1. [15 marks altogether]

   (a) [4 marks] Write regular expressions for the following languages over $\{a, b, c\}$.

      i. Strings such that every 'a' is followed immediately by a 'b'.

      ii. Strings that contain 'c' an even number of times.

   (b) [6 marks] Draw a DFA for the language of all strings over $\{a, b, c\}$ that do *not* contain either 'ab' or 'bc'.

COMP3630/6363 Theory of Computation 2007

   (c) [5 marks] Draw an NFA that accepts the following language over $\{a, b\}$:

$$(b \cup ab^+)^* \cup a^*$$

2. [20 marks]

Consider the following true statement:

Theorem. *Let L be a context-free language. Let L_{even} be the subset of L consisting of all the strings in L that have even length. Then L_{even} is context-free.*

There are at least three proofs of this theorem that use techniques studied in the course:

(a) A proof using grammars.

(b) A proof using PDAs.

(c) A proof using a theorem about language intersections.

Your task for this question is to provide **any two** of these proofs.

3. [10 marks altogether]

(a) [5 marks] The following alleged theorem is incorrect.

Alleged Theorem. *Let L be a context-free language. Then the complement of L is context-free.*

Alleged Proof. *Let P be a PDA that accepts L by final state. Modify P by changing each final (accepting) state into a non-final state and each non-final state into a final state. Then the new PDA accepts the complement of L.* $\diamond$

Explain why this proof is incorrect. (Note: you need to explain carefully why the logic used in this proof is invalid, not just provide evidence that the theorem is incorrect.)

(b) [5 marks] Write an *unambiguous* context-free grammar for this language. You don't have to prove the unambiguity.

The language of *functional expressions* is defined thus:

- A *variable name* consists of "x" followed by one or more digits.
- A *function name* consists of "f" followed by one or more digits.
- An *functional expression* consists of a function name followed by an argument list enclosed in parentheses. The argument list consists of one or more variable names or functional expressions separated by commas.

Examples of functional expressions are $f1(x2, x37)$ and $f35(f17(x7), x1, f9(x4, x5, f111(x99)))$.

4. [10 marks altogether] Classify each of the following languages as either "regular", "context-free but not regular", or "not context-free". Prove your assertions.

(a) [5 marks] $\{a^i b^j c^k \mid i \leq k\}$

(b) [5 marks] $\{a^i b^j c^k \mid i \leq j \leq k\}$

5. [20 marks altogether]

To the left of each of the following, write either "T" for "true" or "F" for "false".

(a) For a given input alphabet, there is a bijection (one-to-one correspondence) between the set of all Turing machines and the set of natural numbers.

(b) Nondeterminism does not increase the power of Turing machines because a nondeterminisitc Turing machine can be converted to a deterministic one that has the same running time and accepts the same language.

(c) When a Turing machine makes one move, its ID (instantaneous description) cannot change at more than two positions (unit increase in length counting as one change).

(d) Although a counter is a restricted version of a stack, one-counter machines accept the same set of languages as (nondeterministic) pushdown automata.

(e) Two-counter machines accept the same set of languages as Turing machines.

(f) It is always possible to convert a k-tape Turing machine to a single-tape one without increasing its running time exponentially.

(g) One can use the "storage in the state" technique to implement a counter in a Turing machine to decide whether the input has more than 10 symbols.

(h) The set of decidable languages is closed under union, intersection, and complementation.

(i) The set of recursively enumerable languages is closed under union and intersection.

(j) A language is decidable if and only if it and its complement are both recursively enumerable.

(k) Using diagonalization, one can exhibit finite languages that are not recursively enumerable.

(l) There exists an algorithm that can analyze the encoding of a given Turing machine and decide whether it accepts all strings (over the input alphabet of the Turing machine).

(m) A language is $\mathcal{NP}$-complete if SAT reduces to it in polynomial time.

(n) An arbitrary Boolean formula can be converted to one in conjunctive normal form such that the two formulas have the same set of variables and the same set of satisfying assignments.

(o) The set of (encoded) pairs of equivalent Boolean formulas is known to be in $\mathcal{NP}$ because one can reduce equivalence checking to SAT.

(p) Given a Boolean formula, computing the number of its satisfying assignments is in $\mathcal{PS}$ (polynomial space).

(q) Given a quantified Boolean formula with no free variables, one can always convert it to a Boolean formula (without quantifiers) in conjunctive normal form such that the former is true if and only if the latter is satisfiable.

(r) Every Turing machine whose space is bounded by a polynomial accepts a decidable language.

(s) If SAT is in co-$\mathcal{NP}$, then $\mathcal{NP}$ = co-$\mathcal{NP}$.

(t) The set of primes is in co-$\mathcal{NP}$.

6. [6 marks altogether]

This is a multiple-choice question. For each of the following three parts, select exactly one of the alternatives.

(a) Which of the following restrictions will reduce the set of languages accepted by Turing machines (multiple tapes allowed unless stated otherwise):

i. Tape head cannot move to the left of the input.

ii. Each tape cell can be altered at most once.

iii. Tape cells containing the input cannot be altered and only a single tape is used.

(b) The number of satisfying assignments for the Boolean formula $(A \vee B) \rightarrow C$ is:

i. 1

ii. 5

iii. 7

iv. 8

v. None of the above.

(c) Recall that a language L is in $\mathcal{RP}$ iff it is accepted by a randomized Turing machine M in the sense that (1) strings not in L are rejected. (2) each string in L is accepted with probability at least 1/2, and (3) M halts in polynomial time on all inputs. Let L_1 be a language in $\mathcal{RP}$ and M_1 be a corresponding Turing machine satisfying the above definition. If we change 1/2 to 3/4 in the above definition, then:

i. $\mathcal{RP}$ may now include more languages.

ii. L_1 may no longer be in $\mathcal{RP}$.

iii. M_1 may accept a language that is a strict superset of L_1.

iv. M_1 may accept a language (possibly empty) that is a strict subset of L_1.

v. M_1 may not accept any language at all.

7. [19 marks]

Design a Turing machine that accepts the set of strings with an equal number of 0's and 1's (mixed in any order). Describe the Turing machine in words and draw its transition diagram.

8. [20 marks]

COMP6363 and PhD only.
Your total mark will be scaled to a mark out of 100.

A subset U of the nodes of a graph is a *dominating set* if every node not in U is connected to some node in U. Show that it is $\mathcal{NP}$-complete to determine if a graph has a dominating set of size k. You can assume that the node cover (set of nodes that includes at least one endpoint of every edge) problem is $\mathcal{NP}$-complete.

Question 1. [15 marks altogether]

Construct for each of the following languages over the alphabet $\Sigma = \{a, b\}$ a (nondeterministic) finite automaton **as well as** a regular expression describing the language.

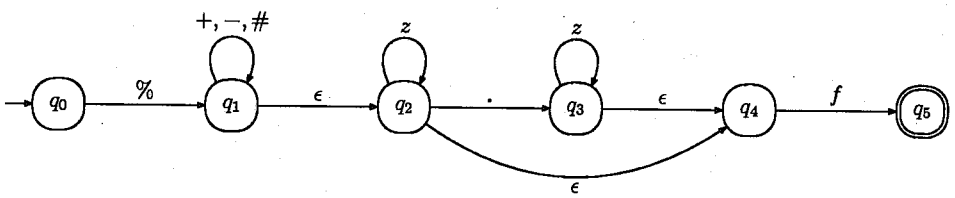
(a) [5 marks] $\{a^n b^m \mid n, m \in \mathbb{N}\}$

(b) [5 marks] $\{w \in \Sigma^* \mid w$ contains at least one a and at least one $b\}$

(c) [5 marks] $\{w \in \Sigma^* \mid$ the third last character in w is $b\}$

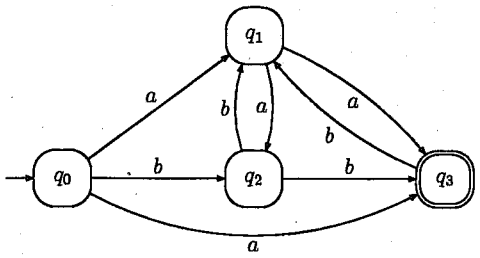
Question 2. [20 marks altogether]

(a) [10 marks] Let $\mathcal{A} = \{\Sigma, Q, q_0, \delta, F\}$ be the nondeterministic finite automaton with $\Sigma = \{f, +, -, \#, z, ., \%\}$, $Q = \{q_0, \ldots, q_5\}$, $F = \{q_5\}$, and δ according to the following diagram:



(The automaton accepts the format string for the real numbers of the C-library function `printf`.) Construct an equivalent (nondeterministic) automaton without ϵ-transitions!

(b) [10 marks] Let $\mathcal{A} = \{\Sigma, Q, q_0, \delta, F\}$ be the ϵ-free nondeterministic finite automaton with $\Sigma = \{a, b\}$, $Q = \{q_0, \ldots, q_3\}$, $F = \{q_3\}$, and δ according to the following diagram:



Construct an equivalent (ϵ-free) deterministic finite automaton!

Question 3. [25 marks altogether]

(a) [10 marks] Consider the following true statement:

Theorem. *Let L be a context-free language. Let $L_{a..a}$ be the subset of L consisting of all the strings in L that start and finish with the same symbol. Then $L_{a..a}$ is context-free.*

There are at least three proofs of this theorem that use techniques studied in the course:

i. A proof using grammars.

ii. A proof using PDAs.

iii. A proof using a theorem about language intersections.

Provide **any two** of these proofs.

(b) [7 marks] The fictional computer language `baby-C` has variable declarations that consist of a type name (only `int` and `char` exist) followed by a comma-separated list of one or more variable names and a terminating semicolon. Variable names consist of a lowercase letter followed by one or more digits. An example of a variable declaration is

```
int x23,a5,y7665;
```

Write **two** context-free grammars for `baby-C` variable declarations. The first grammar should be ambiguous: demonstrate by an example that it is ambiguous. The second grammar should be unambiguous (but you don't need to prove it).

(c) [8 marks] When the designers of the language `baby-C` decided to upgrade it to `toddler-C`, they added a restriction to the syntax of variable declarations:

Rule: No variable shall appear in the list more than once!

Prove that the language of `toddler-C` variable declarations is no longer context-free.