International Baccalaureate Computer Science for the Diploma Program

Higher Level Year One (140 - 150 Contact Hours)

Course Overview and Goals

The International Baccalaureate (IB) Diploma Program for Computer Science is designed to facilitate the learning and exam taking processes for students who are enrolled in the Higher Level (HL) section of their IB Diploma Program. This course covers programming components and critical skills, which align with each of the 10 characteristics of the IB Learner Profile. Students will learn how programming and cybersecurity are applicable to nearly all fields of study and how to implement these skills in real-life.

IB Computer Science Higher Level is a 2-year sequence. Currently the CodeHS course includes the first year of content.

Learning Environment

This course adapts the blended learning approach, giving teachers more autonomy in their pedagogical approach. The course content is a combination of web-based and offline activities. However, students will access lessons through the CodeHS platform.

Programming Environment

Students will demonstrate their programming skills by learning to program in Java, which aligns with the standards established by the International Baccalaureate Organization. Students will also learn concepts in Networking, Digital Information, Computer Organization, and Design Thinking. Students will also be able to practice exam problems throughout the course.

Quizzes

Each lesson includes at least one multiple choice assessment. At the end of each module, students will take a summative multiple choice assessment to test their understanding of the concepts covered throughout.

Prerequisites

The International Baccalaureate Computer Science for Diploma Program course uses the Java programming language and is designed for complete beginners with no previous background in computer science. The course is highly visual, dynamic, and interactive, making it engaging for new coders.

More Information

Browse the content of this course at: https://codehs.com/course/13497/explore

Course Breakdown

Unit 1: Introduction to Programming with Karel the Dog

(2.5 weeks or 8 hours)

In this module, students learn the basics of Java commands, control structures, and problem solving by solving puzzles with Karel.

Topics Covered	● Commands ● Defining vs. Calling Methods ● Designing methods ● Program entry points ● Control flow ● Looping ● Conditionals ● Classes ● Commenting code ● Preconditions and Postconditions ● Top Down Design

Unit 2: Design Thinking

(2.5 weeks or 8 hours)

This module introduces students to the theory and practice of user centered design. Students learn about what makes an engaging and accessible product and will employ an iterative design process including rapid prototyping and user testing to design and develop their own product.

Topics Covered	● Design Cycle ● Accessibility ● Prototyping ● Testing

Unit 3: Put it in Writing

(2.5 weeks or 8 hours)

This module introduces students to the theory and practice of user centered design. Students learn about what makes an engaging and accessible product and will employ an iterative design process including rapid prototyping and user testing to design and develop their own product.

Topics Covered	● Planning and implementing a new IT system ● Change management ● Data migration challenges and solutions ● Cloud-based systems ● User documentation and training ● Managing updates ● Causes and impacts of data loss ● Data loss prevention

Unit 4: Digital Information

(3 weeks or 12 hours)

Students learn about the various ways to represent information digitally including number systems, encoding data, programmatically creating pixel images, comparing data encodings, compressing and encrypting data.

Topics Covered	- How digital data is represented - Encoding data - Converting and using different number systems - Binary - Hexadecimal - Manipulating images at the pixel level

Unit 5: Networking

(3 weeks or 12 hours)

This module explores the structure and design of the internet and networks, and how this design affects the reliability of network communication, the security of data, and personal privacy. Students will learn how the Internet connects computers all over the world by use of networking protocols.

Topics Covered	- Introduction to the Internet - Notational Systems - Data Representation - Internet Hardware - Internet Addresses - Domain Name System (DNS) - Routing - Packets and Protocols - The Internet and Cybersecurity - Impact of the Internet

Unit 6: Computer Organization

(2 weeks or 6 hours)

Students learn about the physical elements of computers and networking, such as motherboards, RAM, routers, and the use of port numbers, ethernet, and wireless devices.

Topics Covered	- Hardware - Software - Computer Organization - Input/Output

Unit 7: Primitive Types

(2.5 weeks or 8 hours)

This unit introduces students to the Java programming language and the use of classes, providing students with a firm foundation of concepts that will be leveraged and built upon in all future units. Students will focus on writing the main method and will start to call pre-existing methods to produce output. Students will start to learn about three built-in data types and learn how to create variables, store values, and interact with those variables using basic operations. The ability to write expressions is essential to representing the variability of the real world in a program and will be used in all future units. Primitive data is one of two categories of variables covered in this course. The other category, reference data, will be covered in the next unit.

Topics Covered	- Why programming? Why Java? - Variables and Data Types - Expressions and Assignments Statements - Compound Assignment Operators - User Input - Casting and Ranges of Variables

Unit 8: Using Objects

(3 weeks or 13 hours)

In the first unit, students used primitive types to represent real-world data and determined how to use them in arithmetic expressions to solve problems. This unit introduces a new type of data: reference data. Reference data allows real-world objects to be represented in varying degrees specific to a programmer's purpose. This unit builds on students' ability to write expressions by introducing them to `Math` class methods to write expressions for generating random numbers and other more complex operations. In addition, strings and the existing methods within the `String` class are an important topic within this unit.

Topics Covered	- Objects: Instances of Classes - Creating and Storing Objects (Instantiation) - Calling a Void Method - Calling a Void Method with Parameters - Calling a Non-void Method - `String` Objects: Concatenation, Literals, and More - `String` Methods - Wrapper Classes: `Integer` and `Double` - Using the `Math` class

Unit 9: Boolean Expressions and `if` Statements

(3 weeks or 13 hours)

Algorithms are composed of three building blocks: sequencing, selection, and iteration. This unit focuses on selection, which is represented in a program by using conditional statements. Conditional statements give the program the ability to decide and respond appropriately and are a critical aspect of any nontrivial computer program. In addition to learning the syntax and proper use of conditional statements, students will build on the introduction of Boolean variables by writing Boolean expressions with relational and logical operators. The third building block of all algorithms is iteration, which you will cover in Unit 4. Selection and iteration work together to solve problems.

Topics Covered	- Boolean Expressions - `if` Statements and Control Flow - `if-else` Statements - `else if` Statements - Compound Boolean Expressions - Equivalent Boolean Expressions - Comparing Objects

Unit 10: Iteration

(4 weeks or 16 hours)

This unit focuses on iteration using while and for loops. Boolean expressions are useful when a program needs to perform different operations under different conditions. Boolean expressions are also one of the main components in iteration. This unit introduces several standard algorithms that use iteration. Knowledge of standard algorithms makes solving similar problems easier, as algorithms can be modified or combined to suit new situations. Iteration is used when traversing data structures such as arrays, ArrayLists, and 2D arrays. In addition, it is a necessary component of several standard algorithms, including searching and sorting, which will be covered in later units.

Topics Covered	● `while` loops ● `for` loops ● Developing Algorithms Using Strings ● Nested Iteration ● Informal Code Analysis
Associated Lab: Consumer Review Lab	

Unit 11: Writing Classes

(3 weeks or 13 hours)

This unit will pull together information from all previous units to create new, user-defined reference data types in the form of classes. The ability to accurately model real-world entities in a computer program is a large part of what makes computer science so powerful. This unit focuses on identifying appropriate behaviors and attributes of real-world entities and organizing these into classes. Students will build on what they learn in this unit to represent relationships between classes through hierarchies, which appear in Unit 9. The creation of computer programs can have extensive impacts on societies, economies, and cultures. The legal and ethical concerns that come with programs and the responsibilities of programmers are also addressed in this unit.

Topics Covered	● Anatomy of a Class ● Constructors ● Documentation with Comments ● Accessor Methods ● Mutator Methods ● Writing Methods ● Static Variables and Methods ● Scope and Access ● `this` Keyword ● Ethical and Social Implications of Computing Systems

Unit 12: Array

(2 weeks or 8 hours)

This unit focuses on data structures, which are used to represent collections of related data using a single variable rather than multiple variables. Using a data structure along with iterative statements with appropriate bounds will allow for similar treatment to be applied more easily to all values in the collection. Just as there are useful standard algorithms when dealing with primitive data, there are standard algorithms to use with data structures. In this unit, we apply standard algorithms to arrays; however, these same algorithms are used with ArrayLists and 2D arrays as well. Additional standard algorithms, such as standard searching and sorting algorithms, will be covered in the next unit.

Topics Covered	• Array Creation and Access • Traversing Arrays • Enhanced for Loop for Arrays • Developing Algorithms Using Arrays

Unit 13: `ArrayList`

(2.5 weeks or 10 hours)

As students learned in Unit 6, data structures are helpful when storing multiple related data values. Arrays have a static size, which causes limitations related to the number of elements stored, and it can be challenging to reorder elements stored in arrays. The ArrayList object has a dynamic size, and the class contains methods for insertion and deletion of elements, making reordering and shifting items easier. Deciding which data structure to select becomes increasingly important as the size of the data set grows, such as when using a large real-world data set. In this unit, students will also learn about privacy concerns related to storing large amounts of personal data and about what can happen if such information is compromised.

Topics Covered	• Introduction to `ArrayList` • `ArrayList` Methods • Traversing ArrayLists • Developing Algorithms Using ArrayLists • Searching • Sorting • Ethical Issues Around Data Collection

Unit 14: 2D Array

(3 weeks or 13 hours)

In Unit 6, students learned how 1D arrays store large amounts of related data. These same concepts will be implemented with two-dimensional (2D) arrays in this unit. A 2D array is most suitable to represent a table. Each table element is accessed using the variable name and row and column indices. Unlike 1D arrays, 2D arrays require nested iterative statements to traverse and access all elements. The easiest way to accomplish this is in row-major order, but it is important to cover additional traversal patterns, such as back and forth or column-major.

Topics Covered	• 2D Arrays • Traversing 2D Arrays

Unit 15: Inheritance

(3 weeks or 13 hours)

Creating objects, calling methods on the objects created, and being able to define a new data type by creating a class are essential understandings before moving into this unit. One of the strongest advantages of Java is the ability to categorize classes into hierarchies through *inheritance*. Certain existing classes can be extended to include new behaviors and attributes without altering existing code. These newly created classes are called *subclasses*. In this unit, students will learn how to recognize common attributes and behaviors that can be used in a *superclass* and will then create a hierarchy by writing subclasses to extend a superclass. Recognizing and utilizing existing hierarchies will help students create more readable and maintainable programs.

Topics Covered	- Creating Subclasses and Superclasses - Writing Constructors for Subclasses - Overriding Methods - `super` Keyword - Creating References Using Inheritance Hierarchies - Polymorphism - `Object` Superclass

Unit 16: Recursion

(1 week or 4 hours)

Sometimes a problem can be solved by solving smaller or simpler versions of the same problem rather than attempting an iterative solution. This is called recursion, and it is a powerful math and computer science idea. In this unit, students will revisit how control is passed when methods are called, which is necessary knowledge when working with recursion. Tracing skills introduced in Unit 2 are helpful for determining the purpose or output of a recursive method. In this unit, students will learn how to write simple recursive methods and determine the purpose or output of a recursive method by tracing.

Topics Covered	- Recursion - Recursion Searching and Sorting

Unit 17: Exam Practice

(1 week or 4 hours)

Students apply their knowledge to complete IB-related exam practice questions.

Topics Covered	- Recursion - Recursion Searching and Sorting