1. Briefly explain the essential difference between
   a) Call-by-value and call-by-reference. How are parameters passed in C and Java?
   b) Static scope and dynamic scope. How is scoping done in C and Java?

2. Java compilation.
   a) Draw a block diagram showing how programs are compiled and executed in Java.
   b) What is a Java just-in-time compiler?

3. Let \( L \) be the set of strings of the form \( abxba \) where \( x \) is a string of \( a \)'s, \( b \)'s, and \( c \)'s that does not contain \( ba \) as a substring.
   a) Write a regular expression for \( L \).
   b) Show how your regular expression generates the string \( ababcba \).
   c) Construct a deterministic finite automaton for \( L \).
   d) Show how your automaton processes the input \( ababcba \).

4. Consider the context-free grammar \( G \):
   \[
   S \rightarrow S + S \mid S * S \mid a.
   \]
   a) Show that \( G \) is ambiguous by constructing all parse trees for \( a + a * a \).
   b) Construct an unambiguous grammar for \( L(G) \) in which \( + \) is left associative, \( * \) is nonassociative and of higher precedence than \( + \). Draw the parse tree in your grammar for the input string \( a + a * a \).

5. Syntax-directed translation.
   a) Construct an SDTS that maps postfix expressions containing the digits 0, 1, …, 9 and the binary arithmetic operators + and * into equivalent infix expressions.
   b) Show how your SDTS translates the expression 123+*.
   c) [Extra credit, 10 pts] Modify your SDTS so that it uses the fewest possible number of parentheses in the output.