Problem 1 – Regular Expressions

You are asked to provide different definitions for identifiers for potential use in your new programming language, each with different required structures. The alphabet, i.e., finite set of symbols over which to define your “identifier language” ID is

Σ = {a, b, c, d, e, A, B, C, D, E, 0, 1, 2, 3, 4, $, &, _} with ID ⊂ Σ*

You may use the following regular expressions for your definitions of ID:

- letter_lower := (a | b | c | d | e)
- letter_upper := (A | B | C | D | E)
- digit := (0 | 1 | 2 | 3 | 4)
- special := ($ | & | _)

Give regular expressions for ID that satisfy the following conditions. Note: You may want to define your own regular expressions to help you define ID. All identifiers need to have at least one symbol. **You must use the regular expression syntax as presented on slide 9 in lecture 10. However, you may use the “+” operator, i.e., positive Kleene closure.**

1. All identifiers that start with an upper case letter and are not more than 8 symbols long.

2. All identifiers that start with an underscore, followed by at least one digit and one letter, and exactly one special symbol that is not an underscore. Note: Follow does not mean immediately follow within a string, but means “occurs later in the string”, which includes immediately follow.

3. All identifiers that do not have more than 3 digits.

4. All identifiers that have an odd number of upper case letters, or an even number of lower case letters, and no special symbols.
Problem 2 – DFA

Give a DFA for the following languages over the alphabet \{0, 1\}:

1. The set of all strings with an odd number of 0’s and an odd number of 1’s. The empty string is not accepted.

2. The set of all strings beginning with a 1 which, interpreted as the binary representation of an integer, is congruent to zero modulo 5, i.e., \langle \text{bitstring} \rangle \mod 5 = 0.

Problem 3 – NFA and DFA

1. Construct a nondeterministic finite automaton (NFA) for the regular expression \( d|a(bc)^* \) using Thompson’s Construction Algorithm.

2. Convert your NFA with \( \epsilon \) transitions into a DFA using the subset construction.

3. Is the DFA minimal? If not, give the minimal DFA.