# **Coding Languages Discussion Guide (for use during or after reading)**

1. What are coding languages and why are they necessary? (What Is a Coding Language?, p. 4-5)
   1. Coding languages, also known as programming languages, are used by programmers to give instructions to computers. Coding languages are necessary because computers only understand things written in code, not in human language.
2. Compare high-level languages and low-level languages. (What Is a Coding Language?, p. 6-7)
   1. High-level languages are those programmers use to write computer codes. They use words and symbols that are familiar or similar to human language and mathematics, so they are easier for programmers to write, edit, and fix if needed. Low-level languages, like machine language or assembly language, are read by computers. Low-level languages involve binary numbers and can be difficult for humans to work with and manipulate. Programmers rely on special programs called compilers and assemblers in order to translate from a high-level language to a low-level one.
3. Machine language is made up of bits. How do these bits work with circuits to transmit data? (Machine Language, p. 8-11)
   1. Machine language is made up of bits, or the tiniest pieces of data a computer can store. Bits are represented in binary, by either a *0* or a *1*. Computer programs read bits to determine their instructions and switch certain circuits on and off. When a computer reads a *0* bit, a circuit switches off, and when it reads a *1* bit, a circuit switches on. Many circuits work together to carry information to processors within the computer.
4. Why is coding in a high-level language considered to be more efficient than coding in a low-level language? (High-Level Languages, p. 12-13)
   1. Coding in a high-level language is considered to be more efficient that coding in a low-level language because high-level languages can be used on a variety of machines, whereas low-level languages can be very specific to one type of machine, or even a single machine! If programmers only used low-level languages, they would need to write huge amounts of complex code specific to the hardware of each machine that might run the program. High-level languages are therefore more efficient because they must only be written once and can work for a variety of machines. In addition, high-level languages involve familiar words and symbols and are easier for programmers to code, edit, and fix, making them more efficient to use.
5. If high-level languages can be used to write programs that work on a variety of machines, why are there so many different types of high-level programming languages? (High-Level Languages, p. 14-15)
   1. Even though high-level languages can be used on a variety of machines, different programming languages are better suited for different situations. For example, some high-level languages are great for building websites, whereas others might be better for designing applications or making video games.
6. Describe the high-level programming language C. (C, p. 16-17)
   1. One high-level programming language is called C. It is closer to machine language (a low-level language) than other languages, giving the programmer more control over the program itself. C allows the program to easily access a computer’s memory as well as its hardware. Most operating systems, programs that control the main functions of a computer, use C. Many video games use a version of C called C++.
7. Describe the high-level programming language Python. (Python, p. 18-19)
   1. Another high-level programming language is called Python. Python is considered to be easy to read compared to other coding languages. It is powerful and can be used to quickly make simple computer programs. Even companies like Google and YouTube use Python!
8. Describe the high-level programming language Scratch. (Scratch, p. 20-21)
   1. Both C and Python use lines of text in their code, but Scratch works a little bit differently. Scratch is a block-based language often used to teach beginning programmers how to create code. Block-based languages like Scratch rely on visual elements that fit together like puzzle pieces rather than text alone. This can make it easier for the programmer to create, edit, and fix code as needed. Scratch programs, called scripts, can be used to create simple games and are great practice for beginning programmers.
9. What is syntax? What frequently causes syntax errors? (So, How Do They Work?, p. 22-23)
   1. Syntax in coding is similar to syntax in any language. In English, syntax refers to the order and arrangement of words in a sentence. In code, syntax refers to the order in each line of code as well as the order of the lines themselves. Syntax helps programmers create codes that computers can understand. Syntax errors can occur when there is not proper syntax. For example, if a code is written in the wrong order, one might get a syntax error warning. Coding can be a complicated and detailed process where mistakes that cause syntax errors are bound to happen, even to the world’s best programmers!
10. How do compilers and assemblers work together to translate programs from high-level language into machine language? (Translating a Program, p. 26-27)
    1. Compilers and assemblers work together to translate programs from high-level language into machine language. First, the compiler translates a high-level language into an assembly language that will match its machine’s processors. Next, the assembler turns that assembly language into machine code so information can be sent to the machine’s circuits. Compilers and assemblers have made coding much more efficient because programmers no longer needed to enter binary digits into every single code. Not only did this cut down on programming time, it also helped eliminate coding errors.