Cerro Coso College

# Course Outline of Record Report

**10/13/2021**

## CSCIC252 : Introduction to Computer Science

### General Information

| | |
|---|---|
| **Author:** | - Tech Support |
| **Course Code (CB01) :** | CSCIC252 |
| **Course Title (CB02) :** | Introduction to Computer Science |
| **Department:** | Business Information Technolog |
| **Proposal Start:** | Spring 2014 |
| **TOP Code (CB03) :** | (0706.00) Computer Science (transfer) |
| **SAM Code (CB09) :** | Non-occupational |
| **Distance Education Approved:** | Yes |
| **Course Control Number (CB00) :** | CCC000360376 |
| **Curriculum Committee Approval Date:** | 05/03/2013 |
| **Board of Trustees Approval Date:** | 06/13/2013 |
| **External Review Approval Date:** | 06/13/2013 |
| **Course Description:** | This course explores topics that provide students with a foundation in computer science. This course covers the fundamental issues of algorithms, computer organization, software, computational theory, fundamental object oriented programming and social and ethical issues of computing. |
| **Submission Type:** | New Course |
| **Author:** | No value |

### Faculty Minimum Qualifications

| | |
|---|---|
| **Master Discipline Preferred:** | - Computer Science |
| **Alternate Master Discipline Preferred:** | No value |
| **Bachelors or Associates Discipline Preferred:** | No value |
| **Additional Bachelors or Associates Discipline Preferred:** | No value |

### Course Development Options

**Basic Skills Status (CB08)**

Course is not a basic skills course.

**Course Special Class Status (CB13)**

Course is not a special class.

**Grade Options**

□ Allow Students to Gain Credit by Exam/Challenge

**Allowed Number of Retakes**

0

**Course Prior To College Level (CB21)**

Not applicable.

**Rationale For Credit By Exam/Challenge**

**Retake Policy Description**

□ Allow Students To Audit C

| | | |
|---|---|---|
| No value | No value | □ Allow Students To Audit Course |

**Course Support Course Status (CB26)**

No value

## Associated Programs

☑ Course is part of a program (CB24)

| Associated Program | Award Type | Active |
|---|---|---|
| CC Liberal Arts: Mathematics & Science | A.A. Degree Major | Summer 2018 to Fall 2020 |
| Associate in Science Degree In Mathematics for Transfer | A.A. Degree for Transfer | Summer 2018 |
| Liberal Arts: Mathematics & Science Associate in Arts Degree | A.A. Degree Major | Fall 2020 |

## Transferability & Gen. Ed. Options

**Course General Education Status (CB25)**

No value

| Transferability | Transferability Status |
|---|---|
| Transferable to both UC and CSU | Approved |

## Units and Hours:

### Summary

| | |
|---|---|
| **Minimum Credit Units (CB07)** | 3 |
| **Maximum Credit Units (CB06)** | 3 |
| **Total Course In-Class (Contact) Hours** | 90 |
| **Total Course Out-of-Class Hours** | 72 |
| **Total Student Learning Hours** | 162 |
| **Faculty Load** | 0 |

### Credit / Non-Credit Options

**Course Credit Status (CB04)**

Credit - Degree Applicable

**Course Non Credit Category (CB22)**

Credit Course.

**Non-Credit Characteristic**

No Value

**Course Classification Status (CB11)**

Credit Course.

**Funding Agency Category (CB23)**

Not Applicable.

☐ Cooperative Work Experience Education Status (CB10)

☐ Variable Credit Course

## Weekly Student Hours

| | In Class | Out of Classs |
|---|---|---|
| Lecture Hours | 2 | 4 |
| Laboratory Hours | 3 | 0 |
| Activity Hours | 0 | 0 |

## Course Student Hours

| | |
|---|---|
| **Course Duration (Weeks)** | 18 |
| **Hours per unit divisor** | 0 |
| **Course In-Class (Contact) Hours** | |
| Lecture | 0 |
| Laboratory | 0 |
| Activity | 0 |
| **Total** | 90 |
| **Course Out-of-Class Hours** | |
| Lecture | 0 |
| Laboratory | 0 |
| Activity | 0 |
| **Total** | 72 |

## Time Commitment Notes for Students

No value

## Faculty Load

**Extra Duties:** 0

**Faculty Load:** 0

## Units and Hours: - Weekly Specialty Hours

| Activity Name | Type | In Class | Out of Class |
|---|---|---|---|
| No Value | No Value | No Value | No Value |

## Pre-requisites, Co-requisites, Anti-requisites and Advisories

### Prerequisite

MATHC055 - Intermediate Algebra

Math skills are required for the algorithmic definitions, conversions between decimal and other number systems, and logical skills required.

**AND**

### Advisory

CSCIC101 - Introduction to Computer Information Systems

Students must be able to install software, understand the basic of networking, databases, and an idea of what programming is. All of this is discussed in the CSCI C101 course.

## Entrance Skills

| Entrance Skills | Description |
|---|---|
| No value | No value |

## Limitations on Enrollment

| Limitations on Enrollment | Description |
|---|---|
| No value | No value |

## Specifications

### Methods of Instruction

| | |
|---|---|
| **Methods of Instruction** | Computational Work |
| **Rationale** | No value |

| | |
|---|---|
| **Methods of Instruction** | Demonstration |
| **Rationale** | No value |

| | |
|---|---|
| **Methods of Instruction** | Discussion |
| **Rationale** | No value |

| | |
|---|---|
| **Methods of Instruction** | Laboratory |
| **Rationale** | No value |
| **Methods of Instruction** | Lecture |
| **Rationale** | No value |
| **Methods of Instruction** | Outside reading |
| **Rationale** | No value |
| **Methods of Instruction** | Problem Solving |
| **Rationale** | No value |
| **Methods of Instruction** | Project-based learning |
| **Rationale** | No value |
| **Methods of Instruction** | Skills Development and Performance |
| **Rationale** | No value |

### Assignments

A. Reading Text – Preparing for class by reading the chapters assigned
B.Programming Assignments – Programming assignments every 2 weeks
C. Homework Assignments – Problem sets as handouts or from the text to practice concepts,
preparing for presenting topics on the course
D. Group work – Group work time for a group project

| Methods of Evaluation | Rationale |
|---|---|
| Tests | Objective tests/quizzes demonstrating student's knowledge of algorithmic problem solving methods, components of a computer system, and networking concepts. Examples include: asking them to create an algorithm which uses addition to multiply to numbers, describe the parts of the von Neumann architecture |
| Research Paper | Essays demonstrating students' understanding of social computing issues. Example: Discuss a social computing issue from a current event. What is happening and how will it impact the future of computing? |
| Homework | Programming assignments demonstrating student's ability to design a basic application using an object-oriented programming language. Example: Create a program in C++ which creates a pay stub for employees. The inputs are the employee id, hours worked, and hourly wage. You must output their gross pay (including any overtime defined as over 40 hours a week), taxes taken out and net pay. |

### Equipment

Microsoft.
Microsoft Visual Studio C++ Express Edition
, Express 2010 ed.

-- Free C++ compiler for Windows machines

### Textbooks

| Author | Title | Publisher | Date | ISBN |
|---|---|---|---|---|
| Kubica, J | Computational Fairy Tales | Course Technology | 2012 | |
| Kubica Gersting, Judith L., and Michael Schneider | Invitation to Computer Science | Course Technology | 2010 | |

### Other Instructional Materials

No Value

### Materials Fee

No value

## Learning Outcomes and Objectives

### Course Objectives

No value

### CSLOs

**Design, implement, test, and debug a program using fundamental programming and fundamental data structures including basic computation, simple I/O, conditional and iterative structures, and functions.**
Expected SLO Performance: 70.0

**Choose professional behavior in response to ethical issues inherent in computing.**
Expected SLO Performance: 70.0

**Differentiate between object oriented, structured, and functional programming methodologies using the history of programming languages.**
Expected SLO Performance: 70.0

**Use pseudo code, trace the execution, test, and debug algorithms for solving simple problems.**
Expected SLO Performance: 70.0

**Demonstrate different forms of binding, visibility, scoping, and lifetime management.**
Expected SLO Performance: 70.0

## Outline

### Course Outline

A. History of Computing
  a. Prehistory – the world before 1946

- b. History of computer hardware, software networking
    - c. Pioneers of computing
    - d. Overview of Operating Systems
- B. Societal and Professional Issues and Ethical Conduct
    - a. Codes of ethics and responsible conduct
    - b. Intellectual property, copyright, plagiarism
    - c. Computing and the internet
    - d. Social impact of computing
    - e. Privacy
- C. Machine and Assembly level representation of data
    - a. Von Neumann Architecture
    - b. Bits, Bytes, and words
    - c. Numeric data representation and number bases
    - d. Representation of nonnumeric data
    - e. Control unit; instruction fetch, decode and execution
    - f. Instruction sets and types
- D. Basic Computability
    - a. Finite-state machines
    - b. Turing machine
- E. Algorithms and Problem-solving
    - a. Problem solving strategies
    - b. The role of algorithms in the problem solving process
    - c. Implementation strategies for algorithms and debugging strategies
    - d. Concept and properties of algorithms
    - e. Simple algorithm development
- F. Program Development
    - a. Software lifecycle
- i.Program development phases
    - b. Fundamental design concepts and principles
    - c. Requirements elicitation
    - d. Test case design
        - i. Black and White-box testing
        - ii. Unit , integration, validation testing
    - e. Characteristics of maintainable software
- G. Overview of Programming Languages
    - a. History of programming languages
    - b. Brief survey of programming paradigms
        - i. Scripting languages
        - ii. Procedural languages
        - iii. Functional Languages
        - iv. Object oriented languages
- H. Software tools and environment
    - a. Setting up software tool
    - b. Compiling, interpreting, linking, executing, testing and debugging in tool
- I. Fundamental Programming
    - a. Basic syntax and semantics of a higher-level language
    - b. Data Types
        - i. Concept of types as values and operations
        - ii. Binding, Visibility, scope and lifetime
    - c. Variables
    - d. Expressions
    - e. Assignment
    - f. Commenting
- J. Conditional Control Structures
    - a. Comparisons
    - b. Boolean Logic
    - c. If
    - d. If/Else
    - e. Switch
- K. Iterative Control Structures
    - a. While
    - b. For
    - c. Do While
    - d. Input validation
- L. Fundamental Data Structures
    - a. Arrays
    - b. Strings and string processing

    - c. Data representation in memory
- M. Functions
    - a. Functions as abstraction
    - b. Parameters
        - i. Reference vs value
    - c. Data Structures and function
    - d. Reusability
- N. Object Oriented Programming
    - a. Objects
    - b. Classes
    - c. Methods
    - d. Separation of behavior and implementation
    - e. Inheritance
- i. Encapsulation and information-hiding
    - f. Polymorphism
    - g. Reusability
- O. Graphical Programming
    - a. Creating a GUI
    - b. Event-Driven programming

### Lab Outline

- A. Machine and Assembly level representation of data
    - a. Control unit and Von Neumann architecture.
    - b. Number Base Conversion
- B. Basic Computability
    - a. Finite-state machines
    - b. Turing machine
- C. Algorithms and Problem-solving
    - a. Implementation and debugging strategies
    - b. Simple algorithm development
- D. Software tools and environment
    - a. Setting up software tool
    - b. Compiling, interpreting, linking, executing, testing and debugging in tool
- E. Fundamental Programming
    - a. Basic syntax and semantics of a higher-level language
    - b. Data Types
    - c. Variables
    - d. Expressions
    - e. Assignment
    - f. Commenting
- F. Conditional Control Structures
    - a. Comparisons
    - b. Boolean Logic
    - c. If
    - d. If/Else
    - e. Switch
- G. Iterative Control Structures
    - a. While
    - b. For
    - c. Do While
    - d. Input validation
- H. Fundamental Data Structures
    - a. Arrays
    - b. Strings and string processing
    - c. Structs
- I. Functions
    - a. Function creation
    - b. Parameters
        - i. Reference vs value
    - c. Data Structures and function
    - d. Reusability
- J. Object Oriented Programming
    - a. Objects
    - b. Classes

c. Methods

K. Graphical Programming

a. Creating a GUI

b. Event-Driven programming

## Delivery Methods and Distance Education

**Delivery Method: Please list all that apply -Face to face -Online (purely online no face-to-face contact) -Online with some required face-to-face meetings (“Hybrid”) -Online course with on ground testing -iTV – Interactive video = Face to face course with significant required activities in a distance modality -Other**

Face 2 Face
Online
Hybrid
Interactive

**Rigor Statement: Assignments and evaluations should be of the same rigor as those used in the on-ground course. If they are not the same as those noted in the COR on the Methods of Evaluation and out-of-class assignments pages, indicate what the differences are and why they are being used. For instance, if labs, field trips, or site visits are required in the face to face section of this course, how will these requirements be met with the same rigor in the Distance Education section?**

No Value

**Effective Student-Instructor Contact: Good practice requires both asynchronous and synchronous contact for effective contact. List the methods expected of all instructors teaching the course. -Learning Management System -Discussion Forums -Moodle Message -Other Contact -Chat/Instant Messaging -E-mail -Face-to-face meeting(s) -Newsgroup/Discussion Board -Proctored Exam -Telephone -iTV - Interactive Video -Other (specify)**

contact_moodle_forums
contact_moodle_message
contact_email
contact_phone
contact_itv
contact_other

**Software and Equipment: What additional software or hardware, if any, is required for this course purely because of its delivery mode? How is technical support to be provided?**

No Value

**Accessibility: Section 508 of the Rehabilitation Act requires access to the Federal government's electronic and information technology. The law covers all types of electronic and information technology in the Federal sector and is not limited to assistive technologies used by people with disabilities. It applies to all Federal agencies when they develop, procure, maintain, or use such technology. Federal agencies must ensure that this technology is accessible to employees and the public to the extent it does not pose an "undue burden". I am using -iTV—Interactive Video only -Learning management system -Publisher course with learning management system interface.**

s508_itv
s508_moodle
s508_publisher

**Class Size: Good practice is that section size should be no greater in distance ed modes than in regular face-to-face versions of the course. Will the recommended section size be lower than in on-ground sections? If so, explain why.**

No Value