

## CS 1521: Computer Science II (5)

### Catalog Description:

Continuation of introduction to computer science. Methods for procedural and data abstraction. Focus on abstract data types. Software design, algorithm analysis and issues in ethical use of computers. Requires implementation of significant programming projects.

**Textbook:** Caranno, Frank M. & Janet J. Prichard. *Data Abstraction and Problem Solving with C++: Walls and Mirrors*. 3<sup>rd</sup> ed. Addison Wesley, 2002.

### References:

### Course Goals:

The course begins with an introduction to object-oriented design (OOD) including: the Unified Process (UP) development cycle, iterative development techniques, use case analysis, Unified Modeling Language (UML) sequence and static diagrams. The basic principles of object-oriented programming (OOP) encapsulation, information hiding, inheritance, and polymorphism, are taught. Discussion of the importance of procedural and data abstraction leads to the concept of an ADT (Abstract Data Type). The ADT concept is then elaborated on in a series of assignments and lectures covering the basic ADTs: lists, stacks, queues, trees, priority queues, tables, and graphs. Advanced C++ programming techniques are taught and practiced. Student understanding of recursion and algorithm analysis is extended. By the end of the course, a student should have mastered the main concepts of OOD, OOP, and have successfully completed major programming assignments in C++ on each of the basic ADTs.

### Prerequisites by Course & Topic

CS 1511: Computer Science I – Working knowledge of the C++ language, basics of software design, basics of algorithm analysis, introduction to computer ethics

### Major Topics Covered in the Course

- Advanced features of the C++ language
- Object-oriented design principles
- Algorithmic analysis tools, focusing on Big-O, including analysis of recursive algorithms
- Introduction to Abstract Data Types, including: lists, stacks, queues, trees, tables, priority queues, and graphs
- Intermediate searching and sorting routines, including: binary search, mergesort, quicksort, radix sort, tree sort, and heapsort

**Class/Laboratory Schedule:** Lecture: 3 hours per week, Discussion: 1 hour per week, Laboratory: 1 hour per week

### Laboratory Projects

- Lab Assignments: One per week:
  1. Creating C++ Projects Using Visual C++.NET
  2. Object-Oriented Design Using Violet
  3. Recursive Solution
  4. Linked List ADT
  5. Standard Template Library Class list
  6. Advanced Recursion
  7. Stack ADT
  8. Queue ADT
  9. Polymorphism
  10. Advanced C++ Programming
  11. Algorithm Analysis
  12. Tree ADT
  13. Binary Search Tree
  14. Table ADT
  15. Graph ADT

Programming Assignments: Seven biweekly:

1. Object-Oriented Design
2. Recursive Solution
3. List ADT
4. Stack ADT
5. Queue ADT
6. Tree ADT
7. Priority Queue ADT

**Course Contribution to Program Objectives and Outcomes:**

1. Proficiency in object-oriented design and coding. (*b*)
2. Ability to analyze the run-time characteristics of any given algorithm. (*b, c*)
3. Understanding of software engineering issues. (*b*)
4. Understanding of issues related to abstract data types. (*b, c*)
5. Understanding ethical issues in software. (*g*)

**Estimate CSAB Category Content**

	CORE	ADVANCED		CORE	ADVANCED
Data Structures	2		Computer Organization and Architecture	0	
Algorithms	0.5		Concept of Programming Languages	1.5	
Software Design	1				

**Social and Ethical Issues**

Please list the topics that address the social and ethical implications of computing covered in all course sections. Estimate the class time spent on each topic. In what ways are the students in this course graded on their understanding of these topics (e.g., test questions, essays, oral presentations, and so forth)?

Students receive 1.5 lecture hours on the ethical implications of faulty software. Understanding of this material is evaluated through homework, quiz and examination questions.

**Theoretical Content**

Please list the types of theoretical material covered, and estimate the time devoted to such coverage.

1. Design (6 hours).
2. Algorithm analysis (4 hours).
3. Language theory (2 hours).

**Problem Analysis**

Please describe the analysis experiences common to all course sections.

Students receive 3 lecture hours on algorithm analysis focusing on Big-Oh analysis of sorting routines, including recursive quicksort. Understanding of this material is evaluated through homework, lab assignments, quiz and examination questions.

**Solution Design**

Please describe the design experiences common to all course sections.

In early programming and lab assignments, students are required to individually develop specifications to a problem given by the instructor. Later programming and lab assignments will involve students implementing the ADT portion of a solution specified in the form of use cases, UML sequence and static diagrams. Solution design is also evaluated through quiz and examination questions.

**Coordinator/Prepared by:** S. Holtz