# 

# Department of Computing

**CS 354: Compiler Construction**

**Class:** BSCS-8AB

# Lab [01]: [DFA](https://lms.nust.edu.pk/portal/mod/resource/view.php?id=68232) Simulation

**Date:** 17th Sep, 2021

# Time: Friday (10:00 – 1:00)

# Instructor: Dr Syed Atif Ali Shah

# Lab Instructor: M . Danyal Sadiq

# Lab [01] : [DFA](https://lms.nust.edu.pk/portal/mod/resource/view.php?id=68232) Simulation

**Introduction**

A deterministic finite automaton (DFA)—also known as deterministic finite state machine—is a finite state machine that accepts/rejects finite strings of symbols and only produces a unique computation (or run) of the automaton for each input string. 'Deterministic' refers to the uniqueness of the computation. – Wikipedia

Construction of DFA from RE:

https://www.gatevidyalay.com/construction-of-dfa-examples-dfa-solved-examples/

**Objectives**

1. Successful understanding/implementation DFA in C/C++/Java

**Tools/Software Requirement**

1. gcc, g++, GNU Make or Visual Studio C++

**Description**

A DFA is defined as an abstract mathematical concept, but due to the deterministic nature of a DFA, it is implementable in hardware and software for solving various problems. For example, a DFA can model software that decides whether or not online user-input such as email addresses are valid. DFAs recognize exactly the set of regular languages which are, among other things, useful for doing lexical analysis and pattern matching.

**Lab Tasks**

1. Consider the following Regular Expression:

**a(bb)\*bc**

* 1. Draw a DFA for the above RE.
  2. Determine the language accepted by this automaton
  3. Implement this DFA in C/C++/Java.
     1. Your implementation should validate the input string for alphabet i.e. **∑ = {a, b, c},** before using it in the DFA.
     2. Your first implementation of DFA should use **goto** statements only.
     3. Your second implementation of DFA should use **switch** statement instead of the goto’s.
  4. Test your implementation using the following inputs:

**abc, abbc, abcd, abbbc, abbbbc**

1. Consider the following language:

**L(M) = {w | w € {a, b}\* and contains even number of a’s and b’s}**

* 1. Draw a DFA for the above language
     1. Your implementation should validate the input string for alphabet i.e. **∑ = {a, b},** before using it in the DFA.
     2. Your first implementation of DFA should use **goto** statements only.
     3. Your second implementation of DFA should use **switch** statement instead of the goto’s.
  2. Test your implementation using the following inputs:

**aa, abba, abab, aabbaabb, empty string**

**Deliverables**

You are required to upload your task (Sources & Word/PDF document) using the link created on LMS followed by a viva.