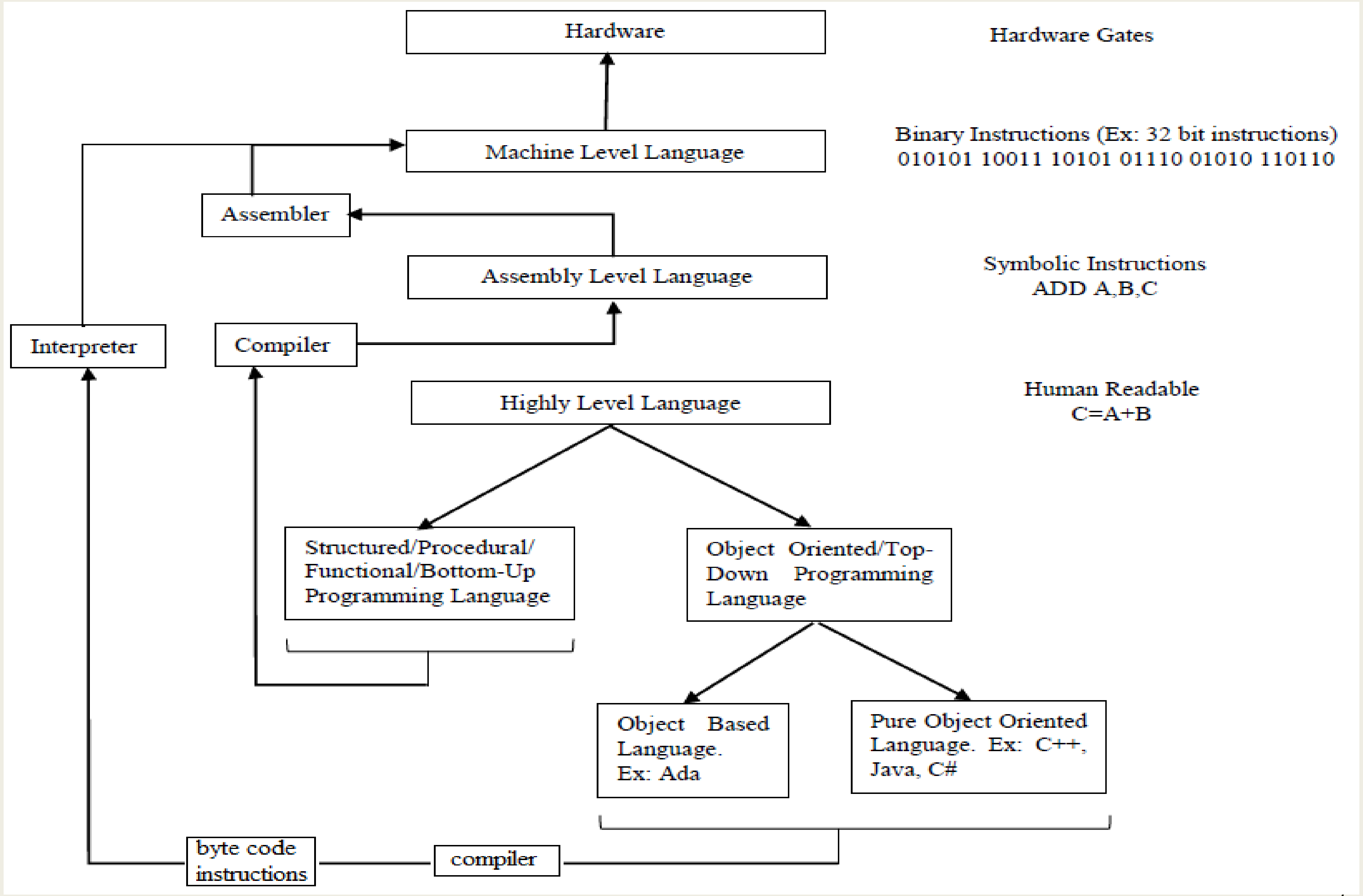
Introduction

Rathinaraja Jeyaraj

Introduction to Programming


Hardware
Hardware Gates
Machine Level Language
Binary Instructions (Ex: 32 bit instructions)
010101 10011 10101 01110 01010 110110
Assembler
Assembly Level Language
Symbolic Instructions
ADD A,B,C
Interpreter
Compiler
Highly Level Language
Human Readable
C=A+B
Structured/Procedural/ Functional/Bottom-Up Programming Language
Object Oriented/Top-Down Programming Language
Object Based Language. Ex: Ada
Pure Object Oriented Language. Ex: C++, Java, C#
byte code instructions
compiler

- Over the decades different programming languages have been developed to ease writing programs.

- Imperative programming - writing programs in English as statements line by line. Ex: C, C++, Java, PHP, Python, Ruby…

- Structured programming - A style of imperative programming with explicit control-flow (such as for, while, switch…) structures rather than jumping (go-to) directly from instruction to instruction. Ex: C, C++, Java, Python…

- Procedural programming - Derived from structured programming, based on the concept of modularity (define methods and call it from anywhere in the program). Ex: C, C++, Lisp, PHP, Python…

- Functional programming - basic building block is functions. Functions can be passed as argument, received as return values… unlike the above programming methods that use statements as the basic building blocks. Ex: Clojure, Elixir, Erlang, F#, Haskell, Lisp, Python, Ruby, Scala, SequenceL, SML…

- Object-oriented programming - every element/entity in the world is represented by data and methods. Ex: data for fan is: color, price, length… methods for fan are: rotation(), speed()… To access members (data and methods) of an entity, we need to create a handle (object) of a corresponding entity. Ex: C++, C#, Java, PHP, Python, Ruby, Scala…

- Declarative/Querying - defines computation logic without detailed control flow structures (for, while…). Ex: SQL, CSS, Prolog, OWL, SPARQL… File access loads entire file into memory, which wastes memory space. Query language is used on formatted data like RDBMS. Query (pre-defined function calls) retrieves only required data.

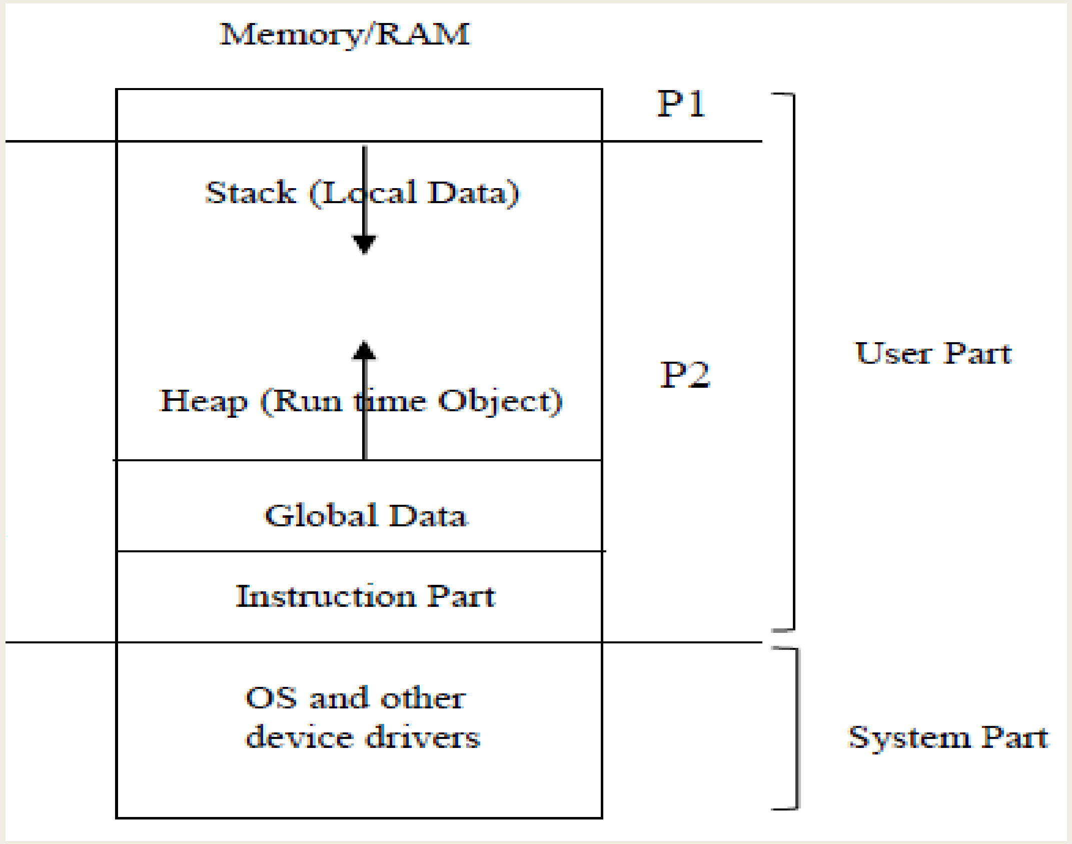
- Scripting language - it is not compiled, but interpreted on the fly at runtime. Scripting languages can be embedded within HTML to add functionality to a web page such as different menu styles or graphic displays. These types of languages are client-side scripting languages affecting the data that the end user sees in a browser window. Other scripting languages are server-side scripting languages that manipulate the data in a database. Ex: ASP, JSP, PHP, Perl, Python, Pig…

- Dataflow language - programming that helps to achieve execution in Directed Acyclic Graph model is called data flow language. Because, next level gets input only from previous level. Ex: Pig…

Basic terminologies

- **Physical** – components in real (memory/CPU/storage)
- **Logical** – group of physical components can be logically seen as one component. Ex: 4 computers, each with dual core, 4 GB memory, 1 TB storage can be said as a distributed system having 8 cores, 16 GB memory, and 4 TB storage.
- **Virtual** – software behaving like a hardware is called virtual in computer science. A software simulates all behavior of a physical component. Ex: Virtual Machines. There is no real hardware for Virtual Machines. But, hypervisor imitates all behavior of real hardware.
- **Element/Entity** – can be physical/logical/virtual component.
- **Program** – program is a passive collection of instructions stored in a file in secondary storage.
- **Computation** – a generic term that denotes sequence of calculations/activities. Process and computation are interchangeably used.
- **Event** – any computational element that gets input and produces output is called event. Collection of events is also called a program.

- **Object** – can be any event or data.
- **Process:** a process is an instance of a program that is being executed. It is talked with respect to the program loaded into memory. Figure shows structure of a process in memory.
- Basically, there are two logical divisions in memory. One is for user programs and another part is for system programs such as OS and other system software’s.
- User part holds user processes, say, P1, P2… each process has four logical parts.

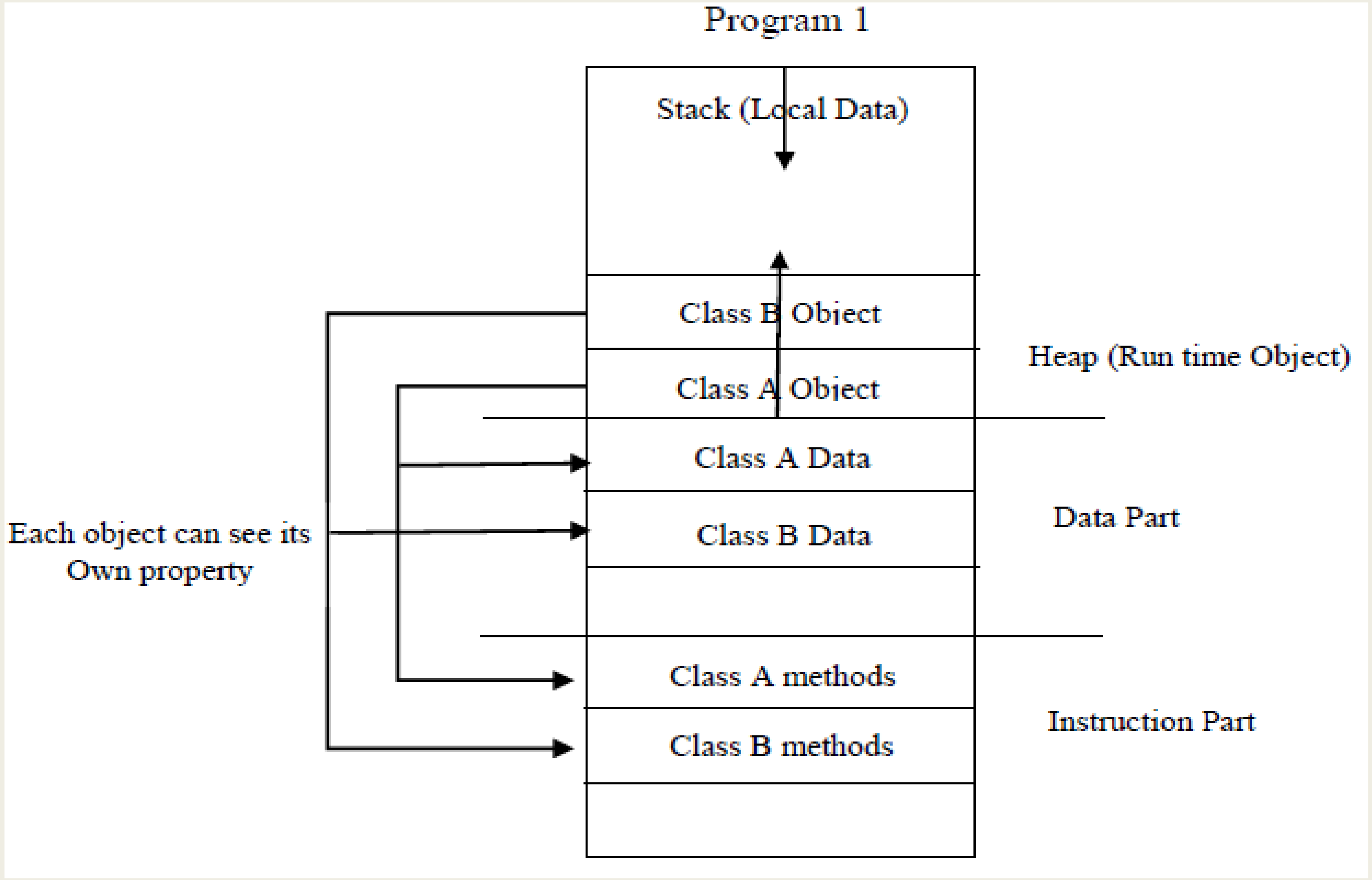


- Instruction part – all instructions of a program such as methods.
- Global data – data that are accessed by all parts of our program.
- Stack – data structure where local data of functions are loaded.
- Heap – run time objects are loaded in heap memory.

you can infer encapsulation, inheritance and abstraction of OOP. Ex: There are two classes A, B. Objects are created for them and loaded in heap. Object of class A can access only its methods and data. Class B can have access to class A properties using inheritance or by creating object for class A.

Object is an Abstract Data Type (**ADT**) which contain properties of a class in heap memory.

- **Software** – contains set of programs running together for a specific purpose, also called as application.
- **Application programmers/client:** application programmers write programs to access data from/to database by using queries.
- **End users:** end users use applications. Ex: check list of trains run between cities.


Program 1
Stack (Local Data)
Class B Object
Class A Object
Heap (Run time Object)
Class A Data
Class B Data
Data Part
Each object can see its Own property
Class A methods
Class B methods
Instruction Part