Glossary

For Introduction to Programming Using Python
By Y. Daniel Liang

Chapter 1

- **.py** Python script file extension name.
- **assembler** A software used to translate assembly-language programs into machine code.
- **assembly language** A low-level programming language in which a mnemonic is used to represent each of the machine language instructions.
- **bit** A binary number 0 or 1.
- **bus** The components are connected through a subsystem called a *bus* that transfers data or power between them.
- **byte** A unit of storage. Each byte consists of 8 bits. The size of hard disk and memory is measured in bytes. A megabyte is roughly a million bytes.
- **cable modem** Uses the TV cable line maintained by the cable company. A cable modem is as fast as a DSL.
- **calling a function** In programming terminology, when you use a function, you are said to be "invoking a function" or "calling a function."
- **central processing unit** (**CPU**) A small silicon semiconductor chip with millions of transistors that executes instructions.
- **comment** Comments document what a program is and how it is constructed. They are not programming statements and are ignored by the compiler. In Python, comments are preceded by the pound sign (#) in a line or enclosed between ''' and ''' in multiple lines.
- **compiler** A software program that translates program source code into a machine language program.
- **console** refers to text entry and display device of a computer.
- **dot pitch** The amount of space between pixels. The smaller the dot pitch, the better the display.
- **DSL** (**digital subscriber line**) Uses a phone line and can transfer data in a speed 20 times faster than a regular modem.
- **encoding scheme** An encoding scheme is a set of rules that govern how a computer translates characters, numbers, and symbols into data the computer can actually work with. Most schemes translate each character into a predetermined string of numbers.
- **function** A function performs actions.

- **hardware** The physical aspect of the computer that can be seen.
- **high-level programming language** Are English-like and easy to learn and program.
- **indentation** The use of tabs and spaces to indent the source code.
- **interactive mode** Typing a statement at the >>> prompt and executing it is called running Python in interactive mode.
- **interpreter** An interpreter reads one statement from the source code, translates it to the machine code or virtual machine code, and then executes it right away.
- **invoking a function** Same as calling a function.
- **IDLE** Stands for Interactive DeveLopment Environment, a tool for developing Python programs.
- **line comment** Text after the pound sign # in a line is a line comment.
- **logic error** An error that causes the program to produce incorrect result.
- **low-level language** Assembly language is referred to as a low-level language, because assembly language is close in nature to machine language and is machine dependent.
- **machine language** Is a set of primitive instructions built into every computer. The instructions are in the form of binary code, so you have to enter binary codes for various instructions.
- **memory** Stores data and program instructions for CPU to execute.
- **modem** A regular modem uses a phone line and can transfer data in a speed up to 56,000 bps (bits per second).
- **module** A text file for storing Python code is called a module, script, or source file.
- **motherboard** is a circuit case that connects all of the parts of a computer together.
- **network interface card** (**NIC**) A device to connect a computer to a local area network (LAN). The LAN is commonly used in business, universities, and government organizations. A typical type of NIC, called 10BaseT, can transfer data at 10 Mbps.
- **operating system** (**OS**) A program that manages and controls a computer’s activities (e.g., Windows, Linux, Solaris).
- **pixel** Tiny dots that form an image on the screen.

- **program** same as software
- **programming** writing programs is called programming.
- **resolution** Specifies the number of pixels per square inch. The higher the resolution, the sharper and clearer the image is.
- **runtime error** An error that causes the program to terminate abnormally.
- **software** The invisible instructions that control the hardware and make it work.
- **source code** A program written in a programming language such as Java.
- **source file** A file that stores the source code.
- **storage devices** The permanent storage for data and programs. Memory is volatile, because information is lost when the power is off. Program and data are stored on secondary storage and moved to memory when the computer actually uses them.
- **statement** A unit of code that specifies an action or a sequence of actions.
- **syntax error** An error in the program that violates syntax rules of the language.
- **syntax rules** Same as syntax.

Chapter 2

- **algorithm** Statements that describe how a problem is solved in terms of the actions to be executed, and specifies the order in which these actions should be executed. Algorithms can help the programmer plan a program before writing it in a programming language.
- **assignment operator** (=) Assigns a value to a variable.
- **camelCase** A convention for naming variables and functions. Use lowercase letters for variable names, as in radius and area. If a name consists of several words, concatenate them into one, making the first word lowercase and capitalizing the first letter of each subsequent word.
- **data type** the type for a value such as integers or strings.
- **expression** Represents a computation involving values, variables, and operators, which evaluates to a value.
- **floating-point number** A number that includes a fractional part.
- **identifier** A name of a variable, function, class.
- **incremental development and testing** A programming methodology that develop and test program incrementally. This approach is efficient and productive. It help eliminate and isolate errors.
- **input-process-output (IPO)** The essence of system analysis and design is input, process, and output. This is called IPO.
- **keyword** (or **reserved word**) A word defined as part of Java language, which have a specific meaning to the compiler and cannot be used for other purposes in the program.
- **line continuation symbol (\)** tells the interpreter that the statement is continued on the next line.
- **literal** A constant value that appears directly in the program. A literal may be numeric, character, string, etc.
- **operand** The operands are the values operated by an operator.
- **operator** Operations for primitive data type values. Examples of operators are +, -, *, /, and %.
- **pseudocode** natural language mixed with some programming code
- **simultaneous assignment** assign multiple values to multiple variable in one statement.
- **system analysis** seeks to analyze the data flow and

to identify the system's input and output.

- **system design** is the stage when programmers develop a process for obtaining the output from the input.
- **type conversion** Conversion of a value from one type to the other.
- **variable** Variables are used to store data and computational results in the program.

Chapter 3

- **backslash** (\) -- A character that precedes another character to denote the following character has a special meaning. For example, ‘\t’ denote a tab character. The backslash character is also used to denote a Unicode character like ‘\u00FF’.

- **character encoding** Mapping a character to its binary representation is called character encoding.

- **end-of-line** A character, which signifies the end of a line.

- **escape sequence** An escape sequence is a special syntax that begin with the character \ followed by a letter or a combination of digits to represent special characters, such as '\'', '\"', '\t', and '\n'.

- **method** A function invoked from an object.

- **newline** Same as end-of-line character.

- **object** In Python, a number is an object, a string is an object, and every data is an object. Objects of the same kind have the same type.

- **string** a sequence of characters.

- **whitespace** Characters ' ', '\t', '\f', '\r', and ‘\n' are whitespaces characters.

Chapter 4

- **Boolean expression** An expression that evaluates to a Boolean value.
- **Boolean value** true or false.
- **operator associativity** Defines the order in which operators will be evaluated in an expression if the operators has the same precedence order.
- **operator precedence** Defines the order in which operators will be evaluated in an expression.
- **selection statement** A statement that uses if statement to control the execution of the program.
- **short-circuit evaluation** Evaluation that stops when the result of the expression has been determined, even if not all the operands are evaluated. The evaluation involving and or or are examples of short-circuit evaluation.

Chapter 5

- **break statement** Break out of the current loop.
- **condition-controlled loop** The while loop is a condition-controlled loop; it is controlled by a true/false condition.
- **continue statement** Break out of the current iteration.
- **count-controlled loop** The for loop is a count-controlled loop that repeats a specified number of times.
- **infinite loop** A loop that runs indefinitely due to a bug in the loop.
- **input redirection** Getting input from a file rather from the console using the < symbol in the command line.
- **iteration** One time execution of the loop body.
- **loop** A structure that control repeated executions of a block of statements.
- **loop body** The part of the loop that contains the statements to be repeated.
- **loop-continuation-condition** A Boolean expression that controls the execution of the body. After each iteration, the loop-continuation-condition is reevaluated. If the condition is true, the execution of the loop body is repeated. If the condition is false, the loop terminates.
- **nested loop** Consists of an outer loop and one or more inner loops. Each time the outer loop is repeated, the inner loops are reentered, and all required iterations are performed.
- **off-by-one error** A common in the loop because the loop is executed one more or one less time than it should have been.
- **Output redirection** send the output to a file rather than displaying it on the console.
- **sentinel value** A special input value that signifies the end of the input.
- **step value** used in the range function to increase or decrease the current value.

Chapter 6

- **actual parameter** (i.e., argument) The variables or data to substitute formal parameters when invoking a function.
- **argument** Same as actual parameter
- **caller** The main program or a function that calls another function.
- **default argument** Default value for a parameter if the argument is not specified when the function is invoked.
- **divide and conquer** The concept of function abstraction can be applied to the process of developing programs. When writing a large program, you can use the “divide and conquer” strategy to decompose it into subproblems. The subproblems can be further decomposed into smaller, more manageable problems.
- **formal parameter (i.e., parameter)** The variables defined in the function signature.
- **information hiding** A software engineering concept for hiding the detail implementation of a function for the client.
- **function** A collection of statements grouped together to perform an operation. See class function; instance function.
- **function abstraction** A technique in software development that hides detailed implementation. Function abstraction is defined as separating the use of a function from its implementation. The client can use a function without knowing how it is implemented. If you decide to change the implementation, the client program will not be affected.
- **function header** The combination of the name of a function and the list of its parameters.
- **global variable** A variable declared outside all functions and are accessible to all functions in its scope.
- **immutable objects** an object whose contents cannot be changed.
- **information hiding** The details of the implementation are encapsulated in the function and hidden from the client that invokes the function. This is known as information hiding or encapsulation.
- **keyword arguments** pass arguments associated with the parameter names.
- **local variable** A variable declared inside a function.
- **None** A special value assigned to a variable, meaning

that the variable does not hold any value.

- **pass-by-value** (i.e., call-by-value) A term used when a copy of the value of the argument is passed to the function. For a parameter of a primitive type, the actual value is passed; for a parameter of a reference type, the reference for the object is passed.
- **parameter** Variables defined in the function signature.
- **positional arguments** Pass arguments to parameters according based on the positions.
- **return value** A value returned from a function using the return statement.
- **scope of variable** The portion of the program where the variable can be accessed.
- **static local variable** A variable declared in a function as static, whose value is retained for use in the next function call.
- **stepwise refinement** When writing a large program, you can use the “divide and conquer” strategy, also known as stepwise refinement, to decompose it into subproblems. The subproblems can be further decomposed into smaller, more manageable problems.
- **stub** A simple, but not a complete version of the function. The use of stubs enables you to test invoking the function from a caller.

Chapter 7

- **abstract data type (ADT)** A class is known as an abstract type.
- **accessor** (**getter**) The method for retrieving a private field in an object.
- **action** defines what an object can do.
- **attribute** Same as property.
- **behavior** Same as action.
- **class** An encapsulated collection of data and methods that operate on data. A class may be instantiated to create an object that is an instance of the class.
- **class abstraction** is the separation of class implementation from the use of a class.
- **class encapsulation** The details of implementation are encapsulated and hidden from the user. This is known as class encapsulation.
- **class's contract** The collection of methods together with the description of how these methods are expected to behave, serves as the class's contract.
- **client** The program that uses the class is often referred to as the client for the class.
- **constructor** A special method for initializing objects when creating objects. The constructor is defined using __init__ in the class.
- **data-field encapsulation** To prevent direct modifications of properties through the object reference, you can declare the field private, using the private modifier. Data field encapsulation makes the class easy to maintain.
- **dot operator** (.) An operator used to access members of an object. If the member is static, it can be accessed through the class name using the dot operator.
- **indentity** An object's identity is like a person's social security number. Python automatically assigns each object a unique id for identifying the object at runtime.
- **initializer** The initializer is always named __init__ for initializing the object.
- **instance** An object of a class.
- **instantiation** The process of creating an object of a class.
- **mutator** (**setter**) A method that changes the value of a private field in an object.
- **object-oriented programming (OOP)** An approach to

programming that involves organizing objects and their behavior into classes of reusable components.

- **private data field** cannot be accessed from the outside of the class.
- **private method** cannot be invoked from the outside of the class.
- **property** same as data field.
- **state** same as property.
- **Unified Modeling Language** (**UML**) A graphical notation for describing classes and their relationships.

Chapter 8

- **index operator** using the syntax [] to access the elements in a sequence.
- **concatenate operator** concatenate two sequences into a new sequence.
- **operator overloading** A container that contains a component is called a parent container for the UI component.
- **slicing operator** accessing a subsequence in a sequence.
- **repetition operator** concatenate the sequences multiple times.

Chapter 9

- **callback function** a function that is invoked when a binding event occurs.
- **geometry manager** for managing how the components are layout in a parent container.
- **grid manager** lays out the components in a grid.
- **handler** Same as a callback function.
- **pack manager** lays out the components in rows or columns.
- **parent container** A container that contains a component is called a parent container for the UI component.
- **place manager** lays out the components in absolute positions.
- **widget classes** The classes that define that user interface components in Tkinter.
- **event bindings** binds an event with a component.

Chapter 10

- **anonymous list** A list without a name.
- **binary search** An efficient function to search a key in a list. Binary search first compares the key with the element in the middle of the array and reduces the search range by half. For binary search to work, the array must be pre-sorted.
- **garbage collection** An object that is not referenced is a garbage. Garbage is automatically collected by Python.
- **index** An integer value used to specify the position of an element in the array. The array index is an int value starting with 0 for the first element, 1 for the second, and so on.
- **insertion sort** An approach to sort array. Suppose that you want to sort a list in ascending order. The insertion-sort algorithm sorts a list of values by repeatedly inserting a new element into a sorted sublist until the whole list is sorted.
- **linear search** A function to search an element in a list. Linear search compares the key with the element in the array sequentially.
- **selection sort** An approach to sort array. It finds the largest number in the list and places it last. It then finds the largest number remaining and places it next to last, and so on until the list contains only a single number.

Chapter 11

- **column index** You can think of a two-dimensional list as a list that consists of rows. Each row is a list that contains the values. The rows can be accessed using the index, conveniently called a row index. The values in each row can be accessed through another index, called a column index.
- **multidimensional list** A list that contains another list that may contain another list.
- **nested list** Same as multidimensional list
- **row index** See column index
- **two-dimensional list** A list that contains another list.

Chapter 12

- **aggregation** A special form of association that represents an ownership relationship between two classes.
- **association** A general binary relationship that describes an activity between two classes.
- **composition** A form of relationship that represents exclusive ownership of the class by the aggregated class.
- **dynamic binding** A method may be implemented in several classes along the inheritance chain. Python decides which method is invoked at runtime. This is known as dynamic binding.
- **inheritance** Python allows you to derive a subclass from several classes. This capability is known as multiple inheritance.
- **is-a relationship** Same as inheritance.
- **multiple inheritance** Same as inheritance.
- **override** Implement the method in a subclass that is declared in a superclass.
- **polymorphism** Refers to the feature that an object of a subclass can be used by any code designed to work with an object of its superclass.

Chapter 13

- **absolute filename** An absolute filename contains a filename with its complete path and drive letter.
- **binary file** Files can be classified into text and binary. A file that can be processed (read, created, or modified) using a text editor such as Notepad on Windows or vi on Unix is called a text file. All the other files are called binary files.
- **deserializing** Deserializing is the opposite process that extracts an object from a stream of bytes.
- **directory path** The complete directory name.
- **file pointer** When a file is opened for writing or reading, a special marker called a file pointer is positioned internally in the file.
- **raw string** When a file is opened for writing or reading, a special marker called a file pointer is positioned internally in the file.
- **serializing** Serializing is the process of converting an object into a stream of bytes that can be saved to a file or transmitted on a network.
- **text file** See binary file.
- **tracback** The traceback gives information on the statement that caused the error by tracing back to the function calls that led to this statement. The line numbers of the function calls are displayed in the error message for tracing the errors.

Chapter 14

- **dictionary** A dictionary is a collection that stores the elements along with the keys.
- **dictionary entry** A key/value pair is called a dictionary entry.
- **dictionary item** Same as dictionary entry.
- **hashable** An object is hashable if its hash value never changes during its lifetime.
- **immutable tuple** If a tuple contains immutable objects, the tuple is said to be immutable.
- **key/value pair** See dictionary entry.
- **map** Same as dictionary.
- **set** Sets are like lists in that you use them for storing a collection of elements. Unlike lists, however, the elements in a set are nonduplicates and are not placed in any particular order.
- **set difference** The difference between set1 and set2 is a set that contains the elements in set1 but not in set2.
- **set intersection** The intersection of two sets is a set that contains the elements that appear in both sets.
- **set union** The union of two sets is a set that contains all the elements from both sets. You can use the union method or the | operator to perform this operation.
- **set symmetric difference** The symmetric difference (or exclusive or) of two sets is a set that contains the elements in either set, but not in both sets.
- **tuple** Tuples are like lists, but their elements are fixed; that is, once a tuple is created, you cannot add new elements, delete elements, replace elements, or reorder the elements in the tuple.

Chapter 15

- **base case** A simple case where recursion stops.
- **direct recursion** A function invokes itself.
- **indirect recursion** A function A invokes function B, and B invokes A.
- **infinite recursion** Recursion never stops.
- **recursive function** A function that invokes itself directly or indirectly.
- **recursive helper function** Sometimes the original function needs to be modified to receive additional parameters in order to be invoked recursively. A recursive helper function can be declared for this purpose.
- **stopping condition** Same as base case.
- **tail recursion** A recursive function is said to be tail recursive if there are no pending operations to be performed on return from a recursive call.