

- Understand the **expectations**, **resources**, and **policies** associated with 15-110
- Define the essential components of computer science, **algorithms** and **abstraction**
- Construct **plain-language algorithms** to solve basic tasks
- Recognize and use the basic **data types** in programs
- Interpret and react to basic **error messages** caused by programs
- Use **variables** in code and trace the different values they hold
- Understand how different **number systems** can represent the same information
- Translate **binary numbers** to decimal, and vice versa
- Interpret binary numbers as abstracted types, including **colors** and **text**
- Identify the **argument(s)**, **returned value**, and **side effect(s)** of a function call
- Use **function definitions** when reading and writing algorithms to implement procedures that can be repeated on different inputs
- Recognize the difference between local and global **scope**
- Trace the **call stack** to understand how Python keeps track of nested function calls
- Use **libraries** to import functions in categories like math, randomness, and graphics
- Recognize that the process of **tokenizing**, **parsing**, and **translating** converts Python code into instructions a computer can execute
- Recognize how the different types of **errors** are raised at different points in the Python translation process
- Use **logical operators** on Booleans to compute whether an expression is True or False
- Use **conditionals** when reading and writing algorithms that make choices based on data
- Use **nesting** of control structures to create complex control flow
- Debug **logical errors** by using the scientific method