Computer Programming IA

Levels: 10-12
Units of Credit: 0.50
CIP Code: 11.0201
Core Code: 35020000030
Prerequisites: Algebra I, Keyboarding Proficiency, Computer Technology
Skill Test: #820 Computer Programming 1A

COURSE DESCRIPTION
An introductory course in computer programming/software engineering and applications. The course introduces students to the fundamentals of computer programming. Students will learn to design, code, and test their own programs while applying mathematical concepts. Teachers introduce concepts and problem solving skills to beginning students through a programming language such as Delphi, C++, C#, Java, Python, or VB.

CORE STANDARDS, OBJECTIVES AND INDICATORS

STANDARD 1
Students will be familiar with and use a programming environment.

Objective 1: Demonstrate knowledge of external and internal computer hardware.
a. Describe the functions of basic computer hardware devices (monitor, printer, CD-ROM drive, floppy drive, keyboard, mouse, adapters, other devices).
b. Describe the functions of the internal components of computers (CPU, RAM, ROM, Motherboard, graphics card, hard drive).
c. Utilize the binary numbering system (translate from binary to decimal and vice-versa).

Objective 2: Demonstrate knowledge of software concepts.
a. Define the distinction between computer software and hardware.
b. Identify software categories such as Application Software, Web Based Software, OS, Utility Software (anti virus, system tools).
c. Describe the difference between an interpreted language vs a compiled language
d. Describe the difference between a low level and high level language

Objective 3: Develop the ability to use a current operating system.
a. Demonstrate how to open and save files.
b. Demonstrate how to move, rename, copy, compress and delete files.
c. Demonstrate how to display and print files.
d. Create and use appropriate directory and path structures
e. Demonstrate how to execute a program.

Objective 4: Demonstrate the ability to use the editor to write code.
a. Demonstrate the process of selecting a block of text.
b. Demonstrate how to move, copy, and delete blocks of text.

Objective 5: Demonstrate the ability to compile, debug, and execute programs.
a. Demonstrate how to use the editor to compile and run programs.
b. Understand the difference between syntax, run-time, and logic errors.

c. Demonstrate how to debug programs.
d. Optional -- Use a debugger to set break-points, and step through code to track down errors at runtime

STANDARD 2
Students will employ accepted programming methodology.

Objective 1: Demonstrate the ability to use good programming style.
a. Demonstrate how to use white space properly.
b. Employ proper naming conventions (such as Camel Case and Pascal Case).
c. Construct programs with meaningful identifiers.

Objective 2: Follow the major steps of a Software Development Life Cycle (SDLC).
a. Prepare specifications and requirements for computer programs.
b. Design solutions using algorithms such as flow charts, pseudo code, and basic UML.
c. Implement the code for a program.
d. Test programs for effectiveness and completeness.
e. Provide documentation for a program (such as internal and external documentation).

Objective 3: Identify the syntactical components of a program
a. Identify keywords, identifiers, operators, operands, and literals
b. Identify the entry-point of a program
c. Identify statements and expressions in a program
d. Identify subroutines in a program

STANDARD 3
Students will properly use language-fundamental commands and operations.

Objective 1: Demonstrate the ability to use basic elements of a specific language.
a. Write programs using a language-specific template.
b. Declare, initialize, and assign values to constants and variables.
c. Output text with formatting.
d. Demonstrate the ability to use input/output commands.
e. Output values stored in identifiers.

Objective 2: Employ basic arithmetic expressions in programs.
a. Use basic arithmetic operators (addition, subtraction, modulus, multiplication, division)
b. Understand order of operation of expressions
c. Write expressions that mix floating-point and integer expressions.
d. Write expressions to accumulate values.

Objective 3: Demonstrate the ability to use data types in programs.
a. Declare and use primitive data types (integer, floating point, Boolean)
b. Declare and use reference (non-primitive) types
c. Declare and use constants.
d. Optional -- Declare and use enumerators as a list of constants

Objective 4: Demonstrate the ability to use strings in programs.
a. Declare string identifier.
b. Input string identifiers.
c. Output string identifiers.

STANDARD 4
Students will properly employ control structures.

Objective 1: Demonstrate the ability to use relational and logical operators in programs.
a. Compare values using relational operators.
b. Form complex expressions using logical operators.

Objective 2: Demonstrate the ability to use decisions in programs.
a. Employ simple IF structures.
b. Use IF-ELSE structures.
c. Write programs with nested IF-ELSE structures.
d. Make multiple-way selections (switch, case).

Objective 3: Demonstrate the ability to use loops in programs.
a. Use initial, terminal, and incremental values in loops.
b. Construct both pre-test and post-test loops.
c. Demonstrate how to use counted loops.
d. Describe the use of flagged (sentinel-controlled) loops.
e. Utilize nested loops.
f. Explain how to avoid infinite loops.
g. Accumulate running totals using loops.

Objective 4: Demonstrate the ability to use modularity in programs.
a. Demonstrate how to use language-defined subroutines.
b. Develop and utilize subroutines.
c. Utilize value and reference parameters.
d. Understand the scope of identifiers in subroutines.
e. Return values from subroutines.

STANDARD 5
Students will demonstrate knowledge of current ethical issues dealing with computers and information in society.

Objective 1: Understand ethical responsibility of software developers
a. Explain the ethical reasons for creating reliable and robust software.
b. Explain the impact software can have on society.
c. Show how security concerns can be addressed in a program.

Objective 2: Demonstrate knowledge of the social and ethical consequences of computers.
a. Describe how computer-controlled automation affects a workplace and society.
b. Explain the ramifications of society's dependence on computers.
c. Identify advantages and disadvantages of changing workplace environments.

Objective 3: Demonstrate knowledge of the right to privacy.
a. Explain how computers can compromise privacy.
b. Exhibit knowledge of privacy laws.
c. Describe responsibilities of people who control computer information.

Objective 4: Demonstrate knowledge of computer, information and software security.

a. Exhibit knowledge of copyright laws.
b. Explain how computers could erroneously be used to compromise copyright laws.
c. Give examples of ways to protect information on computer systems.
d. Identify ways to protect against computer viruses.

STANDARD 6

Students will develop an awareness of career opportunities in the Computer Programming/Software Engineering industry and of its history.

Objective 1: Identify personal interests and abilities related to Computer Programming/Software Engineering careers

a. Identify personal creative talents
b. Identify technical/programming talents
c. Identify organizational and leadership skills
d. Explore aptitude for innovation
e. Determine aptitude for working as a member of a Computer Programming/Software Engineering team

Objective 2: Identify Computer Science career fields

a. Understand the work of a Software Engineer
b. Understand what a Systems Analyst does
c. Understand the kind of work performed by a Applications Programmer (Gaming, Multimedia Etc.)

Objective 3: Investigate career opportunities, trends, and requirements related to Computer Programming/Software Engineering careers

a. Identify the members of a Computer Programming/Software Engineering team: Team Leader, Analyst, Sr. Developer, Jr. Developer, and Client/Subject Matter Expert
b. Describe work performed by each member of the Computer Programming/Software Engineering team
c. Investigate trends associated with Computer Programming/Software Engineering careers
d. Develop a realistic Student Education Occupation Plan (SEOP) to help guide further educational pursuits

Objective 4: Identify factors for employ ability and advancement in Computer Programming/Software Engineering careers

a. Survey existing Computer Programming/Software Engineering businesses to determine what training is required
b. Survey universities and colleges to determine higher education options
c. Develop employ ability competencies/characteristics: responsibility, dependability, respect, and cooperation
d. Achieve high standards of personal performance
e. Develop a positive work ethic
f. Compile a portfolio of the individual and group programs developed during the course

Objective 5: Discuss relevant history of software development

a. Discuss relevant history of computer technology
b. Identify key points in the history of the Computer Programming/Software Engineering industry