CompuScholar, Inc.

Alignment to Ohio **"145060 - Programming"** Course Standards

Ohio Course Details:

Course Title:	Information Technology
Course Code(s):	145060 - Programming
Credit:	1
Grade Level:	9th-12th
State Standards Link:	http://education.ohio.gov/Topics/Career-Tech/Information-Technology-Career-Field

CompuScholar Course Details:

Course Title:	Windows Programming with C#
Course ISBN:	978-0-9887070-0-9
Course Year:	2019

Note 1: Citation(s) listed may represent a subset of the instances where objectives are met throughout the course.

Note 2: Citation(s) for a "Lesson" refer to the "Lesson Text" elements and associated "Activities" within the course, unless otherwise noted. The "Instructional Video" components are supplements designed to introduce or re-enforce the main lesson concepts, and the Lesson Text contains full details.

Course Description

In this course, students will learn the basics of building simple interactive applications. Students will learn the basic units of logic: sequence, selection, and loop. Students will apply algorithmic solutions to problem- domain scenarios. Students will gain experience in using commercial and open source languages, programs, and applications.

Course Standards

Strand 2 - IT Fundamentals	
Learners apply fundamental principles of IT, including the history of IT and its impact on society, common industry terms, systems theory, information storage and retrieval, database management, and computer hardware, software, and peripheral device configuration and installation. This base of knowledge and skills may be applied across the career field.	
2.3. Data Encoding: Explain and describe data encoding basics.	**CITATION(S)**
2.3.1. Identify and explain coding information and representation of characters (e.g., American Standard Code for Information Interchange [ASCII], Extended Binary Coded Decimal Interchange Code [EBCDIC],	Chapter 4, Lesson 4 Chapter 8 Activity
2.3.2. Convert between numbering systems (e.g., binary, hexadecimal, decimal).	Chapter 4, Lesson 5

2.9. Project Concept Proposal: Develop a project concept proposal.	**CITATION(S)**
2.9.1. Identify and incorporate branding strategies.	N/A
2.9.2. Determine the scope and purpose of the project.	Suppl. Chapter 2, Lesson 1
2.9.3. Determine the target audience, client needs, expected outcomes, objectives, and budget.	Suppl. Chapter 2, Lesson 1
2.9.4. Develop a conceptual model and design brief for the project.	Suppl. Chapter 2, Lesson 1
2.9.5. Develop a timeline, communication plan, task breakdown, costs (e.g., equipment, labor), deliverables, and responsibilities for completion.	Suppl. Chapter 2, Lesson 1
2.9.6. Develop and present a comprehensive proposal to stakeholders.	Suppl. Chapter 2, Lesson 1
2.11. Troubleshooting: Select and apply troubleshooting methodologies for problem solving.	**CITATION(S)**
2.11.1. Identify the problem.	Chapter 10, Lesson 2-4
2.11.2. Select troubleshooting methodology (e.g., top down, bottom up, follow the path, spot the differences).	Chapter 10, Lesson 4
2.11.3. Investigate symptoms based on the selected methodology.	Chapter 10, Lesson 4
2.11.4. Gather and analyze data about the problem.	Chapter 10, Lesson 4 Suppl. Chapter 1, Lesson 6
2.11.5. Design a solution.	Chapter 10 Activtiy Suppl. Chapter 1, Lesson 6
2.11.6. Test a solution.	Chapter 10 Activtiy Suppl. Chapter 1, Lesson 6
2.11.7. Implement a solution.	Chapter 10 Activtiy Suppl. Chapter 1, Lesson 6
2.11.8. Document the problem and the verified solution.	Suppl. Chapter 1, Lesson 6
2.12. Performance Tests and Acceptance Plans: Develop performance tests and acceptance plans.	**CITATION(S)**
2.12.1. Create a written procedure agreed by the stakeholders and project team for determining the acceptability of the project deliverables.	Suppl. Chapter 2, Lesson 1
2.12.2. Develop a test system that accurately mimics external interfaces.	Suppl. Chapter 1, Lesson 6
2.12.3. Develop test cases that are realistic, compare with expected performance, and include targeted platforms and device types.	Chapter 10, Lesson 4
2.12.4. Develop, perform, and document usability and testing integration.	N/A
2.12.5. Make corrections indicated by test results.	Chapter 10, Lesson 4

2.12.6. Seek stakeholder acceptance upon successful completion of the test plan.	N/A
2.13. Rollout and Handoff: Plan rollout and facilitate handoff to customer.	**CITATION(S)**
2.13.1. Include overall project goals and timelines in the rollout plan.	Suppl. Chapter 2, Lesson 1
2.13.2. Communicate rollout plans to key stakeholders in a timely manner.	Suppl. Chapter 2, Lesson 1
2.13.3. Conduct final review and approvals according to company standards.	N/A
2.13.4. Identify support staff, training needs, and contingency plans in the rollout plan.	N/A
2.13.5. Test delivered application to assure that it is fully functional for the customer or user and meets all requirements.	Suppl. Chapter 2, Lesson 1
2.13.6. Deliver support and training materials.	N/A
Strand 5 - Programming and Software System	
Learners apply principles of computer programming and software development to develop code; build, test, and debug programs; create finished products; and plan, analyze, design, develop, implement, and support software applications.	
5.1. Programming Concepts: Describe programming concepts.	**CITATION(S)**
5.1.1. Describe how computer programs and scripts can be used to solve problems (e.g., desktop, mobile, enterprise).	Chapter 7, Lesson 3
5.1.2. Explain how algorithms and data structures are used in information processing.	Chapter 7, Lesson 3 Chapter 11 Suppl. Chapter 1, Lesson 6
5.1.3. Model the solution using both graphic tools (e.g., flowcharts) and pseudocode techniques.	Chapter 7, Lesson 3
5.1.4. Describe, compare, and contrast the basics of procedural, structured, object-oriented (OO), and event-driven programming.	Chapter 2, Lesson 3 Chapter 3, Lesson 3
5.1.5. Describe the concepts of data management through programming languages.	Chapter 4, Lessons 1 - 5 Chapter 11, Lessons 1-2
5.1.6. Analyze the strengths and weaknesses of different languages for solving a specific problem.	Chapter 1, Lesson 4
5.1.7. Compare and contrast the functions and operations of compilers and interpreters.	Chapter 1, Lesson 4 Chapter 2, Lesson 1
5.1.8. Describe version control and the relevance of documentation.	Suppl. Chapter 2, Lesson 1
5.2. Computational and String Operations: Develop code that performs computational and string operations.	**CITATION(S)**
5.2.1. Compare and contrast primitive types of numeric and nonnumeric data (e.g., integers, floats, Boolean, strings).	Chapter 4, Lessons 1 - 4

5.2.2. Identify the scope of data (e.g., global versus local, variables, constants, arrays).	Chapter 4, Lesson 2
5.2.3. Write code that uses arithmetic operations.	Chapter 7, Lesson 1
5.2.4. Write code that uses subtotals and final totals.	Chapter 8, Lesson 1 - 3 Suppl. Chapter 1, Lesson 6
5.2.5. Write code that applies string operations (e.g., concatenation, pattern matching, substring).	Chapter 8, Lesson 1 - 3
5.3. Logical Operations and Control Structures: Develop code that uses logical operations and control structures.	**CITATION(S)**
5.3.1. Explain Boolean logic.	Chapter 5, Lesson 1
5.3.2. Solve a truth table.	N/A
5.3.3. Write code that uses logical operators (e.g., and, or, not).	Chapter 5, Lesson 1
5.3.4. Write code that uses relational operators and compound conditions.	Chapter 5 and subsequent activities
5.3.5. Write code that uses conditional control structures (e.g. if, if-then-else).	Chapter 5 and subsequent activities
5.3.6. Write code that uses repetition control structures (e.g., while, for).	Chapter 5 and subsequent activities
5.3.7. Write code that uses selection control structures (e.g., case, switch).	N/A
5.3.8. Write code that uses nested structures and recursion.	Chapter 14, Lessons 2 - 3
5.3.9. Write code that creates and calls functions.	Chapter 9
5.3.10. Code error-handling techniques.	Chapter 10, Lesson 3
5.3.11. Write code to access data repositories.	Suppl. Chapter 1, Lesson 6
5.3.12. Write code to create classes, objects, and methods.	Chapters 13, 15, 16
5.4. Integrated Development Environment: Build and test a program using an integrated development environment (IDE).	**CITATION(S)**
5.4.1. Configure options, preferences, and tools.	Chapter 2, Lesson 2
5.4.2. Write and edit code in the IDE.	Chapter 2 and all subsequent activities
5.4.3. Compile or interpret a working program.	Chapter 2 and all subsequent activities
5.4.4. Define test cases.	Chapter 10, Lesson 4 Suppl. Chapter 1, Lesson 6

5.4.5. Test the program using defined test cases.	Chapter 10 Activity Suppl. Chapter 1, Lesson 6
5.4.6. Correct syntax and runtime errors.	Chapter 10 and throughout the course
5.4.7. Debug logic errors.	Chapter 10 and throughout the course
5.5. Programming Conventions: Develop programs using applications security practices.	**CITATION(S)**
5.5.1. Develop programs using data validation techniques.	Suppl. Chapter 1, Lesson 6
5.5.2. Develop programs that use reuse libraries.	Chapter 7, Lesson 2
5.5.3. Develop programs using operating system calls.	.NET Framework used throughout the course
5.5.4. Develop programs that call other programs.	N/A
5.5.5. Use appropriate naming conventions and apply comments.	Chapter 2, Lesson 3
5.5.6. Format output (e.g., desktop, mobile, enterprise, reports, data files).	Chapter 8, Lesson 2
5.6. Software Development Lifecycle: Apply the software development lifecycle (SDLC).	**CITATION(S)**
5.6.1. Determine requirements specification documentation.	Suppl. Chapter 2, Lesson 1
5.6.2. Identify constraints and system processing requirements.	Suppl. Chapter 1, Lesson 6
5.6.3. Develop and adhere to timelines.	Suppl. Chapter 2, Lesson 1
5.6.4. Identify a programming language, framework, and an integrated development environment (IDE).	Suppl. Chapter 2, Lesson 1
5.6.5. Identify input and output (I/O) requirements.	Suppl. Chapter 1, Lesson 6 Suppl. Chapter 2, Lesson 1
5.6.6. Design system inputs, outputs, and processes.	Suppl. Chapter 1, Lesson 6
5.6.7. Document a design using the appropriate tools (e.g., program flowchart, dataflow diagrams, Unified Modeling Language [UML]).	Chapter 7, Lesson 3 Suppl. Chapter 2, Lesson 1
5.6.8. Create documentation (e.g., implementation plan, contingency plan, data dictionary, user help).	Suppl. Chapter 1, Lesson 6
5.6.9. Review the design (e.g., peer walkthrough).	Suppl. Chapter 1, Lesson 6
5.6.10. Present system design to stakeholders.	Suppl. Chapter 1, Lesson 6
5.6.11. Develop the application.	Suppl. Chapter 1, Lesson 6

5.6.12. Compare and contrast software methodologies (e.g., agile, waterfall).	Suppl. Chapter 2, Lesson 1
5.6.13. Perform code reviews (e.g., peer walkthrough, static analysis).	Chapter 10, Lesson 4 Suppl. Chapter 1, Lesson 6
5.6.14. Ensure code quality by testing and debugging the application (e.g., system testing, user acceptance testing).	Chapter 10, Lesson 4 Suppl. Chapter 1, Lesson 6
5.6.15. Train stakeholders.	N/A
5.6.16. Deploy the application.	N/A
5.6.17. Collect application feedback and maintain the application.	N/A
5.7. Configuration Management: Describe configuration management activities.	**CITATION(S)**
5.7.1. Explain version management and interface control.	Suppl. Chapter 2, Lesson 1
5.7.2. Explain baseline and software lifecycle phases.	Suppl. Chapter 2, Lesson 1
5.7.3. Analyze the impact of changes.	N/A

Strand 6 - Web Development	
Learners apply principles of design and technology, including programming standards and protocols, to create, test, host, and maintain webpages and websites with text, graphics, multimedia, scripting, linking, and data integration in a structure that is easy to navigate and accessible for all users via a variety of hardware and software platforms.	
6.3. Scripting: Integrate scripting into a webpage.	**CITATION(S)**
6.3.1. Select and apply scripting languages used in web development.	See our Web Design course