SUBJECT DESCRIPTION FORM

Subject Title: Computational Models **Subject Code:** COMP 434

Number of Credits: 3 **Hours Assigned:** Lecture 42 hours
Tutorial 7 hours

Pre-requisite: COMP 210 (not applicable for 61025), COMP 305, COMP307 (only for 61025), COMP 309 (not applicable for 61025) **Co-requisite:** Nil **Exclusion:** Nil

---

Objectives:

This subject provides students knowledge on:

- computational models and theoretical computer science;
- fundamental concepts behind computing and problem solving.

Student Learning Outcomes:

After taking this subject, the students should be able to:

Professional/academic knowledge and skills

(1) acquire fundamental knowledge and concepts in computational models and theoretical computer science;
(2) understand the limitation of computers and algorithms in problem solving, in the presence of unsolvable and intractable problems;
(3) appreciate existence and development of smart algorithms that solve problems effectively;
(4) evaluate the effectiveness of computer algorithms employed in different applications;
(5) apply the knowledge in specific applications such as algorithm design, compiler construction.

Attributes for all-roundedness

(6) develop critical thinking on evaluating solution models and approaches;
(7) describe, express and solve problems through formalism and precise formulation.

Alignment of Programme Outcomes:

Programme Outcome 4: This subject contributes to developing student critical thinking through tutorial exercises on solving comparatively theoretical problems. They will also practice more in written assignments.
Programme Outcome 8: This subject contributes to important concepts underneath algorithms and computer science, and addresses the fundamental limitations to the ability of performing computing on different types of problems.

Syllabus:

Topic	Duration of Lectures
1. Formal languages and automata Strings and languages; grammars; regular languages; context-free languages; context-sensitive languages; recursively enumerable languages; final automata; pushdown automata; deterministic and non-deterministic automata; pumping lemma for regular and context-free languages; Turing machines.	15
2. Unsolvable problems Technique of diagonalization; undecidable or unsolvable problems, e.g., halting problem, Post's correspondence problem.	3
3. Computational complexity Complexity of algorithms; algorithm analysis techniques; complexity classes.	6
4. NP-hard and NP-complete problems Intractable problems; definition of the class NP; problem reduction; well-known NP-complete problems, e.g., 3SAT, bin packing, office hours scheduling, travelling salesperson problem.	6
5. Dynamic programming and approximation Principle of dynamic programming; branch-and-bound; approximated solutions; simulated annealing.	6
6. Applications Complexity implication; lexical analyzer (lex); parser (yacc); real-life NP-complete applications, e.g., TSP variant, scheduling.	6
Total	**42**

Laboratory Experiment: Nil

Case Study:

Existence of real-life undecidable problems (e.g. Halting Problem) and NP-hard problems (e.g. Travelling Salesperson Problem) and approximated and practical solutions to NP-hard problems.

Method of Assessment:

Continuous Assessment	55%
Examination	45%

Method of Assessment for Learning Outcomes:

Assessment method / task	% weighting	Intended subject learning outcomes to be assessed (Please check as appropriate)								
		1	2	3	4	5	6	7	8	9
Assignments	55	x	x	x	x	x	x	x		
Mid-term		x	x		x	x	x	x		
Examination	45	x	x		x	x	x	x		
Total	100									

Reference Books:

1. J.E. Hopcroft, R. Motwani, and J.D. Ullman. Introduction to Automata Theory, Languages, and Computation. Second Edition, Addison-Wesley, 2001.
2. D.C. Kozen. Automata and Computability. Springer, 1997.
3. D. Kelley. Automata and Formal Languages: An Introduction. Prentice Hall, 1995.
4. S.Y. Yan. An Introduction to Formal Languages and Machine Computation. World Scientific, 1998.
5. R. Sedgewick and P. Flajolet. An Introduction to the Analysis of Algorithms. Addison-Wesley, 1996.
6. M.R. Garey and D. Johnson. Computers and Intractability: A Guide to the Theory of NP-completeness. W.H. Freeman, 1979.
7. A.V. Aho, J.E. Hopcroft, and J. Ullman. Data Structures and Algorithms. Addison-Wesley, 1983.