TAFTCOLLEGE

Prepared by: P. Blake
Reviewed by: M. Mayfield
Date prepared: September 2022
C & GE approved: October 14, 2022
Board approved: November 9, 2022
Text Updated: September 2022
Semester effective:

Computer Science (COSC) 1545 Programming Concepts and Methods II with Lab (4 Units) CSU

Prerequisite: Successful completion of ENGR 1540 Introduction to Programming Concepts and Methodologies for Engineering with Lab with a 'C' or better

Prerequisite knowledge/skills: before entering the course the student should be able to:

1. Describe the basics of the architecture of a computer and its components,
2. Describe the principles of structured programming,
3. Describe, design, implement, and test structured programs using currently accepted methodology, and in particular to be able to do so for programs that control or otherwise interfaces with hardware by means of software,
4. Explain what an algorithm is and its importance in computer programming, and
5. Apply numerical techniques to analyze and solve engineering-related problems.

Hours and Unit Calculations:
48 hours lecture. (96 hours outside of class); 48 hours lab (192 Total Student Learning Hours) 4 Units

Catalog Description: This course builds on the foundation provided by ENGR 1540 and introduces the concepts of data abstraction, structures and algorithms used in object-oriented programming framework. Topics include recursion, fundamental data structures (including lists, stacks, queues, hash tables, trees and graphs) and basics of algorithm analysis. This course uses the C++ programming language.

Type of Class/Course: Transfer Degree Credit

Text: Gaddis, Walters and Muganda. *Starting out with C++ Early Objects*. 10th ed. Pearson 2017.

Laboratory Manual: Programming exercises and problem sets from the text are completed in the lab.

Course Objectives:

By the end of the course a successful student will be able to:

1. Describe, design and implement the appropriate data abstraction and structure methods and procedures including: call by value and reference, arrays, records, strings, stacks, queues, hash tables, pointers and tables,
2. Apply effective algorithms and object-oriented programming design and debugging methods including, class hierarchy, inheritance, encapsulation and overloading,
3. Apply proper design principles and techniques to analyze complexity bounds, big "O" notation, empirical performance measurements and times and space tradeoff decisions in algorithms,

4. Describe, design and implement the concept of recursion, its functions and procedures, divide and conquer strategies, backtracking and implementation in stacks and trees,
5. Utilize proper searching and sorting algorithms in arrays, link, and binary trees, and
6. Evaluate tradeoffs in structured analysis and design vs. object-oriented design and software lifetime management.

Course Scope and Content (Lecture):

Unit I — Fundamental Data Structures
- A. Arrays and records
- B. Strings and string processing
- C. Data representation in memory
- D. Pointers and references
- E. Linked lists and structures

Unit II — Implementation Strategies
- A. Stacks, queues, and hash tables
- B. Trees
- C. Correct data structure

Unit III — Recursion
- A. Concept and mathematical principles
- B. Simple procedures
- C. Divide and conquer approaches
- D. Backtracking
- E. Implementation

Unit IV — Declarations and Types
- A. Variable addresses
- B. Scope and persistence
- C. Binding and visibility
- D. Safety and security

Unit V — Abstraction Methods
- A. Functions and modules
- B. Pass by value or reference
- C. Memory storage management
- D. Type parameters
- E. Templates and standard containers

Unit VI — Object Oriented Programming
- A. Object-oriented design
- B. Encapsulation and information hiding
- C. Separation of behavior and implementation
- D. Classes and subclasses
- E. Inheritance and polymorphism
- F. Class hierarchies
- G. Collection classes and iteration protocols
- H. Method tables

Unit VII — Algorithm Analysis

A. Implementation
B. Limits
C. Timing
D. Tradeoffs

Unit VIII Searching and Sorting
A. Arrays
B. Hash tables
C. Binary trees

Course Scope and Content: Laboratory

Unit I Introduction
A. Development toolkit
B. Debugging
C. Naming conventions
D. Lab procedures
E. Group work and projects
F. Fundamental concepts and principles
G. Design strategy

Unit II Data Structures
A. Data types
B. Declaration models
C. Memory representation and allocation
D. Type checking
E. Garbage collection
F. Scope and persistence
G. User defined
H. Arrays and records
I. Strings
J. Static, stack and heaps
K. Pointers and references
L. Structures
M. Stacks, queues and hash tables
N. Trees

Unit III Algorithm Analysis
A. Design
B. Problem solving techniques
C. Tradeoff decisions
D. Bounds
E. Methods
F. Time vs. space
G. Big 'O' notation

Unit IV Computing Algorithms
A. Sorting
B. Searching
C. Link and binary trees
D. Tables

Unit V Object Oriented Programming
- A. Design
- B. Problem solving
- C. Encapsulation and security
- D. Behavior vs. implementation
- E. Inheritance
- F. Subclasses and hierarchies
- G. Polymorphism
- H. Collection classes and iteration

Unit VI Recursion
- A. Concept
- B. Mathematical functions
- C. Simple procedures
- D. Divide and conquer strategies
- E. Backtracking
- F. Implementation methods

Unit VII Abstraction Mechanisms
- A. Parameterization (value vs. reference)
- B. Records and storage
- C. Template or generic type parameters
- D. Modules

Learning Activities Required Outside of Class:

The students in this class will spend a minimum of 6 hours per week outside of the regular class time doing the following:

1. Studying assigned text, handout materials and class notes
2. Reviewing and preparing for quizzes, midterm and final exams
3. Completing individual homework assignments following coding guidelines and proper documentation.

Methods of Instruction:

1. Lecture, demonstrations and discussions
2. Individual homework and lab assignments

Methods of Evaluation:

1. Quizzes
2. Exams
3. Participation
4. Individual assignments and group assignments
5. Design Project and Presentation

Laboratory Category: Extensive Laboratory

Pre delivery criteria: All of the following criteria are met by this lab.

1. Curriculum development for each lab.
2. Published schedule of individual laboratory activities.
3. Published laboratory activity objectives.
4. Published methods of evaluation.
5. Supervision of equipment maintenance, laboratory setup, and acquisition of lab materials and supplies.

During laboratory activity of the laboratory: All of the following criteria are met by this lab.

1. Instructor is physically present in lab when students are performing lab activities.
2. Instructor is responsible for active facilitation of laboratory learning.
3. Instructor is responsible for active delivery of curriculum.
4. Instructor is required for safety and mentoring of lab activities.
5. Instructor is responsible for presentation of significant evaluation.

Post laboratory activity of the laboratory: All of the following criteria are met by this lab.

1. Instructor is responsible for personal evaluation of significant student outcomes (lab exercises, exams, practicals, notebooks, portfolios, etc.) that become a component of the student grade that cover the majority of lab exercises performed during the course.
2. Instructor is responsible for supervision of laboratory clean up of equipment and materials.

TOP Code:	070600: Computer Science
SAM Priority Code:	E: Non-Occupational
Distance Education:	Not Applicable
Funding Agency:	Y: Not Applicable (funds not used)
Program Status:	1: Program Applicable
Noncredit Category:	Y: Not Applicable, Credit Course
Special Class Status:	N: Course is not a special class
Basic Skills Status:	N: Course is not a basic skills course
Prior to College Level:	Y: Not applicable

Cooperative Work Experience:	N: Is not part of a cooperative work experience education program
Eligible for Credit by Exam:	E: Credit By Exam
Eligible for Pass/No Pass:	NO
Taft College General Education:	NONE
Discipline	Computer Science OR Engineering