Title (Units): **COMP 3790 Advanced Algorithm Design, Analysis and Implementation (3,2,2)**

Course Aims: This course aims to help students develop advanced algorithm design, analysis skills as well as problem solving techniques for implementing solutions for a variety of challenging problems.

Prerequisite: COMP 1150 Object Oriented Programming
COMP 1210 Data Structures and Algorithms

Course Intended Learning Outcomes (CILOs):
Upon successful completion of this course, students should be able to:

No.	Course Intended Learning Outcomes (CILOs)
	Knowledge
1	Explain the concepts of automata, language theory, and computational complexity
2	Describe the concepts of collections and generic programming
	Professional Skill
3	Develop concise and reusable codes based on collections and generics
4	Design efficient algorithms and develop efficient and correct codes for solving computational problems
	Attitude
5	Build team spirit in solving challenging problems

Calendar Description: This course aims to help students develop advanced algorithm design and analysis skills as well as problem solving techniques for implementing solutions for a variety of challenging problems. The course has two major components: (1) theory of computation: automata, language theory, and computational complexity; and (2) problem solving: programming for a variety of algorithms for real challenging problems.

Teaching & Learning Activities (TLAs):

CILOs	TLAs
1, 2, 3, 4	Students will learn the fundamental principles and key concepts via lectures and tutorials.
3, 4, 5	Students will work on written assignments and programming assignments to consolidate and apply what they have learnt.

Assessment:

No.	Assessment Methods	Weighting	CILOs to be addressed	Remarks
1	Continuous Assessment	50%	1-5	Continuous assessments are designed to measure how well students have learned the basic concepts of formal methods, collections, generic programming and algorithm design techniques. A set of assignments are designed to measure how well students have learned the concepts and mastered the required problem solving skills.
2	Examination	50%	1-4	Final examination questions are designed to see how far students have achieved in understanding of formal methods, collections, generic programming and algorithm design techniques.

Rubrics:

	Excellent (A)	Good (B)	Satisfactory (C)	Marginal Pass (D)	Fail (F)
Formal methods	• Evidence of a thorough understanding of complexity, automata and language theory	• Evidence of a good understanding of complexity, automata and language theory	• Evidence of some understanding of complexity, automata and language theory	• Evidence of a limited understanding of complexity, automata and language theory	• Fail to show evidence of some understanding of complexity, automata and language theory
Collections and generics	• Excellent understanding of collections and generics • Fluent use of collection frameworks and generic types	• Good understanding of collections and generics • Can use collection frameworks and generic types	• Average understanding of collections and generics • Use collection frameworks and generic types if asked	• Some understanding of collections and generics • Use collection frameworks and generic types with some errors	• Little understanding of collections and generics • Cannot use collection frameworks and generic types
Problem solving	• Has a high degree of efficiency and correctness in applying a variety of algorithm design techniques for problem solving	• Has a considerable degree of efficiency and correctness in applying a variety of algorithm design techniques for problem solving	• Has a moderate degree of efficiency and correctness in applying some algorithm design techniques for problem solving	• Has some degree of efficiency and correctness in applying some algorithm design techniques for problem solving	• Unable to apply correct algorithm design techniques for problem solving

Course Intended Learning Outcomes and Weighting:

Content	CILO No.
I. Computational Complexity	1
II. Automata and Language Theory	1
III. Discrete Structures	1
IV. Collections	2,3
V. Generic Programming	2,3
VII. Problem Solving	4,5

References:

M. Sipser. Introduction to the Theory of Computation, 2nd Edition, Course Technology, 2005.
J. E. Hopcroft, R. Motwani and J. D. Ullman. Introduction to Automata Theory, Languages, and Computation, 3rd Edition, Addison Wesley, 2006.
T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. Introduction to Algorithms, 3rd Edition, The MIT Press, 2009.
S. S. Skiena and M. Revilla. Programming Challenges, the Programming Contest Training Manual, Springer, 2003.

Course Content in Outline:

Topic

I. Computational Complexity
   A. Upper and lower asymptotic bounds of specific algorithms
   B. NP-completeness

II. Automata and Language Theory
   A. Models of computation
   B. Formal languages and grammars

III. Discrete Structures
   A. Mathematical logic
   B. Discrete probability
   C. Recurrence relations

IV. Collections
   A. Overview of list, set, map and queue
   B. Use collections framework
   C. Sort and Search

V. Generic Programming
   A. Generics and generic collections
   B. Generic methods
   C. Generic declarations

VI. Problem Solving
   A. Algorithm design methodologies
   B. Number theory
   C. High-precision arithmetic
   D. Combinatorics
   E. Graph algorithms
   F. Computational geometry