


<b>Course Outline</b>	
<b>COMP6049</b> <b>Algorithm Design and Analysis</b> <b>(4)</b>	
<b>Effective Date</b> 01 September 2018	<b>Study Program</b> <b>Computer Science</b>
	<b>Revision 3</b>

### 1. Course Description

The course describes fundamental concept of design and analysis of algorithms in order to calculate time and space computation, complexity, and compare design algorithm methods. It gives the students knowledge of several algorithm that enable students for designing a good algorithm.

### 2. Graduate Competency

Each course in the study program contributes to the graduate competencies that are divided into employability and entrepreneurial skills and study program specific outcomes, in which students need to have demonstrated by the time they complete their course.

BINUS University employability and entrepreneurial skills consist of planning and organizing, problem solving and decision making, self management, team work, communication, and initiative and enterprise.

#### 2.1. Employability and Entrepreneurial Skills

Aspect	Key Behaviour

#### 2.2. Study Program Specific Outcomes

Study Program Specific Outcomes

### 3. Topics

- Introduction of design and analysis of algorithms
- Mathematical induction and recursive function
- Algorithms and complexity
- Complexity of algorithms
- Stack and queue
- Tree and binary tree
- Priority queue and heap
- Graph
- Divide and conquer
- Greedy methods
- Dynamic Programming: Fibonacci Sequence Problem
- Dynamic Programming: Coin Change Problem
- Dynamic Programming: Multistage Graph
- Dynamic Programming: Travelling Salesman
- Dynamic Programming: Knapsack Problem
- String Matching
- Huffman Code
- Graph Colouring



23. <http://visualgo.net/en/dfsdfs>
24. <http://whocouldthat.be/visualizing-string-matching/>
25. [http://www.algorithmist.com/index.php/Coin\\_Change](http://www.algorithmist.com/index.php/Coin_Change)
26. [http://www.algorithmist.com/index.php/Dynamic\\_Programming](http://www.algorithmist.com/index.php/Dynamic_Programming)
27. <http://www.cs.cmu.edu/~adamchik/15-121/lectures/Stacks%20and%20Queues/Stacks%20and%20Queues>.
28. <http://www.cs.cmu.edu/~adamchik/15-121/lectures/Trees/trees.html>
29. <http://www.cs.cmu.edu/afs/cs/project/jair/pub/volume4/vanbeek96a-html/node6.html>
30. <http://www.cs.oswego.edu/~odendahl/coursework/csc465/notes/09-c-multistage.html>
31. <http://www.csd.uoc.gr/~hy583/papers/ch11.pdf>
32. <http://www.dgp.toronto.edu/~jstewart/378notes/01intro/>
33. <http://www.geeksforgeeks.org/graph-and-its-representations/>
34. <http://www.hbmeyer.de/backtrack/achtdamen/autoacht.htm#up>
35. <http://www.huffmancoding.com/my-family/my-uncle/huffman-algorithm>
36. <http://www.ics.uci.edu/~eppstein/161/960215.html>
37. [http://www.imsc.res.in/~vraman/pub/intro\\_notes.pdf](http://www.imsc.res.in/~vraman/pub/intro_notes.pdf)
38. <http://www.personal.kent.edu/~rmuhamma/Algorithms/MyAlgorithms/divide.htm>
39. <http://www.personal.kent.edu/~rmuhamma/Algorithms/MyAlgorithms/Dynamic/knapsackdyn.htm>
40. <http://www.seas.gwu.edu/~ayoussef/cs6212/greedy.html>
41. <http://www.vogella.de/articles/ComplexityAnalysis/article.html>
42. <https://binus.ac.id/bits/learning-object/Complexity-of-Bubble-Sort-1217/index.html#/>
43. Huffman Code
44. Introduction of design and analysis of algorithms
45. Mathematical induction and recursive function
46. Priority queue and heap
47. Review
48. Stack and queue
49. String Matching
50. Strongly Connected Components
51. Tree and binary tree

## 7. Schedule

### Lecture

Session/Mode	Related LO	Topics	References
1 F2F	LO 1	Introduction of design and analysis of algorithms <ul style="list-style-type: none"> <li>□ Definition of Algorithm</li> <li>□ Definition of pseudocode</li> <li>□ Introduction to analysis of algorithms</li> </ul>	<ul style="list-style-type: none"> <li>□ Introduction of design and analysis of algorithms</li> <li>□ Design and analysis of algorithms</li> <li>□ Some Introductory Notes on Design and Analysis of Algorithms <a href="http://www.imsc.res.in/~vraman/pub/intro_notes.pdf">http://www.imsc.res.in/~vraman/pub/intro_notes.pdf</a></li> </ul>

2 F2F	LO 1	<p>Mathematical induction and recursive function</p> <ul style="list-style-type: none"> <li>□ Mathematics induction</li> <li>□ Recursive function</li> </ul>	<ul style="list-style-type: none"> <li>□ Mathematical induction and recursive function</li> <li>□ Design and analysis of algorithms</li> <li>□ Induction and Recursion <a href="http://www.imsc.res.in/~vraman/pub/intro_notes.pdf">http://www.imsc.res.in/~vraman/pub/intro_notes.pdf</a></li> </ul>
3 F2F	LO 1	<p>Algorithms and complexity functions</p> <ul style="list-style-type: none"> <li>□ Calculating processing time and growth rate</li> <li>□ Complexity function</li> <li>□ Steps to develop an algorithm</li> </ul>	<ul style="list-style-type: none"> <li>□ Algorithms and complexity functions</li> <li>□ Design and analysis of algorithms</li> <li>□ Computational Complexity Theory <a href="http://cgi.csc.liv.ac.uk/~ped/teachadmin/algorithm/complex.html">http://cgi.csc.liv.ac.uk/~ped/teachadmin/algorithm/complex.html</a></li> </ul>
4 F2F	LO 1 LO 2	<p>Complexity of algorithms analysis</p> <ul style="list-style-type: none"> <li>□ Algorithm of prime number using conventional Loop technique</li> <li>□ Algorithm of prime number using flagging Technique (sieve)</li> <li>□ Comparing both Algorithms and their complexity</li> </ul>	<ul style="list-style-type: none"> <li>□ Complexity of algorithms analysis</li> <li>□ Design and analysis of algorithms</li> <li>□ INTRODUCTION TO COMPLEXITY ANALYSIS <a href="http://www.dgp.toronto.edu/~jstewart/378notes/01intro/">http://www.dgp.toronto.edu/~jstewart/378notes/01intro/</a></li> <li>□ Complexity Analysis of Algorithms <a href="http://www.vogella.de/articles/ComplexityAnalysis/article.html">http://www.vogella.de/articles/ComplexityAnalysis/article.html</a></li> <li>□ Complexity of Bubble Sort <a href="https://binus.ac.id/bits/learning-object/Complexity-of-Bubble-Sort-1217/index.html#/">https://binus.ac.id/bits/learning-object/Complexity-of-Bubble-Sort-1217/index.html#/</a></li> </ul>
5 GSLC	LO 1	<p>Stack and queue</p> <ul style="list-style-type: none"> <li>□ Concept of ADT (abstract data type)</li> <li>□ Queue data structure</li> <li>□ Stack data structure</li> </ul>	<ul style="list-style-type: none"> <li>□ Stack and queue</li> <li>□ Design and analysis of algorithms</li> <li>□ Stack &amp; Queue <a href="http://www.cs.cmu.edu/~adamchik/15-121/lectures/Stacks%20and%20Queues/Stacks%20and%20Queues.html">http://www.cs.cmu.edu/~adamchik/15-121/lectures/Stacks%20and%20Queues/Stacks%20and%20Queues.html</a></li> </ul>

6 GSLC	LO 1	<p>Tree and binary tree</p> <ul style="list-style-type: none"> <li>□ Binary tree characteristics</li> <li>□ Definition of tree</li> <li>□ Operations in tree</li> <li>□ Tree traversal</li> </ul>	<ul style="list-style-type: none"> <li>□ Tree and binary tree</li> <li>□ Design and analysis of algorithms</li> <li>□ Binary Tree <a href="http://www.cs.cmu.edu/~adamchik/15-121/lectures/Trees/trees.html">http://www.cs.cmu.edu/~adamchik/15-121/lectures/Trees/trees.html</a></li> </ul>
7 F2F	LO 1 LO 2 LO 3	<p>Priority queue and heap</p> <ul style="list-style-type: none"> <li>□ Concept of heap</li> <li>□ Concept of priority queue</li> <li>□ Example of priority queue in our life</li> <li>□ Heapsort</li> <li>□ Operations in heap</li> </ul>	<ul style="list-style-type: none"> <li>□ Priority queue and heap</li> <li>□ Design and analysis of algorithms</li> <li>□ Priority Queue <a href="http://pages.cs.wisc.edu/~vernon/cs367/notes/11.PRIORITY-Q.html">http://pages.cs.wisc.edu/~vernon/cs367/notes/11.PRIORITY-Q.html</a></li> </ul>
8 F2F	LO 1 LO 2 LO 3	<p>Graph</p> <ul style="list-style-type: none"> <li>□ Implementation of graph using cost adjacency list</li> <li>□ Implementation of graph using cost adjacency matrix</li> <li>□ Types of graph</li> </ul>	<ul style="list-style-type: none"> <li>□ Graph</li> <li>□ Design and analysis of algorithms</li> <li>□ Graph - MST - Prim's Algorithm <a href="http://lmscontent.binus.ac.id/digitalcontent/DC%20COMP6226%20DAN%20COMP6049%20GRAPH%20-%20MST%20-%20Prim%20Algorith.zip">http://lmscontent.binus.ac.id/digitalcontent/DC%20COMP6226%20DAN%20COMP6049%20GRAPH%20-%20MST%20-%20Prim%20Algorith.zip</a></li> <li>□ Graph and its representation <a href="http://www.geeksforgeeks.org/graph-and-its-representations/">http://www.geeksforgeeks.org/graph-and-its-representations/</a></li> </ul>
9 F2F	LO1 LO2	<p>Divide and conquer</p> <ul style="list-style-type: none"> <li>□ Binary search</li> <li>□ Comparing merge sort, quick sort, and selection sort</li> <li>□ Concept of divide and conquer</li> <li>□ Merge sort</li> <li>□ Quick sort</li> <li>□ Selection sort</li> </ul>	<ul style="list-style-type: none"> <li>□ Divide and conquer</li> <li>□ Design and analysis of algorithms</li> <li>□ Divide and Conquer Algorithm <a href="http://www.personal.kent.edu/~rmuhamma/Algorithms/MyAlgorithms/divide.htm">http://www.personal.kent.edu/~rmuhamma/Algorithms/MyAlgorithms/divide.htm</a></li> </ul>
10 F2F	LO 1 LO 2	<p>Greedy methods</p> <ul style="list-style-type: none"> <li>□ Disjoint Set data structure (MST)</li> <li>□ Fractional Knapsack problem</li> <li>□ Greedy method</li> <li>□ Minimum Spanning Tree</li> <li>□ Shortest Path</li> </ul>	<ul style="list-style-type: none"> <li>□ Greedy methods</li> <li>□ Design and analysis of algorithms</li> <li>□ The Greedy Method</li> <li>□ <a href="http://www.seas.gwu.edu/~ayoussef/cs6212/greedy.html">http://www.seas.gwu.edu/~ayoussef/cs6212/greedy.html</a></li> </ul>

11 F2F	LO 1 LO 2	Greedy methods <ul style="list-style-type: none"> <li>□ Disjoint Set data structure (MST)</li> <li>□ Fractional Knapsack problem</li> <li>□ Greedy method</li> <li>□ Minimum Spanning Tree</li> <li>□ Shortest Path</li> </ul>	<ul style="list-style-type: none"> <li>□ Greedy methods</li> <li>□ Design and analysis of algorithms</li> <li>□ The Greedy Method <a href="http://www.seas.gwu.edu/~ayoussef/cs6212/greedy.html">http://www.seas.gwu.edu/~ayoussef/cs6212/greedy.html</a></li> </ul>
12 F2F	LO 2 LO 3	Dynamic Programming: Fibonacci Sequence Problem <ul style="list-style-type: none"> <li>□ Concept of Dinamic Programming</li> <li>□ Fibonacci sequence problem</li> <li>□ Solving Fibonacci Sequence problem</li> </ul>	<ul style="list-style-type: none"> <li>□ Dynamic Programming: Fibonacci Sequence Problem</li> <li>□ Design and analysis of algorithms</li> <li>□ Dynamic Programming <a href="http://www.algorithmist.com/index.php/Dynamic_Programming">http://www.algorithmist.com/index.php/Dynamic_Programming</a></li> </ul>
13 F2F	LO 2 LO 3	Dynamic Programming: Coin Change Problem <ul style="list-style-type: none"> <li>□ Coin change problem</li> <li>□ Solving of Coin Change problem</li> </ul>	<ul style="list-style-type: none"> <li>□ Dynamic Programming: Coin Change Problem</li> <li>□ Design and analysis of algorithms</li> <li>□ Coin Change <a href="http://www.algorithmist.com/index.php/Coin_Change">http://www.algorithmist.com/index.php/Coin_Change</a></li> </ul>
14 F2F	LO 2 LO 3	Dynamic Programming: Multistage Graph <ul style="list-style-type: none"> <li>□ Backward technique</li> <li>□ Forward technique</li> <li>□ Multistage Graph problem</li> </ul>	<ul style="list-style-type: none"> <li>□ Dynamic Programming: Multistage Graph</li> <li>□ Design and analysis of algorithms</li> <li>□ Multistage Graphs <a href="http://www.cs.oswego.edu/~odendahl/coursework/csc465/notes/09-c-multistage.html">http://www.cs.oswego.edu/~odendahl/coursework/csc465/notes/09-c-multistage.html</a></li> </ul>
15 F2F	LO 2 LO 3	Dynamic Programming: Travelling Salesman <ul style="list-style-type: none"> <li>□ Solving of traveling salesman problem</li> <li>□ Traveling Salesman problem</li> </ul>	<ul style="list-style-type: none"> <li>□ Dynamic Programming: Travelling Salesman</li> <li>□ Design and analysis of algorithms</li> <li>□ The Traveling Salesman Problem <a href="http://www.csd.uoc.gr/~hy583/papers/ch11.pdf">http://www.csd.uoc.gr/~hy583/papers/ch11.pdf</a></li> </ul>

16 F2F	LO 2 LO 3	Dynamic Programming: Knapsack Problem <ul style="list-style-type: none"> <li>□ Comparing Greedy Method and Dynamic Programming</li> <li>□ Review of knapsack problem</li> <li>□ Solving knapsack problem using dynamic programming</li> </ul>	<ul style="list-style-type: none"> <li>□ Dynamic Programming: Knapsack Problem</li> <li>□ Design and analysis of algorithms</li> <li>□ Dynamic-Programming Solution to the 0-1 Knapsack Problem <a href="http://www.personal.kenet.edu/~rmuhamma/Algorithms/MyAlgorithms/Dynamic/knapsackdyn.htm">http://www.personal.kenet.edu/~rmuhamma/Algorithms/MyAlgorithms/Dynamic/knapsackdyn.htm</a></li> </ul>
17 GSLC	LO 2 LO 3	String Matching <ul style="list-style-type: none"> <li>□ Naïve String Matching Algorithm</li> <li>□ The Knuth-Morris-Pratt Algorithm</li> </ul>	<ul style="list-style-type: none"> <li>□ String Matching</li> <li>□ Design and analysis of algorithms</li> <li>□ Knuth-Morris-Pratt String Search <a href="http://people.ok.ubc.ca/ylocet/DS/KnuthMorrisPratt.html">http://people.ok.ubc.ca/ylocet/DS/KnuthMorrisPratt.html</a></li> <li>□ String Algorithm <a href="http://whocouldthat.be/visualizing-string-matching/">http://whocouldthat.be/visualizing-string-matching/</a></li> <li>□ String <a href="http://algo.is/aflv16/aflv_11_strings.pdf">http://algo.is/aflv16/aflv_11_strings.pdf</a></li> </ul>
18 GSLC	LO 2 LO 3	Huffman Code <ul style="list-style-type: none"> <li>□ Building Huffman tree</li> <li>□ Concept of compression technique</li> <li>□ Creating Huffman Code table based on Huffman tree</li> <li>□ Introduction to Huffman code</li> </ul>	<ul style="list-style-type: none"> <li>□ Huffman Code</li> <li>□ Design and analysis of algorithms</li> <li>□ Huffman Coding <a href="http://www.huffmancoding.com/my-family/myuncle/huffman-algorithm">http://www.huffmancoding.com/my-family/myuncle/huffman-algorithm</a></li> </ul>
19 F2F	LO 2 LO 3	Graph Colouring <ul style="list-style-type: none"> <li>□ Chromatic number</li> <li>□ Edge Colouring</li> <li>□ Graph Colouring</li> <li>□ Map colouring</li> <li>□ Node Colouring</li> <li>□ Region Colouring</li> </ul>	<ul style="list-style-type: none"> <li>□ Graph Colouring</li> <li>□ Design and analysis of algorithms</li> <li>□ Graph Colouring with Simple Backtracking, Part Three <a href="http://blogs.msdn.com/b/ericlippert/archive/2010/07/22/graph-colouring-with-simple-backtracking-part-three.aspx">http://blogs.msdn.com/b/ericlippert/archive/2010/07/22/graph-colouring-with-simple-backtracking-part-three.aspx</a></li> </ul>

20 F2F	LO 2 LO 3 LO 4	Basic Search and Traversal <ul style="list-style-type: none"> <li>□ BFS in Tree and Graph</li> <li>□ Breadth First Search</li> <li>□ Depth First Search</li> <li>□ DFS in Tree and Graph</li> <li>□ Tree traversal</li> </ul>	<ul style="list-style-type: none"> <li>□ Basic Search and Traversal</li> <li>□ Design and analysis of algorithms</li> <li>□ BFS dan DFS <a href="http://www.ics.uci.edu/~eppstein/161/960215.html">http://www.ics.uci.edu/~eppstein/161/960215.html</a></li> </ul>
21 GSLC	LO 2 LO 3	Backtracking <ul style="list-style-type: none"> <li>□ Concept of Backtracking</li> <li>□ N-Queen Problem</li> </ul>	<ul style="list-style-type: none"> <li>□ Backtracking</li> <li>□ Design and analysis of algorithms</li> <li>□ Backtracking Algorithm <a href="http://www.hbmeyer.de/backtrack/achtdamen/autoacht.htm#up">http://www.hbmeyer.de/backtrack/achtdamen/autoacht.htm#up</a></li> <li>□ Backtracking Algorithm <a href="http://www.cs.cmu.edu/afs/cs/project/jair/pub/volume4/vanbeek96a-html/node6.html">http://www.cs.cmu.edu/afs/cs/project/jair/pub/volume4/vanbeek96a-html/node6.html</a></li> </ul>
22 GSLC	LO 2 LO 3	<ul style="list-style-type: none"> <li>□ Branch and Bound</li> <li>□ Definition of branch and bound</li> <li>□ FIFO branch and bound</li> <li>□ LC branch and bound</li> <li>□ LIFO branch and bound</li> <li>□ Solving traveling salesman problem</li> </ul>	<ul style="list-style-type: none"> <li>□ Branch and Bound</li> <li>□ Design and analysis of algorithms</li> </ul>
23 F2F	LO 2 LO 3	<ul style="list-style-type: none"> <li>□ Branch and Bound</li> <li>□ Definition of branch and bound</li> <li>□ FIFO branch and bound</li> <li>□ LC branch and bound</li> <li>□ LIFO branch and bound</li> <li>□ Solving traveling salesman problem</li> </ul>	<ul style="list-style-type: none"> <li>□ Branch and Bound</li> <li>□ Design and analysis of algorithms</li> </ul>
24 F2F	LO 2 LO 3	Strongly Connected Components <ul style="list-style-type: none"> <li>□ Strongly connected definition</li> <li>□ Tarjan's strongly connected component algorithm</li> </ul>	<ul style="list-style-type: none"> <li>□ Strongly Connected Components</li> <li>□ Design and analysis of algorithms</li> <li>□ Strongly Connected Components Algorithm <a href="http://visualgo.net/en/dfsbfs">http://visualgo.net/en/dfsbfs</a></li> </ul>



25 F2F	LO 2 LO 3 LO 4	Review <ul style="list-style-type: none"> <li><input type="checkbox"/> Dynamic Programming</li> <li><input type="checkbox"/> String Matching</li> <li><input type="checkbox"/> Strongly Connected Component</li> </ul>	<ul style="list-style-type: none"> <li><input type="checkbox"/> Review</li> <li><input type="checkbox"/> Design and analysis of algorithms</li> <li><input type="checkbox"/> Practices of algorithm analysis <a href="http://algo.is/competitive-programming-course/">http://algo.is/competitive-programming-course/</a></li> </ul>
26 F2F	LO 2 LO 3 LO 4	Review <ul style="list-style-type: none"> <li><input type="checkbox"/> Dynamic Programming</li> <li><input type="checkbox"/> String Matching</li> <li><input type="checkbox"/> Strongly Connected Component</li> </ul>	<ul style="list-style-type: none"> <li><input type="checkbox"/> Review</li> <li><input type="checkbox"/> Design and analysis of algorithms</li> <li><input type="checkbox"/> Practices of algorithm analysis <a href="http://algo.is/competitive-programming-course/">http://algo.is/competitive-programming-course/</a></li> </ul>

**8. Evaluation**

**Lecture**

Assessment Activity	LO			
	1	2	3	4
ASSIGNMENT	✓	✓	✓	✓
FINAL EXAM		✓	✓	✓
MID EXAM	✓	✓	✓	




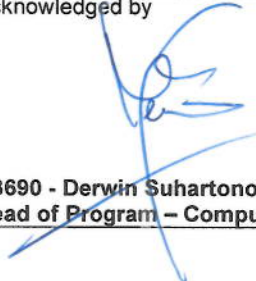
**Final Evaluation Score**

Aspects	Weight
Theory	100%

**9. Assessment Rubric (Study Program Specific Outcomes)**

LO	Indicators	Proficiency Level			
		Excellent (85 - 100)	Good (75 - 84)	Average (65 - 74)	Poor (<= 64)
LO 1	1.1. Ability to define the concept of analysis algorithm	The concept of analysis algorithm is clearly defined and relevant	The concept of analysis algorithm is clearly defined	The concept of analysis algorithm is not clearly defined	The concept of analysis algorithm is defined incorrectly
	1.2. Ability to demonstrate the stack and Queue applications and operations	Correctly and effectively demonstrating the stack and queue applications and operations	Correctly demonstrating the stack and queue applications and operations	Partially correct in demonstrating the stack and queue applications and operations	Incorrectly demonstrating the stack and queue applications and operations

LO 2	2.1. Ability to explain the algorithm techniques and methods	Correct explanation is given with relevant examples	Correct explanation is given with partially relevant examples	Correct explanation	Incorrect explanation
	2.2. Ability to give examples how to use the algorithm techniques and methods	All the examples are align to the algorithm techniques and methods	Only some of the examples are align to the algorithm techniques and methods	Only few of the examples are align to the algorithm techniques and methods	none of the examples are align to the algorithm techniques and methods
LO 3	3.1. Ability to calculate the complexity of an algorithm	Correctly and effectively calculating the complexity of an algorithm	Correctly calculating the complexity of an algorithm	Partially correct in calculating the complexity of an algorithm	Incorrectly calculating the complexity of an algorithm
	3.2. Ability to design algorithm with specific complexity	Correctly and effectively designing algorithm with specific complexity	Correctly designing algorithm with specific complexity	Partially correct in designing algorithm with specific complexity	Incorrectly designing algorithm with specific complexity
LO 4	4.1. Ability to interpret the algorithm design methods	Correctly interpreting all of the algorithms design methods	Correctly interpreting many of the algorithms design methods	Correctly interpreting some of the algorithms design methods	Incorrectly interpreting the algorithms design methods
	4.2. Ability to choose the algorithm design method for a real case	Correctly choosing the most effective algorithm design method for a real case	Correctly choosing effective algorithm design method for a real case	Correctly choosing algorithm design method for a real case	Incorrectly choosing algorithm design method for a real case

Prepared by  D5542 - Fidelson Tanzil, S.Kom., M.TI	Checked by  D5542 - Fidelson Tanzil, S.Kom., M.TI Acting as Subject Content Specialist
Approved by  D5542 - Fidelson Tanzil, S.Kom., M.TI Acting as Subject Content Coordinator	Acknowledged by  D3690 - Derwin Suhartono, S.Kom., M.T.I. Head of Program – Computer Science