IB Computer Science SL Resources Syllabus

High School (100 Contact Hours of Content)

Course Overview and Goals

IB Computer Science SL Resources is a collection of CodeHS units and lessons that align to the IB Computer Science guide for first assessment in 2027. It gathers existing CodeHS content across computational thinking, Python programming, algorithms, computer systems, networks, databases, and artificial intelligence so that teachers can locate and assign material that maps to specific IB topics. Students who work through this content learn to design and trace algorithms, write and debug Python programs, work with files and object-oriented design, and explain how computer systems, networks, cryptography, databases, and machine learning models work. The collection covers subject content only.

How to Use This Collection

This is a resource library, not a paced course. The units are sequenced so that prerequisites always come before the lessons that depend on them, but they are not scheduled against the IB two-year timeline. A teacher copying this content into their own course should plan when each topic is taught, which lessons to assign in full, and which to pull from selectively. There is no content in this collection that prepares students for IB assessments — there is no Internal Assessment support, no exam-style practice, and no paper-specific preparation. Everything here exists to give teachers CodeHS material that aligns to the IB subject content.

Development Environment

Students write and run Python programs and SQL queries in the browser using the CodeHS editor.

More Information

Browse the content of this course at https://codehs.com/course/27487/explore

Course Breakdown

Module 1: Computational Thinking (2 hours)

Students learn what algorithms are, how the same process can be represented in natural language, flowcharts, pseudocode, and code, and how to move between those representations.

Topics Covered	• Algorithms in Computing • Algorithm Representations • Flowcharts • Pseudocode • Decomposition • Testing Instructions
Example Assignments	• **Algorithms in Life and Technology** ○ Students describe a real-world algorithm they use and analyze how the same steps appear in technology.

	• **Real-World Flowchart**
	  ○ Students build a flowchart for a multi-factor decision scenario.
	• **Flow Chart to Pseudocode**
	  ○ Students convert a given flowchart into pseudocode without worrying about syntax.
	• **Pseudocode to Program**
	  ○ Students convert their pseudocode into a working Python program and reflect on whether drafting first helped.

Module 2: Basic Python and Console Interaction (5 hours)

Students write their first Python programs, working with printing, variables, user input, and operators, and learn how programming languages are executed.

Objectives / Topics Covered	• Printing in Python • Variables and Types • User Input • Mathematical Operators • String Operators • Comments • Compiled vs. Interpreted Languages

Example Assignments / Labs	• **Age**
	  ○ Students ask a user for their age and print how many birthday candles they need.
	• **Rectangle, Part 3**
	  ○ Students rewrite a rectangle program so the user supplies the length and width.
	• **Recipe**
	  ○ Students build a salad recipe scaler that takes ingredient names and amounts and prints quantities for a given number of servings.
	• **Compiled vs. Interpreted Languages**
	  ○ Students compare how compiled and interpreted languages execute and give examples of each.

Module 3: Conditionals (4 hours)

Students use Booleans, comparison operators, and logical operators to write programs that make decisions, and learn why floating point values require careful comparison.

Objectives / Topics Covered	• Booleans • If Statements • Comparison Operators • Logical Operators • Floating Point Numbers and Rounding
Example Assignments / Labs	• **Positive, Zero, or Negative?** ○ Students use an if-elif-else statement to classify a number entered by the user. • **Presidential Eligibility**

	○ Students combine age, citizenship, and residency conditions with logical operators to report eligibility. ● **Table Reservation** ○ Students compare a user-entered name against a stored reservation name. ● **Transaction** ○ Students automate a bank deposit and withdrawal system that validates the customer's request before processing it.

Module 4: Looping (3 hours)

Students write while loops and for loops, control loop flow with break and continue, and nest control structures to solve more complex problems.

Objectives / Topics Covered	● While Loops ● For Loops ● Break and Continue ● Nested Control Structures
Example Assignments / Labs	● **Average Test Score** ○ Students collect three scores from the user and report the average. ● **Higher / Lower** ○ Students build a guessing game that tells the user whether each guess is too high or too low. ● **Rolling Dice** ○ Students use nested loops to print every combination of two dice. ● **Categories** ○ Students use nested loops to collect three items in each of three user-chosen categories.

Module 5: Functions and Exceptions (4 hours)

Students write functions with parameters and return values, reason about scope and namespaces, and handle runtime errors with exceptions.

Objectives / Topics Covered	● Functions ● Parameters and Default Parameters ● Namespaces and Scope ● Return Values ● Exceptions
Example Assignments / Labs	● **Area of a Square with Default Parameters** ○ Students write a function with a default parameter to calculate the area of a square. ● **Add, Subtract, or Multiply** ○ Students write functions for three operations and call the one the user selects. ● **Temperature Converter, Part 2** ○ Students extend a conversion function to handle both Celsius to Fahrenheit and Fahrenheit to Celsius from user input. ● **Enter a Positive Number**

	○ Students write a function that repeatedly prompts for input and uses exception handling to reject invalid entries.

Module 6: Strings (5 hours)

Students index, slice, and traverse strings, work around string immutability, and use built-in string methods to search and transform text.

Objectives / Topics Covered	• Indexing • Slicing • Immutability • Strings and For Loops • The in Keyword • String Methods
Example Assignments / Labs	• **Part 2, Replace a Letter** ○ Students write a function that returns a copy of a string with the character at a given index replaced. • **Spelling Bee** ○ Students use a for loop to print a word one letter at a time. • **Keeping Count** ○ Students count how many times a single character appears in a string. • **Part 2, Remove All From String** ○ Students take two strings from the user and remove every instance of the second from the first.

Module 7: Creating and Altering Data Structures (4 hours)

Students store collections of data in tuples and lists, traverse them with loops, and modify list contents using built-in methods.

Objectives / Topics Covered	• Tuples • Lists • For Loops and Lists • List Methods
Example Assignments / Labs	• **Coordinate Pairs** ○ Students complete a distance function that takes two coordinate tuples. • **Max In List** ○ Students write a function that returns the largest integer in a list. • **Librarian, Part 2** ○ Students collect author last names from the user and print them in sorted order. • **Word Ladder** ○ Students build a list of words where each word differs from the previous one by a single letter.

Module 8: Extending Data Structures (7 hours)

Students work with 2D lists, list comprehensions, dictionaries, and the stack and queue abstractions, applying each to a realistic scenario.

Objectives / Topics Covered	• 2D Lists • List Comprehensions • Packing and Unpacking • Dictionaries • Stacks (Last-In, First-Out) • Queues (First-In, First-Out)

Example Assignments / Labs	• **Checkerboard, v3**
	  ○ Students build a 2D grid that alternates values in a checkerboard pattern.
	• **Tic Tac Toe**
	  ○ Students complete a working tic tac toe simulation built on a 2D list.
	• **Phone Book**
	  ○ Students build a dictionary of names and phone numbers and look up entries by name.
	• **Textile Loom**
	  ○ Students implement a stack to manage threads added to and unraveled from a loom.
	• **Movie Night**
	  ○ Students implement a queue so student movie suggestions are watched in the order submitted.

Module 9: Algorithms (6 hours)

Students analyze how long algorithms take to run, express efficiency in Big O notation, and implement and compare the standard search and sort algorithms.

Objectives / Topics Covered	• Informal Runtime Analysis • Statement Execution Counts • Big O Notation • Linear Search • Binary Search • Bubble Sort • Selection Sort

Example Assignments / Labs	• **Evaluating Algorithms with Statement Execution Counts**
	  ○ Students use trace tables to count executions and compare the efficiency of competing algorithms.
	• **Draft Day Search**
	  ○ Students build a fantasy-football draft tool that searches an available-players list.
	• **Smart Closet**
	  ○ Students implement binary search to find clothing items by ID and return style recommendations.
	• **Sort Your Playlist**
	  ○ Students sort song titles and reason about how Python compares strings character by character.

Module 10: File I/O (5 hours)

Students read data from files character by character, line by line, and all at once, write and append to files, and control the file pointer position.

Objectives / Topics Covered	• What is File I/O • Reading Characters from a File • Reading Lines from a File • Writing to a File • Moving the File Pointer • TXT vs. CSV Formats

Example Assignments / Labs	• **Choosing a File Format: TXT vs. CSV**
	  ○ Students justify when each format is the better choice with specific examples.
	• **Finding the Needle in the Haystack**
	  ○ Students read a word list with readlines() and search it for a target word.
	• **Guest Book**
	  ○ Students append a user's name and message to a running guest book file.
	• **Reverse a File**
	  ○ Students read a story from a file and write it back out with the character order reversed.

Module 11: Classes and Objects (10 hours)

Students design their own classes, add methods and built-in methods, overload operators, apply inheritance and encapsulation, and document designs with UML class diagrams.

Objectives / Topics Covered	• Classes and Objects • Methods • Built-In Methods • Operator Overloading • Class vs. Instance Variables • Inheritance • Hidden Attributes • Namespaces and Modules • UML Class Diagrams

Example Assignments / Labs	• **The Rectangle Class, Parts 1-9**
	  ○ Students build one class across nine steps, adding a constructor, methods, string and equality methods, operator overloading, and class variables.
	• **Food, Parts 1-3**
	  ○ Students build a three-level inheritance hierarchy and test instances against each class.
	• **Names In a Hat**
	  ○ Students write a NameHat class that inserts, draws, and reports on names.
	• **Designing Classes with UML**

	○ Students produce a UML class diagram for a design and submit it for review.

Module 12: Computer Fundamentals (10 hours)

Students examine the internal components and peripherals of a computer, how data is represented in binary and hexadecimal, how images and files are compressed, and what operating systems and distributed systems do.

Objectives / Topics Covered	● Internal Components ● Peripheral Devices ● Storage Options ● Cloud Computing ● Binary and Hexadecimal ● Encoding Images in Binary ● Data Compression ● Operating Systems ● Sequential, Parallel, and Distributed Computing ● Impact of the Internet
Example Assignments / Labs	● **Lab: Computer Disassembly** ○ Students safely disassemble a computer and document each internal component they find. ● **Storage Suggestions** ○ Students build a comparison of the major storage options and recommend one for a given need. ● **Project: Cloud Computing Roles** ○ Students create a one-pager summarizing the customer, provider, and partner roles in cloud computing. ● **Create a Color Pixel Image** ○ Students use a bit interpreter to build a color image and explain the RGB encoding behind it. ● **Decreasing Resolution** ○ Students lower an image's resolution and analyze the trade-off between file size and quality.

Module 13: Logic Gates (4 hours)

Students work with the seven logic gates, move between truth tables, Boolean expressions, and logic diagrams, and simplify circuits using Boolean algebra and Karnaugh maps.

Objectives / Topics Covered	● Logic Gates ● Boolean Operators ● Truth Tables ● Boolean Expressions ● Logic Diagrams ● Combinational Circuits ● Boolean Algebra ● Karnaugh Maps
Example Assignments / Labs	● **Build the Seven Gates**

	○ Students implement each of the seven logic gates as a function that returns the correct output for every input. ● **Design the Alarm** ○ Students translate a description of a system that has to work into a Boolean expression. ● **Build a Full Adder** ○ Students extend a half adder into a full adder that accepts an incoming carry. ● **Simplify, Verify, Redraw** ○ Students simplify an expression using both Boolean algebra and a Karnaugh map and confirm both approaches agree.

Module 14: Networks (11 hours)

Students study how the internet is built and operated, from network devices and protocols through IP addressing, routing, network design, and network security and management.

Objectives / Topics Covered	● The Internet ● Network Types and Options ● Network Devices ● Protocols and Standards ● OSI Model ● IP Addresses ● Routing and Packets ● Network Design and Subnetting ● Private Networks ● Network Communication and Management
Example Assignments / Labs	● **Wireless Network Setup** ○ Students diagram a wireless network for a fictitious house or office. ● **Troubleshooting with the OSI Model** ○ Students act as a network engineer for a small business and isolate faults by OSI layer. ● **Create a Subnet Network** ○ Students design a subnetted network for a school with multiple locations. ● **Coffee Shop: Design an Intranet** ○ Students plan an intranet for a local coffee shop, choosing devices and access rules. ● **Packet Analysis Response** ○ Students identify anomalies in captured traffic and explain what each could represent.

Module 15: Project: The Inside Scoop on LANs (2 hours)

Students design a magazine that explains local area network methodologies, topologies, Ethernet standards, and cabling for a general audience.

Objectives / Topics Covered	● LAN Methodologies ● Network Topologies ● Perimeter Networks (DMZ)

	• Ethernet Standards • LAN Cabling • Technical Communication

Example Assignments / Labs	• **Advice Column: LAN Methodologies**
	  ○ Students explain three common LAN methodologies and diagram how each works.
	• **Product Review: Network Topologies**
	  ○ Students compare three to four LAN topologies and evaluate the trade-offs of each.
	• **Advertisement: Perimeter Network (DMZ)**
	  ○ Students make the case for a DMZ and explain its security benefits.
	• **Your Choice: LAN Cabling**
	  ○ Students describe the three cabling types used in LANs in a format of their choosing.

Module 16: Cryptography (4 hours)

Students trace cryptography from classical ciphers to modern public-key systems, and examine hashing, digital certificates, and authentication methods.

Objectives / Topics Covered	• History of Cryptography • Symmetric Encryption • Substitution and Transposition Ciphers • Asymmetric Encryption • Diffie-Hellman Key Exchange • Digital Certificates • Hashing • Authentication Methods

Example Assignments / Labs	• **Enhanced Caesar Cipher**
	  ○ Students strengthen a Caesar cipher by adding a transposition step.
	• **Rail Fence Cipher**
	  ○ Students encrypt and decrypt messages using a transposition cipher built on rows and columns.
	• **Public-Key Encryption Explorer**
	  ○ Students work with a partner through an interactive public-key exchange and compare symmetric and asymmetric approaches.
	• **Digital Certificates Response**
	  ○ Students explain how to identify a site using a certificate and what the certificate guarantees.

Module 17: Databases (7 hours)

Students query relational databases with SQL, filter, order, rename, and join results, and design a database schema for a business scenario.

Objectives / Topics Covered	• Databases and Cybersecurity • Getting Started in SQL • Basic Querying • Filtering Queries

	• Advanced Filters • Ordering Results • Renaming Fields • Joining Tables • Database Design

Example Assignments / Labs	• **R First Names**
	  ○ Students combine a pattern match with a multi-value condition in a single query.
	• **List All Gryffindors**
	  ○ Students use a JOIN to combine person and house data in one result set.
	• **Quidditch Matchups: Ravenclaw Away Games**
	  ○ Students join across tables to return home and away teams for a filtered set of matches.
	• **Design Your Database Schema**
	  ○ Students design a visual schema for a business of their choosing and populate it with sample data.
	• **Present Your Database!**
	  ○ Student teams present their design and justify the entities and attributes they chose.

Module 18: AI (7 hours)

Students examine how AI models are trained, where bias and hallucinations come from, the risks of deepfakes and misinformation, and the hardware that makes machine learning possible.

Objectives / Topics Covered	• Emerging Technology Ethics • Human and Artificial Intelligence • Training AI Models • Bias in AI • Hallucinations and Security Risks • Deepfakes and Misinformation • Machine Learning • Deep and Transfer Learning • Machine Learning Hardware

Example Assignments / Labs	• **Explore & Reflect: Test the Intelligence of an AI**
	  ○ Students probe a public chatbot and evaluate what its responses reveal about machine intelligence.
	• **Turning on a Light: Pt 2, Apply the Model**
	  ○ Students train a gesture model, apply it, and evaluate how well it categorizes their inputs.
	• **Gender Shades Reflection**
	  ○ Students analyze a facial recognition bias study and explain how the bias arose.
	• **Making an AI Hallucinate**
	  ○ Students construct prompts that produce hallucinations and describe what caused them.
	• **Project: AI on Trial**
	  ○ Students research generative AI lawsuits and argue a mock trial from an assigned role.