Department of Electrical and Computer Engineering
University of Puerto Rico
Mayagüez Campus

Syllabus for ICOM 4035 – Data Structures (CS2)
Spring 2006

1. Faculty

Dr. Pedro I. Rivera Vega
Office: T-213
Phone: (787)-832-4040, x-2191
E-mail: pedro.rivera@ece.uprm.edu
Office hours: TBA (to be announced), or by appointment

2. Teaching Assistant

Isabel Najeras
Office: TBA
Phone: TBA
E-mail: TBA
Office hours: TBA, or by appointment

3. Course Description

Introduction to the design, analysis and implementation of data structures and sorting algorithms, using object-oriented programming techniques. Study of computational complexity and Big-O notation. Design and implementation of abstract data types and containers classes: Vector, List, Set, Sequence, Table, Stack, Queue, Priority Queue, Tree, and Graph. The list of concrete structures to be studied includes: dynamic arrays, linked lists, Bit vectors, binary trees, binary search trees, 2-3 Trees, heaps, hash tables, adjacency matrices and adjacency lists. Recursion, generics and inheritance will be used extensively. This course is often called **Computer Science II (CS2)** in many prominent U.S. universities.

4. Pre-requisites

ICOM 4015 or equivalent. Proficiency with Java and basic knowledge of UNIX.

5. Time and Place

Lecture: **Section 090 Monday & Wednesday 4:10-5:25 PM** ROOM: S-228
Section 100 Monday & Wednesday 5:55-7:10 PM ROOM: S-228

Laboratory:
Section 090: Thursday 12:55-2:50 PM ROOM S-121
Section 100: Tuesday 12:55-2:50 PM ROOM S-121

6. Credits

4 credits

7. Class Web Page

http://www.ece.uprm.edu/~pirvos/...

You are responsible to read this Web page periodically to obtain class materials, and other important announcements about this course.

8. Textbooks

Required:

Data Abstraction and Problem Solving with Java: Walls and Mirrors, Updated Ed.
Frank M. Carrano and Janet J. Prichard
Addison-Wesley, 2004
ISBN: 0-321-19717-8

Recommended References:

……

9. Grading

Your grade will be based **exclusively** on the scores that you obtain in the class projects, exams and laboratory assignments. The curve to be used to assign a grade to your score will be as follows:

Score	Grade
100 – 90	A
89 – 80	B
79 – 65	C
64 – 55	D
54 – 0	F

Your final score will be computed from your individual scores in the projects, exams and laboratory assignments. The weights assigned to each of these categories are as follows:

Class Attendance	05%
Midterm Exams (3)	45%
Final Exam (Comprehensive)	25%
Programming Projects (at least 4)	15%
Laboratory Work	10%

There will be no special project, no special homework, no special exam, nor any other kind of "*special work*" to improve grades. However, each project or exam might have an extra credit problem that you can use to help improve your score in that corresponding category.

10. Class Attendance

Class attendance is required by university rules. Attendance will be verified for every lecture by each one of you signing a sheet where your name appears. In the past some cases have been reported of individuals

signing for other students different from themselves. Signing for another person different from yourself will be considered as a violation to academic integrity rules, and possible sanctions may be applied to both persons involved. Avoid any trouble, just sign your name. On random days I will do a double check of the attendance list. The final grade corresponding to class attendance (which has a weight of 5% in the final grade) will be assigned using the following formula: `max(0,100-x*20)`, where `x` is the numbers of lectures in which the student is no present. **You should notice that 5 of more absences represent a score of 0 in this category.**

11. Exams

In this course, there will be three midterm exams and a comprehensive final exam. Unless otherwise indicated, all exams will be taken with closed books and closed notes. The midterm exams will be administered outside the regular class time. The date and time for each midterm exam will be as follows:

Exam Number	Date	Time	Place
I	February 17, 2006	5:00 PM – 8:00 PM	TBA
II	March 24, 2006	5:00 PM – 8:00 PM	TBA
III	April 28, 2006	5:00 PM – 8:00 PM	TBA

If for any other course, an exam has already been scheduled on one of the dates and hours given, you must notify it immediately, so that such exam can be rescheduled for you, the same day at an earlier hour. No excuse about conflicts with another exam will be accepted after today, January 11, 2006. The exam lasts 2 hours, but a period of 3 hours has been reserved. The exam will begin at 5:00pm, but those of you with a reasonable excuse, and after proper notification, may begin at 6:00pm.

Remember, the exams are offered out of regular hours in order for you to be able to have more time available to answer it. If for any reason (storm, bad weather conditions, etc.) we cannot offer a particular exam on the date shown, unless otherwise established, that particular exam will be automatically rescheduled for the first Wednesday that we are able to meet afterward at regular lecture hours of the respective section.

The final exam will be administered in accordance with the schedule specified by the Registrar of the University of Puerto Rico, Mayagüez Campus.

The lowest score in the midterm exams can be replaced with the score in the final exam, provided that the score in the final exam is higher. Otherwise, the scores in the midterm exams will neither be replaced nor dropped.

Each question included in each exam (midterm or final) will fall into one of the following categories:

- Explanation of a technical concept.
- Proof of a mathematical proposition.
- Solution to a problem using the concepts discussed in class.
- Tracing of either Java code segments or algorithms.
- Analysis of either Java code segments or algorithms.
- Implementation of Java classes and code segments.

10.1 Exam Reposition Policy

In this course, there will be **NO repositions** for missed midterm exams. If a student misses one midterm exam then that score will be replaced with the score obtained in the final exam. Any other missed midterm exam will carry a score of 0.

12. Programming Projects

In this course, you are expected to complete four programming projects that are designed with the following objectives**:**

1) Test your knowledge of the data structures presented in class.
2) Test your individual skills for engineering a programming solution to a particular problem.
3) Provide experience in the design and implementation of complex software modules using object-oriented techniques.

You will be given at least **two weeks** to complete each programming project. You must implement your project using the Java programming language, and you must work individually. You might discuss with your peers general aspects about the project and/or programming environment. However, you cannot share your code with any student, nor use code written by someone else. Failure to comply with this requirement will be considered as an act of academic dishonesty and you may receive a grade of F in the class (**read section below titled Academic Integrity**).

You must submit your project electronically following a procedure that will be discussed in class. For each project, you will be given a **tar file** containing a directory, called the *project directory*, which **may include** some of the following items:

1) A document explaining the tasks to be completed for the programming project.
2) A document indicating a minimal set of operations that your program must execute to be considered a *running program*. **If your program neither compiles nor performs this minimal set of operations it will receive a score of 0.**
3) A set of "`.java`" files containing **empty** implementations of the methods associated with classes that you will need to implement, algorithms and other tasks related with the particular programming project. **It is your job in these cases to implement the Java code that executes the tasks these methods are designed to perform.**
4) A set of test input files and their corresponding test output files. You should use these to help you decide whether your program is working correctly or not, based on what type of output your program produces out of these test input files. **NOTE**: This set of input files will not be the only one to be used to grade your project. Hence, a program might pass all the tests in these test files, and yet fail some of the extra tests used for grading. However, if your program executes correctly on the test input files you will receive at least 70% of the total score for a given project.

Your project directory should contain all the files associated with your project. Once you have completed your project, you will create a **new** tar file that **must** contain everything that you have created in your project directory. You will submit this tar file for us to grade your project. Again, you will receive further instructions on how to submit your project electronically.

You are expected to work in the Amadeus UNIX lab provided by the University of Puerto Rico, Mayagüez Campus. But, you are free to use your own UNIX computer, or any other compatible system, if you prefer to do so. **However, you must ensure that the programs that you submit for grading do compile and execute correctly on the machines available in the Amadeus UNIX lab, since the projects will be graded there using the current JVM installed there**. Failure to comply with this requirement will result in a score of 0 for the project being evaluated. **NOTE: This policy will be strictly enforced**.

Late Project Policy:

Each project will have a due date composed of an hour, month and day (i.e. 4:00 PM-September 12). A project will be considered late if it is submitted for grading one minute after its due date. For example, if the due date for a project is 3:00 PM-October 31, then a project submitted at 3:01 PM-October 31 is considered as one day late. Any late project will receive the following penalty:

1 day past due date	-15%
2 days past due date	-30%

No project will be accepted for grading if submitted 3 or more days after its due date, and any such project will receive a score of 0. Any project that is not submitted for grading will automatically receive a score of 0.

NOTE: We will not debug your code via e-mail. We shall only look at your program source code listings, or login to see your code files during the allotted office hours.

13. Laboratory

In this course there will be a weekly 3-hour laboratory session in which the Teaching Assistant will further discuss the concepts presented in class, and the programming projects. There will be several problem sets, and various small programming exercises as part of the assignments associated with the laboratory. Information regarding each laboratory assignment will be given well in advance of the laboratory session.

14. Incomplete Grade Policy

A student will receive an incomplete grade if and only if the student misses the final exam, has a valid excuse, and if with the reposition of that exam (assuming the largest possible grade) the student has a chance to obtain a final passing grade in the course. Such excuse must be one of the following:

- Medical certificate indicating illness.
- Legal certificate indicating an appointment to attend a Court of Law.
- Certificate from a hospital or a physician indicating the death of either: parent, child, husband, wife or sibling.

15. Academic Integrity

Each student is expected to work individually on all projects, exams and laboratory assignments. You may not share your answers to the laboratory assignments. You may not use code from another student, or code that you find on the Internet or any similar resources. You may not share your code with another student. Failure to comply with these requirements will result in a grade of F in the course for the

student(s) breaking these rules. Unauthorized group efforts, particularly during exams, will be considered academic dishonesty and the students involved will receive an F in the course. You should read Article 10 of the "Reglamento General de Estudiantes de la Universidad de Puerto Rico" to learn more about the possible sanctions that you might experience if caught in an act of academic dishonesty.

16. List of Topics

The following is a list of the course topics in the order in which they will be presented. This list is subject to change and it will vary depending on the pace of the lectures.

TOPICS:

1. Discussion of the Course Syllabus
2. The Phases of Software Development
    a. Specification, Design and Implementation
    b. Testing and Debugging
3. Computational Complexity
    a. Big-O Notation
    b. Worst-Case Analysis
    c. Average-Case Analysis
    d. Best-Case Analysis
4. General review about recursion
    a. Definition of recursive method
    b. Fundamental parts of valid recursive methods
    c. Applications o solution of classical problems: factorial, searching, Hanoi Towers
5. Data Abstraction
    a. Abstract Data Types
        i. Specification
        ii. Implementation using Java classes
    b. List ADT
        i. Specification
        ii. Array-based implementation
6. Linked structures
    a. Dynamic structures and implementation using linked components
    b. Linked list and variants of it (simple, doubly, circular, dummy head, etc.)
    c. Linked list implementation of the list ADT (array-based and dynamic)
7. More advanced discussion about recursion and its applications
    a. Backtracking technique and its use in exhaustive exploration

b. Simple computer language grammars and simple interpreter

8. Stacks
    a. Stack Interface and Applications
    b. Dynamic Array Implementation
    c. Linked List Implementation
9. Queues
    a. Queue Interface and Applications
    b. Dynamic Array Implementation
    c. Linked List Implementation
10. Relationships among classes and the Object-Oriented approach
    a. Inheritance
    b. Dynamic binding
    c. Abstract classes
    d. Interfaces
    e. Iterators
    f. Advantages of an object-oriented approach
11. Sorting
    a. Study of different sorting algorithms (selection, insertion, merge, quicksort)
    b. Efficiency
12. Trees
    a. Binary and General Trees
    b. Array Representation
    c. Pointer Representation
    d. Tree Traversals
        i. Pre-order
        ii. In-order
        iii. Post-order
13. Binary Search Trees
14. Tables
15. Heaps and Priority Queues
16. 2-3 trees
17. Maps and Searching
    a. Serial Search
    b. Binary Search

c. Open-Address Hashing

d. Chained Hashing

18. Graphs

a. Undirected Graphs

b. Directed Graphs

c. Adjacency Matrix Implementation

d. Adjacency List Implementation

e. Graph Traversals