CompuScholar, Inc.

Alignment to Utah's **Computer Programming I** Standards

Course Title: **TeenCoder: Windows Programming**
Course ISBN: **978-0-9887070-0-9**
Course Year: **2015**

Note: Citation(s) listed may represent a subset of the actual instances where objectives are met throughout the course.

Computer Programming I (July, 2013 revision)
Levels: 10-12
Units of Credit: 1.0
CIP Code: 11.0201
Core Code: 35-02-00-00-030
Prerequisites: Secondary Math I, Keyboarding Proficiency, Computer Literacy requirement

Semester 1 Skill Test: #820 Computer Programming 1A
Semester 2 Skills Tests: #822 Computer Programming IB (C++)
#824 Computer Programming IB (Java)
#826 Computer Programming IB (VB)
#827 Computer Programming IB (Python)
#828 Computer Programming IB (C#)

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 C++, C#, Java, Python, or VB.

The second half of the year reviews and builds on the concepts introduced in the first semester. This semester introduces students to more complex data structures and their uses, including sequential files, arrays, and classes. Students will learn to create more powerful programs.

(*Semester 2 objectives)

CORE STANDARDS, OBJECTIVES AND INDICATORS

STANDARD 1	**CITATION(S)**
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 external computer hardware devices (monitor, printer, keyboard, mouse, adapters, other devices).	Chapter 1, Lesson 2 (except floppy drive)
b. Describe the functions of the internal components of computers (CPU, RAM, ROM, motherboard, graphics card, hard drive, optical drive).	Chapter 1, Lesson 2
c. Understand what a bit and a byte is and how it relates to memory storage.	Chapter 4, Lesson 1
Objective 2: Demonstrate knowledge of software concepts.	
a. Define the distinction between computer software and hardware.	Chapter 1, Lesson 2 Chapter 1, Lesson 3
b. Identify software categories such as application software, web-based software, or OS.	Chapter 1, Lesson 3 Chapter 1, Lesson 5
c. Describe the difference between an interpreted language vs a compiled language	Chapter 2, Lesson 1
Objective 3: Demonstrate the ability to compile, debug, and execute programs.	
a. Demonstrate how to use an editor/IDE to compile and run programs.	Chapter 2, Lesson 1 Chapter 2, Lesson 3
b. Understand the difference between syntax, run-time, and logic errors.	Chapter 10, Lesson 3
c. Demonstrate how to debug programs.	Chapter 10, Lesson 2
d. Optional -- Use a debugger to set break-points, and step through code to track down errors at runtime	Chapter 10, Lesson 1 Chapter 10, Lesson 2

STANDARD 2	**CITATION(S)**
Students will employ accepted programming methodology.	
Objective 1: Demonstrate the ability to use good programming style.	
a. Demonstrate how to use white space properly.	Chapter 2, Lesson 3
b. Employ an appropriate naming convention	Chapter 4, Lesson 2
c. Construct identifiers with meaningful format (ie: camelCase, under_scores,PascalCase, and ALLCAPS).	Chapter 4, Lesson 2

Objective 2: Understand that software development is a process and use a variety of creation techniques to develop 21st Century Skills.(www.p21.org).	
a. Understand specifications and requirements for computer programs.	Chapter 7, Lesson 3
b. Decompose the problem into appropriate components.	Chapter 7, Lesson 3
c. Design solutions using algorithms and other problem solving techniques	Chapter 7, Lesson 3
d. Write the code for a program.	Chapter 7, Activity 1; Plus, every chapter has a hands-on coding project.
e. Test programs for errors and proper functionality.	Chapter 10, Lesson 3 Chapter 10, Lesson 4
f. Provide internal and external documentation for a program during development.	n/a
g. Redo all steps as needed.	Supplemental Activity 2
Objective 3: Identify the syntactical components of a program	
a. Identify keywords, identifiers, operators, operands, and literals	Chapter 4, Lesson 1 Chapter 4, Lesson 2 Chapter 5, Lesson 1, and other keywords introduced as needed in other lessons
b. Identify the entry-point of a program	Chapter 2, Lesson 3
c. Identify statements and expressions in a program	Chapter 2, Lesson 3 Chapter 5, Lesson 1
d. Identify program components such as functions, methods, or procedures.	Chapter 9

STANDARD 3	**CITATION(S)**
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 formatted based on the conventions of the utilized language.	Chapter 2, Lesson 3 Chapter 3, Lesson 1 (Plus students write programs in all chapter activities)

b. Declare, initialize, and assign values to constants and variables.	Chapter 4, Lesson 2
c. Demonstrate the ability to use input and output commands.	Chapter 3, Lesson 3 Chapter 6
d. Communicate clearly with output values stored in identifiers. (www.p12.org).	Chapter 2, Lesson 3 Chapter 8, Lesson 2
e. Demonstrate the ability to use strings in programs.	Chapter 8
Objective 2: Employ basic arithmetic expressions in programs.	
a. Use basic arithmetic operators (modulus, multiplication, division, addition, subtraction).	Chapter 7, Lesson 1
b. Understand order of operation of expressions	Chapter 5, Lesson 1
c. Write expressions that mix floating-point and integer expressions.	Chapter 4, Lesson 2 (casting)
Objective 3: Demonstrate the ability to use data types in programs.	
a. Declare and use variable types (primitives, reference, or object).	Chapter 4, Lesson 1 Chapter 4, Lesson 2 Chapter 4, Lesson 3 Chapter 13, Lesson 1
b. Declare and use constants.	Chapter 4, Lesson 2
c. Know the difference between data types and their application (boolean, integer, floating point, strings).	Chapter 4, Lesson 1 Chapter 4, Lesson 2 Chapter 4, Lesson 4
d. Optional -- Declare and use enumerators as a list of constants	Chapter 4, Lesson 1
Objective 4: Demonstrate the ability to use strings in programs.	
a. Declare string identifier.	Chapter 2, Lesson 4 Chapter 4, Lesson 4
b. Input string identifiers.	Chapter 2, Lesson 4
c. Output string identifiers.	Chapter 2, Lesson 3 Chapter 6 Activity

STANDARD 4	**CITATION(S)**
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.	Chapter 5, Lesson 1
b. Form complex expressions using logical operators.	Chapter 5, Lesson 1
Objective 2: Demonstrate the ability to use decisions in programs.	
a. Employ simple IF structures.	Chapter 5, Lesson 2
b. Use IF-ELSE structures.	Chapter 5, Lesson 2
c. Write programs with nested IF-ELSE structures.	Chapter 17, Activity 3
d. Make multiple-way selections (switch, case).	n/a

Objective 3: Demonstrate the ability to use loops in programs.	
a. Use initial, terminal, and incremental values in loops.	Chapter 5, Lesson 3
b. Construct while, do-while, and for loops	Chapter 5, Lesson 3 Chapter 5, Lesson 4
c. Describe the various ways that loops can end.	Chatper 5, Lesson 3 Chapter 5, Lesson 4
d. Utilize nested loops.	Chapter 14 Activity
e. Explain how to avoid infinite loops.	Chapter 5, Lesson 4
f. Accumulate running totals using loops.	Chapter 5, Lesson 4
Objective 4: Demonstrate the ability to use modularity in programs using functions or methods.	
a. Demonstrate how to use language-defined components.	Chapter 3, Lesson 4 Chapter 7, Lesson 2
b. Utilize value and reference parameters.	Chapter 9, Lesson 2 Chapter 9, Lesson 3
c. Understand the scope of identifiers (local, class variables).	Chapter 4, Lesson 2 Chapter 13, Lesson 2
d. Return values.	Chapter 9, Lesson 2 Chapter 9, Lesson 3

STANDARD 5	**CITATION(S)**
Students will demonstrate knowledge of current ethical issues dealing with computers and information in a global society using 21st Century Skills..	
Objective 1: Understand ethical responsibility of software developers	
a. Explain the ethical reasons for creating reliable and robust software.	Chapter 1, Lesson 5
b. Explain the impact software can have on society.	Chapter 1, Lesson 5
c. Show how security concerns can be addressed in a program.	Chapter 1, Lesson 5
Objective 2: Demonstrate knowledge of the social and ethical consequences of computers.	
a. Describe how computer-controlled automation affects a workplace and society.	n/a
b. Explain the ramifications of society's dependence on computers.	n/a
c. Use 21st Century Skills to understand and address global issues	n/a
d. Identify advantages and disadvantages of changing workplace environments.	n/a
e. Be aware of changing tools in technology and adapt to a changing environment	n/a
Objective 3: Demonstrate knowledge of the right to privacy.	

a. Explain how computers can compromise privacy.	Chapter 1, Lesson 5
b. Exhibit knowledge of privacy laws.	Chapter 1, Lesson 5
c. Describe responsibilities of people who control computer information.	Chapter 1, Lesson 5
Objective 4: Demonstrate knowledge of computer, information and software security.	
a. Exhibit knowledge of copyright laws.	Chapter 1, Lesson 5
b. Explain how computers could erroneously be used to compromise copyright laws.	Chapter 1, Lesson 5
c. Give examples of ways to protect information on computer systems.	Chapter 1, Lesson 5
d. Identify ways to protect against computer viruses.	Chapter 1, Lesson 5

STANDARD 6	**CITATION(S)**
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	Supplemental Lesson 3
b. Identify technical/programming talents	Supplemental Lesson 3
c. Identify organizational and leadership skills	Supplemental Lesson 3
d. Explore aptitude for innovation	Supplemental Lesson 3
e. Determine aptitude for working as a member of a Computer Programming/Software Engineering team	Supplemental Lesson 2 (Group project) Supplemental Lesson 3
Objective 2: 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, senior developer, junior developer, and client/subject matter expert	Supplemental Lesson 3
b. Describe work performed by each member of the computer programming/software engineering team	Supplemental Lesson 3
c. Investigate trends associated with computer programming/software engineering careers	Supplemental Lesson 3

d. Discuss related career pathways.	Supplemental Lesson 3
e. Compile a portfolio of the individual and group programs developed during the course	Student projects in every chapter are stored and submitted in an organized fashion.
Objective 3: Discuss relevant history of software development	
a. Discuss relevant history of computer technology	Chapter 1, Lesson 1
b. Identify key points in the history of the computer programming/software engineering industry	Chapter 1, Lesson 4 Chapter 12, Lesson 2

(Semester 2 Standards*)

STANDARD 7	**CITATION(S)**
Students will employ arrays.	
Objective 1: Demonstrate the ability to use arrays in programs.	
a. Declare arrays of all applicable types.	Chapter 11, Lesson 1
b. Initialize arrays.	Chapter 11, Lesson 1
c. Perform data input to and output from arrays.	Chapter 11, Lesson 1
d. Perform operations on arrays including sequential searches	Chapter 11, Lesson 1
e. Iterate through the structure (i.e. foreach loop)	Chapter 11, Lesson 1 Chapter 11, Lesson 3
Objective 2: Demonstrate the ability to use dynamic arrays (i.e. vectors, arraylists, or generic lists)	**Note: Linked Lists**
a. Declare a dynamic array	Chapter 11, Lesson 2
b. Add and remove items from the array	Chapter 11, Lesson 2
c. Output data from arrays.	Chapter 11, Lesson 2
d. Perform operations on arrays.	Chapter 11, Lesson 2
e. Iterate through the loop (i.e. foreach loop)	Chapter 11, Lesson 3
Objective 3: Demonstrate the ability to use strings in programs.	
a. Compare string identifiers.	Chapter 4, Lesson 4
b. Find the length of a string.	Chapter 8, Lesson 1
c. Copy part or all of string identifiers into other strings.	Chapter 8, Lesson 1
d. Concatenate string identifiers.	Chapter 4, Lesson 4
e. Locate substring positions	Chapter 8, Lesson 1
f. Insert strings into other strings.	Chapter 8, Lesson 1 Chapter 8, Lesson 2

STANDARD 8	CITATION(S)
Students will properly employ object-oriented programming techniques.	
Objective 1: Demonstrate the ability to use classes.	
a. Instantiate objects.	Chapter 13, Lesson 1
b. Use object data members.	Chapter 13, Lesson 2
c. Use object member functions (methods).	Chapter 13, Lesson 2
Objective 2: Demonstrate the ability to create user-defined classes.	
a. Create and use data members.	Chapter 13, Lesson 2
b. Create a constructor to initialize the data members.	Chapter 13, Lesson 4
c. Create and use instance functions (methods).	Chapter 9
Objective 3: Demonstrate proper design principles with classes	
a. Create classes that are well encapsulated (data members private).	Chapter 13, Lesson 3
b. Properly use modifiers and accessors (getters and setters).	Chapter 13, Lesson 3
c. Understand appropriate private and public modifiers according to program design.	Chapter 13, Lesson 3

STANDARD 9	CITATION(S)
Students will properly use sequential files.	
Objective 1: Demonstrate the ability to use sequential files in programs.	
a. Create and initialize sequential files.	n/a
b. Store data to sequential files.	n/a
c. Retrieve data from sequential files.	n/a
d. Update sequential files.	n/a

STANDARD 10	CITATION(S)
Students will apply appropriate programming skill as an effective member of a team demonstrating the ability to collaborate with others (www.p21.org).	
Objective 1: Demonstrate the ability to apply knowledge to a programming project.	
a. Formalize specifications.	Supplemental Lesson 2 / Activity 2
b. Choose proper input parameters.	Supplemental Lesson 2 / Activity 2
c. Choose appropriate data structures and processing.	Supplemental Lesson 2 / Activity 2
d. Design appropriate output.	Supplemental Lesson 2 / Activity 2
e. Use appropriate test data.	Supplemental Lesson 2 / Activity 2

f. Write good documentation.	Supplemental Lesson 2 / Activity 2
Objective 2: Demonstrate the ability to use teamwork and collaboration in a programming project.	
a. Divide a project among programmers.	Supplemental Lesson 2 / Activity 2
b. Present work to a group.	Supplemental Lesson 2 / Activity 2
c. Coordinate work with others in the group.	Supplemental Lesson 2 / Activity 2
d. Complete assigned work according to predetermined deadlines.	Supplemental Lesson 2 / Activity 2 (all coding projects have classroom due dates)
e. Participate in a peer performance evaluation.	Supplemental Lesson 2 / Activity 2
f. Demonstrate professionalism in team relationships, communication, timeliness, and attitude.	Supplemental Lesson 2 / Activity 2