#1 Language Complement

The *complement* of a language A is defined to be the set of strings *not* in the language:

$$\overline{A} = \{ w \mid w \notin A \}.$$

Prove that if A is regular, then $\overline{A}$ is also regular.

N.b. You may prove this either by exhibiting a DFA, NFA, or Regular Expression that recognizes $\overline{A}$. One of these should be very straightforward; part of this question is just getting you to think about which one that is.

#2 Thinking inductively

Another way of defining the natural numbers is as the least set $\mathbb{N}$ such that

1. $0 \in \mathbb{N}$, and
2. if $n \in \mathbb{N}$, then $n + 1 \in \mathbb{N}$.

Alternatively, we can express the same by the following grammar:

$$\text{Naturals } n ::= 0 \mid n + 1$$

1. Draw an analogy between the definition of the naturals above and our formal definition of regular expressions (Sipser def. 1.52). Which cases are non-recursive? Which are recursive?
2. Draw an analogy between proof by induction over $\mathbb{N}$ and proof by induction over regular expressions. Which are the base cases? Which are the step cases? In each of these cases, where do you invoke the inductive hypothesis? That is, over what data?

#3 Syntactic sugar (is sweet)

Lab 1A asked you to show that regular expressions such as R^+ and $R^?$ can be *elaborated* to the primitively defined operators of union, concatenation, star, and so forth.

1. What would be the consequence of instead modifying our formal definition of regular expressions? Would Theorem 1.54 (Sipser pg. 66) still be true? If not, how would we need to modify the proof?
2. Draw an analogy between changing the definition of regular expressions and changing the definition of data types in your programming. In each case, what must you refactor?

#4 Ending this lab with a bang

Define the *bang* (!) of a regular expression over the alphabet $\Sigma = \{0, 1\}$ by the following inductive definition.

$$0^! = 1 \tag{1}$$
$$1^! = 0 \tag{2}$$
$$\epsilon^! = \epsilon \tag{3}$$
$$\emptyset^! = \emptyset \tag{4}$$
$$(A \cup B)^! = A^! \cup B^! \tag{5}$$
$$(AB)^! = A^! B^! \tag{6}$$
$$(A^*)^! = (A^!)^* \tag{7}$$

N.b. that Sipser uses both the notation $A \circ B$ and AB to denote the concatenation of A and B. I have omitted the former.

Intuitively, the bang operator simply flips the 0's and 1's in a regular expression. Here are some examples.

$$(01)^! = 10 \tag{8}$$
$$(00 \cup 1 \cup \epsilon)^! = 11 \cup 0 \cup \epsilon \tag{9}$$
$$((01)^*)^! = (10)^* \tag{10}$$

For illustration, consider the following sample computation.

$$(00 \cup 1^* \cup \epsilon)^! \tag{11}$$
$$= ((00)^! \cup (1^*)! \cup \epsilon^!) \tag{12}$$
$$= (0^! 0^! \cup (1^!)^* \cup \epsilon^!) \tag{13}$$
$$= (11 \cup 0^* \cup \epsilon) \tag{14}$$

For this exercise, we will use *structural induction* to show that that the bang operator preserves regularity. Suppose R is a regular expression and perform a *case analysis* on the following cases.

(a) $R = 0$

(b) $R = 1$

(c) $R = \epsilon$

(d) $R = \emptyset$

(e) $R = (R_1 \cup R_2)$

(f) $R = R_1 R_2$

(g) $R = R_1^*$

In each case, show that $R^!$ is regular.