El Paso Community College Syllabus (Part II) Official Course Description

SUBJECT AREA	**Computer Science**
COURSE RUBRIC AND NUMBER	**COSC 2336**
COURSE TITLE	**Programming Fundamentals III**
COURSE CREDIT HOURS	**3 Credits 3 Lec 1 Lab**

I. Catalog Description

Provides further applications of programming techniques, introducing the fundamental concepts of data structures and algorithms. Includes topics on recursion, fundamental data structures (including stacks, queues, linked lists, hash tables, trees, and graphs), and algorithmic analysis. It is highly recommended that students take COSC 1436, COSC 1437, and COSC 2425 before enrolling in COSC 2336. **Prerequisite: INRW 0311 or ESOL 0340 (can be taken concurrently) or by placement exam or ENGL 1301 with a "C" or better or ENGL 1302 with a "C" or better. (3:1). Lab fee.**

II. Course Objectives

A. **Unit I. Object-Oriented Programming Concepts**
1. Develop and effectively use object-oriented programming basics such as inheritance, encapsulation, and abstract data types (ADTs).
2. Use a Java Integrated Development Environment (IDE) (e.g., Eclipse or IntelliJ) effectively.
3. Determine the signature and pre- and post-conditions for operations on an abstract data type.
4. Develop and use interfaces and iterators.
5. Describe the characteristics of static, stack, and heap allocation.
6. Describe the state of the call stack when calling non-recursive and recursive subroutines.

B. **Unit II. Fundamental Data Structures**
1. Identify, select, design, and use a variety of data structures including stacks, queues, priority queues, references, linked structures, resizable arrays, trees, graphs, and hash tables.
2. Determine which data model (list, set, hash table, tree, or graph) is appropriate for solving a problem.
3. Compare and contrast approaches for resolving collisions in hash tables, e.g., linear probing, quadratic probing, double hashing, rehashing, and chaining.
4. Design and implement a recursive solution to a problem.

C. **Unit III. Basic Analysis-Introduction to Algorithm Performance and Big-O Notation**
1. Perform asymptotic analysis and empirical measurements of an algorithm.
2. Analyze time and space trade-offs in algorithms.
3. Explain the differences among best, average, and worst case behaviors of an algorithm.
4. Differentiate between complexity classes such as constant, logarithmic, linear, quadratic, and exponential.

D. **Unit IV. Algorithmic Strategies**
1. Categorize algorithms based on programming strategy, i.e., brute force, divide-and-conquer, greedy, backtracking, and dynamic programming strategies.
2. Design simple algorithms using the different algorithmic strategies.

E. **Unit V. Fundamental Data Structures and Algorithms**
1. Identify, select, design, and use a variety of algorithms and data structures, including binary search, insertion sort, selection sort, shellsort, quicksort, mergesort, heapsort, binary heaps, binary search trees, and hashing.

2. Create and use the adjacency-matrix and adjacency-list based representations of graphs.
3. Implement graph algorithms, i.e., graph search, union-find, minimum spanning trees, and shortest paths.
4. Apply graph algorithms for determining shortest paths (Dijkstra's and Floyd's algorithms) and minimal spanning trees (Prim's and Kruskal's algorithms).

F. Unit VI. Advanced Data Structures and Algorithms

1. Implement and use balanced trees and B-trees.
2. Use and implement an algorithm that identifies strongly connected components.

III. THECB Learning Outcomes (ACGM)

Upon successful completion of this course, students will:

1. Design and develop programs that implement basic data structures, including stacks, queues, linked lists, hash tables, trees, and graphs.
2. Apply recursive techniques and algorithms to solve problems.
3. Implement searching and sorting algorithms.
4. Understand algorithm efficiency, Big-O notation, and why it should be considered in programming.
5. Analyze and select appropriate data structures to implement a solution to a problem.
6. Design and implement data structures using classes and incorporating object-oriented concepts.
7. Demonstrate best practices of software development including testing, validation, and documentation.

IV. Evaluation

A. Preassessment

None

B. Postassessment

1. There will be four (4) written examinations. The final exam will be comprehensive.
2. Homework assignments will be assigned at the instructor’s discretion and will be averaged on a 100-point scale.
3. Lab assignments will be assigned at the instructor’s discretion and will be averaged on a 100-point scale.

C. Remediation

The instructor may provide the students with means of improving a grade. The instructor will determine the timing, form, and method of remediation.

D. Final Grade

The final grade report will be based on the percentage of the total points earned.

V. Disability Statement (Americans with Disabilities Act [ADA])

EPCC offers a variety of services to persons with documented sensory, mental, physical, or temporary disabling conditions to promote success in classes. If you have a disability and believe you may need services, you are encouraged to contact the Center for Students with Disabilities to discuss your needs with a counselor. All discussions and documentation are kept confidential. Offices located: VV Rm C-112 (831-2426); TM Rm 1400 (831-5808); RG Rm B-201 (831-4198); NWC Rm M-54 (831-8815); and MDP Rm A-125 (831-7024).

VI. 6 Drop Rule

Students who began attending Texas public institutions of higher education for the first time during the Fall 2007 semester or later are subject to a 6-Drop limit for all undergraduate classes. Developmental, ESL, Dual Credit and Early College High School classes are exempt from this rule. All students should consult with their instructor before dropping a class. Academic assistance is available. Students are encouraged to see Counseling Services if dropping because exemptions may apply. Refer to the EPCC catalog and website for additional information.