ICS 104 - Introduction to Programming in Python and C

Introduction (Chapter 1: Sections 1-6)

Learning Outcomes

- Learn about computers and programming
- write and run your first Python program
- recognize compile-time and run-time errors

Computer Programs

- The **computer** is a machine that **stores** data (numbers, words, pictures), **interacts** with devices (the monitor, the sound system, the printer, the scanner), and **executes** programs.

- A computer **program** tells a computer, in minute detail, the sequence of steps that are needed to fulfill a task.
    - *The act of designing and implementing computer programs* is called **programming**.

- The physical computer and peripheral devices are collectively called the **hardware**.
- The programs the computer executes are called the **software**.

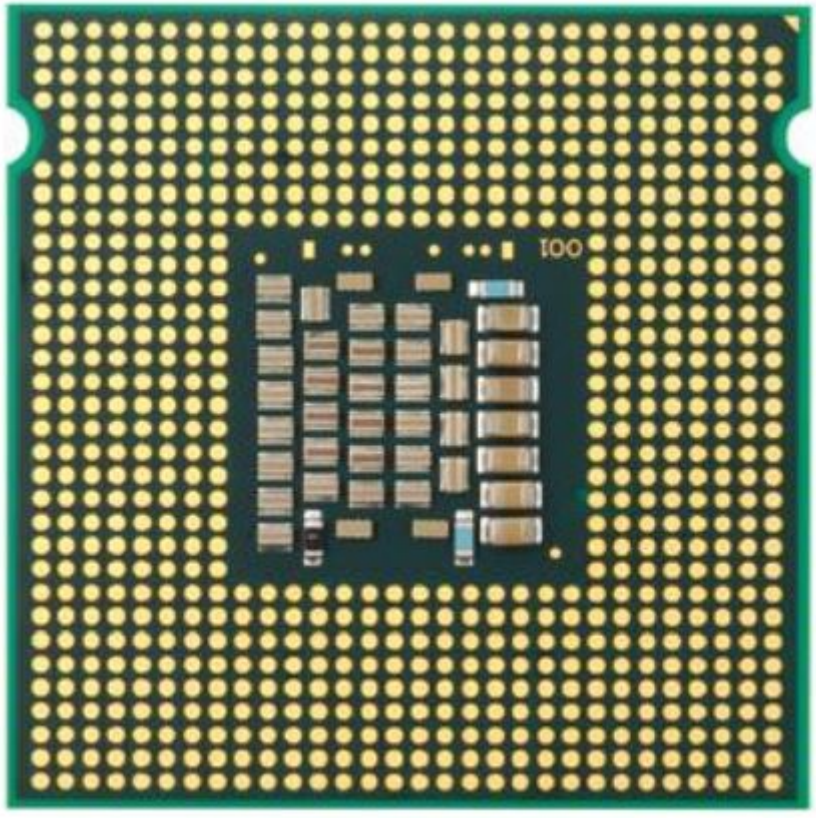
The Anatomy of a Computer

Computer **Hardware** consists of the physical elements in a computer system.

- The **central processing unit (CPU)** performs program control and data processing
- Storage devices include
    - **Primary memory**: Consists of memory chips (electronic circuits that can store data as long as it is provided electric power).
        - Fast and more expensive.
        - e.g., RAM and ROM
        -
    - **Secondary storage**: Provides slower, less expensive storage that is persistent (without elective power)
        - e.g., Hard disks, flash drives, CD/DVD drives.
    - Computers store both data and programs
        - Both are located in secondary storage and are loaded into primary storage when programs are executed.
- **Input/output** devices allow the user to interact with the computer
    - Mouse, keyboard, printer, screen

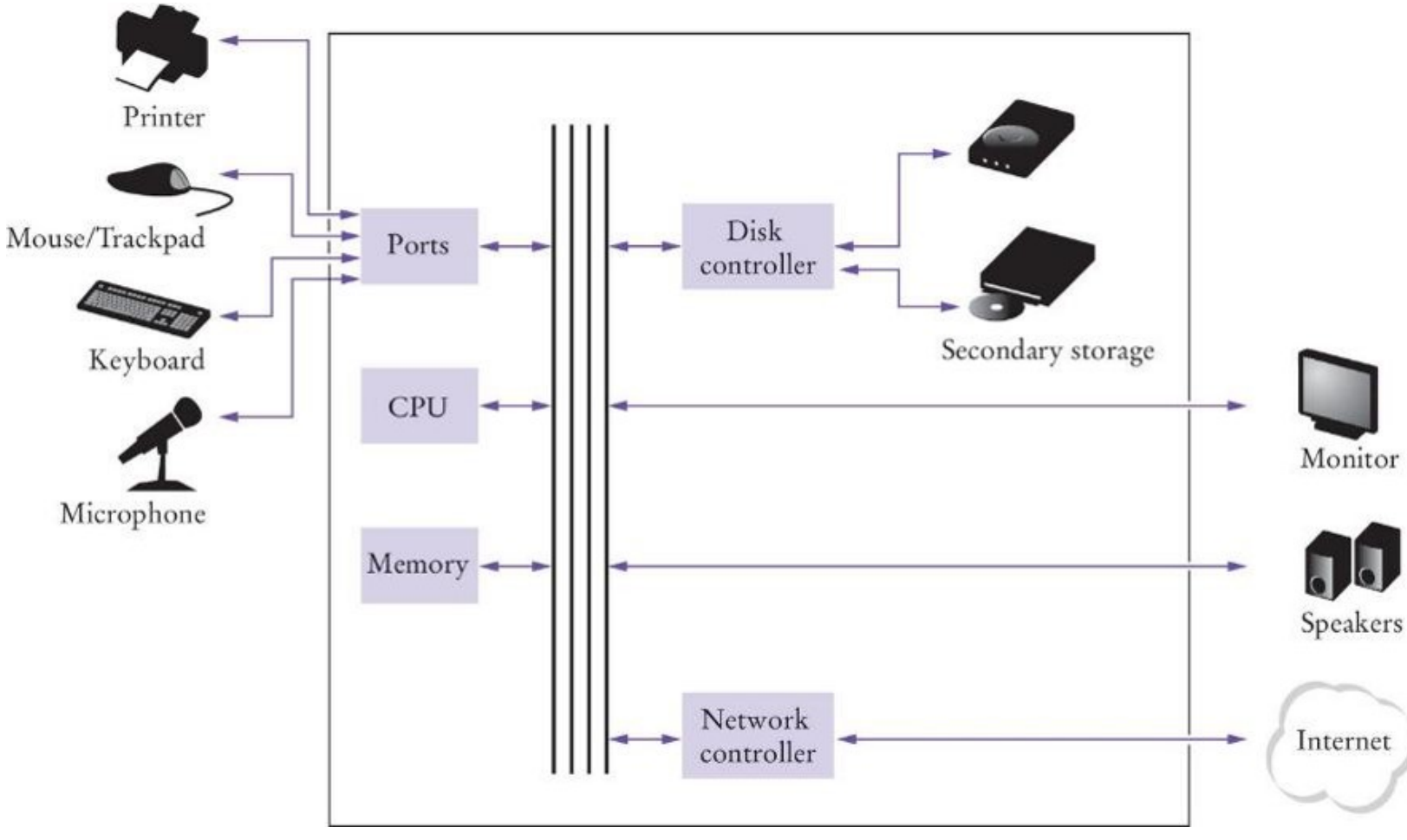
The Central Processing Unit (CPU)

- The CPU has two components, the **control unit** and the **arithmetic logic unit**



- The **control unit** directs operation of the processor.
    - Computer resources are managed by the control unit.
    - Controls communication and co-ordination between input/output devices.
    - Reads and interprets instructions and determines the sequence for processing the data.
    - 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.



- Program instructions and data (such as text, numbers, audio, or video) are stored on the hard disk, on a compact disk (or DVD), or elsewhere on the network.
- When a program starts, it is brought into memory, where the CPU can read it.
    - one instruction at a time.
    - The CPU reads, modifies and writes data back to memory or the hard disk.

- Some program instructions will cause the CPU to place dots on the display screen or printer or to vibrate the speaker.
- Some program instructions read user input from the keyboard or mouse.
    - The program analyzes the nature of these inputs and then executes the next appropriate instruction.

The Python Programming Language

- High-level programming languages specify a large number of simple CPU instructions with much fewer understandable statements.
    - Specifying CPU instructions can be tedious and error-prone.

- One very popular high level language is Python.
    - Developed by Guido van Rossum in the early 1990s.
    - Wanted a language suitable for writing smaller programs that may not run at optimum speed.
        - As opposed to other languages like Java or C.

Sauria Associates, LLC/FlickrVision/Getty Images, Inc.

- Some reasons for Python's succsess:
    - Simpler and cleaner syntax than Java, C, and C++.
    - Can try out short Python programs in an interactive environment.
    - Portable between computer systems.

Becoming Familiar with Your Programming Environment

- There are several ways of creating a computer program
    - Using an Integrated Development Environment (IDE)
    - Using a text editor
- We will use the **Jupyter Notebook** (Details on installing it will be provided in the first lab)

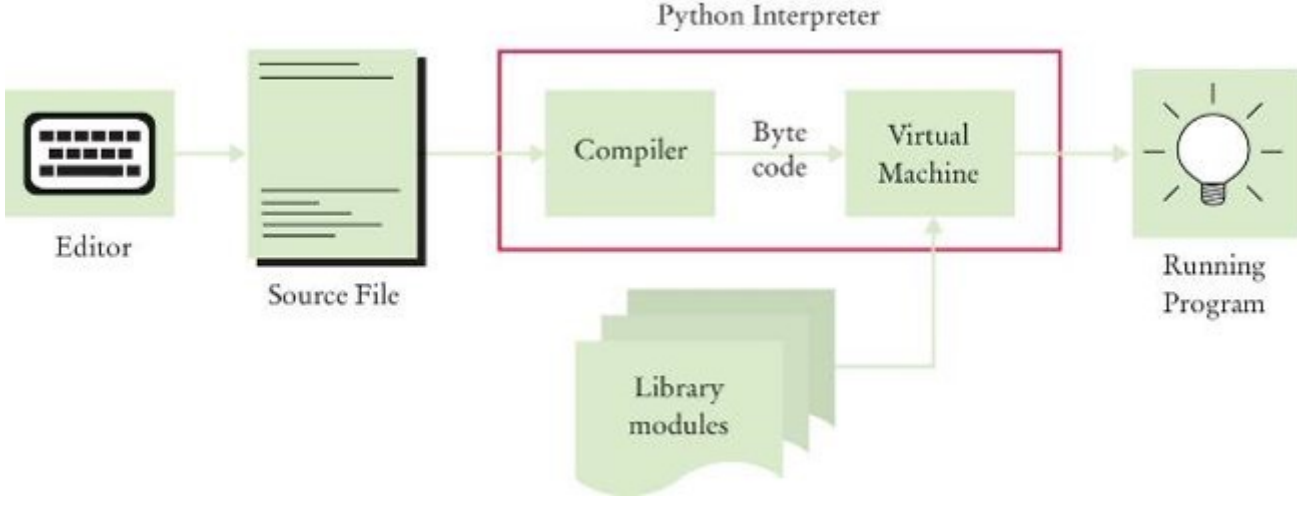
- **Jupyter Notebook** is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text.
- Slides of this course have been developed using Jupyter.
- A big advantage is that we use the same platform to **present** and **run** Python code.
- Let us look at our first Python program

In [1]:
```
# My first Python program.
print("Hello, Saudi World!")
print("Hello Sultan")
```

```
Hello, Saudi World!
Hello Sultan
```

How do Python programs run?

1. The compiler reads your source code (that is, the Python instructions that you wrote) all at once.
2. The compiler translates the instructions into **byte code**.
    - **Byte codes** are very simple instructions understood by the **virtual machine** (a separate program that is similar to the CPU of a computer).
3. Any necessary libraries are automatically located and included by the virtual machine.
    - For example, the implementation of the **`print`** function.
4. The virtual machine executes your byte code.



Analyzing Your First Program

In [3]:
```
# My first Python program.
print("Hello, World!")
```

```
Hello, World!
```

- The first line is a comment.
- Comments start with a # and are not considered statements (ignored by the interpreter).

- The second line displays a line of text, viz., **`Hello, World!`** using the **`print`** function.
- A **function** is a *collection* of programming instructions that carry out a particular task.
- **`"Hello, World!"`** is called a **string**.

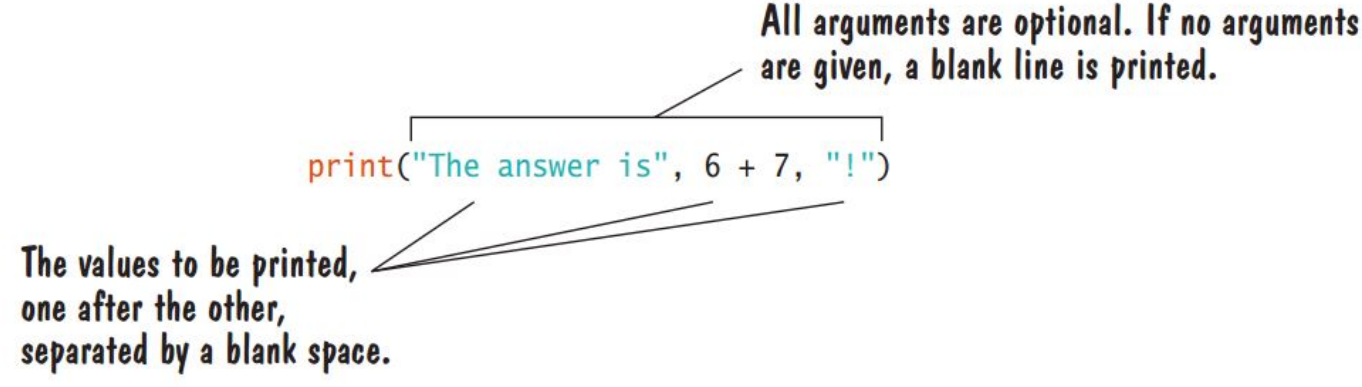
- To use, or call, a function in Python, you need to specify:
- The name of the function you want to use (in this case, **`print`**).
- Any values the function needs to carry out its task (in this case, **`"Hello, World!"`**).
    - The technical term for such a value is an argument.
    - Arguments are enclosed in parentheses.
    - Multiple arguments are separated by commas.
    - The number of arguments required depends on the function.

- Syntax of the **`print`** statement

print Statement

Syntax

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



More Examples of the Print Statement

- Printing numerical values

In [ ]:
```
print(3 + 4)
```

- Passing multiple values to the function

In [ ]:
```
print("The answers of adding", "and multiplying are", 6 + 7, 6 * 7, ", respectively")
```

- Note that each value passed to the function is displayed, one after another, with a blank space after each value.

- By default the print function starts a new line after its arguments are printed

In [ ]:
```
print("Hello")
print("World!")
```

Our Second Program (printtest.py)

In [ ]:

```
##
#  Sample program that demonstrates the print function.
#

# Prints 7.
print(3 + 4)

# Prints "Hello World!" in two lines.
print("Hello")
print("World!")

# Prints multiple values with a single print function call.
print("My favorite numbers are", 3 + 4, "and", 3 + 10)

# Prints three lines of text with a blank line.
print("Goodbye")
print()
print("Hope to see you again")
```

Errors

- There are two Categories of Errors:
    - Compile-time Errors (Syntax Errors)
    - Run-time Errors (Logical Errors)

Compile-time Errors

- Spelling, capitalization, punctuation
- Ordering of statements, matching of parenthesis, quotes, ...etc.

In [8]:
```
## Uncomment each statement.
#print("Hello, World!)
#print("Hello, World!")
#print("Hello World!')
#print('Hello'
```

- No executable program is created by the compiler
- Correct first error listed, then compile again.
- Repeat until all errors are fixed

Run-time Errors

- The program runs, but produces unintended results
- The program may **crash**

In [ ]:
```
## Uncomment each statement.
#print("Hello, Word!")
#print(1/0)
```

- Note that run-time errors are more troublesome. They are the harder to find and fix because the interpreter cannot flag them for us.

Problem Solving (Algorithm Design)

- Self Reading
- The topic will be discussed in Lab 2.

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)

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.
- 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.

Errors

- 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.