COLLEGE OF THE DESERT

Course Code CS-007A

Course Outline of Record

Course Code: CS-007A

Long Course Title: Computer Science I - Programming in C++
Short Course Title: COMPUTER SCIENCE I

Catalog Course Description: This course is an introduction to computer programming and is designed primarily for computer science and related transfer major. Its primary objective is to teach problem solving using the C++ programming language. Topics include structured procedural programming with program control structures (sequence, selection, iteration), modular program structures (functions and parameter passing), data types (primitive types, arrays, files and structures) and an intro to object-oriented programming.
Class Schedule Course Description: An introduction to computer programming using the C++ language for students with no programming experience.
Semester Cycle (*if applicable*): *N/A*

Total Units: 4.00 Total Semester Hrs: 108.00
Lecture Units: 3 Semester Lecture Hrs: 54.00
Lab Units: 1 Semester Lab Hrs: 54.00
Class Size Maximum: 35 Allow Audit: No
Repeatability No Repeats Allowed
Justification

Prerequisite or Corequisite Courses or Advisories:
Course with requisite(s) and/or advisory is required to complete Content Review Matrix (CCForm1-A)
Prerequisite: MATH 005 with a minimum grade of C or any other math course with math 40 as a prerequisite

Textbooks, Required Reading or Software: *(List in APA or MLA format.)*
Felleisen, M., Findler, R., B., Flatt, M., Krishnamurthi, s. (2010). *How to Design Programs* (2nd/e). MIT. ISBN: 0-262-06218-6
College Level: Yes
Flesch-Kincaid reading level: 9
Gaddis, Tony (2009). *Starting Out with C++: From Control Structures through Objects* (6th/e). Addison-Wesley. ISBN: 0321545885
College Level: Yes
Flesch-Kincaid reading level: 12
Tony Gaddis, Judy Walters, Godfrey Muganda (2011). *Starting Out with C++: Early Objects* (7th/e). Addison-Wesley. ISBN: 0136077749
College Level: Yes
Flesch-Kincaid reading level: 12
Harvey M. Deitel and Paul J. Deitel (2010). *C++ How to Program* (7th/e). Deitel & Associates, Inc. . ISBN: 0136117260
College Level: Yes
Flesch-Kincaid reading level: 12
Racket. http://racket-lang.org/, (First/e).

Entrance Skills: *Before entering the course students must be able:*
Demonstrate proficiency with polynomial analysis including the fundamental theorem of algebra and its implications.
MATH 005 - Demonstrate that previously learned fundamental skills and knowledge from arithmetic, algebra, and geometry prior learning have been maintained or restored.
MATH 005 - Develop an appreciation for the use of proof in mathematics, with an emphasis on its use in geometry, including the ability to create mathematical proofs of geometric properties.
MATH 005 - Develop deductive reasoning skills in mathematics with emphasis in geometry.
Demonstrate proficiency with analysis of trigonometry functions including amplitude, periodicity, phase shifts, and basic identities.
MATH 005 - Demonstrate problem solving skills in application problems, with emphasis on the concepts of distance and angles

MATH 005 - Demonstrate problem solving skills in application problems, with an emphasis on periodic phenomena.
MATH 005 - Create, analyze, and interpret graphs of trigonometric functions.
MATH 005 - Develop an appreciation for the use of proof in mathematics, with an emphasis on its use in geometry, including the ability to create mathematical proofs of geometric properties.
MATH 005 - Develop deductive reasoning skills in mathematics with emphasis in geometry.
Use basic methods of modeling with exponential functions and the use logarithms to solve exponential equations.
MATH 005 - Demonstrate that previously learned fundamental skills and knowledge from arithmetic, algebra, and geometry prior learning have been maintained or restored.
Demonstrate proficiency with a wide variety of mathematical relations including rational functions, root functions, symmetric functions and the quadratic relations used in modeling shifted conics.
MATH 005 - Demonstrate that previously learned fundamental skills and knowledge from arithmetic, algebra, and geometry prior learning have been maintained or restored.
MATH 005 - Demonstrate problem solving skills in application problems, with emphasis on the concepts of distance and angles
MATH 005 - Demonstrate problem solving skills in application problems, with an emphasis on periodic phenomena.
MATH 005 - Create, analyze, and interpret graphs of trigonometric functions.
MATH 005 - Develop an appreciation for the use of proof in mathematics, with an emphasis on its use in geometry, including the ability to create mathematical proofs of geometric properties.
MATH 005 - Develop deductive reasoning skills in mathematics with emphasis in geometry.
Solve word problems leading to systems of linear and/or non-linear equations in 2 or more variables using methods of elimination and substitution.
MATH 005 - Demonstrate that previously learned fundamental skills and knowledge from arithmetic, algebra, and geometry prior learning have been maintained or restored.
MATH 005 - Demonstrate problem solving skills in application problems, with emphasis on the concepts of distance and angles
MATH 005 - Demonstrate problem solving skills in application problems, with an emphasis on periodic phenomena.
MATH 005 - Create, analyze, and interpret graphs of trigonometric functions.
MATH 005 - Develop deductive reasoning skills in mathematics with emphasis in geometry.

Course Content and Scope:
Lecture:

1. Learn the simple, consistent, easy-to-use Scheme language.
2. Understanding of the concepts of design patterns, which are step-by-step "recipes" for getting from a vague English-language description of a problem to a working computer program.
3. How to distinguish a "good program" from a "bad program."

Lab: *(if the "Lab Hours" is greater than zero this is required)*

1. Complete programming assignments incorporating design elements and code.
2. Consult with the teacher and classmates in small groups to tackle special programming tasks that arise as part of (1), above.

Course Student Learning Outcomes:
After completion of this course, students will be able to design and code small-scale interactive programs in C++, in a well-documented and efficient manner, using the basic looping and decision structures and functions. Students will be able to write basic file input and output routines, use data types ranging from simple variables to strings, arrays, and pointers, and create simple classes.
Students will demonstrate critical thinking skills. Students will interpret and analyze information by categorizing, clarifying meaning in context, identifying ideas, detecting arguments and analyzing arguments into component elements. Students will evaluate ideas by assessing claims and arguments and justifying procedures. Students will draw inferences by questioning evidence, selecting alternatives, and drawing conclusions. Students will demonstrate inductive reasoning skills. Students will demonstrate deductive reasoning skills.

Course Objectives: *Upon completion of this course, students will be able to:*
Locate and operate the basic components in the structure of a computer system.
Experiment with and test various computer software and hardware
Demonstrate the grammar, punctuation, and vocabulary of a programming language by composing original programs.
Demonstrate good program design principles to do problem solving, creating programs that are correct, easy to write, read, modify, and repair.
Describe the sequence of steps to go through, in designing, writing, testing and debugging a program.
Synthesize application-specific knowledge (e.g. to write a program that draws geometric shapes on the screen, you have to know something about geometry.)

Methods of Instruction: *(Integration: Elements should validate parallel course outline elements)*
Laboratory
Lecture

Assignments: *(List samples of specific activities/assignments students are expected to complete both in and outside of class.)*
In Class Hours: 108.00
Outside Class Hours: 108.00
Out-of-class Assignments

1. Read the text.
2. Write descriptions of programs in pseudocode.
3. Finish unfinished lab work.
4. Take quizzes.

In-class Assignments

1. Take quizzes.
2. Take tests.
3. Participate in discussion.
4. Develop original programs to solve given problems.

Methods of Evaluating Student Progress: *The student will demonstrate proficiency by:*
College level or pre-collegiate essays
Written homework
Critiques
Portfolios
Term or research papers
Reading reports
Laboratory projects
Computational/problem solving evaluations
Group activity participation/observation
True/false/multiple choice
Mid-term and final evaluations
Poor attendance/repetitive tardiness

Need/Purpose/Rationale -- *All courses must meet one or more CCC missions.*
PO-GE C4.b - Language & Rationality (Communication & Analytical Thinking)
Raise questions and problems, formulating them clearly and precisely.
Gather, assess, and interpret relevant information.
Express solutions to complex problems using language and logic.
Apply logical and critical thinking to solve problems; explain conclusions; and evaluate, support, or critique the thinking of others.
IO - Critical Thinking and Communication
Apply principles of logic to problem solve and reason with a fair and open mind.
Apply standard conventions in grammar, mechanics, usage and punctuation.
Conduct research, gather and evaluate appropriate information, organize evidence into oral and written presentation, using proper MLA, APA, and other discipline-specific formats to cite sources.
Utilizing various communication modalities, display creative expression, original thinking, and symbolic discourse.

Special Materials and/or Equipment Required of Students:

Materials Fees: ☐ Required Material?

Material or Item	Cost Per Unit	Total Cost

Provide Reasons for the Substantial Modifications or New Course:
This is part of a revamped Computer Science curriculum that will be more robust.

Cross-Listed Course *(Enter Course Code)*: *N/A*
Replacement Course *(Enter original Course Code)*: *N/A*

Grading Method *(choose one)*: Letter Grade Only

MIS Course Data Elements
Course Control Number [CB00]: *N/A*

T.O.P. Code [CB03]: 70710.00 - Computer Programming
Credit Status [CB04]: D - Credit - Degree Applicable
Course Transfer Status [CB05]: A = Transfer to UC, CSU
Basic Skills Status [CB08]: 2N = Not basic skills course
Vocational Status [CB09]: Possibly Occupational
Course Classification [CB11]: I - Career-Technical Education
Special Class Status [CB13]: N - Not Special
Course CAN Code [CB14]: *N/A*
Course Prior to College Level [CB21]: Y = Not Applicable
Course Noncredit Category [CB22]: Y - Not Applicable
Funding Agency Category [CB23]: Y = Not Applicable
Program Status [CB24]: 1 = Program Applicable
Name of Approved Program *(if program-applicable)*: COMPUTER SCIENCE
Attach listings of Degree and/or Certificate Programs showing this course as a required or a restricted elective.)

Enrollment - Estimate Enrollment
First Year: 40
Third Year: 50

Resources - Faculty - Discipline and Other Qualifications:
Sufficient Faculty Resources: Yes
If No, list number of FTE needed to offer this course: *N/A*

Additional Equipment and/or Supplies Needed and Source of Funding.

N/A

Additional Construction or Modification of Existing Classroom Space Needed. *(Explain:)*

N/A

FOR NEW OR SUBSTANTIALLY MODIFIED COURSES
Library and/or Learning Resources Present in the Collection are Sufficient to Meet the Need of the Students Enrolled in the Course:
No

Originator Geoffrey Hagopian Origination Date 24/06/10