Course Description

COT4400 | Design and Analysis of Algorithms |4.00 credits

This upper division course is for students majoring in the B.S. in Information Systems Technology or the B.S. in Electrical and Computer Engineering Technology programs. This course covers general techniques in algorithm design (such as divide-and-conquer, greedy method, dynamic programming, search and traversal techniques, branch-and-bound) in the context of problem domains like graph, sorting and optimization problems. Prerequisite(s): COP1334 or COP2270.

Course Competencies:

Competency 1: The student will demonstrate knowledge of algorithm analysis by:

1. Generating a function which bounds the algorithm's computing time (a priori analysis)
2. Using asymptotic notation to determine the order of magnitude of the frequency of execution of statements

Competency 2: The student will demonstrate an understanding of elementary data structures by:

1. Implementing ordered or linear lists stacks and queues
2. Implementing trees: B-Trees, binary trees, heaps
3. Designing and implementing solutions using graphs
4. Implementing symbol tables and using hashing functions

Competency 3: The student will demonstrate an understanding of divide-and-conquer by:

1. Designing and implementing binary search solutions
2. Implementing merge sort algorithm
3. Implementing quicksort algorithm

Competency 4: The student will demonstrate an understanding of the greedy method by:

1. Applying the solution to solve complex problems including the knapsack and job scheduling problems
2. Designing and implementing an optimal merge pattern that will reduce the number of operations when merging records
3. Applying binary trees with minimal weighted external path lengths to obtain an optimal set of codes for messages
4. Developing minimum spanning trees used in graph traversal

Competency 5: The student will demonstrate an understanding of dynamic programming approaches by:

1. Developing a dynamic programming formulation for a k-stage graph problem
2. Developing and implementing optimal binary search trees
3. Apply dynamic programming algorithms to solve the 0/1 knapsack problem
4. Find the minimum cost path to solve the traveling salesperson problem

Competency 6: The student will demonstrate an understanding of basic search and traversal techniques by:

1. Developing recursive and non-recursive binary tree traversal algorithms
2. Implementing breadth first search traversal algorithms
3. Implementing depth first search traversal algorithms
4. Implementing game trees for game simulation including tic-tac-toe, chess, checkers, etc.

Competency 7: The student will demonstrate an understanding of backtracking methods by:

1. Creating a tree structure that defines the problems state space of the problem
2. Systematically generating the problem states, determining which are solution states, and which solution states are answer states
3. Implementing a depth first node and breadth first node generation with bounding functions

Competency 8: The student will demonstrate an understanding of branch-and-bound methods by:

1. Developing a systematic enumeration of candidate solutions by means of state space search
2. Enumerating the candidate solutions of a branch by checking against upper and lower estimated bounds on the optimal solution
3. Applying the B&B method to the solution of the zero-one knapsack and traveling salesman problems

Competency 9: The student will demonstrate an understanding of NP-Hard and NP-complete problems by:

1. Defining what types of problems are NP- Hard or NP-complete problems
2. Describing the characteristics of approximation algorithms for NP-hard problems