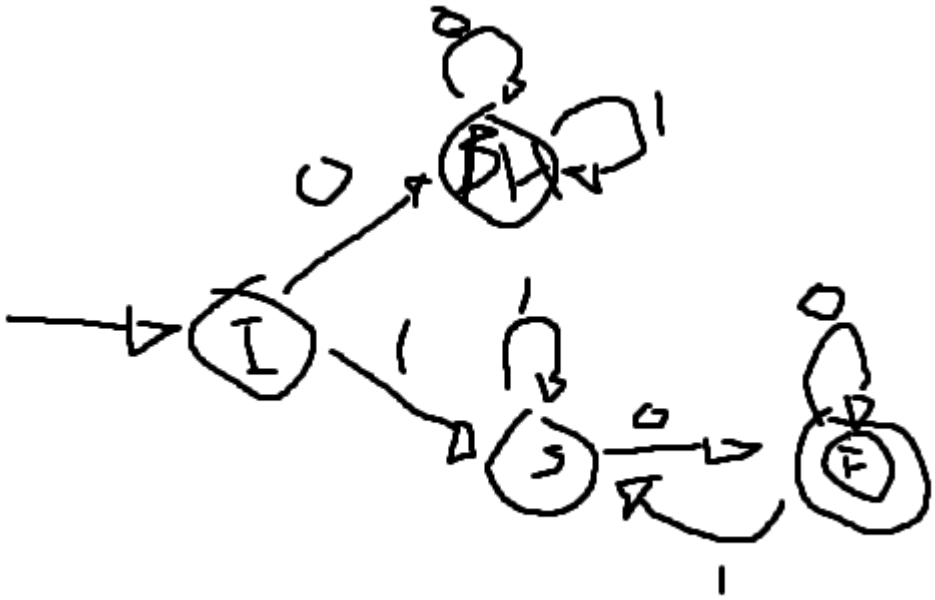
labeled F.



(c)



(d)



(e)

(f) They are language equivalent but not isomorphic. That is because the graph is not minimal for the complemented union version, but the graph is minimal for the version I created. Minimizing both DFAs will make them isomorphic.

——— END OF QUESTION 3 ———

4. **25 points:** (Vinu)

   This whole question is answered **by hand** *only.*

   (a) Prove by the Pumping Lemma that the language of strings over $\{0, 1\}$ with an equal number of 0s and 1s is not regular. Clearly show all your steps as per the template below:

      i. Follow the outline of the theorem in Section 4.7.1
      ii. Assume L is regular
      iii. State the initial string s chosen in L
      iv. State the length of this string as a function of the presumed number of states N of the DFA. Show an example string of this length that involves N also. ("We pick a string containing this many 0s and this many 1s and such." Involve N in "this many.")
      v. Now interpret x, y, and z. That is, $s = xyz$ must be true.

      Note that I forgot to mention in the book that $|xy| \leq N$ (here $|xy|$ is the length of string xy). I'll add it in the book later. This means that when you consider the first N characters that have been processed by

a "supposed DFA of N states," it would definitely have repeated a state in that march.[1]

vi. Show that you can "pump y and get out." **Argue in a sentence or two that every x, y, z split of y has been considered.**

vii. State why this is a contradiction that L is regular. Be very clear.

(b) Now show that the language of strings of the form 0^i1^j where $i \neq j$ is **not regular**. Again mention all the above steps per the above template. *Do not use any language closure properties. You must get a contradiction.*

i. State the initial string s chosen in L. **Strong hint:** You must choose $s = 0^N1^{N+N!}$—yes, I'm not joking. Try something else and ask us why that won't work, please!

ii. State the length of this string as a function of the presumed number of states N of the DFA. Show an example string of this length that involves N also. ("We pick a string containing this many 0s and this many 1s and such." Involve N in "this many.")

iii. Now interpret x, y, and z. That is, $s = xyz$ must be true.

iv. Show that you can "pump y and get out." **Argue in a sentence or two that every x, y, z split of y has been considered.**

v. State why this is a contradiction that L is regular. Be very clear.

Solution:

(a) Prove for eq 0,1, meaning we are going to argue that the language

$$eq_{01} = \{w \ : \ \#_0(w) = \#_1(w)\}$$

is not regular

i. The pumping theorem states that any regular language L has a DFA D_L whose language is L and any string $s \in L(D_L)$ that is at least N long can be read as xyz such that y is nonempty and $xy^*z \in L$ is true

ii. Let us assume the language $L = \{w : \#_0(w) = \#_1(w)\}$ is regular. Because L is regular, it has an associated minimal DFA. Let us call the number of states this DFA has N.

iii. We choose a string $s = 0^N1^N$. In fact, we can choose s to be any string in L and arrive at a contradiction with infinite descent; however, it is easier to simply choose $s = 0^N1^N$.

iv. The length of s is $2 * N$.

v. By the pumping lemma, we have that s must be able to be split into xyz such that $xy^*z \in L$. Moreover, because $|xy| \leq N$, we have that y is a substring of 0^N.

[1] Memory aid: think of duck digits and webs as in the book.

vi. Because y is guaranteed to be nonempty consists of only zeros, adding/removing copies of y will increase/decrease the number of 0s in s without changing the number of 1s. Every x, y, z split of y has been considered because it does not matter the length of y, only that it consists of only zeros and is nonempty, which is given by the pumping lemma.

vii. This is a contradiction because xy^*z cannot be in L for the string s we chose under any possible split, contrary to the pumping lemma. Therefore, our assumption that L is regular must be wrong, and L must be an irregular language.

(b) Now show that the language of strings of the form 0^i1^j where $i \neq j$ is **not regular**.

i. We choose $s \in L$ such that $s = 0^N1^{N+N!}$, where N is the number of states in the minimal DFA for L (we assume L is regular).

ii. The length of s is $2N + N!$.

iii. By the pumping lemma, s must be able to be split into x, y, z such that $xy^*z \in L$ and $xy \leq N$. Because $xy \leq N$, we have again that y must be a substring of 0^N.

iv. Because y is nonempty and y is a substring of 0^N, we have that $len(y) \in \{1, 2, 3, ...N\}$. We know that $\frac{N!}{len(y)} \in \mathbb{Z}$ because $N!$ evenly divides $\{1, 2, 3, ...N\}$. Therefore, by pumping y $\frac{N!}{len(y)}$ times, we can have a string with $N! + N$ 0s and $N! + N$ 1s (since y consists of only 0s). This string cannot be in L, which consists only of strings with unequal 0s and 1s.

v. We see that for any y we choose, it is not possible that $xy^*z \in L$. Therefore, our assumption that L is regular must be incorrect.

——— END OF QUESTION 4 ———

5. **25 points:** (Saeed)

(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.

```
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
''')
```

(b) **By Jove** (in your Jove notebook cells), check your answer and show that the minimization is correct. Write your conclusions in the markdown part of Jove.

Solution:

(a) We first create pairs between all states, and mark their distinguishability distance with a -1. Thusly, we create the pairs (IF - A, -1), (IF - A1, -1), (IF - B1, -1), (IF, B, -1), (A, A1, -1), (A, B1, -1), (A, B, -1), (A1, B1, -1), (A1, B, -1), (B1, B, -1).

We mark all final - non final state pairs as having distinguishability distance zero, giving (IF - A, 0), (IF - A1, 0), (IF - B1, 0), (IF, B, 0), (A, A1, -1), (A, B1, -1), (A, B, -1), (A1, B1, -1), (A1, B, -1), (B1, B, -1). Because IF is the only final state, all (IF, x) pairs are marked as 0-distinguishable.

We take all remaining indistinguishable pairs and check their output when fed both $0, 1$ to see if they lead to 0-distinguishable pairs and mark these as 1-distinguishable, giving (IF - A, 0), (IF - A1, 0), (IF - B1, 0), (IF, B, 0), (A, A1, -1), (A, B1, 1), (A, B, 1), (A1, B1, 1), (A1, B, 1), (B1, B, -1). (A, B1) was marked as 1-distinguishable because feeding a 0 leads to the pair (IF, B1) which is zero distinguishable. Similarly, feeding a zero to (A, B) leads to the pair (IF, B1) which is 0-distinguishable. Feeding a zero to (A1, B1) leads to (IF, B) and feeding a zero to (A1, B) also leads to (IF, B), which is 0-distinguishable. These pairs are marked as 1-distinguishable.

Out of the remaining indistinguishable pairs, we repeat this process to generate the 2-distinguishable pairs. However, there are no 2-distinguishable pairs, so the pairs remain the same (IF - A, 0), (IF - A1, 0), (IF - B1, 0), (IF, B, 0), (A, A1, -1), (A, B1, 1), (A, B, 1), (A1, B1, 1), (A1, B, 1), (B1, B, -1).

Because there were no changes after iterating once more, we see that the indistinguishable pairs are (A, A1) and (B1, B). These form the two equivalence classes (A, A1) and (B1, B). We combine these to get the states A1A and B1B, and remake the DFA to be

$\{IF : 0 \Rightarrow A1A, IF : 1 \Rightarrow IF, A1A : 0 \Rightarrow B1B, A1A : 1 \Rightarrow A1A, B1B : 0 \Rightarrow IF, B1B : 1 \Rightarrow B1B\}$

We essentially removed the states A, A1, B, and B1 from the graph, and instead replaced all mentions of A or A1 with A1A, and all mentions of B or B1 with B1B.

(b) This answer appears in a Jove cell also, after you apply the `min_dfa` function.

END OF QUESTION 5

6. **25 points:** (Mike)

   (Answer by Jove)

   (a) Run `DFAUnit2` cell by cell, and write a summary of your understanding of the contents of these cells in a markdown cell consisting of a few bullets. The bullets must reflect your understanding, and must be about 20-30 words per cell.

   Note: TA Mike Bentley has merged some cells for reducing your tedium. Also clear answers count more than the number of words, although with the merged cells you will likely need 20 words.

   (b) We want a DFA for the language with an odd number of 0s **or** an even number of 1s. The alphabet is $\{0, 1\}$. Obtain this DFA by designing it in the markdown syntax of Jove, and set it aside, calling it `DFAOdd0OREven1`.

   (c) Now obtain the same DFA via DeMorgan's law, and show the whole construction. This construction will obtain the **complement of the respective languages,** obtain the intersection, and then complement the whole result. Prove the isomorphism of this machine and the machine `DFAOdd0OREven1` after minimizations.

   Note: Complement of the respective languages followed by DeMorgan will need this:

   - Obtain an additional DFA for EVEN0
   - Obtain an additional DFA for ODD1
   - AND these (intersect these)
   - NEGATE (Complement)

   Then you are, by DeMorgan, going to get
   DFAOdd0OREven1

   Solution:

> Answer in the Jove cells of *your* `DFAUnit2.ipynb` for 6(a).
>
> For 6(b) and 6(c), answer in A2J.ipynb.

END OF QUESTION 6

END OF ASSIGNMENT-2.

- I will also be issuing remarks about the assignment plus additions and clarifications on the relevant chapters of the book in these pages. I will refrain from editing the book chapters directly.
- If there are any changes to the assignment, it will be in this pages or the pages before this.

General Notes

I will in general type-up hints or suggestions here. This way everyone can see it. Whenever I make an entry here, you will of course get an announcement.

1. **Note that I forgot to mention in the book that $|xy| \leq N$ (here $|xy|$ is the length of string xy). I'll add it in the book later.** This means that when you consider the first N characters that have been processed by a "supposed DFA of N states," it would definitely have repeated a state in that march.[2]

2. In the book, where I say N goes to $2 \cdot N + b$, I mean b is a single bit (either a 0 or a 1).

3. MSB stands for most-significant bit and LSB stands for least-significant bit.

   Thus in a binary number 100 which is decimal 4, 1 is the MSB and 0 is the LSB.

[2]Memory aid: think of duck digits and webs as in the book.