|  |  |
| --- | --- |
| Probl  # | Pts |
| 1 |  |
| 2 |  |
| 3 |  |
| 4 |  |
| 5 |  |
| 6 |  |
| 7 |  |
| Sum | 100% |

GRADE CALCULATION = round (

(pts#1/.x1+ pts#2/.x2+ pts#3/.x3+ pts#4/.x4+ pts#5/.x5+ pts#6/.x6+ pts#1/.x7)/7, 0)

Problem #1

Given the following NFA: N = (Q, , , S0, {S4}) where S0 is the starting state, {S4} is the set of final (accepting) states (one state), Q = {S0, S1, S2, S3, S4},  = {1, 2, 3} and the transition summary table is:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | 1 | 2 | 3 |
|  | S0 | {S0, S1} | {S0, S2} | {S0, S3} |
|  | S1 | {S1, S4} | {S1} | {S1} |
|  | S2 | {S2} | {S2, S4} | {S2} |
|  | S3 | {S3} | {S3} | {S3, S4} |
|  | S4 | Ø | Ø | Ø |

1. Draw the NFA. Make it as clean and neat as possible.
2. Use the -closure algorithm to find its DFA. The DFA must be given as a table, not as a “messy” picture. Do not forget to *underline ALL accepting states*.
3. Show all *possible* -derivations for the strings given below to determine if they are or are not in L(N)?. (USE the abbreviation nafs = “not accepting final state”, where appropriate).
   1. 31232
   2. 33221

Problem #2

Given  = {a}. Let L1 = {w : |w|mod 3 = 1} and L2 = {w : |w|mod 5 = 1}.

1. Design a DFA for L1.
2. Design a DFA for L2.
3. Design a NFA for L3 = L1 ∪ L2
4. Design a NFA for L4 = L1 ∩ L2.

(3a): Given  = {a, b}, write a regular definition for the following language:

L(w) = {w | w ends in *aa* and contains the substring *aba*}

(3b): Given  = {0, 1}, describe in English the languages denoted by the following regular expression: (0 + 1)\*101(0 + 1)\*

(3c): Given  = {a, b, c}, write a regular expression for the set: "All strings of a’s and b’s and c’s containing exactly two a's "

(3d): Given  = {a, b}, write a regular expression for the set: "All strings of a’s and b’s beginning with *bb* and not having three consecutive *a*'s "

(3e): Given  = {0, 1}, describe in English the languages denoted by the following regular expression: (0+1)\*(00+01+11)++0+1

Problem #4

Languages La, Lb, Lc and Ld are all regular languages. Using the properties of the Regular Languages prove/disprove the following statements:

~~The set ((L~~~~1~~~~)~~~~c~~∩ ~~L~~~~2~~~~)~~~~c~~ ~~is a not a regular language~~

1. The set (La)c∩(Lb)R – (Lc∪Ld)La is a regular language.
2. The set (La – ) ∪ {} is a regular language.
3. L = {anbn: n > 0 & n < 2000} is not a regular ~~ranguage~~ language.

First, say that the statement is True or False and then explain (justify) in detail your answer.

Problem #5

Given the regular expression (01\*0)(01 + 10)\*010 build its corresponding NFA

Problem #6

Consider the following alphabet:

 = {[0 0], [0 1], [1 0], [1 1] }

Here,  contains all rows of 0s and 1s of size 1 (one row) ~~2~~. A string of symbols in  gives two columns of 0 and 1’s. Consider each column to be part of a binary number, that is, all the first columns form a binary number and all the second columns form the other binary number. Let ~~and let~~:

L = {w ∈  | the binary number formed by the last columns of w is twice the binary number formed by the first columns}

For example:

w = [0 0] [0 1] [1 0] [0 0] ∈ L (because the first columns form the binary number 0010 = 2 and the second columns form the binary number 0100 = 4, and 4 is twice 2) ~~and~~ while

w = [0 0] [0 1] [1 0] [0 1] ∉ L (because the first columns form the binary number 0010 = 2 and the second columns form the binary number 0101 = 5, and 5 is not twice 2).

Show that L is regular.

Problem #7

~~Let Ln = {w | w is a binary number that is multiple of n ≥ 1}. Show that for each n, Ln is regular.~~

Let Ln = {w | w = ak, a ∈ , where (k mod n = 0) for all n ≥ 1}. Carefully design a DFA for the case when n = 6. Then generalize your result for any arbitrary value of n ≥ 1 to show that Ln is regular for all n.