UNCA CSCI 431
Exam 1 Fall 2019
17 October 2019 – 3:15 pm to 4:55 pm

You may use your notes, printouts, scratch paper, and your textbook. You many not use any calculators, electronic devices, or help from any other source or person.

Anyone needing a break during the exam must leave their exam with the instructor.

This exam must be turned in before 4:55 PM.

Name:_____

Note: textbook versus `grep`

- The textbook uses $E \cup F$ for union of E and F, grep uses E|F .
- The textbook uses $E \circ F$ for concatenation of E and F. **grep** uses EF.
- The textbook uses E^* for the Kleene star. **grep** uses E*.
- The textbook uses Σ to match any character. **grep** uses the period.

Use whichever you wish in your answer.

Also, notice the subtle difference between u, U, and ∪. The first two are letters. The last is the union operator (or \cup in LaTeX).

Each problem is worth 20 points. The first problem is easy!

Problem 1: Regular expressions
Describe what is matched by the following regular expressions? For each of the following two expressions, ***give two examples of strings*** that belong to **each of** the corresponding regular languages.

borr(ow ∪ y) *or in* ***grep***, `borr(ow|y)`

There are only two strings that match: `borrow` *or* `borry`.

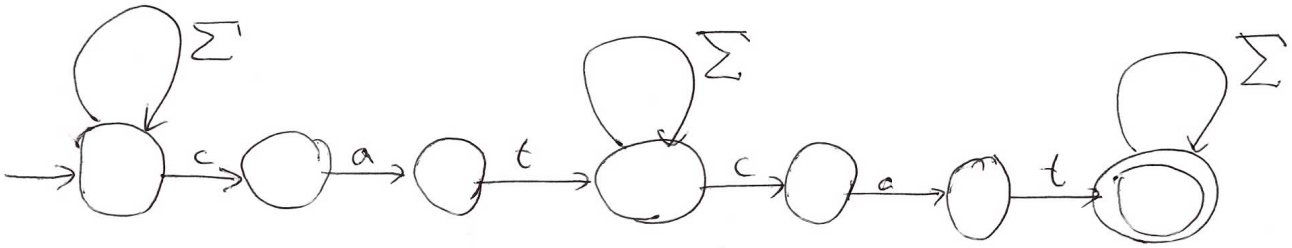
hissss* *or in* ***grep***, `hissss*`

Matching strings start with `hisss`. Then you can add all the `s`'s you want. Examples are `hisss` and `hisssssss`.

Several people didn't "describe" what is matched. If someone only gave `hisss` and `hissss` as examples, it wasn't clear that the number of `s`'s was unlimited.

Problem 2: First NFA

Draw a state diagram for an NFA (nondeterminate finite automata) that would accept strings from the alphabet Σ = {a, c, t} which contain the three-letter substring cat *at least two* times.



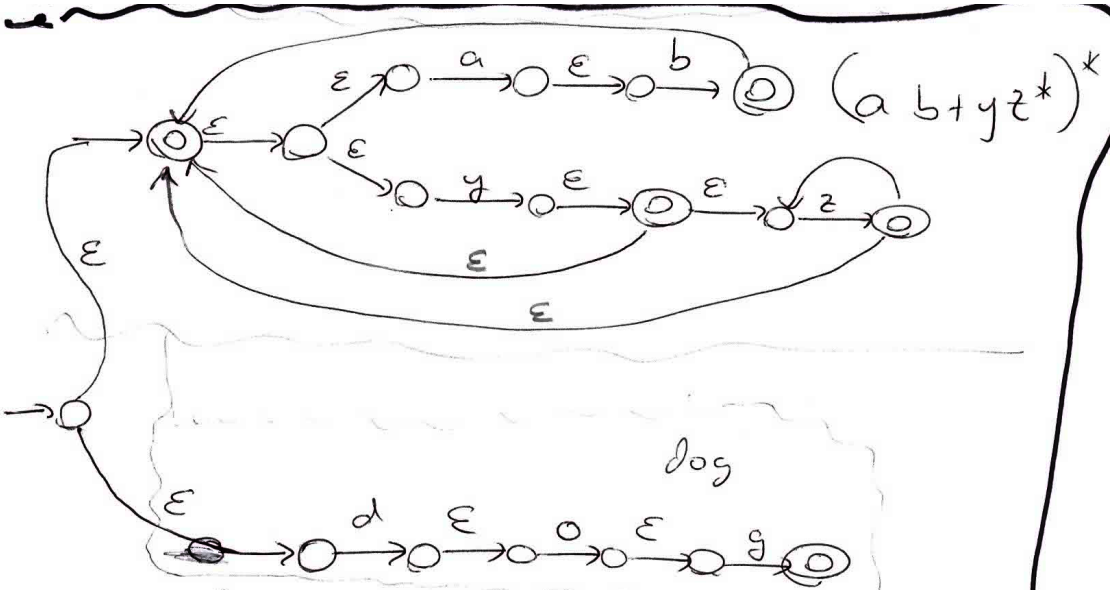
A common problem was leaving out the loops for Σ

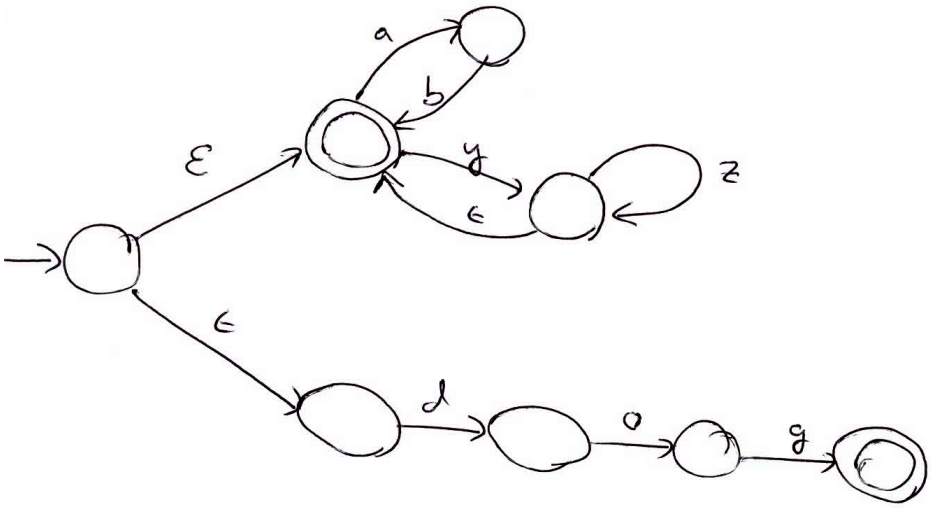
Problem 3: Second NFA

Use the mechanical RE-to-NFA conversion algorithm described in Chapter 1 of the textbook to construct an NFA for the following regular expression over the alphabet Σ = {a, …, z}. This may take a while.

(ab u yz*)* u dog or in **grep**: (ab|yz*)*|dog

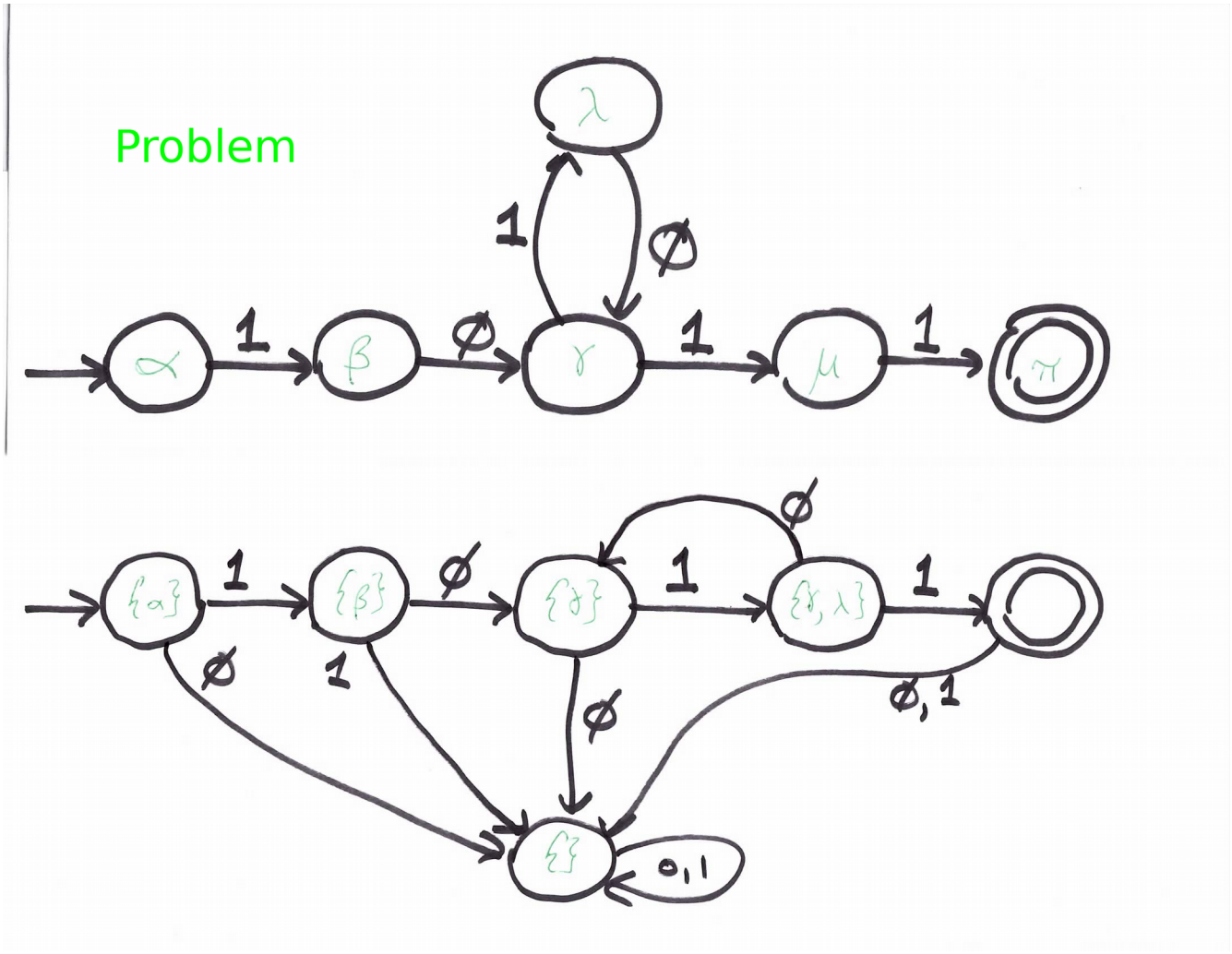
The top one (tries to) follow the textbook construction strictly.
The bottom one makes some optimizations.





Problem 4: NFA to DFA

Draw a DFA equivalent to the NFA (which has a relation to the preamble to Ethernet packets) shown below.



Problem 5: Pumping Lemma (Theorem 1.70)

If A is a regular language then there is a number p (the pumping length) where if s is any string in A of length at least p, then s may be divided into three pieces $s = xyz$, satisfying the following conditions:

1. for each $i \geq 0$, $xy^iz \in A$,
2. $|y| > 0$, and
3. $|xy| \leq p$.

Problem 5A: A tricky question for a simple RE

The language generated by the regular expression

- `borr(ow u y)`

mentioned in Problem 1 **must** obey the Pumping Lemma even though there is nothing to pump! *How can this be the case!!??* What value of p could make be appropriate for this two-element regular language? (**Explain your answer.** You may have to use a bit of Chapter 0 reasoning!)

If p were anything greater than 6, then no string in this finite language would be "of length at least p." Consequently, the lemma holds trivially in this situation. Many people seemed to have trouble with this one.

Problem 5B: A more interesting regular language

Now consider the other regular expression of Problem 1

- `hissss*`

In this case, there is something to pump. What would be an appropriate pumping length p for this regular language? **Justify your answer!**

If p were 5, we could take a string similar to `hisss` and pump it down to something like `hiss` which is not in the language. However, any p greater than 5 would work well. You'd just be pumping `s`'s. I'd prefer 6, because it is the smallest possible pumping length.

Problem 5C:

Using your value of p from Problem 5B, use the pumping lemma to show that the string

- `hisssssssssssssssssssssssssssssssssssssssssssssssssss`

(that's 50 s's) belongs to this language.

Ugh. I simplified Problem 5B by replacing "smallest possible pumping length" with "an appropriate pumping length".

If p was 6, then x could be `hisss` and y could be `s` and z could be the empty string. In that case $xy^{45}z$ would give the target string. I will be understanding in grading this one.