Teaching Tomorrow's Technology.
CompuScholar

Windows Programming C#

Course Syllabus and Planner

Course Overview

The CompuScholar **Windows Programming C#** curriculum is a one or two-semester course covering topics typically found in **Computer Science I** or similar courses. This course has been aligned to specific course standards in in many states. Please visit our course description page for a video tour and alignment information.

https://www.compuscholar.com/schools/courses/intro-csharp/

Other introductory programming courses are not required; students merely need to have typical computer usage skills prior to starting this course.

Teaching Strategies

The course material is designed to appeal to a variety of students, from traditional learners who thrive on written text to audio-visual students who enjoy a multi-media format. All content is delivered through an online system that allows students to work seamlessly both in the classroom and at home.

Labs and Programming Environment

Every chapter contains one or more hands-on programming labs where students will design or implement programs to demonstrate understanding of the lesson topics. Students will get the opportunity to work on individual and group projects and will experience all phases of a project lifecycle, including requirements, design, implementation, and testing.

The primary language and programming environments are Microsoft C# and the free Visual Studio Community Edition IDE. The course contains detailed download, installation, and usage instructions.

Course Planner

When used as a single-semester course, students will move at a fast pace to complete most core lessons in approximately 90 days (but skipping some chapters marked below), assuming daily class-times. Each "day" listed below represents one typical class period of 45 – 60 minutes, so students will typically work 3-5 hours per week. Schools may choose to offer this class as a relaxed two-semester or 180 day course by meeting less frequently (e.g. 2-3 times a week), moving at a slower pace or including the all chapters, the team project and numerous Supplemental Lessons. The suggested number of days factors in 1 day per lesson & quiz, 1 day per lab, and 1 day per test.

Each chapter contains multiple lessons, quizzes and a chapter test in addition to the listed Lab assignments. Teachers may utilize Supplemental Lessons as desired to meet state standards or student interest. Team Projects and other assignments may be adjusted to fit the available time.

Days	Reading and Objectives	Labs
8	Chapter One: Understanding Your Computer • Computers Past and Present • Survey of Computer Hardware • Introduction to Computer software • Common Programming Languages • Computer Ethics and Security (Classes may skip this chapter if using a 1-semester schedule, but will complete the lab.)	**Establish Development Environment** - Install Microsoft Visual Studio, create working directory, and practice submitting projects through the online interface. Class discussion and review of a sample EULA terms and conditions.
6	Chapter Two: Fundamentals of C# • Introduction to C# • Your Integrated Development Environment • Console Programs • Getting Console Input	**Echo, Echo** – The student's first C# program will use the console to ask the user for a name and print a message as output.

Days	Reading and Objectives	Labs
6	Chapter Three: Windows Programming Concepts • Your First GUI Program • Common Windows Elements • Event-Driven Programming • Namespaces	**A More Personal Hello** – The student will create an event-driven GUI program to display their name in a message box in response to a button click.
7	Chapter Four: Data Types and Variables • Value Data Types • Variables • Reference Data Types • Introducing Strings • The Binary Number System	**Experiment with Data Types** – The student will demonstrate declaring, initializing, and printing variables of different data types.
6	Chapter Five: Basic Flow Control • Logical Expressions • Using the "if" Statement • For Loops • While Loops	**Jeepers, Beepers** – The student will create loops of different types to produce a specific number of beeps or pop-ups entered by the user.
5	Chapter Six: User Input • Text Boxes • List Boxes and Combo Boxes • Radio Buttons and Check Boxes	**Telling Tall Tales** – The student will create a "mad-lib" style program to demonstrate a variety of user input controls.

Days	Reading and Objectives	Labs
6	Chapter Seven: Math Functions in C# • Math Operators (+, -, *, /, and %) • .NET Framework Math Functions • Common Algorithms • A Simple Calculator	**Algorithms Practice** – The student will flowchart and implement two simple algorithms. **Divide and Multiply** – The student will add multiplication and division buttons to the calculator created in the last lesson.
5	Chapter Eight: Working with Strings • Common String Operations • Formatting Strings • Converting Between Strings and Numbers	**Caesar's Cipher** – The student will write a program that translates a message to a cipher code and then back to plain form again.
5	Chapter Nine: Methods • Writing and Calling Methods • Method Parameters and Return Values • Calling Methods	**What's Your Birthday?** – The student will write a function to translate a date into a day of the week.
6	Chapter Ten: Debugging and Exceptions • The Visual Studio Debugger • Debugging Demonstration • C# Runtime Exceptions • Finding Runtime Errors	**Divide by Zero** – In this lab the student will identify and resolve a hidden error left in an earlier program.

Days	Reading and Objectives	Labs
5	Chapter Eleven: Collections • Arrays • Linked Lists • Enumerations and ForEach	**Your ToDo List** – The student will create a program that allows users to add and remove text items from a linked list.
5	Chapter Twelve: Object-Oriented Programming • Object-Oriented Concepts • History of OOP • Designing an Object	**Creating Songs** – The students will design on paper Song and Note classes and assemble Songs from multiple Notes.
7	Chapter Thirteen: Classes in C# • Defining a Class • Properties and Methods • Public vs. Private • Constructors • Static Members	**Your Song Player** – The student will implement a Song class to play music defined in the previous chapter.
5	Chapter Fourteen: Sorting and Recursion • Simple Sorting • Recursion • Recursive Sorting	**The Number Sort** – The student will write an Insertion Sort function to sort numbers in a list.

Days	Reading and Objectives	Labs
5	Chapter Fifteen: Vector Graphics • Screen Coordinates • Drawing Lines • Drawing Circles and Rectangles (Classes may skip this chapter if using a 1-semester schedule.)	**Graphing** – The student will create simple line graphs from input data sets.
7	Chapter Sixteen: Inheritance and Polymorphism • Base Classes and Derived Classes • Using References to Base and Derived Classes • Virtual Base Methods • The "Object" Base Class • Using Base Features from Derived Classes	**Creating the Chess Pieces** – The student will create a small object hierarchy of standard chess pieces in preparation for the final project.
8	Chapter Seventeen: Final Project For the final project the student will complete a Chess game. The student will create the abstract hierarchy of pieces (AbstractChessPiece, Pawn, Knight, Rook, etc.) and write other logic to complete the game. The project consists of 5 guided lab steps. Each guided step contains a checkpoint for testing to ensure code meets the requirements at each step.	**Starting Your Chess Project** – Ensure the student can build the starter project. **Initializing the Game()** – Write logic to initialize the game and set pieces in the starting position. **Finishing handleClick()** – Write game logic to allow selection and de-selection of game pieces. **Moving Pawns** – Write game logic to control Pawn movement. **Moving Other Pieces and Testing for Check** – Write game logic to control movement and capture of other pieces. Also complete logic to test for check.

Days	Reading and Objectives	Labs
As desired	Chapter Eighteen: Team Project In this activity, students will form small groups or work individually on unique projects by applying a traditional software development lifecycle.	**Requirements** **Design** **Implementation** **Testing**
As desired	Supplemental Lessons, make-up work, review Teachers can assign any mixture of Supplemental Lessons and labs as required by state standards or based on class interest. Three full chapters of supplemental material are available, each with either a lab or a quiz in addition to the lesson text. **Suppl. Chapter 1: Enrichment Topics (8 lessons)** **Suppl. Chapter 2: Software and Industry (4 lessons)** **Suppl. Chapter 3: Computers and Modern Society (4 lessons)**	