Nr.	Datum	Inhalt
1	9.4.	Organisatorisches, Scheinbedingungen, Credit points Einführung: Definition von Algorithmen und Datenstrukturen, Geschichte Fundamentale Algorithmen (create, destroy, assign, swap, copy, identity, equality) (call by) value vs. reference, deep vs. shallow copy/compare Fundamentale Datenstrukturen: Zahlen und Container
Ü 1		Python-Tutorial Sieb des Eratosthenes Wert- und Refernzsemantik
2	10.4.	Anforderungen der Algorithmen an Container Einteilung der Container Listen, Arrays, Stacks und Queues
3	16.4.	Sortieralgorithmen: selection sort, merge sort Zeitmessung in Python
Ü 2		Sortieren: Implementation und Geschwindigkeitsvergleich (Kurven in Abhängigkeit von Problemgröße) Entwicklung eines effizienten Algorithmus: Bruchfestigkeit von Gläsern
4	17.4.	Quicksort und seine Varianten
5	23.4.	Korrektheit, Korrektheitsbeweise Programming by contract, Vor- und Nachbedingungen, Invarianten Testen, Unit Tests, Execution paths
Ü 3		Experimente zur Effektivität von Unit Tests Deque-Datenstruktur: Vor- und Nachbedingungen der Operationen, Implementation und Unit Tests
6	24.4.	Exceptions und Exception Handling Geschwindigkeit und Optimierung: Innere Schleife, Caches, locality of reference Komplexität versus Geschwindigkeit
7	30.4.	Komplexitätsklassen Bester, schlechtester, durchschnittlicher Fall Amortisierte Komplexität
Ü 4		Theoretische Aufgaben zur Komplexität Amortisierte Komplexität von array.append() Optimierung der Matrizenmultiplikation
	1.5.	*Himmelfahrt*
8	7.5.	Suchen: Binäre Suche, Suchbäume, balancierte Bäume Komplexität der Suche
Ü 5		Implementation und Analyse eines Binärbaumes Anwendung: einfacher Taschenrechner
9	8.5.	Erhaltung der Balance beim Einfügen, Rotationen, selbst-balancierende Bäume
10	14.5.	Prioritätswarteschlange, Heap
Ü 6		Treap-Datenstruktur: Verbindung von Suchbaum und Heap
11	15.5.	Mengen Hashing Dictionaries als Baum und als Hashtabelle
12	21.5.	Iteration vs. Rekursion Typen der Rekursion und ihre Umwandlung in Iteration Iteratoren
Ü 7		Übungen zu Rekursion und Iteration: Fakultät, Koch-Schneeflocke, Auflösung rekursiver Formeln, Umwandlung von Rekursion in Iteration
	22.5.	*Fronleichnam*
13	28.5.	Abstrakte Datentypen Generizität: Required Interface, Adapter und Typattribute Funktoren (Beispiel: Ordnung für Sortieren)
Ü 8		Übungen zur Generizität: Sortieren mit veränderter Ordnung, Adapterklassen
14	29.5.	Zahlendatentypen und Arithmetik, Byteorder, IEEE floating point Algebraische Konzepte Operator Overloading, Anwender-definierte Zahlentypen (z.B mit physikalischen Einheiten)

15	4.6.	Graphen und Graphendatenstrukturen Adjazenzlisten und Adjazenzmatrizen Gerichtete und ungerichtete Graphen Vollständiger Graph Pfade, Zyklen
Ü 9		Elementare Graphenaufgaben (Adjazenzlisten aufschreiben, Beweis e <= 3n-6 für planare Graphen, Zyklen eines vollständigen Graphen) Implementation einer Graphklasse Iteratoren für Tiefensuche und Breitensuche
16	5.6.	Tiefensuche und Breitensuche Zusammenhang, mehrfacher Zusammenhang, Komponenten
17	11.6.	Gewichtete Graphen Minimaler Spannbaum
Ü 10		Anwendung: Weg aus einem Labyrinth Anwendung: Erzeugen einer perfekten Hashfunktion
18	12.6.	Best-first search (Dijkstra) Kürzeste Wege Most-Promising-first search (A*)
19	18.6.	Prinzipien des Algorithmenentwurfs Repetition Orthogonale Zerlegung Divide and Conquer (inklusive branch-and-bound) Randomisierung Optimierung, Zielfunktionen Systematisierung von Algorithmen aus der bisherigen Vorlesung
Ü 11		Anwendung: Routenplaner Beispiele für Divide and Conquer: pow-Funktion Beispiel für Methode der kleinsten Quadrate: Approximation von Kreisen
20	19.6.	Analytische Optimierung: Methode der kleinsten Quadrate, Approximation von Geraden
21	25.6.	Zufallszahlen, Zyklenlänge, Pitfalls Zufallseverteilungen, Box-Muller Transformation Randomisierte vs. deterministische Algorithmen Las Vegas vs. Monte Carlo Algorithmen Beispiel für Las Vegas: Randomisiertes Quicksort
Ü 12		Naiver (deterministischer) und randomisierter Primzahltest, Laufzeitvergleich RANSAC für Kreise
22	26.6.	Beispiel für Monte Carlo: randomisierte Integration, randomisierter Primzahltest RANSAC-Algorithmus, Erfolgswahrscheinlichkeit Beispiel: Linien finden (Deterministisch vs. RANSAC)
23	2.7.	Greedy-Algorithmen, Bedingung für Optimalität Beispiele für Greedy-Algorithmen
Ü 13		Theoretische und praktische Aufgaben zur dynamische Programmierung
24	3.7.	Dynamische Programmierung (1)
25	9.7.	Dynamische Programmierung (2)
Ü 14		Sudoku-Löser
26	10.7.	Exhaustive Search, Backtracking NP-Vollständigkeit, Algorithmen mit exponentieller Laufzeit
27	16.7.	Approximation bei NP-vollständigen Problemen
28	17.7.	Quantum computing