EL CAMINO COLLEGE
COURSE OUTLINE OF RECORD

I. COURSE DESCRIPTION

Course Title and Number: Computer Science 1

Descriptive Title: Problem Solving and Program Design using C++

Discipline: Computer Science

Division: Mathematical Sciences

Course Length: ☒Full Term ☐Other (specify):

Hours Lecture: **3** Hours Laboratory: **3** Course Units: **4**

Grading Method: ☒Letter ☐Credit/No Credit ☐Both ☐No Grade

Course Type: ☒Credit, Degree Applicable ☐Credit, Not Degree Applicable ☐Non-Credit

Transfer CSU: ☒Yes Effective Date May 19, 1997 ☐No

Transfer UC: ☒Yes Approval Date Fall, 1997 ☐Pending ☐No

Conditions of Enrollment:
Specify Prerequisite Corequisite, Recommended Preparation, Enrollment Limitation or None.

Prerequisite: Mathematics 170 with a minimum grade of C or equivalent

Catalog Description:

This course is an introduction to problem solving and program design using structured, top-down, algorithmic development techniques, applied to the solution of numeric and non-numeric problems. Software engineering topics such as analysis, design, implementation, testing, documentation, and maintenance of software are discussed. Laboratory work will be done using the C++ computer language.

II. COURSE OBJECTIVES

List the major objectives of the course. These must be stated in behaviorally measurable terms.

1. Define simple software engineering terminology, such as software life cycle, problem analysis, program design, testing and top-down design. Utilize software engineering terminology properly when describing program design.

2. Utilize problem analysis and design techniques in developing solutions to programming problems. In particular, break programming problems down into chunks, leading to efficient use of top-down design in order to create and implement modular solutions.

3. Represent data utilizing simple numeric and character data types in a program and use them with input-output processes of the particular implementation of C++ being used.

4. Design solutions requiring translation of mathematical and algebraic steps into a C++ program, using appropriate mathematical operators and math library functions of the C++ implementation in use.

5. Design programming solutions requiring decision-making, using appropriate C++ selection statements, such as if-then, if-then-else and switch.

6. Design programming solutions requiring the use of repeated processes, using appropriate C++ iteration statements, such as for, while, and do-while loops.

7. Design, implement and manipulate C++ structure data types in order to store and manipulate data efficiently.

8. Design programming solutions requiring the storage and manipulation of large amounts of data (with random access ability during execution cycle), using single and multi-dimensional arrays, such as numerical, string, char, and structure types.

9. Design, implement and manipulate string class data types as objects in order to store string type data.

10. Implement skills required for reading data from and writing results to text files in C++.

III. OUTLINE OF SUBJECT MATTER

The topics should be detailed enough to enable an instructor to determine the major areas that should be covered and so that the course may have consistency from instructor to instructor and semester to semester.

Approximate Time (measured in hours)	Major Topics
15	Fundamentals of the C++ language ~ Use of the computer and computer languages ~ Problem analysis ~ Meaning of an algorithm.
9	Elementary data types and operations
18	Design with control structures ~ Design with decision making steps: use of if, if-else, nested if, multi-alternative if, and switch statements

	~ Design with repetitions steps: use of iteration statements
24	Design with sub programs (block-structured programming style) ~ Functions with and without parameters/return values ~ Use of control structures in the context of sub programs
15	Input/Output File manipulations ~ Use of text input files and creating output files
21	Design with large block of data in main memory ~ Use of structured data-types: arrays, structures, and operations (such as read, print, search, and sort)
6	Classes and objects ~ Demonstrating design and development ~ Use of a user-defined class

Total: 108 Hours

IV . METHODS OF EVALUATION

A. CREDIT, DEGREE APPLICABLE AND CREDIT, NOT DEGREE APPLICABLE COURSES

Check the PRIMARY method of evaluation for this course.

- ☐ Substantial writing assignments
- ☒ Problem solving demonstrations (computational or non-computational)
- ☐ Skills demonstrations

A minimum of one response in the categories 1, 2, or 3 below, as applicable, is required. However, you may check all that apply.

1. Indicate the types of writing assignments used as primary or secondary methods of evaluation for this course.
   - ☐ Essay exams
   - ☒ Written homework
   - ☐ Term or other papers
   - ☐ Reading reports
   - ☐ Laboratory reports
   - ☐ Other (specify)

2. Indicate the types of problem-solving demonstrations used as primary or secondary methods of evaluation for this course.
   - ☒ Exams
   - ☐ Laboratory reports
   - ☐ Quizzes
   - ☒ Homework problems
   - ☐ Fieldwork
   - ☒ Other (specify) computer programming assignments

3. Indicate the types of skill demonstrations used as primary or secondary methods of evaluation for this course.
   - ☐ Class performance
   - ☐ Performance exams
   - ☐ Fieldwork
   - ☐ Other (specify)

4. If objective exams are also used, check all that apply.
   - ☐ Multiple choice
   - ☒ Completion
   - ☐ Matching items
   - ☐ True/false
   - ☐ Other (specify)

B. NON-CREDIT COURSE

Indicate the methods of evaluation that will be used to determine that stated objectives have been met.

V. COURSEWORK

A. TYPICAL ASSIGNMENT

Provide an example of a typical assignment. This assignment must correspond to the PRIMARY method of evaluation indicated in Section IV, Methods of Evaluation. That is, it must be a writing assignment or, if more appropriate, an assignment involving problem solving or skill demonstration.

Analyze the problem below and develop an algorithm based on your analysis. Convert the algorithm into a menu-driven C++ program that will read the inventory file of a company and present a menu to the user with the options shown below. Create an inventory report. Update the input file before stopping the program.

The menu options should be the following:

	Menu Item	Required Features
1.	Order Products	If the order quantity is below the minimum required, print an error message and allow the user to quit ordering or change the quantity ordered. Allow as many orders as the user likes to be placed. If the order quantity is larger than the supply on-hand, fill the order for the supply on-hand, set the on-hand to zero and indicate a back order for the remaining amount in the end report. After the user orders the products, print the bill.
2.	Display Inventory	Print a list of available products, their names and Product IDs.
3.	Search for Product Details	Given the product ID, print the quantity on-hand, the price, and the minimum order quantity. Prompt the user for Product ID.
4.	Add a New Product	Prompt the user for the information needed to describe the new product. Automatically supply a Product ID.
5.	Remove a Product from Inventory	Prompt the user for the Product ID.
6.	Sort List Based on Product ID	
7.	Quit	

The input file contains information about the products available. The input is from a file (inventory.dat) and the output is to be written to a file (name provided by the user). A sample input file is provided (and can be downloaded from the class account in the lab) to test your program.

The end report will include the details of items in the inventory in a tabular form. The program calculates the sales of each item, marks for re-ordering those items that have fallen below the re-order amount, and presents the results in a tabular form. The program also calculates and prints the total sales for the day. Finally, the program will update the inventory input file.

Other Requirements:

- You must develop a complete and detailed algorithm.
- You must have a maximum of 50 distinct products.
- You must use an array of strings to store the names of the products. Product names may not exceed 10 characters.
- You must use a two-dimensional array of integers to store the Quantity On-Hand, the Re-Order Quantity, the Minimum Order Quantity, and the Product ID.
- You must use an array of floats to store the price of the products.
- You must have a modular program with many functions. Main will declare the necessary variables and will call the functions to do the work.
- You must prompt the user, when appropriate.
- You must read the input and output file names from the keyboard.
- You must use the following data types:
    - Use **int** for the Product ID, the Quantity On-Hand, the Re-Order Quantity, and the Minimum Order Quantity.
    - Use **float** for the Price.
    - Use **string** (maximum 10 characters) for the Product Name.
- You must not use structures, even if you know how to use them. If you do, you will not get credit for this assignment.

B. COLLEGE-LEVEL CRITICAL THINKING ASSIGNMENTS

Cite two specific assignments that demonstrate college-level critical thinking. (Required for degree applicable courses only.)

1. In this assignment you will write two separate programs, so you must create two projects. Both will do the same thing, but they will use different tools. Both programs will request the user to input a line of text. The program must then analyze the text to determine whether or not it is a palindrome. A palindrome reads the same forwards and backwards while ignoring punctuation and spaces. For example, these are all palindromes**:**

   abba
   abccccba
   radar
   Dad
   Madam, I'm Adam
   A man, a plan, a canal, Panama!!!

   The letter **a** is also a palindrome, as is any letter by itself. Finally, using this definition, the empty string is also considered a palindrome. A palindrome is NOT case sensitive. The expression doesn't even need not make any sense, such as with abccccba.

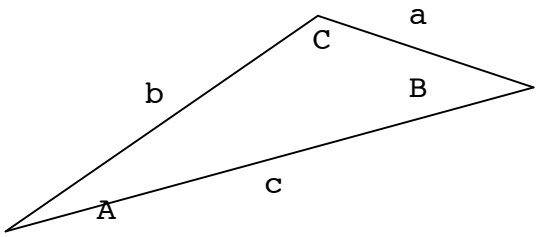
   In the first project, you are restricted to using the <string> library - you may not use <cstring>. In the second project, the situation is reversed; you may only use <cstring>, not <string>. No global variables, global constants or global arrays are allowed. (The one exception is that I will allow a global constant for the maximum size of the array, which I have set at 50). For both programs, you must create a loop that allows the user to test several expressions until the user choose to quit. However, it must allow for any kind of user response such as y, Y, yes or Yes for the response. Functions are required. One function should test an expression to see if it is a palindrome - it should have the prototypes: bool isPal(const string&) and bool isPal(const char[ ]), depending on which program you are doing. You should also have a function that removes punctuation from the expression - how you code this is up to you. I will give suggestions each day in class.

After finishing both programs, compare and contrast the two approaches in a short, but well-written, paragraph.

2. The program you will write will calculate the perimeter and area of any arbitrary triangle. The user has to choose from among three ways to give the input, namely

    1) the length of all three sides,
    2) the lengths of two sides and the measure of the included angle, or
    3) the measures of two angles and the length of the included side.

    Once the user has entered the input, the program should perform the proper operations to calculate the lengths of all three sides of the triangle (a, b and c), the measures of all three angles (A, B and C), the perimeter of the triangle, and the area of the triangle. As you see in the diagram, side a lies opposite the angle A, side b lies opposite the angle B, and side c lies opposite the angle C.



    You must use functions to implement this program.

    Once you have finished, compare the types of inputs offered the user in your program. Which one took the least amount of code to solve? Which one took the greatest amount of code to solve? Write a short paragraph explaining why one approach required more code than the other. Write a second paragraph discussing the strengths and weaknesses inherent in offering the user multiple input options.

C. WORK OUTSIDE OF CLASS

Two hours work outside of class are required for each hour of lecture or equivalent. Each student in this course will be required to participate in the following work outside of class time. Check all that apply.

- ☒ Study
- ☐ Answer questions
- ☐ Skill practice
- ☒ Required reading
- ☒ Problem solving activity
- ☐ Written work (such as essay/composition/report/analysis/research)
- ☐ Journal (done on a continuing basis throughout the semester)
- ☐ Observation of or participation in an activity related to course content (such as theatre event, museum, concert, debate, meeting)
- ☐ Course is lab only - minimum required hours satisfied by scheduled lab time
- ☐ Other (specify)

VI. INSTRUCTIONAL METHODOLOGY

A. Check all planned instructional activities that apply:

- ☒ Lecture
- ☒ Lab
- ☐ Discussion
- ☐ Multimedia presentations
- ☐ Demonstration
- ☐ Group Activities
- ☐ Role play/simulation
- ☐ Guest Speakers
- ☐ Field trips
- ☐ Other (specify)

Note: In compliance with Board Policies 1600 and 3410, Title 5 California Code of Regulations, the Rehabilitation Act of 1973, and Sections 504 and 508 of the Americans with Disabilities Act,

instructional delivery shall provide access, full inclusion, and effective communication for students with disabilities.

VII. TEXTS AND MATERIALS

If multiple selection is offered, only representative texts need be listed. An up-to-date list of required and recommended materials is maintained in the division office.

A. REQUIRED TEXTS (title, author, publisher, year)

C++ Programming: From Problem Analysis to Program Design, Third Edition,
D. S. Malik, Thompson Course Technology, 2007

B. REQUIRED SUPPLEMENTARY READINGS

C. OTHER REQUIRED MATERIALS

VIII. CONDITIONS OF ENROLLMENT

If this course has a Prerequisite or Corequisite, complete section A. If this course has an Enrollment Limitation complete section B.

A. PREREQUISITE AND/OR COREQUISITE

1. Indicate if this course has a prerequisite or corequisite or both.
   ☒Prerequisite ☐Corequisite ☐Both

2. Indicate Type. Check all that apply.
   ☐Sequential ☒Computational/Communication Skills
   ☐Health and Safety ☐Non-Course
   ☐Standard (If this is a Standard prerequisite or corequisite, attach CCC Form D.)
3. Entrance Skills/Knowledge
   List the required skills and/or knowledge without which a student would be highly unlikely to receive a grade of A, B, C, or Credit (or for Health and Safety, would endanger self or others) in this course.

   1. Ability to set up systems of linear equations and knowledge of methods for solving systems of linear equation.
   2. Ability to create trigonometric models and work with trigonometric functions.

B. ENROLLMENT LIMITATION

1. Indicate the category which describes the Enrollment Limitation for this course.
   ☐Band/Orchestra
   ☐Theater
   ☐Speech
   ☐Chorus
   ☐Journalism
   ☐Dance
   ☐Intercollegiate Athletics
   ☐Honors Course
   ☐Blocks of Courses
   ☐Other (specify)

2. List Degree and/or Certificate requirements that are met by this course.

3. List all El Camino College courses that also satisfy the requirements listed above in Section B.2.

Originator: Gregory Scott Submittal Date: February 24, 1997

BOARD APPROVAL DATE: May 19, 1997

Reviewed and/or Revised by:

Massoud Ghyam Date: March 19, 2007

REQUIRED SIGNATURES FOR NON-CREDIT COURSE

College Curriculum Committee Chair

_____

Vice-President - Academic Affairs

_____

CCC Form 1, 5/2006