CSC1102
STRUCTURED AND OBJECT-ORIENTED PROGRAMMING

Contacts (In order of preference)

- Dr. Ntanda Moses
- Email: moses.ntanda@mak.ac.ug
- Office: Block A, Level 3, Networks Department, Room 304
- Mobile: 0752533355
- Whatsapp: +86170948610

Overview and Description

- See course webpage on MUELE

Objectives

See course webpage on MUELE

Mode of Delivery

- See course webpage on MUELE

Assessment

- Coursework and short projects
- Presentations
- Assignments
- Final exam

Reading Materials

- See course webpage on MUELE

Course Outline

– Module 1 (16 hours): Structured Programming

- Introduction to programming
- Program structure
- Branching and Iterations: Control flow statements

– Module 2 (16 hours): Data manipulation

- Functions

Course Outline

- Strings, pointers, and arrays

– Module 3 (28 hours): Object-Oriented Concepts

- Object-Oriented programming: Classes and objects
- Inheritance and visibility modifiers
- Interfaces and abstract classes
- Exception handling

Course Outline

– Module 1 (16 hours): Structured Programming

- Introduction to programming
- Program structure
- Branching and Iterations: Control flow statements

– Module 2 (16 hours): Data manipulation

- Functions

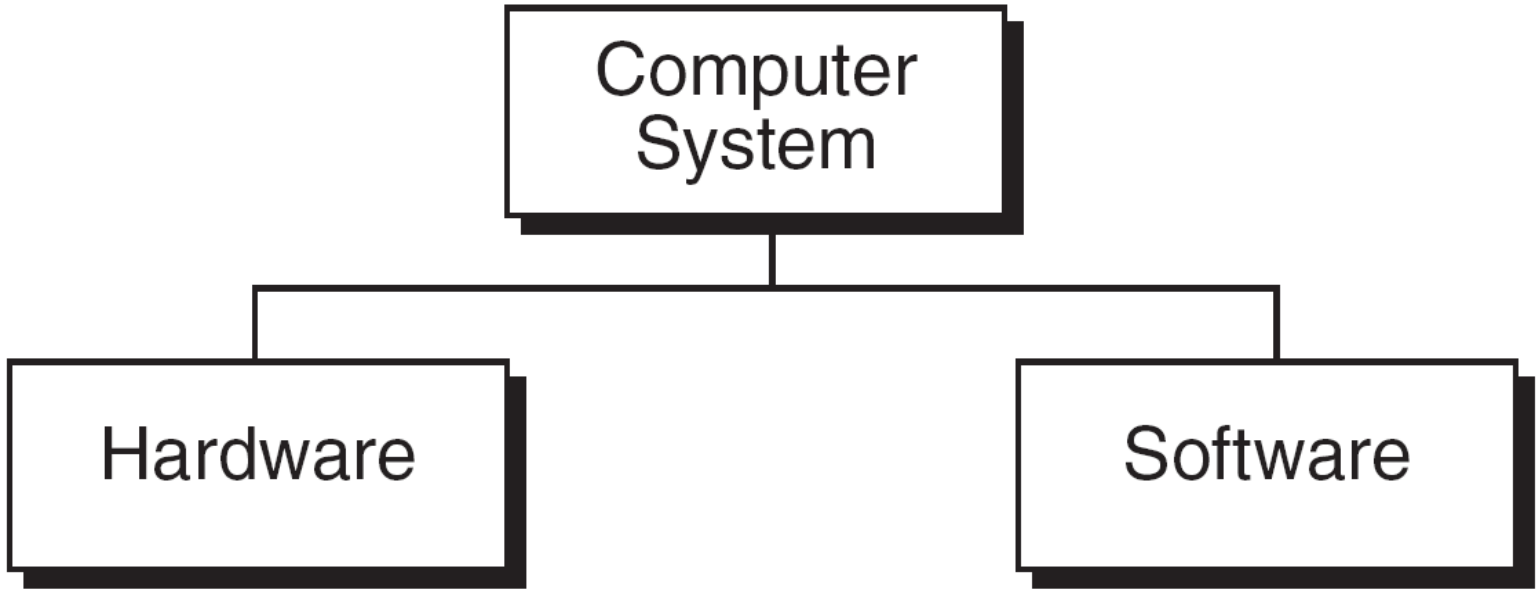
In This Lecture

- Introduction to programming
- Introduction to python

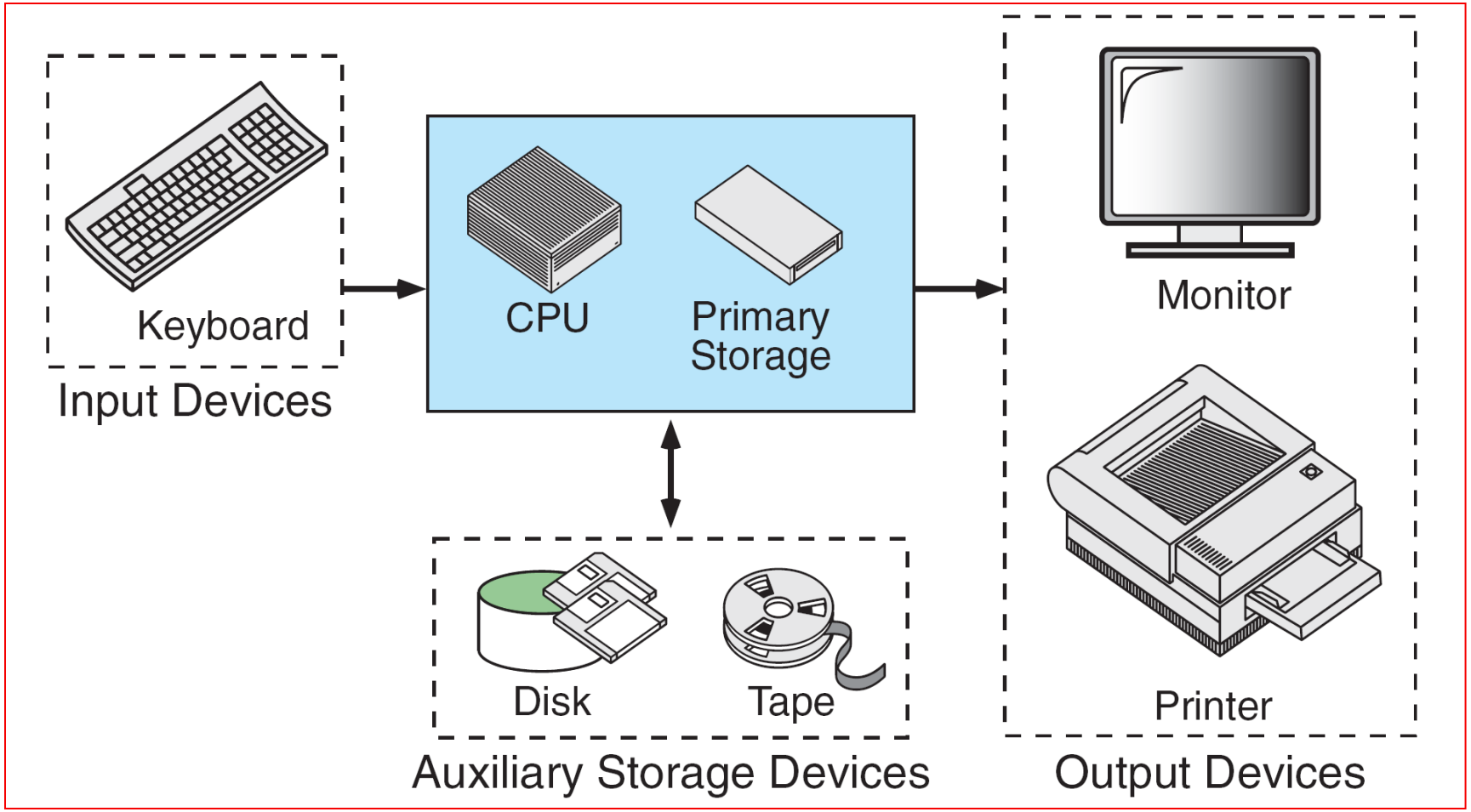
Introduction To Programming

- Computer programming is the process of designing and building an executable computer program for accomplishing a specific computing task (Wikipedia)
- A computer system consists of both Hardware and Software.
- Programming is the art and science of translating a set of ideas into a program - a list of instructions a computer can follow

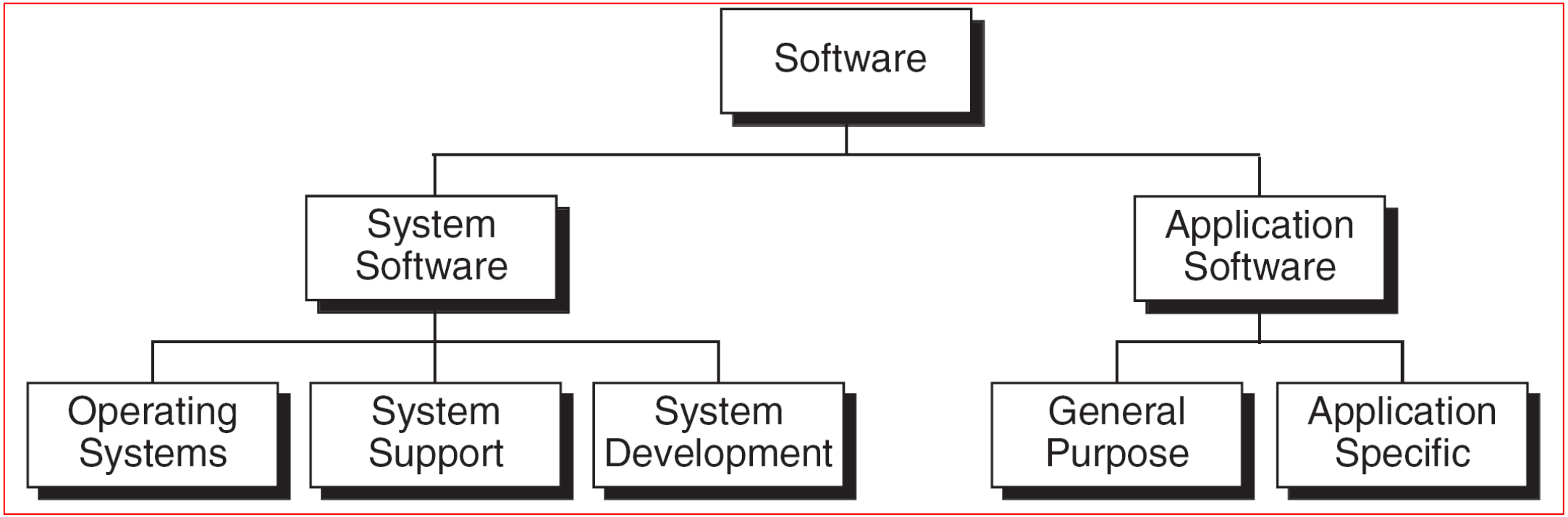
A Computer System



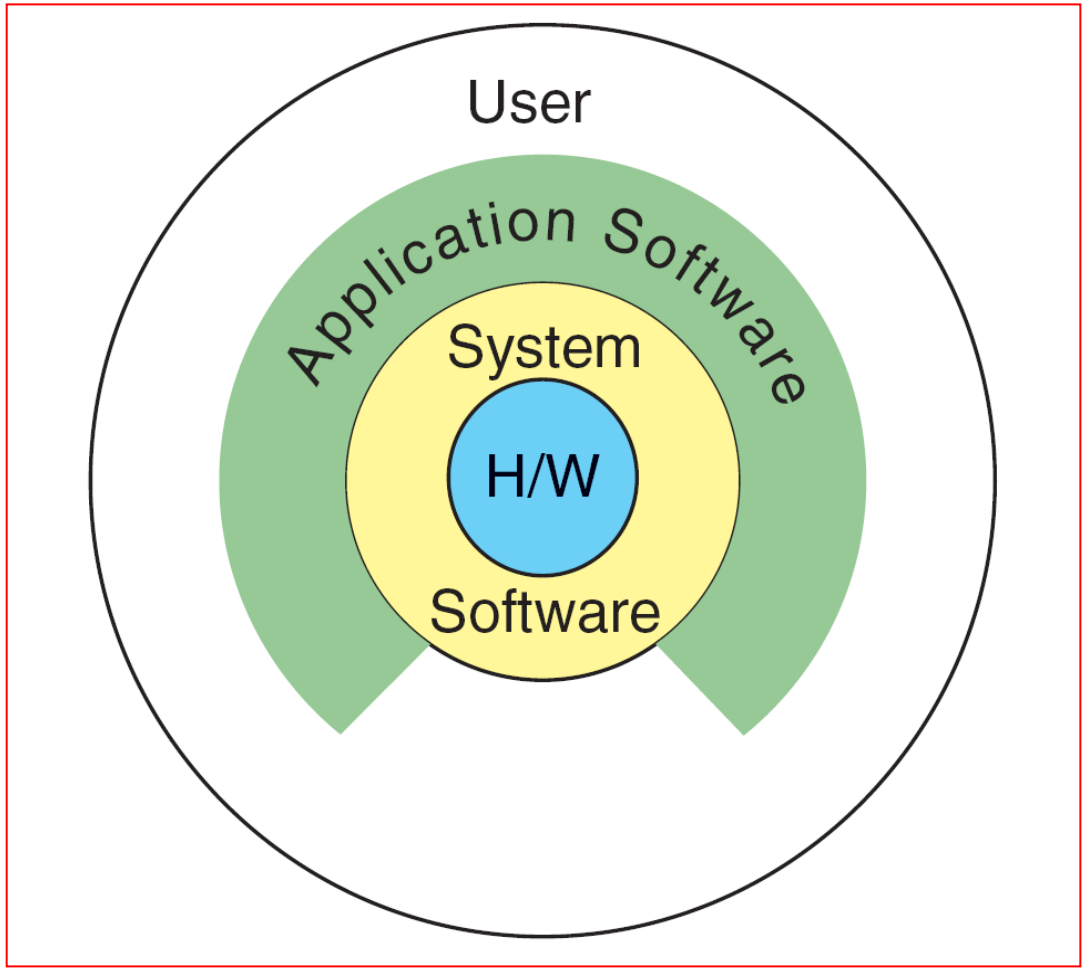
Basic Hardware Components



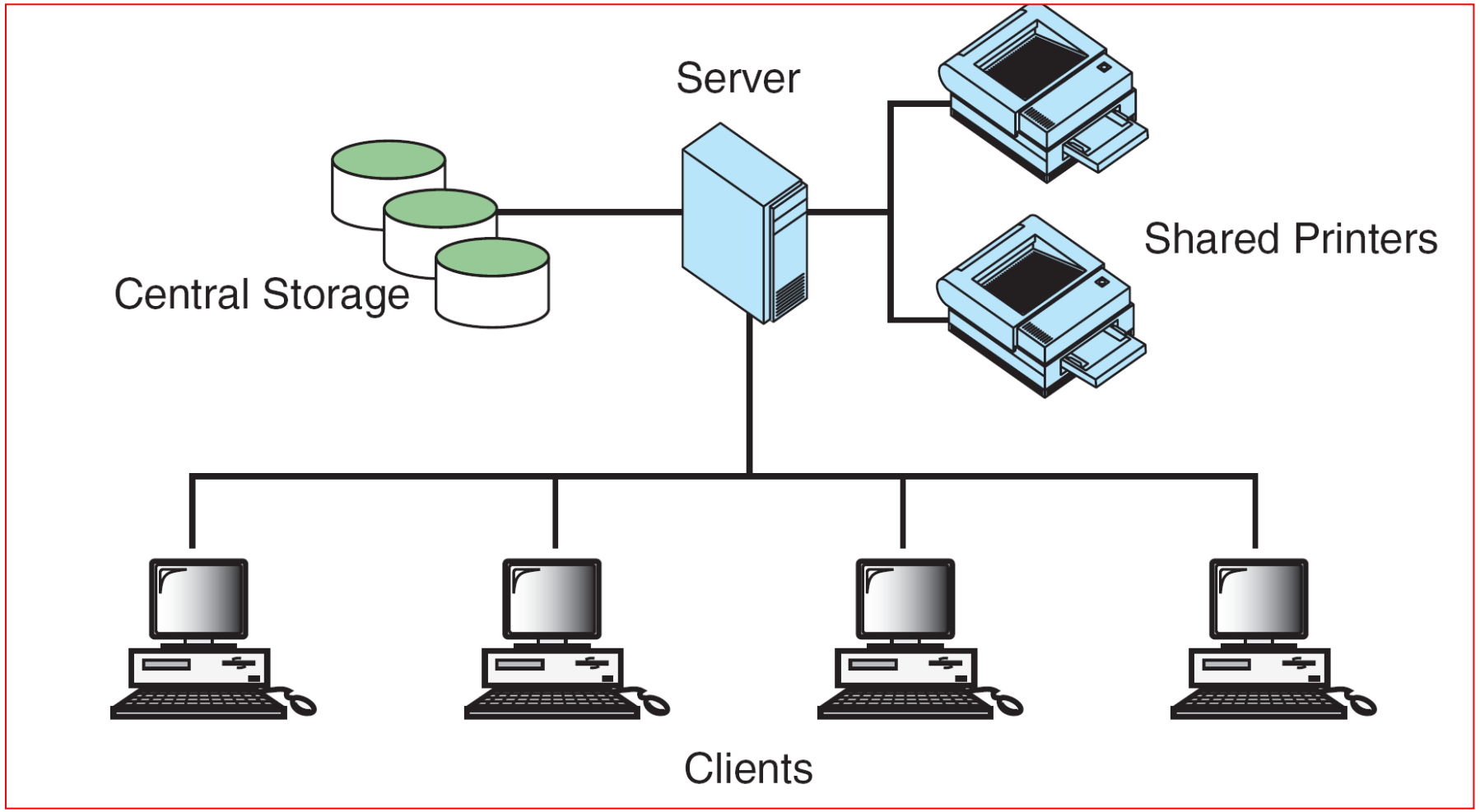
Types of Software



Relationship between system and application software



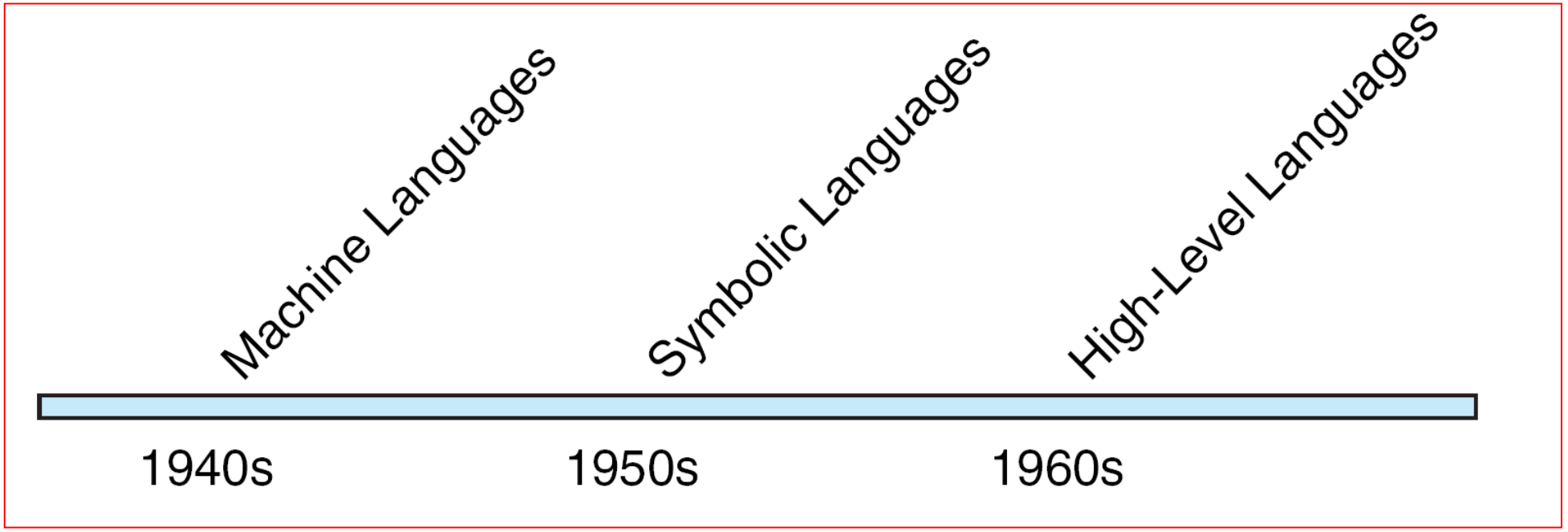
The Client/Server Environment



Computer Languages

- Computers don’t understand human language but computer language.
- The only language understood by computer hardware is machine language
- Computer languages have evolved from machine language to natural languages.

Computer Language Evolution



Sample Multiplication Program in Machine Language

```
 1            00000000 00000100 0000000000000000
 2  01011110  00001100 11000010 0000000000000010
 3            11101111 00010110 0000000000000101
 4            11101111 10011110 0000000000001011
 5  11111000  10101101 11011111 0000000000010010
 6            01100010 11011111 0000000000010101
 7  11101111  00000010 11111011 0000000000010111
 8  11110100  10101101 11011111 0000000000011110
 9  00000011  10100010 11011111 0000000000100001
10  11101111  00000010 11111011 0000000000100100
11  01111110  11110100 10101101
12  11111000  10101110 11000101 0000000000101011
13  00000110  10100010 11111011 0000000000110001
14  11101111  00000010 11111011 0000000000110100
15            01010000 11010100 0000000000111011
16                     00000100 0000000000111101
```

Sample Multiplication Program in Symbolic Language

```
 1        entry   main,^m<r2>
 2        subl2   #12,sp
 3        jsb     C$MAIN_ARGS
 4        movab   $CHAR_STRING_CON
 5
 6        pushal  -8(fp)
 7        pushal  (r2)
 8        calls   #2,SCANF
 9        pushal  -12(fp)
10        pushal  3(r2)
11        calls   #2,SCANF
12        mull3   -8(fp),-12(fp),-
13        pusha   6(r2)
14        calls   #2,PRINTF
15        clrl    r0
16        ret
```

The Multiplication Program in C

```
 1  /* This program reads two integers from the keyboard
 2     and prints their product.
 3        Written by:
 4        Date:
 5  */
 6  #include <stdio.h>
 7
 8  int main (void)
 9  {
10  // Local Definitions
11     int number1;
12     int number2;
13     int result;
14
15  // Statements
16     scanf  ("%d", &number1);
```

continued

The Multiplication Program in C (Continued)

```
17      scanf  ("%d", &number2);
18      result = number1 * number2;
19      printf ("%d", result);
20      return 0;
21  }   // main
```

The Multiplication Program in Python

```
num1 = input("Enter number 1\n")
num2 = input("Enter number 2\n")
result = int(num1)*int(num2)
print(result)
```

Programming Paradigms

- A **programming paradigm** is a style, or "way," of programming.
- Some languages are more suitable for some paradigms but not others.
- Given Language $\boldsymbol{L}$ makes it easy to use programming paradigm $\boldsymbol{P}$, we usually say "$\boldsymbol{L}$ is a $\boldsymbol{P}$ language" e.g:
  - Java is an Object Oriented Programming Language
  - Java is an Object Oriented Programming Language
  - C is a structured Programming Language etc

Some Common Programming Paradigms

- Imperative Programming
  - Control flow is *explicit*: commands show *how* the computation takes place, step by step.
  - Each step affects the global **state** of the computation.

```
    result = []
    i = 0
start:
    numPeople = length(people)
    if i >= numPeople goto finished
    p = people[i]
    nameLength = length(p.name)
    if nameLength <= 5 goto nextOne
    upperName = toUpper(p.name)
    addToList(result, upperName)
nextOne:
    i = i + 1
    goto start
finished:
    return sort(result)
```

Some Common Programming Paradigms

- Structured Programming
  - Similar to imperative programming but control flow is defined by nested loops, conditionals, and subroutines, rather than via gotos.
  - Variables are generally local to blocks

```
result = [];
for i = 0; i < length(people); i++ {
    p = people[i];
    if length(p.name)) > 5 {
        addToList(result, toUpper(p.name));
    }
}
return sort(result);
```

Some Common Programming Paradigms

- Object Oriented Programming (OOP)
  - Based on the sending of **messages** to objects.
  - Objects respond to messages by performing operations, generally called **methods**.

Some Common Programming Paradigms

- Declarative Programming
  - Control flow is *implicit*: the programmer states only *what* the result should look like, **not** how to obtain it.

```
select upper(name)
from people
where length(name) > 5
order by name
```

Some Common Programming Paradigms

- Functional Programming
  - control flow is expressed by combining function calls, rather than by assigning values to variables:

```
sort(
  fix(λf. λp.
    if(equals(p, emptylist),
      emptylist,
      if(greater(length(name(head(p))), 5),
        append(to_upper(name(head(p))), f(tail(p))),
        f(tail(people)))))(people))
```

Programming Errors

Three Major Types:

- **Syntax errors**: errors due to the fact that the syntax of the language is not respected.
- **Semantic errors**: errors due to an improper use of program statements.
- **Logical errors**: errors due to the fact that the specification is not respected.

From the point of view of when errors are detected, we distinguish:

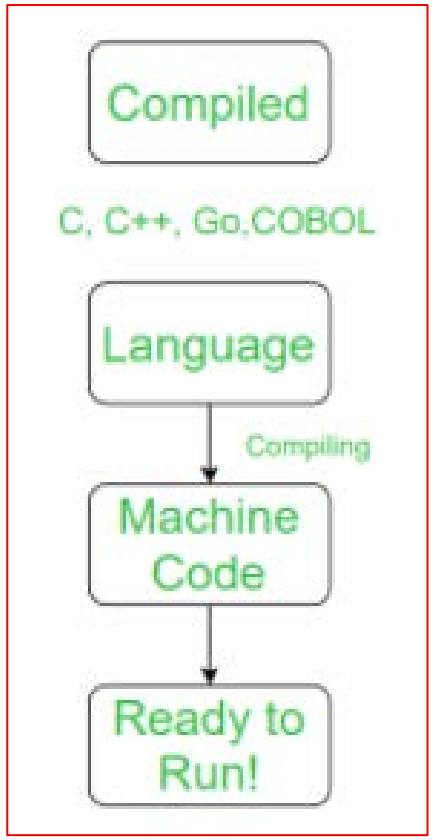
- **Compile time errors**: syntax errors and **static** semantic errors indicated by the compiler.
- **Runtime errors**: **dynamic** semantic errors, and logical errors, that cannot be detected by the compiler (debugging).

Introduction to Python

- In this course, we use Python (though the course is not about Python). Though you will learn Python, more important is that you will learn how to write programs that solve problems. This skill can be transferred to any programming language.
- Why Python?
  - Cross platform (Windows, Mac, Linux, Raspberry Pi, etc).
  - Simple syntax (close to English).
  - Concise (use fewer lines to accomplish a lot).
  - Interpreted language (code can be executed as soon as it is written. Makes prototyping very quick).
  - Can be treated in a procedural way, an object-oriented way or a functional way.

Introduction to Python

- Python is an interpreted Object Oriented Programming (OOP) language .



Introduction to Python: What can python do?

- Create web applications on server side.
- Connect to database systems.
- Read and modify files.
- Handle big data and perform complex mathematics.
- Rapid prototyping.

Introduction to Python

- Considered a high-level language because it allows the programmer to concentrate on the problem at hand and not worry about the machine that the program will be using.
- Convenient for software developers whose applications have to run on many different hardware platforms.
- Python is an Advanced 3rd Generation Language (3GL) thus much more machine independent and more programmer-friendly.
- It obscures non-essential details from the user unlike (2GLs).
- 3GLs are more abstract than previous generations of languages, and thus considered higher level languages than 1GLs and 2GLs.

Python Programming Language

- Other 3GLs include Fortran, ALGOL, and COBOL .
- See https://en.wikipedia.org/wiki/Third-generation_programming_language
- Python improves clarity, quality, and software development time by making extensive use of the structured control flow constructs of:
  - selection (if/then/else),
  - repetition (while and for), and
  - block structures.
- Read more about structured programming: https://en.wikipedia.org/wiki/Structured_programming

Python Programming: Development Environment

- Python is already installed on most PCs and Macs
- To check if already installed, type `python --version` at the terminal. This returns the version of python installed on your machine. Current version of python is 3 and is one we shall use for this course.
- If not installed, download it from https://www.python.org/
- Python was introduced in 1990 by Guido von Rossum in 1990. Python 2 came in 2000 making the language very popular.
- Short python code can be run at command line / terminal. At the terminal, type `python` to start python interpreter

Python Programming: Development Environment

- Large python programs can be written using your favorite text editor to create the source code.
- The source code can be run from the file at command line.
- For complex projects, Integrated Development Environments, IDEs (Such as PyCharm) may be handy. The standard Python installation comes with an IDE called **IDLE.**
- IDEs come with text editors with syntax highlighting, auto completion, and smart indentation.

Python Programming: Hello World!

- Ready to start?
  - Start up python at the terminal.
  - Type print(“Hello World”)
  - Congs on your first python programs
  - Can you create the same program in file?
- Visit the Python Tutorial page at https://www.w3schools.com/python/default.asp and read about:
  - Python Syntax
  - Python comments
  - Python Variables