Rencana Pembelajaran Semester (RPS)

<table>
<tr><td></td><td colspan="6">UNIVERSITAS PENDIDIKAN GANESHA
JURUSAN TEKNIK INFORMATIKA
PROGRAM STUDI ILMU KOMPUTER</td><td>Kode Dokumen</td></tr>
<tr><td colspan="8">RENCANA PEMBELAJARAN SEMESTER</td></tr>
<tr><td colspan="2">MATA KULIAH (MK)</td><td>KODE</td><td colspan="2">Rumpun MK</td><td>BOBOT (sks)</td><td>SEMESTER</td><td>Tgl Penyusunan</td></tr>
<tr><td colspan="2">Desain dan Analisis Algoritma</td><td>KOMS120403</td><td colspan="2">Mata kuliah inti keilmuan</td><td>T=3 P=0</td><td>4</td><td>20/012023</td></tr>
<tr><td colspan="2" rowspan="2">OTORISASI</td><td>Pengembang RPS</td><td colspan="2">Koordinator RMK</td><td colspan="2">Ketua PRODI</td><td></td></tr>
<tr><td>Ni Luh Dewi Sintiari, Ph.D.</td><td colspan="2">A.A. Gede Yudhi Paramartha, S.Kom., M.Kom.</td><td colspan="2">A.A. Gede Yudhi Paramartha, S.Kom., M.Kom.</td><td></td></tr>
<tr><td rowspan="12">Capaian Pembelajaran (CP)</td><td colspan="7">CPL-PRODI yang dibebankan pada MK</td></tr>
<tr><td>S1</td><td colspan="6">Bertakwa kepada Tuhan Yang Maha Esa dan mampu menunjukkan sikap religius;</td></tr>
<tr><td>S2</td><td colspan="6">Menjunjung tinggi nilai kemanusiaan dalam menjalankan tugas berdasarkan agama, moral, dan etika;</td></tr>
<tr><td>S8</td><td colspan="6">Menginternalisasi nilai, norma dan etika akademik;</td></tr>
<tr><td>S9</td><td colspan="6">Menunjukkan sikap bertanggung jawab atas pekerjaan di bidang keahliannya secara mandiri</td></tr>
<tr><td>S10</td><td colspan="6">Menginternalisasi semangat kemandirian, kejuangan, dan kewirausahaan;</td></tr>
<tr><td>P1</td><td colspan="6">Mampu memahami dan menguasai konsep dasar ilmu komputer secara umum seperti matematika, algoritma, pemrograman, dan basis data.</td></tr>
<tr><td>P2</td><td colspan="6">Mampu memahami dan menguasai konsep pengembangan perangkat lunak, mulai dari analisis kebutuhan, perancangan, pengembangan, dan implementasi perangkat lunak.</td></tr>
<tr><td>KU1</td><td colspan="6">Mampu menerapkan pemikiran logis, kritis, sistematis, dan inovatif dalam konteks pengembangan atau implementasi ilmu pengetahuan dan teknologi yang memperhatikan dan menerapkan nilai humaniora yang sesuai dengan bidang ilmu komputer;</td></tr>
<tr><td>KU2</td><td colspan="6">Mampu menunjukkan kinerja mandiri, bermutu, dan terukur;</td></tr>
<tr><td>KK1</td><td colspan="6">Terampil dalam menganalisis kebutuhan, merancang, dan mengimplementasikan rancangan, dan menguji perangkat lunak.</td></tr>
<tr><td colspan="7">Capaian Pembelajaran Mata Kuliah (CPMK)</td></tr>
</table>

	CPMK	Mahasiswa mampu merumuskan desain algoritma untuk menyelesaikan masalah dalam Ilmu Komputer, dan memiliki keterampilan untuk mengimplementasikan algoritma tersebut ke dalam dahasa pemrograman, sehingga mampu menjelaskan metode penyelesaian masalah secara sistematis dalam bentuk verbal dan tulisan.
	Kemampuan akhir tiap tahapan belajar (Sub-CPMK)	
	Sub-CPMK1	Mahasiswa mampu menjelaskan tahapan desain dan analisis algoritma dengan baik
	Sub-CPMK2	Mahasiswa mampu menghitung kompleksitas waktu algoritma (worst-case, best-case, average-case), menggunakan notasi Big-O, Big-Omega, dan Big-Theta, dan mengklasifikasikan algoritma berdasarkan kompleksitas waktunya dengan benar
	Sub-CPMK3	Mahasiswa mampu menjelaskan tentang konsep strategi brute-force/exhaustive search dan teknik heuristik dengan baik, menganalisis kebenaran dan kompleksitas waktu algoritma brute-force, serta mengaplikasikan strategi tersebut dalam pemecahan masalah dengan baik dan benar
	Sub-CPMK4	Mahasiswa mampu menjelaskan konsep algoritma rekursif, menuliskan pseudocode, menganalisis kebenaran, memformulasikan bentuk rekursif dari fungsi kompleksitas waktunya dan menghitung rumus eksplisit fungsi tersebut, serta mengaplikasikan metode rekursif dalam pemecahan masalah dan mengimplementasikannya dalam program komputer dengan baik dan benar
	Sub-CPMK5	Mahasiswa mampu menjelaskan strategi Divide-and-Conquer, Decrease-and-Conquer, dan Transform-and-Conquer, menuliskan pseudocode, menganalisis kebenaran dan menghitung fungsi kompleksitas waktu algoritma, serta mengaplikasikan ketiga strategi tersebut dalam pemecahan masalah dengan baik dan benar
	Sub-CPMK6	Mahasiswa mampu menjelaskan konsep algoritma Greedy, membuktikan optimalitas atau menunjukkan ketak-optimalan algoritma Greedy, mengaplikasikan metode Greedy dalam pemecahan masalah dan mengimplementasikannya dalam program komputer dengan baik dan benar
	Sub-CPMK7	Mahasiswa mampu menjelaskan metode BFS dan DFS dengan baik, menganalisis kompleksitas waktu dan ruang melalui contoh riil, dan mengaplikasikan metode BFS dan DFS dalam pembentukan pohon ruang status pada algoritma graf dinamis dengan baik dan benar
	Sub-CPMK8	Mahasiswa mampu menjelaskan konsep algoritma Backtracking dan Branch-and-Bound, serta mengaplikasikannya dalam pemecahan masalah algoritmik dengan baik dan benar
	Sub-CPMK9	Mahasiswa mampu menjelaskan konsep pemrograman dinamis, melakukan analisis kompleksitas waktu, dan mengaplikasikan pemrograman dinamis dalam pemecahan masalah algortimik dengan baik dan benar
	Sub-CPMK10	Mahasiswa mampu menjelaskan jenis-jenis permasalahan algoritmik dalam Ilmu Komputer, mengklasifikasikan masalah dalam kelas kompleksitas (P, NP, NP-Complete, dan NP-Hard), serta menentukan strategi algoritma yang tepat dalam pemecahan masalah algoritmik dengan baik dan benar
Deskripsi Singkat MK	Mata kuliah ini mempelajari tentang perancangan dan analisis algoritma, yang mencakup pembahasan mengenai jenis-jenis permasalahan algoritmik pada dunia komputer, analisis efisiensi yaitu kompleksitas waktu dan ruang algoritma, strategi-strategi perancangan algoritma, dan keterbatasan	

	setiap strategi algoritma. Strategi-strategi perancangan algoritma yang dibahas mencakup strategi Brute Force, teknik Rekursif, Divide-and-Conquer, Decrease-and-Conquer, Transform-and-Conquer, Greedy, Backtracking, Branch and Bound, Dynamic Programming, serta kelas kompleksitas algoritma (Teori P, NP, dan NP-Complete). Setelah mengikuti mata kuliah ini, mahasiswa diharapkan memahami berbagai macam strategi perancangan algoritma, serta mampu mengaplikasikan teknik perancangan algoritma untuk menyelesaikan masalah dalam kehidupan nyata.	
Bahan Kajian: Materi Pembelajaran	**Bahan Kajian:** Pengenalan macam-macam strategi algoritma, metode pembuktian kebenaran algoritma dan analisis kompleksitas waktu algoritma.	**Materi Pembelajaran:** Modul ajar
Pustaka	**Utama:**	Introduction to the Design & Analysis of Algorithms, Anany Levitin, Pearson Education, Inc.
	Pendukung:	- Slide Kuliah Strategi Algoritma, oleh Rinaldi Munir, Institut Teknologi Bandung - Slide Analysis of Algorithms, Robert Sedgewick
Dosen Pengampu	Ni Luh Dewi Sintiari, Ph.D.	
Matakuliah syarat	Struktur Data dan Algoritma	

Mg Ke-	**Kemampuan akhir tiap tahapan belajar (Sub-CPMK)**	**Penilaian**		**Bantuk Pembelajaran, Metode Pembelajaran, Penugasan Mahasiswa, [Estimasi Waktu]**		**Materi Pembelajaran [Pustaka]**	**Bobot Penilaian (%)**
		Indikator	**Kriteria & Bentuk**	**Luring (*offline*)**	**Daring (*online*)**		
(1)	**(2)**	**(3)**	**(4)**	**(5)**	**(6)**	**(7)**	**(8)**
1	Mahasiswa mampu menjelaskan tahapan desain dan analisis algoritma dengan baik	1. Ketepatan dalam menuliskan algoritma sederhana dengan benar 2. Ketepatan dalam menjelaskan tahapan proses perancangan	Bentuk Penilaian: • Non-tes, tanya-jawab lisan	Bentuk Pembelajaran: Kegiatan Proses Belajar [3x50’] Metode	Media: elearning.undiksha.ac.id	• Kontrak kuliah • Pengantar desain dan analisis algoritma • Jenis-jenis algoritma	**5%**

		algoritma 3. Ketepatan dalam menjelaskan tahapan analisis algoritma 4. Ketepatan dalam menjelaskan permasalahan-permasalahan algoritmik penting dalam Ilmu Komputer 5. Ketepatan dalam menyebutkan macam-macam strategi perancangan algoritma 6. Ketepatan dalam menjelaskan definisi kebenaran algoritma beserta teknik untuk membuktikan kebenaran suatu algoritma		<u>Pembelajaran:</u> Diskusi, tanya-jawab, penugasan <u>Tugas 1:</u> Penulisan makalah ilmiah aplikasi strategi algoritma (Waktu pengerjaan 12 minggu)		• Contoh permasalahan dalam Ilmu Komputer	
2	Mahasiswa mampu menghitung kompleksitas waktu algoritma (worst-case, best-case, average-case), menggunakan notasi Big-O, Big-Omega, dan Big-Theta, dan mengklasifikasikan algoritma berdasarkan kompleksitas waktunya dengan benar	1. Ketepatan dalam menjelaskan konsep dan urgensi dari penghitungan kompleksitas waktu algoritma 2. Ketepatan dalam menghitung fungsi kompleksitas waktu algoritma. 3. Ketepatan dalam menghitung kompleksitas waktu	Bentuk Penilaian: • Tanya-jawab lisan • Quiz • Tugas	<u>Bentuk Pembelajaran:</u> Kegiatan Proses Belajar [3x50’] <u>Metode Pembelajaran:</u> Diskusi, tanya-jawab, presentasi, penugasan <u>Tugas 2:</u> Penghitungan	<u>Media:</u> elearning.undiksha.ac.id	1. Penghitungan kompleksitas waktu worst-case, best-case, dan average-case algoritma 2. Notasi asimptotik (big-O, big-Theta, dan big-Omega) beserta operasinya 3. Kelas algoritma berdasarkan fungsi kompleksitas	**7%**

		worst-case, best-case, dan average-case suatu algoritma 4. Ketepatan dalam menuliskan kompleksitas waktu dengan notasi asimptotik (Big-O, Big-Sigma, Big-Omega) 5. Ketepatan dalam menghitung operasi aritmetik dengan notasi asimptotis 6. Ketepatan dalam membuktikan secara matematis sifat-sifat sederhana dari operasi aritmetik dengan notasi asimptotis 7. Ketepatan dalam mengelompokkan algoritma menjadi kelas algoritma berdasarkan kompleksitas waktunya (linier, polinomial, logaritmik, eksponensial, dsb.)		kompleksitas waktu algoritma dan pembuktian sifat-sifat dengan notasi asimptotik		waktunya	
3	Mahasiswa mampu menjelaskan tentang konsep strategi brute-force/exhaustive search dan teknik heuristik dengan baik, menganalisis	1. Ketepatan dalam menjelaskan prinsip dasar dan karakteristik algoritma brute-force 2. Ketepatan dalam	Bentuk Penilaian: • Tanya-jawab lisan • Presentasi materi • Tugas	<u>Bentuk Pembelajaran:</u> Kegiatan Proses Belajar [3x50’] <u>Metode</u>	<u>Media:</u> elearning.undiksha.ac.id	**1.** Pengenalan strategi brute-force **2.** Pembuktian kebenaran dan penghitungan kompleksitas waktu	**5%**

	kebenaran dan kompleksitas waktu algoritma brute-force, serta mengaplikasikan strategi tersebut dalam pemecahan masalah dengan baik dan benar	merancang algoritma brute-force untuk menyelesaikan permasalahan algoritmik sederhana, seperti: mencari nilai maksimum/minimum pada array, sequential search, menghitung perpangkatan bilangan, menghitung nilai faktorial, perkalian matriks persegi, pengecekan bilangan prima, interpolasi polinom, mencari pasangan titik terdekat, dan pattern matching, dll. 3. Ketepatan dalam menghitung kompleksitas waktu algoritma Brute-force 4. Ketepatan dalam memodifikasi algoritma Brute-force untuk meningkatkan efisiensinya 5. Ketepatan dalam merancang algoritma exhaustive search untuk menyelesaikan permasalahan		<u>Pembelajaran:</u> Diskusi, tanya-jawab, presentasi, penugasan <u>Tugas 3:</u> Mendesain algoritma brute-force untuk menyelesaikan permasalahan algoritmik sederhana, dan menganalisis kebenaran dan kompleksitas waktunya		algoritma brute-force **3.** Strategi exhaustive search untuk permasalahan kombinatorial	

		kombinatorial, seperti: Traveling Salesman Problem dan 1/0 Knapsack problem					
4		1. Ketepatan dalam menjelaskan implementasi teknik heuristik pada algoritma untuk meningkatkan efisiensi strategi Brute-force, seperti pada permasalahan anagram dan magic square 2. Ketepatan dalam menyebutkan kelebihan dan kekurangan dari metode brute force 3. Ketepatan dalam menjelaskan tahapan algoritma pengurutan berbasis Brute-Force, seperti Selection sort, Bubble sort, dan Insertion sort 4. Ketepatan dalam membuktikan kebenaran algoritma Selection sort, Bubble sort, dan Insertion sort dengan menggunakan loop-invariant	Bentuk Penilaian: • Tanya-jawab lisan • Presentasi materi • Tugas	<u>Bentuk Pembelajaran:</u> Kegiatan Proses Belajar [3x50’] <u>Metode Pembelajaran:</u> Diskusi, tanya-jawab, presentasi, penugasan <u>Tugas 4</u>: Implementasi algoritma sorting pada Bahasa pemrograman	<u>Media:</u> elearning.undiksha.ac.id	1. Teknik heuristik untuk peningkatan efisiensi algoritma brute-force 2. Algoritma sorting berbasis brute-force (Selection Sort, Bubble Sort, dan Insertion Sort) 3. Pembuktian dengan metode loop-invariant 4. Program komputer untuk implementasi algoritma brute-force	**4%**

		5. Ketepatan dalam mengaplikasikan algoritma Selection Sort, Bubble Sort, dan Insertion sort dalam pemecahan masalah 6. Ketepatan dalam membuat program implementasi algoritma sorting pada pemecahan masalah					
5	Mahasiswa mampu menjelaskan konsep algoritma rekursif, menuliskan pseudocode, menganalisis kebenaran, memformulasikan bentuk rekursif dari fungsi kompleksitas waktunya dan menghitung rumus eksplisit fungsi tersebut, serta mengaplikasikan metode rekursif dalam pemecahan masalah dan mengimplementasikannya dalam program komputer dengan baik dan benar	1. Ketepatan dalam menjelaskan prinsip strategi rekursif, karakteristik algoritma rekursif, serta perbedaannya dengan algoritma brute-force 2. Ketepatan dalam merancang algoritma rekursif untuk menyelesaikan permasalahan algoritmik sederhana, seperti menghitung faktorial, mencari nilai maksimum pada array, dan menghitung jumlah elemen pada array 3. Ketepatan dalam menjelaskan tahapan	Bentuk Penilaian: • Tanya-jawab lisan • Presentasi materi • Tugas	Bentuk Pembelajaran: Kegiatan Proses Belajar [3x50’] Metode Pembelajaran: Diskusi, tanya-jawab, presentasi, penugasan Tugas 5: Penyelesaian masalah algoritmik sederhana dengan metode rekursif, dan implementasinya dalam Bahasa pemrograman	Media: elearning.undiksha.ac.id	1. Pengenalan strategi rekursif 2. Penyelesaian masalah Menara Hanoi 3. Binary search 4. Pembuktian kebenaran algoritma rekursif dengan induksi matematika 5. Analisis efisiensi waktu algoritma rekursif 6. Implementasi algoritma rekursif dalam bentuk program komputer	**8%**

		algoritma rekursif untuk menyelesaikan masalah Menara Hanoi 4. Ketepatan dalam menjelaskan tahapan algoritma rekursif pada Binary Search 5. Ketepatan dalam menjelaskan tahapan algoritma rekursif untuk menghitung perpangkatan 6. Ketepatan dalam membuktikan kebenaran dari algoritma rekursif dengan menggunakan induksi matematika 7. Ketepatan dalam menyatakan fungsi kompleksitas waktu dalam formula rekursif dan menghitung bentuk fungsi ekplisitnya Ketepatan dalam menjelaskan redundansi algoritma rekursif pada algoritma konstruksi barisan Fibonacci 8. Ketepatan dalam menuliskan pseudocode algoritma rekursif untuk mengatasi tumpang					

		tindih (*overlap*) komputasi 9. Ketepatan dalam membuat program/implementasi algoritma rekursif dan melakukan analisis efisiensi berdasarkan eksperimen					
6	Mahasiswa mampu menjelaskan strategi Divide-and-Conquer, Decrease-and-Conquer, dan Transform-and-Conquer, menuliskan pseudocode, menganalisis kebenaran dan menghitung fungsi kompleksitas waktu algoritma, serta mengaplikasikan ketiga strategi tersebut dalam pemecahan masalah dengan baik dan benar	1. Ketepatan dalam menjelaskan prinsip algoritma divide-and-conquer, dan mengaplikasikannya untuk menyelesaikan permasalahan algoritmik 2. Ketepatan dalam menghitung kompleksitas waktu algoritma divide-and-conquer 3. Ketepatan dalam menjelaskan algoritma pengurutan data berbasis divide-and-conquer, yaitu: merge sort dan quick sort 4. Ketepatan dalam menggunakan Teorema Master untuk	Bentuk Penilaian: • Tanya-jawab lisan • Presentasi materi • Tugas teori	<u>Bentuk Pembelajaran:</u> - Kegiatan Proses Belajar [3x50’] - Tugas mandiri [3x50’] <u>Metode Pembelajaran:</u> Diskusi, tanya-jawab, presentasi, penugasan <u>Tugas 6</u>: Implementasi algoritma divide-conquer pada permasalahan perkalian dua polinomial	<u>Media:</u> elearning.undiksha.ac.id	1. Pengenalan strategi divide-and-conquer 2. Analisis kompleksitas waktu algoritma divide-and-conquer 3. Merge Sort dan Quick Sort 4. Teorema Master 5. Metode Strassen untuk perkalian matriks 6. Metode Karatsuba untuk perkalian bilangan besar	**7%**

		menghitung kompleksitas waktu algoritma divide-and-conquer 5. Ketepatan dalam menjelaskan algoritma Divide-and-Conquer untuk perkalian matriks persegi, serta modifikasi algoritma untuk meningkatkan efisiensi, melalui algoritma perkalian matriks Strassen 6. Ketepatan dalam menjelaskan algoritma Divide-and-Conquer untuk perkalian bilangan besar, serta modifikasi algoritma untuk meningkatkan efisiensi, melalui perkalian Karatsuba					
7		1. Ketepatan dalam menjelaskan prinsip strategi Decrease-and-Conquer, serta perbedaanya dengan strategi Divide-and-Conquer 2. Ketepatan dalam menjelaskan pendekatan	Bentuk Penilaian: • Tanya-jawab lisan • Presentasi materi • Quiz • Tugas	Bentuk Pembelajaran: Kegiatan Proses Belajar [3x50’] Metode Pembelajaran: Diskusi, tanya-jawab, presentasi,	Media: elearning.undiksha.ac.id	1. Pengenalan strategi decrease-and-conquer serta jenis-jenis pendekatannya 2. Pengenalan strategi transform-and-conquer serta jenis-jenis	**6%**

		‘decrease by a constant’, ‘decrease by constant factor’, dan ‘decrease by variable size’, serta mengaplikasikannya dalam pemecahan masalah. 3. Ketepatan dalam menjelaskan prinsip Transform-and-Conquer dan perbedaannya dengan Divide-and Conquer dan Decrease-and-Conquer 4. Ketepatan dalam menjelaskan pendekatan ‘instance ‘representation change’, dan ‘problem reduction’, serta mengaplikasikannya dalam pemecahan masalah		penugasan <u>Tugas 7</u>: Implementasi algoritma decrease-conquer dan transform-conquer pada permasalahan algoritmik		pendekatannya	
8	**Evaluasi Tengah Semester / Ujian Tengah Semester**						**10%**
9	Mahasiswa mampu menjelaskan konsep algoritma Greedy, membuktikan optimalitas atau menunjukkan ketak-optimalan algoritma Greedy, mengaplikasikan metode Greedy dalam	1. Ketepatan dalam menjelaskan prinsip dasar algoritma Greedy 2. Ketepatan dalam menjelaskan dan mengidentifikasi komponen algoritma Greedy melalui contoh	Bentuk Penilaian: • Tanya-jawab lisan • Presentasi materi • Tugas	<u>Bentuk Pembelajaran:</u> Kegiatan Proses Belajar [3x50’] <u>Metode Pembelajaran:</u> Diskusi, tanya-	<u>Media:</u> elearning.undiksha.ac.id	**1.** Pengenalan strategi Greedy **2.** Implementasi strategi Greedy dalam pemecahan masalah **3.** Analisis optimalitas algoritma Greedy	**5%**

	pemecahan masalah dan mengimplementasikannya dalam program komputer dengan baik dan benar	3. Ketepatan dalam menuliskan pseudocode skema algoritma Greedy 4. Ketepatan dalam mengaplikasikan strategi Greedy untuk menyelesaikan permasalahan optimasi sederhana, seperti: *Coin exchange problem*, *Activity selection problem*, *Time minimization in the system*. 5. Ketepatan dalam membuktikan keoptimalan atau menujukkan ketak-optimalan algoritma Greedy yang dirancang secara formal		jawab, presentasi, penugasan Tugas 8: Implementasi algoritma Greedy untuk menyelesaikan masalah optimisasi sederhana			
10		1. Ketepatan dalam menjelaskan strategi Greedy untuk penyelesaian masalah 1/0 Knapsack, dengan pendekatan *Greedy by profit*, *Greedy by weight*, dan *Greedy by density*. 2. Ketepatan dalam menjelaskan strategi Greedy untuk Fractional	Bentuk Penilaian: • Tanya-jawab lisan • Presentasi materi • Tugas tertulis	Bentuk Pembelajaran: Kegiatan Proses Belajar [3x50’] Metode Pembelajaran: Diskusi, tanya-jawab, presentasi, penugasan Tugas 9:	Media: elearning.undiksha.ac.id	**1.** Penyelesaian masalah 1/0 Knapsack dengan strategi Greedy **2.** Penyelesaian masalah Fractional Knapsack dengan strategi Greedy **3.** Konstruksi kode Huffman dengan strategi Greedy **4.** Analisis optimalitas	**5%**

		Knapsack Problem, dengan pendekatan *Greedy by profit*, *Greedy by weight*, dan *Greedy by density*. 3. Ketepatan dalam membuktikan bahwa ketiga pendekatan Greedy pada masalah 1/0 Knapsack tidak selalu memberikan solusi optimal. 4. Ketepatan dalam menjelaskan tahapan konstruksi kode Huffman dengan strategi Greedy 5. Ketepatan dalam menerapkan algoritma Huffman untuk masalah encoding sederhana		Membuktikan keoptimalan/ketak-optimalan algoritma Greedy dalam pemecahan masalah		dari algoritma yang dirancang untuk ketiga permasalahan tersebut	
11		1. Ketepatan dalam menjelaskan tahapan penyelesaian masalah Minimum Spanning Tree dengan algoritma Prim 2. Ketepatan dalam menjelaskan tahapan penyelesaian masalah Minimum Spanning Tree dengan algoritma Kruskal	Bentuk Penilaian: • Tanya-jawab lisan • Presentasi materi • Tugas membuat video	<u>Bentuk Pembelajaran:</u> Kegiatan Proses Belajar [3x50’] <u>Metode Pembelajaran:</u> Diskusi, tanya-jawab, presentasi, penugasan <u>Tugas 10</u>:	<u>Media:</u> elearning.undiksha.ac.id	**1.** Algoritma Prim untuk Minimum Spanning Tree **2.** Algoritma Kruskal untuk Minimum Spanning Tree **3.** Algoritma Dijkstra untuk Shortest Path	**5%**

		3. Ketepatan dalam menjelaskan mengapa algoritma Prim dan Kruskal memberikan hasil optimal untuk masalah Minimum Spanning Tree 4. Ketepatan dalam menjelaskan tahapan algoritma Dijkstra untuk menyelesaikan masalah Shortest Path 5. Ketepatan dalam menjelaskan mengapa algoritma Dijkstra memberikan hasil optimal untuk masalah Shortest Path 6. Ketepatan dalam mengaplikasikan algoritma Prim, Kruskal, dan Dijkstra untuk menyelesaikan masalah terkait pada graf sederhana		Membuat video penjelasan contoh algoritma untuk MST dan Shortest Path			
12	Mahasiswa mampu menjelaskan metode BFS dan DFS dengan baik, menganalisis kompleksitas waktu dan ruang melalui contoh riil, dan mengaplikasikan metode BFS dan DFS dalam	1. Ketepatan dalam menjelaskan tahapan algoritma BFS pada struktur *tree dan* struktur graf yang bukan *tree* 2. Ketepatan dalam menjelaskan tahapan	Bentuk Penilaian: • Tanya-jawab lisan • Presentasi materi • Tugas	<u>Bentuk Pembelajaran:</u> Kegiatan Proses Belajar [3x50’] <u>Metode Pembelajaran:</u> Diskusi, tanya-	<u>Media:</u> elearning.undiksha.ac.id	1. Strategi BFS dan DFS, serta analisis efisiensinya 2. Pembangunan pohon ruang status pada graf dinamis dengan metode BFS dan DFS	**5%**

	pembentukan pohon ruang status pada algoritma graf dinamis dengan baik dan benar	algoritma DFS pada struktur *tree* dan struktur graf yang bukan *tree* 3. Ketepatan dalam menjelaskan definisi dan komponen pohon ruang status pada graf dinamis 4. Ketepatan dalam menjelaskan prosedur pembangunan pohon ruang status dengan strategi BFS dan DFS 5. Ketepatan dalam menyelesaikan masalah *8-puzzle game* melalui pembangunan pohon ruang status. 6. Ketepatan dalam menjelaskan perbandingan efisiensi dari strategi BFS dan DFS		jawab, presentasi, penugasan <u>Tugas 11</u>: Pembangunan pohon ruang status untuk menyelesaikan masalah algoritmik dengan struktur graf dinamis		3. Penyelesaian permainan 8-puzzle	
13	Mahasiswa mampu menjelaskan konsep algoritma Backtracking dan Branch-and-Bound, serta mengaplikasikannya dalam pemecahan masalah algoritmik dengan baik dan benar	1. Ketepatan dalam menjelaskan prinsip dasar strategi backtracking serta keterkaitannya dengan strategi DFS 2. Ketepatan dalam menjelaskan langkah-langkah penyelesaian	Bentuk Penilaian: • Tanya-jawab lisan • Presentasi materi • Tugas	<u>Bentuk Pembelajaran:</u> Kegiatan Proses Belajar [3x50’] <u>Metode Pembelajaran:</u> Diskusi, tanya-jawab, presentasi,	<u>Media:</u> elearning.undiksha.ac.id	1. Pengenalan strategi backtracking 2. Penyelesaian masalah n-ratu dengan strategi backtracking 3. Pencarian sirkuit Hamilton dengan strategi	**6%**

		masalah penempatan *n*-ratu pada papan catur dengan algoritma backtracking 3. Ketepatan dalam menjelaskan Langkah-langkah penyelesaian masalah sirkuit Hamilton dengan algoritma backtracking 4. Ketepatan dalam menjelaskan prinsip dasar strategi branch-and-bound dan perbedaannya dengan strategi backtracking 5. Ketepatan dalam menjelaskan tahapan penyelesaian masalah 1/0 Knapsack dengan algoritma branch-and-bound		penugasan <u>Tugas 12</u>: Membuat program computer secara berkelompok dengan materi algoritma backtracking dan branch-and-bound untuk penyelesaian masalah optimisasi		backtracking 4. Pengenalan strategi branch-and-bound 5. Penyelesaian masalah 1/0 Knapsack dengan strategi branch-and-bound	
14	Mahasiswa mampu menjelaskan konsep pemrograman dinamis, melakukan analisis kompleksitas waktu, dan mengaplikasikan pemrograman dinamis dalam pemecahan masalah algortimik dengan baik dan benar	1. Ketepatan dalam menjelaskan prinsip dasar strategi pemrograman dinamis (*dynamic programming*) 2. Ketepatan dalam menjelaskan menjelaskan “Prinsip Optimalitas” pada pemrograman dinamis 3. Ketepatan dalam	Bentuk Penilaian: • Tanya-jawab lisan • Presentasi materi • Tugas	<u>Bentuk Pembelajaran:</u> Kegiatan Proses Belajar [3x50’] <u>Metode Pembelajaran:</u> Diskusi, tanya-jawab, presentasi, penugasan	<u>Media:</u> elearning.undiksha.ac.id	1. Pengenalan konsep dan tahapan pemrograman dinamis 2. Penggunaan program dinamis untuk pemecahan masalah algoritmik sederhana. 3. Penyelesaian 1/0 Knapsack dengan	**7%**

		menjelaskan tahapan pemrograman dinamis untuk menyelesaikan beberapa masalah sederhana, seperti: “Coin-row problem, Change-making problem, dan Coin collecting problem” 4. Ketepatan dalam menjelaskan tahapan pemrograman dinamis untuk menyelesaikan masalah Knapsack 5. Ketepatan dalam menjelaskan konsep “Memory functions” untuk meningkatkan efisiensi pemrograman dinamis pada Knapsack problem, dan masalah algoritmik lainnya secara umum		Tugas 13: Implementasi pemrograman dinamis pada masalah algoritmik sederhana		pemrograman dinamis, dan konsep “Memory functions” untuk peningkatan efisiensi	
15	Mahasiswa mampu menjelaskan jenis-jenis permasalahan algoritmik dalam Ilmu Komputer, mengklasifikasikan masalah dalam kelas kompleksitas (P, NP, NP-Complete, dan NP-Hard), serta menentukan strategi algoritma yang tepat dalam	1. Ketepatan dalam menjelaskan perbedaan algoritma deterministik dan non-deterministik 2. Ketepatan dalam menjelaskan perbedaan *decision* problem dan *searching* problems, *decidable* dan	Bentuk Penilaian: • Tanya-jawab lisan • Presentasi materi	Bentuk Pembelajaran: Kegiatan Proses Belajar [3x50’] Metode Pembelajaran: Diskusi, tanya-jawab, presentasi, penugasan	Media: elearning.undiksha.ac.id	1. Jenis-jenis permasalahan algoritmik dalam Ilmu Komputer 2. Kelas P, NP, NP-complete, dan NP-Hard 3. Analisis kebutuhan dan batasan antara kecepatan dan	**5%**

	pemecahan masalah algoritmik dengan baik dan benar	*undecidable* problems, serta *tractable* dan *intractable* problems 3. Ketepatan dalam menjelaskan perbedaan kelas P, NP, NP-complete, dan NP-Hard 4. Ketepatan dalam memberikan sebuah contoh permasalahan yang diklasifikasan sebagai P, NP, NP-complete, atau NP-Hard dan menjelaskan mengapa masalah tersebut diklasifikan ke dalam kelas terkait 5. Ketepatan dalam menentukan pilihan algoritma untuk penyelesaian masalah berdasarkan urgensi kebutuhan dan batasan, antara kecepatan dan efisiensi pemakaian ruang memori		Tugas 14: Presentasi makalah ilmiah penerapan strategi algoritma dalam pemecahan masalah		efisiensi pemakaian ruang memori untuk menentukan strategi algoritma	
16	**Evaluasi Akhir Semester / Ujian Akhir Semester**						**10%**

Rujukan:

1. Introduction to The Design & Analysis of Algorithms, Anany Levitin, Pearson Education, Inc.

2. Slide Kuliah Strategi Algoritma, oleh Rinaldi Munir, Institut Teknologi Bandung.

3. Slide Analysis of Algorithms, Robert Sedgewick.

4. Modul Kuliah DAA, Made Windu Antara Kesiman, Universitas Pendidikan Ganesha.