Course Outline for CS 7

INTRODUCTION TO COMPUTER PROGRAMMING CONCEPTS

Effective: Fall 2020

I. CATALOG DESCRIPTION:
  CS 7 — INTRODUCTION TO COMPUTER PROGRAMMING CONCEPTS — 3.00 units

  An introductory course in computer programming concepts and fundamental coding skills using object-oriented languages like Python. Material includes problem-solving techniques, design of algorithms, and common programming constructs such as variables, expressions, input/output, decision-making, loops and arrays.

   2.50 Units Lecture 0.50 Units Lab

Grading Methods:
Letter or P/NP

Discipline:
  • Computer Science

	MIN
Lecture Hours:	45.00
Expected Outside of Class Hours:	90.00
Lab Hours:	27.00
Total Hours:	162.00

II. NUMBER OF TIMES COURSE MAY BE TAKEN FOR CREDIT: 1

III. PREREQUISITE AND/OR ADVISORY SKILLS:

IV. MEASURABLE OBJECTIVES:
Upon completion of this course, the student should be able to:
  A. Design simple algorithms to solve a variety programming problems.
  B. Design and implement programs of short to medium length, using standard elements of programming languages such as variables, input/output, control structures, functions/methods and arrays.
  C. Describe the software development life-cycle.
  D. Describe the principles of structured and object-oriented programming and be able to describe, design, implement, and test structured and object-oriented programs using currently accepted methodology.
  E. Explain what an algorithm is and its importance in computer programming.
  F. Analyze and investigate program behavior to effectively alter or debug existing code.
  G. Design and implement specific program steps and components to achieve desired program behavior.
  H. Design and organize elements of a program using a structured representation such as pseudocode and/or flowcharts.
   I. Design and implement simple graphical and command line user interfaces implementing the students algorithms.

V. CONTENT:
  A. Computer Systems
      1. System overview
      2. Distinction between hardware and software
  B. Programming Concepts
      1. History of Computation and programming languages
      2. Types and purposes of programming languages, including procedural versus object-oriented programming
          a. Survey of current languages
  C. Program Development
      1. Programming design tools and programming environments
      2. Documentation
      3. Software life-cycle including design, development, styles, documentation, testing and maintenance
      4. Principles of testing and designing test data
  D. Programming Language Concepts and Syntax
      1. Data types, variables and expressions
      2. Developing algorithms and program steps for sequential processing
      3. Coding conventions
      4. Arithmetic expressions
      5. Boolean expressions
      6. Control structures

          a. Selective structures, such as **if** and **switch**

          b. Repetitive structures (loops)

     7. Arrays and lists

          a. Declaring, allocating and accessing arrays

          b. Multiple-subscripted (multi-dimensional) arrays

          c. Ordered collection object type: List

          d. Unordered Key-value paired collection object type: Dictionary

          e. Ordered collections of arbitrary objects : Tuple

     8. Error handling

     9. File I/O, including file streams and sequential access

     10. Modular code using functions/methods

VI. METHODS OF INSTRUCTION:

    A. **Lab** -

    B. **Projects** -

    C. **Demonstration** -

    D. **Lecture** -

    E. **Discussion** -

VII. TYPICAL ASSIGNMENTS:

    A. Using a loop and arithmetic expressions, get numerical data from user input and report typical descriptive statistics for that data (e.g., maximum and minimum values, mean, median, frequency of different values).

    B. Write a program to simulate the playing of a simple card game, using arrays/lists to represent a card deck and players' hands.

    C. Create a program to read sequential data from a text file representing students and associated information about each student, providing the user an interface for querying, inserting and editing records.

    D. Create a program to present to the user a simple GUI window asking them for simple input and then, after pressing a button in the UI, manipulate that user input into a measurable result.

VIII. EVALUATION:

Methods/Frequency

    A. Exams/Tests

        Mid-term and Final Exams at least twice per semester (mid-term and end of term)

    B. Quizzes

        Quizzes for each chapter of the text will be given when that chapter's materials are completed in class.

    C. Projects

        Programming Projects testing the students retention of lecture and lab materials after each chapter is completed during class.

    D. Class Work

        Written and/or interactive exercises during the presentation of each chapter's materials during class

IX. TYPICAL TEXTS:

    1. Gaddis, Tony. *Starting Out in Python.* 4th ed., Addison-Wesley, 2018.

    2. John, Zelle. *Python Programming: An Introduction to Computer Science.* 3rd ed., Franklin, Beedle & Associates Inc., 2016.

    3. Downey, Allen. *Think Python: How to Think Like a Computer Scientist.* 2nd ed., O'Reilly, 2016.

X. OTHER MATERIALS REQUIRED OF STUDENTS:

    A. It is recommended that students have a portable data-storage device (i.e. USB drive) or maintain an active cloud-storage account to facilitate saving and transfer of their work.