CPCS-324 Syllabus

Catalog Description

CPCS-324 Algorithms and Data Structures (II)
Credit: 3 (Theory: 3, Lab: 1, Practical: 0)
Prerequisite: CPCS-222 , CPCS-223
Classification: Department Required

This is the second of a two-course sequence on algorithmic solution design and advanced data structures. The objective of this course is to study algorithms from a major application area, to illustrate advanced design techniques, and to introduce main concepts in computational complexity. The course should provide an opportunity to work with complex data structures and develop advanced programming skills. It should include significant discussion of applications and capstone project ideas. Topics include space and time trade-off techniques, limitations of algorithm power, and algorithms that illustrate advanced design methods such as dynamic programming, iterative improvement and greedy techniques. Suggested application areas include: graphs, operations research, computational geometry, bioinformatics, and cryptography.

Class Schedule

Lab/Tutorial 90 minutes 1 times/week

Meet 50 minutes 3 times/week or 80 minutes 2 times/week

Textbook

Anany Levitin, , "Introduction to the Design and Analysis of Algorithms", Pearson Education; 3 edition (2011-12-01)

ISBN-13 9780273764113 **ISBN-10** 027376411X

Grade Distribution

Week	Assessment	Grade %
6	Exam 1	25
12	Exam 2	20
15	Homework Assignments	15
16	Exam	40

Last Articulated

May 8, 2017

Relationship to Student Outcomes

a	b	c	d	e	f	g	h	i	j	k
x	x	x						x	x	

Course Learning Outcomes (CLO)

By completion of the course the students should be able to

1. Describe in formal terms the main characteristics and properties of select (covered) abstract data types. (i)
2. **Demonstrate step-by-step operation of algorithm using a small instance. (i)**
3. Discuss various representations, input formats, and algorithm alternatives related to selected application area. (i)
4. Write code to implement relevant abstract data type(s) in programming language and environment of choice. (i)
5. Analyze select (covered) dynamic programming algorithms given a pseudocode. (j)
6. Write code to implement select (covered) dynamic programming algorithm(s) in programming language and environment of choice. (c)
7. **Analyze select (covered) greedy algorithm given a pseudocode. (j)**
8. **Compare select (covered) algorithms for the same problem. (j)**
9. Write code to implement select (covered) greedy algorithm(s) in programming language and environment of choice. (c)
10. Discuss practical, real world applications of select (covered) algorithms. (a)
11. **Analyze select (covered) iterative improvement algorithms given a pseudocode. (j)**
12. Write code to implement select (covered) iterative improvement algorithm(s) in programming language and environment of choice. (j)
13. **Describe with examples main results regarding intractability, classes of problems, and existence of algorithmic solutions in terms of computational complexity. (b)**
14. Analyze select (covered) space-time tradeoff algorithms given a pseudocode. (j)
15. Compare various data structures with respect to dictionary operations and space requirements. (j)

Coordinator(s)

Dr. Muhammad Al-Hashimi, Assistant Professor

CPCS-324 Syllabus

Topics Coverage Durations

Topics	Weeks
Introduction week	1
Graph Implementation	3
Dynamic programming techniques	1
Greedy technique	4
Iterative improvement	2
Introduction to complexity	1
Trading time for space techniques*	2
Reading week	1