**CS 315 Theory of Automata (Spring 2018)**

**Assignment 1 - Solution**

**Deterministic Finite Automata**

Q1. Draw DFA’s for each of the following sets.

1. The set of strings in of the form where is even and .

|  |
| --- |
| Note: check for corner cases like and . |

1. The set of strings on containing no more than two consecutive occurrences of the same symbol.

|  |
| --- |
| Note: do not forget that this language also accepts empty strings shown by start state as one of the accepting states. |

1. The set .

|  |
| --- |
| Note: do not forget that this language also accepts empty strings shown by start state as one of the accepting states. |

1. The set of all strings such that is any combination of characters from the alphabet, and the first and last symbols of *w* are identical.

|  |
| --- |
|  |

1. The set of all strings over alphabet with even number of 's and even number of 's such that the lengths of the strings are divisible by 3.

|  |
| --- |
| Note: use intersections of the two DFA’s mentioned in the questions. |

**Nondeterministic Finite Automata**

Q2. Convert the NFA’s given below to equivalent deterministic ones. Draw the resulting DFA and show the transition tables for both as well.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | |  |  |  |  | | --- | --- | --- | --- | | *Transition Table for NFA* | | | | |  |  |  |  | | (S) |  | - | - | | (S, F) | - |  | - | | (F) |  |  | - | | |  |  |  | | --- | --- | --- | | *Transition Table for DFA* | | | |  |  |  | | (S,F) |  |  | | (F) |  |  | | (F) |  |  | |  |  | - | | (F) | - |  | | | |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | |  |  |  |  | | --- | --- | --- | --- | | *Transition Table for NFA* | | | | |  |  |  |  | | (S) |  |  |  | |  | - |  |  | | (F) | - | - | - | | |  |  |  | | --- | --- | --- | | *Transition Table for DFA* | | | |  |  |  | | (S,F) |  |  | | (F) |  |  | | (F) | - | - | | | |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | |  |  |  | | --- | --- | --- | | *Transition Table for DFA* | | | |  | **a** | **b** | | (S) |  |  | |  |  |  | |  |  |  | |  |  |  | | (F) |  |  | | |

**Regular Expressions**

Q3. Give regular expressions to describe the following languages:

1. The set of all strings over the alphabet that contain .

|  |
| --- |
|  |

1. The set of all strings over the alphabet that contain an odd number of ’s.

|  |
| --- |
| Making a DFA for this problem is easier. One can then convert it into a regex. |

1. The set of all strings over the alphabet with an equal number of ’s and ’s, such that no prefix has two more ’s than ’s, nor two more ’s than ’s.

|  |
| --- |
|  |

**Properties of Regular Languages**

Q4. Simplify the following expressions:

|  |
| --- |
|  |

|  |
| --- |
|  |

Q5. For each of the following languages, show whether they are regular or not. If they are not regular, use the *pumping lemma* in your proof.

|  |
| --- |
| Assume to the contrary that is a regular language. This implies that pumping lemma holds for . In particular, there is a pumping length . Now consider the string . It is apparent that .  Since , by pumping lemma, there exist and such that , and . This means, for some . It must be true that . But is of the form . Since , . This is a contradiction. Hence, must not be a regular language. |

|  |
| --- |
| The proof follows the same outline as that in part (i). Consider the string , where is the pumping length. |

|  |
| --- |
| The proof follows the same outline as that in part (i). Consider the string , where is the pumping length. |

|  |
| --- |
| It is always possible to split a binary string containing at least one into two substrings such that the two substrings do not have equal number of ’s. E.g. one of them is the original string and the other one is empty. This implies, . Since regular languages are closed under the set difference operator, and , are both regular languages, L is also a regular language. |

**Programming Problems**

Q6. Write a Python program (using regular expressions library **re**) to answer the following questions:

1. Find the sum of all the digits in a string.
2. Find the number of numbers (well-formed, positive or negative integers and floats) in a string.
3. Return True if a string contains only one ‘+’.

Note: please look at the accompanying Python files. Do try the questions yourself first before looking at the solution. Test cases have also been provided for your ease.

|  |
| --- |
| Solution is contained in an accompanying Python file. |

Q7. Given a representation of a DFA, write a Python program to do the following:

1. Find the number of states and the number of transitions.
2. Return a list of all the dead states. A dead state is a non-final state that has either just a self-loop or not no outgoing transitions at all.
3. Write a function to remove all the dead states. You must do the following:
4. Remove dead states them from the given dictionary
5. Return the updated dictionary.

Note: please look at the accompanying Python files. Do try the questions yourself first before looking at the solution. Test cases have also been provided for your ease.

|  |
| --- |
| Solution is contained in an accompanying Python file. |