Chapter One: Introduction

A SHORT INTRODUCTION TO HARDWARE, SOFTWARE, AND ALGORITHM DEVELOPMENT

Chapter Goals

- In this chapter you will earn:
  - About computer hardware, software and programming
  - How to write and execute your first Python program
  - How to diagnose and fix programming errors
  - How to use pseudocode to describe an algorithm

Computer Programs

- A computer program tells a computer the sequence of steps needed to complete a specific task
  - The program consists of a very large number of primitive (simple) instructions
- Computers can carry out a wide range of tasks because they can execute different programs
  - Each program is designed to direct the computer to work on a specific task

Programming:

- The act of designing, implementing, and testing computer programs

Hardware and Software

THE BUILDING BLOCKS THAT MAKE UP A COMPUTER

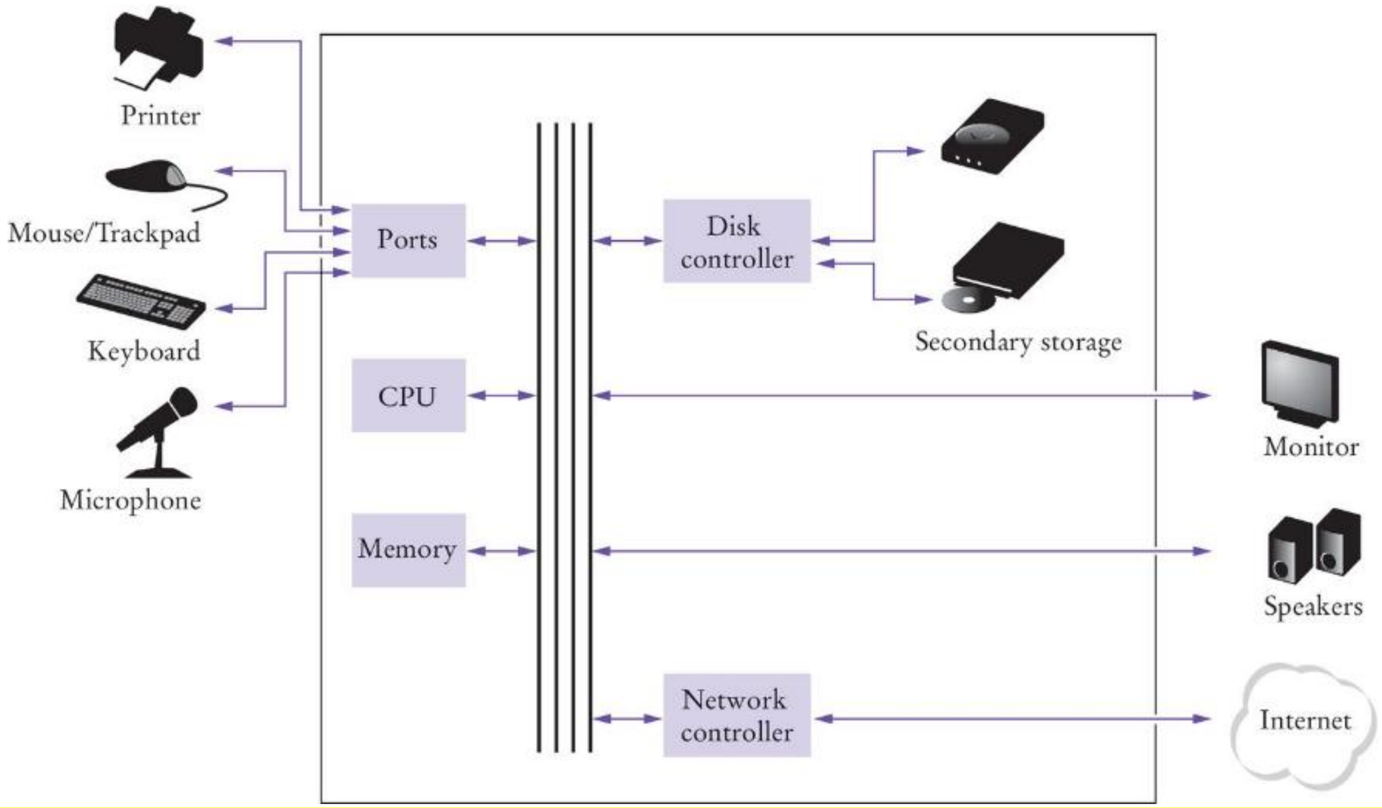
Hardware

- ***Hardware*** consists of the physical elements in a computer system.
  - Some very visible examples are the monitor, the mouse, external storage, and the keyboard.
- The ***central processing unit*** (CPU) performs program control and data processing
- Storage devices include memory (RAM) and secondary storage
  - Hard disk
  - Flash drives
  - CD/DVD drives

process speed:

- Input / output devices allow the user to interact with the computer
  - Mouse, keyboard, printer, screen…

Simple View of a Computer’s Components



The CPU

- The CPU has two components, the **control unit** and the **arithmetic logic unit**
- The ***control unit*** directs operation of the processor.
  - All computer resources are managed by the **control unit**.
  - It controls communication and co-ordination between input/output devices.
  - It reads and interprets instructions and determines the sequence for processing the data.
  - It provides timing and control signals
- The ***arithmetic logic unit*** contains the circuitry to perform calculations and do comparisons.
  - It is the workhorse portion of the computer and its job is to do precisely what the control unit tells it to do.

Storage

- There are two types of storage:
  - Primary Storage
  - Secondary Storage
- Primary storage is composed of memory chips: electronic circuits that can store data as long as it is provided electric power
- Secondary storage provides a slower, less expensive storage that is persistent: the data persists without electric power
- Computers store both data and programs
  - The data and program are located in secondary storage and loaded into memory when the program is executed

secondary storage => memory.

Memory

- A simple way to envision primary memory is a table of cells all the same size, one byte, and each containing a unique address beginning with 0.
  - The "typical" computer has a main memory ranging from 4 gigabytes (GB), to 32 GB.
- How big is a gigabyte?
  - A byte is 8 bits.
  - A kilobyte, KB, is 1024 bytes, or "about 1 thousand bytes."
  - A megabyte, MB, is 1,048,576 bytes, or "about 1 million bytes."
  - A ***gigabyte***, GB, is 1,073,741,824 bytes or "about 1 billion bytes."

Executing a Program

- Program instructions and data (such as text, numbers, audio, or video) are stored in digital format
- When a program is started, it is brought into memory, where the CPU can read it.
- The CPU runs the program one instruction at a time.
    - The program may react to user input.
- The instructions and user input guide the program execution
    - The CPU reads data (including user input), modifies it, and writes it back to memory, the screen, or secondary storage.

Software

- **Software** is typically realized as an application program
  - Microsoft Word is an example of software
  - Computer Games are software
  - Operating systems and device drivers are also software
- Software
  - Software is a sequence of instructions and decisions implemented in some language and translated to a form that can be executed or run on the computer.
- Computers execute very basic instructions in rapid succession
  - The basic instructions can be grouped together to perform complex tasks
- Programming is the act of designing and implementing computer programs

Algorithms

算法.

Introduction to Algorithms

- If you want a computer to perform a task, you start by writing an algorithm
- An ***Algorithm*** is:
  - a sequence (the order mattering) of actions to take to accomplish the given task
  - An algorithm is like a recipe; it is a set of instructions written in a sequence that achieves a goal
- For complex problems software developers write an algorithm before they attempt to write a computer program
- Developing algorithms is a fundamental problem solving skill
  - It has uses in many fields outside of Computer Science

Algorithm: Formal Definition

An ***algorithm*** describes a sequence of steps that is:

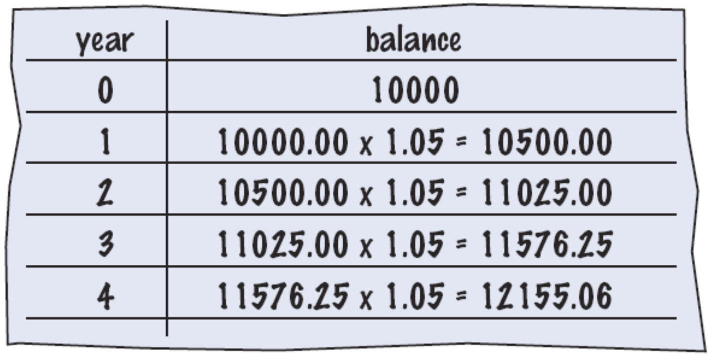
1. Unambiguous
   a. No "assumptions" are required to execute the algorithm
   b. The algorithm uses precise instructions
2. Executable
   a. The algorithm can be carried out in practice
3. Terminating
   a. The algorithm will eventually come to an end, or halt

Problem Solving: Algorithm Design

- Algorithms are simply plans
  - Detailed plans that describe the steps to solve a specific problem
- You already know quite a few
  - Calculate the area of a circle
  - Find the length of the hypotenuse of a triangle
- Some problems are more complex and require more steps
  - Calculate PI to 100 decimal places
  - Calculate the trajectory of a missile

Bank Account Example

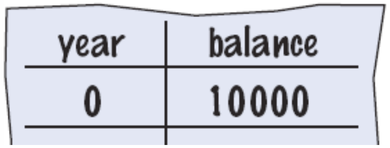
- Problem Statement:
  - You put $10,000 into a bank account that earns 5 percent interest per year. How many years does it take for the account balance to be double the original?
- How would you solve it?
  - Manual method
    - Make a table
    - Add lines until done
  - Use a spreadsheet!
    - Write a formula
      - Per line, based on line above



year	balance
0	10000
1	10000.00 x 1.05 = 10500.00
2	10500.00 x 1.05 = 11025.00
3	11025.00 x 1.05 = 11576.25
4	11576.25 x 1.05 = 12155.06

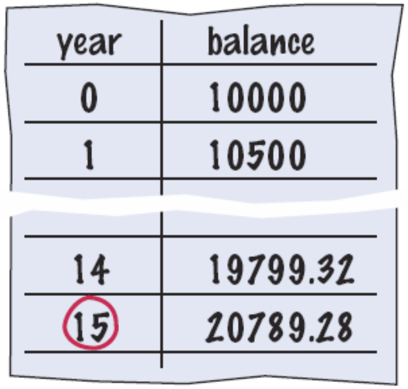
Develop the algorithm steps

- You put $10,000 into a bank account that earns 5 percent interest per year. How many years does it take for the account balance to be double the original?



year	balance
0	10000

- Break it into steps
  - Start with a year value of 0 and a balance of $10,000
  - Repeat the following while the balance is less than $20,000
    - Add 1 to the year value
    - Multiply the balance by 1.05
      - (5% increase)
  - Report the final year value as the answer



year	balance
0	10000
1	10500
14	19799.32
15	20789.28

Translate to pseudocode

- Pseudocode
  - Half-way between natural language and a programming language
- Modified Steps
  - Set the year value of 0
  - Set the balance to $10,000
  - While the balance is less than $20,000
    - Add 1 to the year value
    - Multiply the balance by 1.05
  - Report the final year value as the answer
- The pseudocode is easily translated into Python

Python and Programming Environments

The Python Language

- In the early 1990's, Guido van Rossum designed what would become the Python programming language
- Van Rossum was dissatisfied with the languages available
  - They were optimized to write large programs that executed quickly
- He needed a language that could not only be used to create programs quickly but also make them easy to modify
  - It was designed to have a much simpler and cleaner syntax than other popular languages such as Java, C and C++ (making it easier to learn)
  - Python is interpreted, making it easier to develop and test short programs
- Python programs are executed by the Python interpreter
  - The interpreter reads your program and executes it

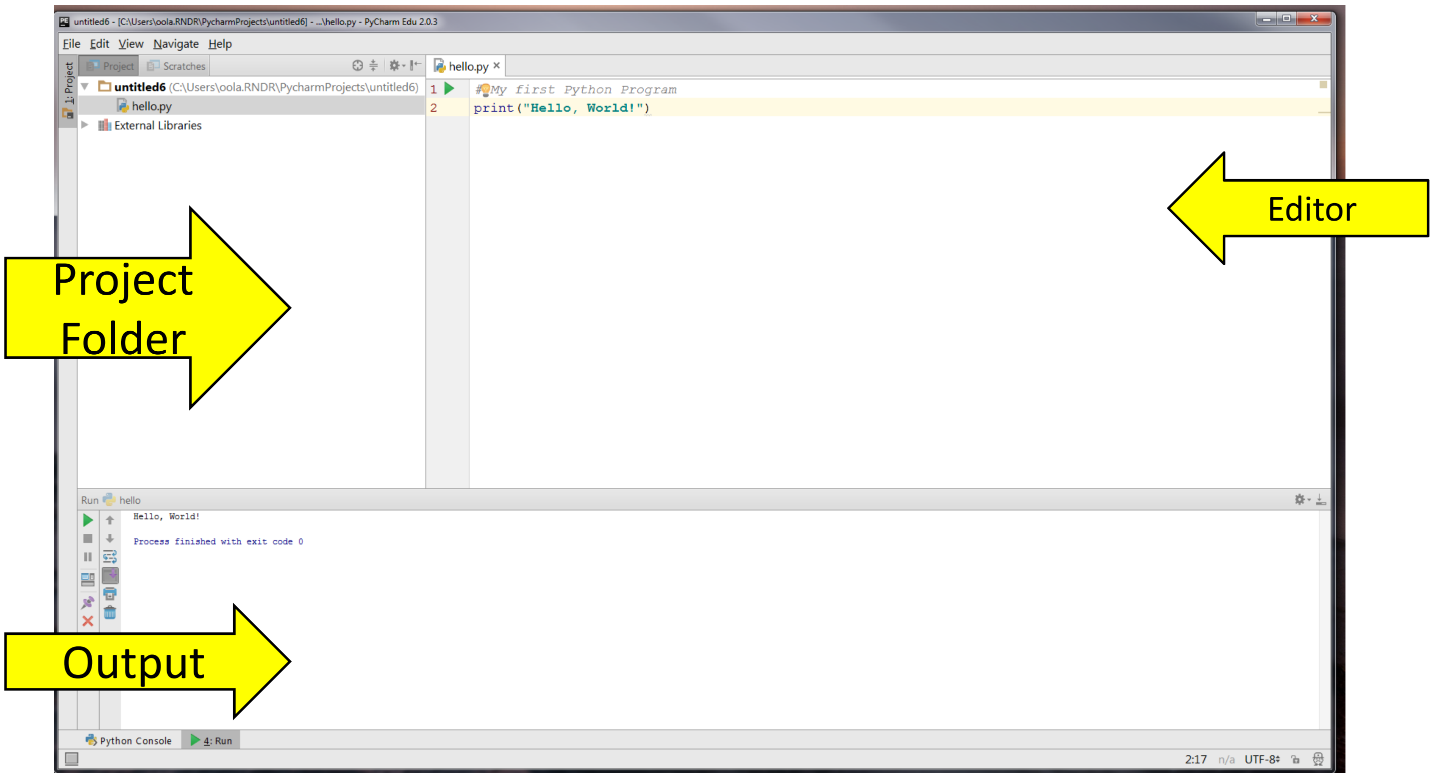
Programming Environments

- There are several ways of creating a computer program
  - Using an Integrated Development Environment (IDE)
  - Using a text editor
- IDE vs. Interpreter
  - Python is the Interpreter
  - PyCharm is the IDE
- You should use the method you are most comfortable with.
  - In this class, I will use the **PyCharm Educational Version**

IDE components

- The source code editor can help programming by:
  - Listing line numbers of code
  - Color lines of code (comments, text…)
  - Auto-indent source code
- Output window
- Debugger

PyCharm IDE


untitled6 - [C:\Users\oola.RNDR\PycharmProjects\untitled6] - ...\hello.py - PyCharm Edu 2.0.3
File Edit View Navigate Help
Project
Scratches
untitled6 (C:\Users\oola.RNDR\PycharmProjects\untitled6)
hello.py
External Libraries
hello.py
Run hello
Python Console
4: Run
2:17 n/a UTF-8
Editor
Project Folder
Output

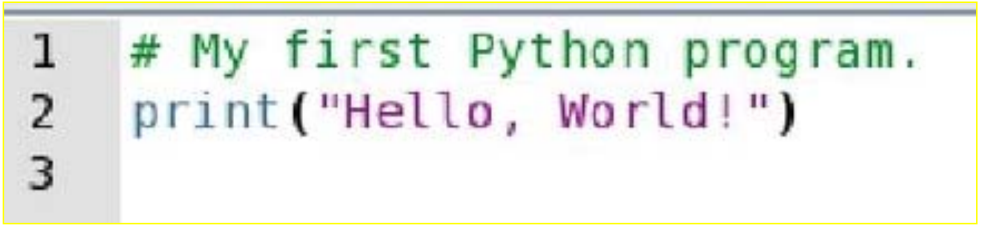
```
#My first Python Program
print("Hello, World!")
```

```
Hello, World!

Process finished with exit code 0
```

Your first program

- Traditional ‘Hello World’ program in Python



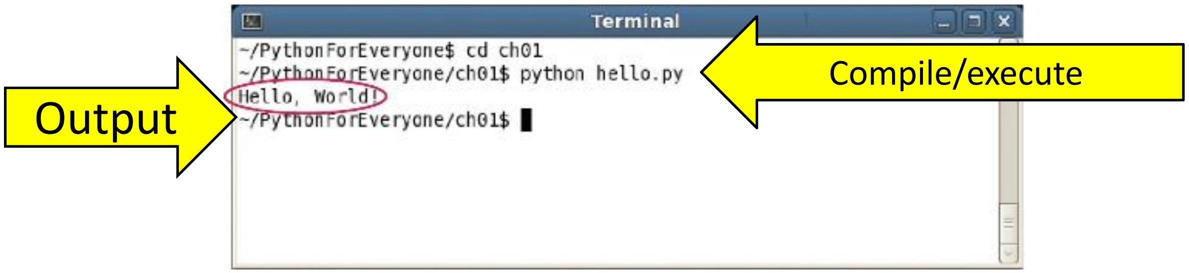
```python
# My first Python program.
print("Hello, World!")

```

- We will examine this program in the next section
  - Typing the program into your IDE would be good practice!
  - Be careful of spelling e.g., ‘print’ vs. ‘primt’
  - PyTHon iS CaSe SeNsItiVe.

Text editor programming

- You can also use a simple text editor to write your source code
- Once saved as Hello.py, you can use a console window to:
  - Compile the program
  - Run the program



```
~/PythonForEveryone$ cd ch01
~/PythonForEveryone/ch01$ python hello.py
Hello, World!
~/PythonForEveryone/ch01$ 
```

Organize your work

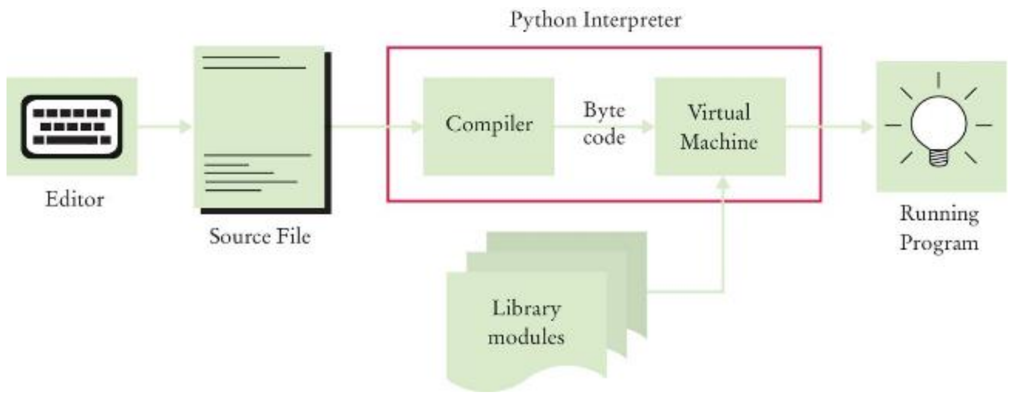
- Your ‘source code’ is stored in .py files
- Create a folder for this course
- Create one folder per program inside the course folder
  - A program can consist of several .py files
- Be sure you know where your IDE stores your files
  - You need to be able to find you files
- **Backup your files**:
  - To a USB flash drive
  - To a network drive

Python interactive mode

- Like other languages you can write/save a complete Python program in a file and let the interpreter execute the instructions all at once.
- Alternatively you can run instructions one at a time using interactive mode.
  - It allows quick ‘test programs’ to be written.
  - Interactive mode allows you to write python statements directly in the console window

Source Code to a Running Program

- The compiler reads your program and generates byte code instructions (simple instructions for the Python Virtual machine)
  - The Python Virtual machine is a program that is similar to the CPU of your computer
  - Any necessary libraries (e.g. for drawing graphics) are automatically located and included by the virtual machine



Analyzing Your First Program

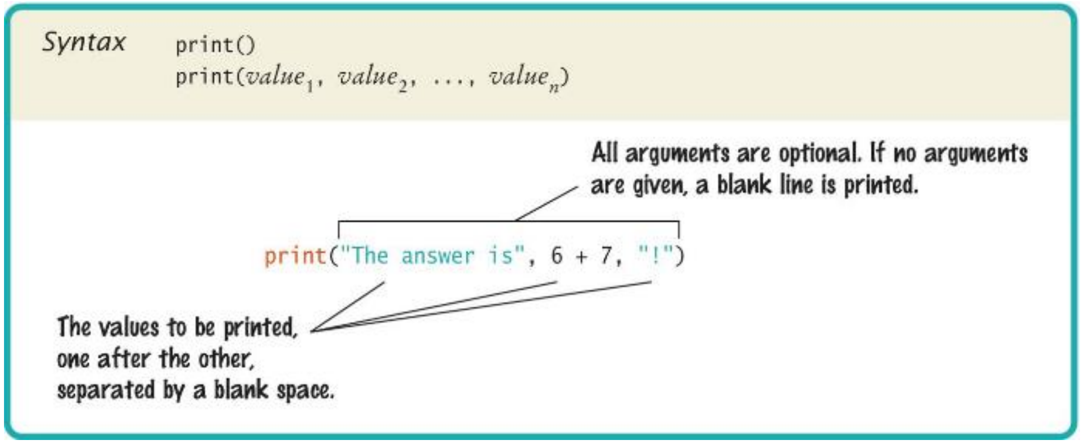
- A Python program contains one or more lines of instructions (statements) that will be translated and executed by the interpreter

```
# My first Python program
Print(“Hello World!”)
```

- The first line is a comment (a statement that provides descriptive information about the program to programmers).
- The second line contains a statement that prints a line of text onscreen “Hello, World!”

Basic Python Syntax: *Print*

- Using the Python ‘print()’ function.
  - A function is a collection of programming instructions that carry out a particular task (in this case to print a value onscreen).
  - It’s code that somebody else wrote for you!



Syntax

```
print()
print(value_1, value_2, ..., value_n)
```

```
print("The answer is", 6 + 7, "!")
```

All arguments are optional. If no arguments are given, a blank line is printed.
The values to be printed, one after the other, separated by a blank space.

Syntax for Python Functions

- To use, or call, a function in Python you need to specify:
  - The name of the function that you want to use (in the previous example the name was print)
  - Any values (arguments) needed by the function to carry out its task (in this case, “Hello World!”).
  - Arguments are enclosed in parentheses and multiple arguments are separated with commas.
  - A sequence of characters enclosed in quotations marks are called a string

Our First Program

```
##
#  Sample Program that demonstrates the print function
#
#  Prints 7

print(3 + 4)

#  Print Hello World! on two lines
print(“Hello”)
print(“World!”)

#  Print multiple values with a single print function call
print(“My favorite number are”, 3 + 4, “and” 3 + 10)

#  Print Hello World! on two lines
print(“Goodbye”)
print()
print(“Hope to see you again”)
```

Errors

- There are two Categories of Errors:
  - Compile-time Errors
    - aka Syntax Errors
      - Spelling, capitalization, punctuation
      - Ordering of statements, matching of parenthesis, quotes…
    - No executable program is created by the compiler
    - Correct first error listed, then compile again.
      - Repeat until all errors are fixed
  - Run-time Errors
    - aka Logic Errors
    - The program runs, but produces unintended results
    - The program may ‘crash’

Syntax Errors

- Syntax error are caught by the compiler
- What happens if you

• Miss-capitalize a word:	Print("Hello World!")
• Leave out quotes	print(Hello World!)
• Mismatch quotes	print("Hello World!')
• Don’t match brackets	print('Hello'

- Type each example above in **PyCharm**
  - What error messages are generated?

Logic Errors

- What happens if you

• Divide by zero	print(1/0)
• Misspell output	print("Hello, Word!")
• Forget to output	Remove line 2

- Programs will compile and run
  - The output may not be as expected
- Type each example above in **PyCharm**
  - What error messages are generated?

Summary: Computer Basics

- Computers rapidly execute very simple instructions
- A *Program* is a sequence of instructions and decisions
- *Programming* is the art (and science) of designing, implementing, and testing computer programs
- The Central Processing Unit (CPU) performs program control and data processing
- Storage devices include memory and secondary storage (e.g., a USB Flash Drive)

Summary: Python

- Python was designed in a way that makes it easier to learn than other programming languages such as Java, C and C++.
- The designers goal was to give Python simpler and cleaner syntax.
- Set aside some time to become familiar with the programming environment that you will use for your class work.
  - It is important to practice with the tool so you can focus on learning Python
- An editor is a program for entering and modifying text, such as a Python program.

Summary: Python

- Python is case sensitive.
  - You must be careful about distinguishing between upper and lowercase letters.
- The Python compiler translates source code into byte code instructions that are executed by the Virtual machine.
- A function is called by specifying the function’s name and its parameters.
- A string is a sequence of characters enclosed in quotation marks.

Summary: Errors and pseudo code

- A compile-time error is a violation of the programming language rules that is detected by the compiler.
- A run-time error causes a program to take an action that the programmer did not intend.
- Pseudo code is an informal description of a sequence of steps for solving a problem.
- An algorithm for solving a problem is a sequence of steps that is unambiguous, executable, and terminating.