Program Course Description Template  
(Required information needed for course submission)

Program Information
Program Name: UC Davis Center for Integrated Computing and STEM Education (C-STEM)

Contact Person
First Name: Harry
Last Name: Cheng
Position/Title: Professor/ C-STEM Center Director
Phone Number: (530) 752 – 9082
E-mail: info@c-stem.ucdavis.edu

SECTION A. BASIC COURSE INFORMATION

Course Title: Computer Programming for Solving Applied Problems (C-STEM)

Transcript Abbreviation(s) / Course Code(s): Computing Programming for Solving Applied Problems / [Course Code]

NOTE: Schools are responsible for providing the above information.

Length of Course:
☐ Half Year (1 semester; 2 trimesters; 2 quarters)
☒ Full Year (2 semesters; 3 trimesters; 4 quarters)
☐ Two Years (4 semesters; 6 trimesters; 8 quarters)

Subject Area / Discipline:
NOTE: See attached [Appendix A] for all subject areas and disciplines.

Subject Area: College-Preparatory Elective
Discipline: Mathematics/Computer Science

UC Honors Designation:

Is this course being submitted for UC honors consideration?
NOTE: 9th grade courses are not eligible for UC Honors consideration.

☐ Yes  ☒ No

Grade Level:
NOTE: Grade level pertains to which grades the course has been designed.

☐ 9  ☒ 10  ☒ 11  ☒ 12

Course Learning Environment:

Is this course, or any separate section of this course, taught in an online learning environment?*

☐ Yes  ☒ No

If “Yes,” has your institution conducted a self-assessment of the online course against the iNACOL Standards for Quality Online Courses?
☐ Yes  ☐ No
If “Yes,” does the course fulfill all 15 UC-identified power standards from the iNACOL Standards for Quality Online Courses?

☐ Yes  ☐ No
If “Yes,” how many of the remaining 37 standards from the iNACOL Standards for Quality Online Courses the course satisfy?  _ ______

*Please ensure to attach a copy of your completed Online Course Self-Assessment Form.

Is this course an integrated course?
NOTE: UC encourages the development of integrated courses that combine and skills of traditional academics with contextualized learning in career technical education.

☒ Yes  ☐ No
If “Yes,” please indicate the Industry Sector and Career Pathway below:
NOTE: See attached [Appendix B] for all industry sectors and career pathways.

Industry Sector: Information and Communication Technologies (ICT)
Career Pathway: Software and Systems Development

SECTION B: COURSE DESCRIPTION

Course Overview:

Briefly (in a short paragraph) provide a brief summary/snapshot of the course’s content:

This course provides students with the fundamental knowledge of computer programming for solving applied problems in C. A user-friendly RoboBlockly and C/C++ interpreter Ch will be used for learning computational thinking and software development. Students learn how a computer works and structured programming in C for software development. The topics include programming constructs, data types and declaration of variables, expressions and operators, selection statements, repetition, flowcharts for algorithm development, functions for modular programming, arrays for statistical data analysis, plotting for visualizing data (using scatter plot, dot plot, bar graph, histogram, Box-and-Whisker plot, etc.), linear regression and curve fitting, processing data files, animation, robotics applications, and applications in math and science. The emphasis of the course is to introduce the students to software development concepts. This course also focuses on algorithm development and computer programming for solving applied problems in science, technology, engineering and math (STEM), such as solving problems in Algebra and robotics. Considerable attention is devoted to program design, task decomposition, testing, debugging, and software reuse. Students write computer programs with graphical plotting in an integrated development environment. Through problem-based projects, students develop critical thinking, problem solving, computational thinking, effective communication, and teamwork skills.

Pre-Requisites: Algebra I or Integrated Mathematics I,  Required x  Recommended ______
NOTE: Laboratory science and Advanced VPA courses require a pre-requisite. Some courses require appropriate pre-requisites. Please refer to the “A-G” Guide for more information.

Co-Requisites: ___________________________ Required ______  Recommended ______
Course Content:

For each unit of the course, provide:

1) A brief description (5-10 sentences) of topics to be addressed that demonstrates the critical thinking, depth and progression of content covered.

2) A brief summary (2-4 sentences) of at least one assignment that explains what a student produces, how the student completes the assignment and what the student learns.

Unit One:  **Introduction to Programming, Variables, Data Types, and Input/Output**

This unit introduces the students to how a computer works, basics of computer programming, and the importance of computing in the 21st century. It leads them to the programming language syntax in C/C++ using the C/C++ interpreter Ch. Students evaluate expressions and practice order of operations in a Ch Command Shell. Students write programs with proper programming language syntax to review and practice basic operations with real numbers, order of operations, and manipulating and evaluating variables in simple algebraic equations. Students learn formatted input and output making their program meet specific design criteria.

To meet the challenges of this unit, students persevere in solving specific problems with attention to precision, construct variable arguments and critique the reasoning of others, and model with mathematics. Throughout this unit, students understand problems that arise in real life context of programming and find solutions of multi-step problems, choose and interpret the problems with formulas and conceptual understanding. These skills are demonstrated in multi-tiered tasks throughout the unit and students apply their knowledge and understanding of basic programming syntax and expressions to create mathematical formulas and models, and then translate the mathematical models into computer programs.

Unit Two:  **Operators and Expressions**

Students develop code to apply arithmetic operators to accomplish addition, subtraction, multiplication, division, and modulus operations in a C program. Students will understand the order of operations as it pertains to mathematical operators. Students will use relational operators to test the relationship between values and variables. Students will use logical operators to test the relationship between the results of two or more relational operators. Students will use compound assignment operators as a shortcut when modifying the value of a variable. Students will use increment and decrement operators to add or subtract values from/to a variable.

For example, students will be able to write a program to check if a person is old enough to drive. It is assumed that a person of 16 years is eligible to obtain a driver license. The program should let the user enter his/her name and age. Students will need to use selection statements to determine specific criteria for obtaining a driver license. Using formatted output, students will notify the user if they are eligible to obtain a driver license.
Unit Three:  **Flowcharts, Decision Making, Loops, and Random Numbers**

Students will understand the concept of visually planning a computer program using graphical symbols to represent the actions and flow of a computer algorithm. Students will understand how to use selection/decision symbols in a flowchart to represent the path the computer program must take when a conditional statement is executed. Students will use repetition statements with “while” and “for” loops to control how many times a series of statements are executed. Students will understand the importance of looping in computer programming. Students will apply the C statement “break” to exit early from “for” and “while” loops.

For example, using random number generation students will create a number guessing game. Students will write a program to randomly generate a number between a given upper and lower bound. The user will be prompted to guess the random number and will be notified if their guess is too high or too low. The program will exit when the correct number is guessed.

Unit Four:  **Modular Programming with Functions**

Students will understand the concept of using functions to modularize a program. Students will incorporate a function prototype that allows the function to be defined after the main section of a program and understands that function prototyping aids in the readability of a complicated C program that contains many user-defined functions. Students will understand the concept of a “void” type when a function either does not require any argument or does not need to return a value after it is executed. Students will utilize a graphical library to plot functions and computer-generated graphs in different graphical formats. Students will recognize that many mathematical functions are included in the standard C library.

Using functions students will write an interactive game program that simulates playing blackjack with the dealer. Students write a function without argument to display a welcome message as well as functions for the face, color, and suit of the respective cards. Full understanding of decision making and looping in C will be necessary to determine if the player or the dealer is closest to 21 without going over.

Unit Five:  **Arrays for Processing, Organizing, and Displaying Data**

Students will write computer code to initialize and assign values to an array. They will understand how the structure of an array is implemented in computer memory. Students will write computer code to initialize and assign values to an array variable and process the data in the array to find the mean, minimum/maximum values, median, sum, and standard deviation of the data. Students will differentiate the behavior of array variables against non-array variables when passed to a function. Students will understand how information located in arrays can be plotted for graphical analysis. The data can be interpreted using statistical models like scatter plots, dot plots, bar graph, histogram, and Box-and-Whisker plots.

Using basic knowledge of arrays, students will aggregate lab data and store it in an array. Once the values are stored in the array, students will perform basic statistical functions on the array. Students will calculate the mean, maximum/minimum, median, sum, and standard deviation for values of the array using C functions. The data in the array will also be analyzed using scatter plots, dot plots, bar graph, histogram, and Box-and-Whisker plots.
Unit Six: **Working with Data Files**

In this unit students will learn how to store and retrieve information from text data files for numerical and graphical analysis. Students will understand that to open a file you need to use a file pointer to create a stream between the program and file to perform input and output functions. Students will manipulate the date in a file, modifying and outputting specific pieces of data into a different or same file. Students will also learn how to graph the data through more basic procedures using loops and arrays, as well specific plotting functions.

Using the array created in unit five, students will first output the data into a text file. Using file processing capability, students will read in the text file for graphical analysis. Using the graphical plotting ability of Ch students will choose an appropriate graph (scatter plot, dot plot, bar graph, histogram, Box-and-Whisker plot, etc.) and plot their data sets.

Unit Seven: **Graphical Plotting and Quick Animation**

In this unit, students will learn the basic concepts of object-oriented programming in C++. In this unit, students will learn plotting various objects including points, lines, circles, arcs, triangles, and quadrilaterals. Students will also learn how to bring those to life through animation. Students will learn common primitives such as points, text, circles, arcs, lines, polygons, and rectangles. Primitives can be fixed in a specific quadrant or at a particular angle. By dictating placement along a movement path, students will create more complex programs with their animations. Students will learn mathematics through animations by programming the path of a projectile given specific initial conditions. Students will dictate more complex programs that will incorporate motion of a projectile. Multiple primitives can be animated simultaneously creating complex movement. Students will also learn how to use previous topics on functions and loops to conveniently create complicated animations for various applications.

Drawing on knowledge of quadratic equations, students will first solve the equation for the trajectory of a soccer ball kicked from the ground. Students will determine the height of the vertex, the distance of travel, and the time of flight. Using quick animation students will animate this trajectory and solve the problem numerically as well as graphically.

Unit Eight: **Robotics Applications**

Students will learn to program robots to solve real-world application problems in math, science, and engineering. Students will learn to program virtual Linkbot and Lego Mindstorm NXT robotic systems through the robot simulation environment RoboBlockly and RoboSim with virtual robots. If hardware Linkbot or NXT robotic systems is available, students will also learn to program these hardware robotic systems. The same code can be used to control both virtual robots and hardware robots. Students will be
able to move specific joints for specific angles or times and change joint speed. For a robot configured as a
two-wheel vehicle, it can move for a given distance and turn for a given angle. Students will learn how to
set up multiple identical robots in an array as a group so one command can be mimicked amongst the
group. Students will learn the difference between blocking and non-blocking functions and how they relate
to controlling multiple individual robots.

Throughout the unit students will work in teams to complete projects which will include several Linkbots or
NXTs grouped in arrays performing synchronized dancing or acting in a play written and controlled by a
program autonomously. Students will use Linkbots or NXTs to simulate real-world mathematical concepts
like solving systems of linear equations.

If the student has access to physical robots of Linkbot or NXT, they will submit a video to the RoboPlay
Video Competition. Students will create a short 2 – 5 minute video showcasing their robot programming
skill aimed at one of the following categories: Best Storyline, Best Choreography, Most Interesting Task,
Best Custom Designed Part, Best Film Promoting Computational Thinking, and Best Overall Video.

If physical robots are not an option, students will create a complex obstacle course using the simulated
environment in RoboSim. Students will exchange obstacles with classmates and will be challenged to
complete the mazes of their peers.

### Unit Nine: Applications in Mathematics and Science

Students will learn systems of linear equations (point slope, standard form, etc.), linear inequalities,
polynomials such as quadratics, cubic's and others polynomials. Through computer programming students
will use the rules for radicals and exponents and solve exponential growth and decay problems. Students
will be able to use general formulas provided, substitution and combination, to write programs that solve
any linear or quadratic system. Students will graph a system of equations in two variables and then visually
obtain the solution of the system. Students will be able to check arithmetic operations performed on
rational expressions using the Ch command window as well as graph rational functions with the correct
domain and their asymptotes. Students will write interactive code that uses the Pythagorean Theorem,
plots scatter plots, answers area and perimeter questions, calculates distance, compound interest, and
finds the midpoint.

Throughout this course many example of math and science application will be used. Students will learn the
relevance of computer programming in math and science. Many programming concepts will be
demonstrated through exercises. For example: students will use the graphs utility in Ch to solve systems of
two linear equations and inequalities. Students will write programs to solve them numerically as well as
graphically. Students will also learn the motion with position and velocity, and measurement in physical
science graphically, numerically, symbolically, verbally, and experimentally.

(Please feel free to add as many unit fields as necessary.)

### SECTION C: COURSE MATERIALS

Primary Textbook:

*NOTE:* Include list of primary and secondary course materials. Course materials help UC understand what materials are used to support student learning and the delivery of the course.

Title: Learning C Programming: An Introduction to Computer Science

Edition: 1st
Supplemental / Secondary Instructional Materials:

**NOTE:** Please list any other course materials here. These may include but are not limited to: literary texts, manuals, periodicals, articles, websites, primary documents, multimedia, etc.

---

**Secondary Textbook:**

**NOTE:** Include list of primary and secondary course materials. Course materials help UC understand what materials are used to support student learning and the delivery of the course.

**Title:** C for Engineers and Scientists: An Interpretive Approach  
**Edition:** 1st  
**Publication Date:** 2010  
**Publisher:** UC Davis C-STEM Center  
**Author(s):** McGraw Hill  
**URL Resource(s):** [http://iel.ucdavis.edu/cfores/](http://iel.ucdavis.edu/cfores/)

**Title:** Learning Computer Programming with Ch for the Absolute Beginner  
**Edition:** 1st  
**Publication Date:** May 2016  
**Publisher:** UC Davis C-STEM Center  
**Author(s):** Harry H. Cheng  
**URL Resource(s):** [http://c-stem.ucdavis.edu](http://c-stem.ucdavis.edu)
Title: Learning Robotics with Linkbot for the Absolute Beginner
Edition: 6th
Publication Date: May 2016
Publisher: UC Davis C-STEM Center
Author(s): Harry H. Cheng
URL Resource(s): http://c-stem.ucdavis.edu

Title: Learning Common Core Mathematics with C/C++ Interpreter Ch for Integrated Mathematics 1
Edition:
Publication Date: June 2016
Publisher: UC Davis C-STEM Center
Author(s): Harry H. Cheng
URL Resource(s): http://c-stem.ucdavis.edu

(Please feel free to add as many unit fields as necessary.)

[APPENDIX A] — SUBJECT AREAS / DISCIPLINES
NOTE: For detailed information on all subject requirements, please visit the A-G Guide.

_____  ("a") – History / Social Science
      _____  U.S. History
      _____  American Government / Civics
      _____  World History / Geography / Cultures

_____  ("b") – English
      _____  English
      _____  English as a Second Language (ESL)/ English Language Development (ELD)

_____  ("c") – Mathematics
      _____  Algebra 1; Yr 1 of 2
      _____  Algebra 1; Yr 2 of 2
      _____  Algebra 1
      _____  Mathematics 1
      _____  Geometry; Yr 1 of 2
      _____  Geometry; Yr 2 of 2
      _____  Geometry
Mathematics II
Algebra 2; Yr 1 of 2
Algebra 2; Yr 2 of 2
Algebra 2
Mathematics 3
Algebra 2 / Trigonometry
Advanced Mathematics
Statistics
Computer Science

(*d*) – Laboratory Science
Biology
Chemistry
Physics
Earth and/or Space
Integrated Science

**NOTE:** Students that enroll in an integrated-science program (ISP) are strongly advised to complete the entire three-year sequence. In most cases, the first year of an integrated-science sequence fulfills only the "g" elective requirement; the second and third years of the sequence then fulfill the two-year "d" laboratory science requirement. Accordingly, if only ISP I is successfully completed, then two courses from the categories of Biology, Chemistry, or Physics in the "d" subject area must be completed. If ISP I and only one of ISP II or ISP III are completed, then one additional course from the categories of Biology, Chemistry, or Physics from the "d" subject area must be taken to fulfill the "d" requirement.

Interdisciplinary Science

**NOTE:** This category demonstrates that the course is cross-disciplinary and is often used for advanced science courses such as AP Environmental Science or Biochemistry.

(*e*) – Language Other than English
LOTE Year 1
LOTE Year 2
LOTE Year 3
LOTE Year 4+
Language:
American Sign Language (ASL)
Arabic
Chinese
French
German
Hebrew
Italian
Japanese
Latin
Other
_____ Russian
_____ Spanish

_____ ("f") – Visual & Performing Arts
   _____ Dance
   _____ Music
   _____ Theater Arts
   _____ Visual Arts
   _____ Interdisciplinary Arts

x_____ ("g") – College-Preparatory Elective
   _____ History / Social Science
   _____ English
   x_____ Mathematics / Computer Science
   _____ Laboratory Science—Biology / Life Science
   _____ Laboratory Science—Physical Science
   _____ Laboratory Science—Earth / Space Science
   _____ Laboratory Science—Interdisciplinary Science
   _____ Laboratory Science—Integrated Science
   _____ Language Other than English
   _____ Visual & Performing Arts
   _____ Interdisciplinary

[APPENDIX B] — INDUSTRY SECTORS / CAREER PATHWAYS

**NOTE:** This applies to integrated courses only.

_____ Agriculture and Natural Resources
   _____ Agricultural Business
   _____ Agricultural Mechanics
   _____ Agriscience
   _____ Animal Science
   _____ Forestry and Natural Resources
   _____ Ornamental Horticulture
   _____ Plant and Soil Science

_____ Arts, Media, and Entertainment
   _____ Media and Design Arts
   _____ Performing Arts
   _____ Production and Managerial Arts

_____ Building and Construction
   _____ Cabinetmaking and Wood Products
   _____ Engineering and Heavy Construction
   _____ Mechanical Construction
Residential and Commercial Construction
Education, Child Development and Family Services

Child Development
Consumer Services
Education
Family and Human Services

Energy and Utilities
Electromechanical Installation and Maintenance
Energy and Environmental Technology
Public Utilities
Residential and Commercial Energy and Utilities

Engineering and Design
Architectural and Structural Engineering
Computer Hardware, Electrical, and Networking Engineering
Engineering Design
Engineering Technology
Environment and Natural Science Engineering

Fashion and Interior Design
Fashion Design, Manufacturing, and Merchandising
Interior Design, Furnishings, and Maintenance

Finance and Business
Accounting Services
Banking and Related Services
Business Financial Management

Health Science and Medical Technology
Biotechnology Research and Development
Diagnostic Services
Health Information
Support Services
Therapeutic Services

Health Science and Medical Technology

Hospitality, Tourism, and Recreation
Food Service and Hospitality
Food, Science, Dietetics, and Nutrition
Hospitality, Tourism, and Recreation

Information Technology
Information Support and Services
Media Support and Services
Network Communications
x Programming and Systems Development

Manufacturing and Product Development
  Graphic Arts Technology
  Integrated Graphics Technology
  Machine and Forming Technology
  Welding Technology

Marketing, Sales, and Service
  E-Commerce
  Entrepreneurship
  International Trade
  Professional Sales and Marketing

Public Services
  Human Services
  Legal and Government Services
  Protective Services

Transportation
  Aviation and Aerospace Transportation Services
  Collision Repair and Refinishing
  Vehicle Maintenance, Service, and Repair