1. Describe the following languages both in terms of regular expressions and in terms of deterministic finite automata:

(a) \( \{ w \in \{0,1\}^* \mid |w| \geq 2, \text{ } w \text{ begins and ends with a 1} \} \)  
7p.

(b) \( \{ w \in \{0,1\}^* \mid |w| \geq 2, \text{ } w \text{ begins and ends with a 1, and each} \) 
\( \text{two consequent 1’s are separated by one or two 0’s} \} \).  
8p.

2. (a) Design a context-free grammar describing balanced sequences of parentheses that may also contain parallel subexpressions, e.g. “(()())()” or “(()()).”. Based on your grammar, give the parse trees for the above sequences.  
8p.

(b) Prove (precisely!) that the language discussed in part (a) can not be recognised (accepted) by a finite automaton.  
7p.

3. Design a (nondeterministic) pushdown automaton that recognises (accepts) the language 
\( L = \{ a^i b^j c^k \mid i = j \text{ or } j = k \} \). 
(Present the automaton preferably as a state diagram rather than a transition table.) Show the accepting computation sequences (“runs”) of your automaton on the inputs \( ab \) and \( abbcc \).  
15p.

4. One of the following:

(a) Design (in outline) algorithms for determining whether the language described by a regular expression \( r \) over the alphabet \( \{0,1\} \) is (a) empty, i.e. \( L(r) = \emptyset \), (b) contains all possible binary strings, i.e. \( L(r) = \{0,1\}^* \).  
15p.

(b) Assume that you are explaining the key contents of the course “Introduction to Theoretical Computer Science” to a friend who has not yet taken the course. Describe the Church-Turing thesis to her, and convince her of the fact that there are problems that cannot be solved by a computer.  
15p.

Total 60p.