Subject Description Form

<table>
<tr><td><b>Subject Code</b></td><td>COMP3011</td></tr>
<tr><td><b>Subject Title</b></td><td>Design and Analysis of Algorithms</td></tr>
<tr><td><b>Credit Value</b></td><td>3</td></tr>
<tr><td><b>Level</b></td><td>3</td></tr>
<tr><td><b>Pre-requisite / Co-requisite / Exclusion</b></td><td><b>Pre-requisite</b>: COMP2011 Data Structures or EIE3320 Object-Oriented Design and Programming or equivalent</td></tr>
<tr><td><b>Objectives</b></td><td>The objectives of this subject are to:
• provide students with in-depth knowledge on algorithm design techniques; and
• introduce and practice advanced algorithms for various data types.</td></tr>
<tr><td><b>Intended Learning Outcomes</b></td><td>Upon completion of the subject, students will be able to:
<i><u>Professional/academic knowledge and skills</u></i>
(a) understand advanced techniques for designing algorithms;
(b) design algorithms for solving computing problems efficiently;
(c) analyze and compare the efficiency of algorithms; and
(d) design and implement efficient algorithms for solving computing problems in a high-level programming language (e.g., C++ or Java).
<i><u>Attributes for all-roundedness</u></i>
(e) solve problems independently; and
(f) think critically for improvement in solutions.</td></tr>
<tr><td><b>Subject Synopsis/ Indicative Syllabus</b></td><td>
<table>
<tr><th>Topic</th><th>Duration of Lectures</th></tr>
<tr><td><b>1. Analysis of algorithms</b>
Mathematical techniques; big-O notation; efficiency analysis; recurring relations.</td><td>2</td></tr>
<tr><td><b>2. Advanced Algorithmic Design Techniques</b>
Dynamic programming, divide-and-conquer, branch-and-bound, greedy algorithm.</td><td>6</td></tr>
<tr><td><b>3. Advanced Analysis Techniques</b>
Introduction to randomized algorithms, probabilistic analysis, amortized analysis.</td><td>6</td></tr>
</table>
</td></tr>
</table>

4. Advanced Data Structures Cache-oblivious data structures, log-structured merge tree, locality sensitive hashing, Bloom filter.	4
5. Computational Geometry Algorithms Spatial range searching, indexing of spatial objects, convex hull, closest pairs	4
6. NP-Complete Problems Complexity classes, NP-completeness, reduction, approximation algorithms.	4
Total	**26**

Teaching/ Learning Methodology

Lectures provide students the main concepts of the topic, together with comprehensive examples for easy understanding.

Tutorials and lab sessions offer an opportunity to students for practicing their algorithmic analysis, design, and implementation techniques.

Both written and programming assignments will be utilized in the course. Written assignments help students develop analysis and design skills, whereas programming assignments emphasize on implementation skills.

Assessment Methods in Alignment with Intended Learning Outcomes

Specific assessment methods/tasks	% weighting	Intended subject learning outcomes to be assessed (Please tick as appropriate)					
		a	b	c	d	e	f
Continuous Assessment	**60%**						
1. Assignments		✓	✓	✓	✓	✓	
2. Lab Exercises		✓	✓	✓	✓	✓	
3. Mid-Term / Tests		✓	✓	✓		✓	✓
Examination	**40%**	✓	✓	✓		✓	✓
Total	100%						

Explanation of the appropriateness of the assessment methods in assessing the intended learning outcomes:

All four items are relevant to the assessment of the use of algorithms advanced data structures for problem solving, as well as their efficiency analysis (for items a, b, c).

In addition, programming exercises in assignments and lab sessions are used to assess implementation skills (for item d); whereas the mid-term / tests and the examination are used to assess independent problem solving and critical thinking skills (for items e, f).

Student Study Effort Expected	Class contact:	
	▪ Lecture	26 Hrs.
	▪ Tutorial/Lab	13 Hrs.
	Other student study effort:	
	▪ Assignments (Written and Programming)	65 Hrs.
	Total student study effort	104 Hrs.
Reading List and References	**Textbook:** 1. Cormen, Thomas H., Leiserson, Charles E., Rivest, Ronald L. and Stein, Clifford, *Introduction to Algorithms*, 3rd Edition, MIT Press, 2009. **Reference Books:** 1. Goodrich, M.T., and Tamassia, R., *Data Structures and Algorithms in Java*, 3rd Edition, John Wiley, 2005. 2. Carrano, Frank M., *Data Abstraction & Problem Solving with C++: Walls & Mirrors*, Addison Wesley, 2007.	