**Theory Solution**

|  |  |
| --- | --- |
| 1.a) | Write down the formal definition of Turing Machine. Describe each notation. |

Definition

A Turing Machine (TM) is a mathematical model which consists of an infinite length tape divided into cells on which input is given. It consists of a head which reads the input tape. A state register stores the state of the Turing machine. After reading an input symbol, it is replaced with another symbol, its internal state is changed, and it moves from one cell to the right or left. If the TM reaches the final state, the input string is accepted, otherwise rejected.

A TM can be formally described as a 7-tuple (Q, X, ∑, δ, q0, B, F) where −

* **Q** is a finite set of states
* **X** is the tape alphabet
* **∑** is the input alphabet
* **δ** is a transition function; δ : Q × X → Q × X × {Left\_shift, Right\_shift}.
* **q0** is the initial state
* **B** is the blank symbol
* **F** is the set of final states

4(a)

What is Context-free language?

**Context-free languages** (CFLs) are generated by [context-free grammars](https://brilliant.org/wiki/context-free-grammars/). The set of all context-free languages is identical to the [set](https://brilliant.org/wiki/sets/) of languages accepted by [pushdown automata](https://brilliant.org/wiki/pushdown-automata/), and the set of [regular languages](https://brilliant.org/wiki/regular-expressions/) is a subset of context-free languages. All regular languages are context-free languages, but not all context-free languages are regular.

A **context-free grammar** is defined as a [grammar](http://en.wikipedia.org/wiki/Formal_grammar) in which every production rule is of the form A→α where A is a variable and α is a sequence of variables and terminals.

Formally, a context-free grammar can be defined as a 4-tuple (V,Σ,R,S), where V  is a finite set consisting of the variables, Σ is a finite set consisting of the terminals, R is a set of production rules (in the form mentioned above), and S∈V is the starting variable.

5(b) Give a formal description of the Pumping Lemma.

Let L be a regular language. Then there exists a constant **‘c’** such that for every string **w** in **L** −

**|w| ≥ c**

We can break **w** into three strings, **w = xyz**, such that −

* |y| > 0
* |xy| ≤ c
* For all k ≥ 0, the string xykz is also in L.

**5 or (B)**

**Pumping Lemma** is used as a proof for irregularity of a **language**. Thus, if a **language** is **regular**, it always satisfies **pumping lemma**. If there exists at least one string made from **pumping** which is not in L, then L is surely not **regular**

|  |  |
| --- | --- |
| **6 b)** | Write down the formal definition of Non-deterministic Finite Automata, and describe each notation.  In a nondeterministic finite automaton (NFA), for each state there can be zero, one, two, or more transitions corresponding to a particular symbol. Formal Definition of an NDFA An NDFA can be represented by a 5-tuple (Q, ∑, δ, q0, F) where −   * **Q** is a finite set of states. * **∑** is a finite set of symbols called the alphabets. * **δ** is the transition function where δ: Q × ∑ → 2Q   (Here the power set of Q (2Q) has been taken because in case of NDFA, from a state, transition can occur to any combination of Q states)   * **q0** is the initial state from where any input is processed (q0 ∈ Q). * **F** is a set of final state/states of Q (F ⊆ Q). |

**6 0r (b)**

1. NFA and Є-NFA

Non-deterministic Finite Automata (**NFA**) is a finite automata having zero, one or more than one moves from a given state on a given input symbol. **Epsilon NFA** is the **NFA** which contains **epsilon** move(s)/Null move(s)

NFA and DFA

