Welcome to "Little Bits to Big Ideas"

Lab 3: Hardware, Low-Level and High-Level Programming Languages

TAs: Ashish & Shivani

CAs: Abhinav, Harry, & Sherry

ILLINOIS

Understanding Computer Hardware

Overview:

Main Components of a Computer:

- **RAM (Random Access Memory):** Stores temporary data for quick access.
- **CPU (Processor):** Executes instructions and processes data.
- **Storage (HDD vs. SSD):** Long-term data storage, with differences in speed and performance.

Levels of Programming Languages

- **Low-Level Languages:**
  a. Machine Code: Direct instructions for the CPU.
  b. Assembly: Human-readable version of machine code, specific to processor architecture.

- **High-Level Languages:**
  a. C, Java, Python: More abstract, easier to read, compiled or interpreted.

Speed vs. Readability in Programming Languages

- **Why do some languages run faster than others?**
  - Compilation vs. Interpretation.
  - Level of abstraction.
  - Memory management.

- **Trade-offs:**
  - Speed vs. ease of development.
  - Performance vs. portability.

Abstraction

- **Two related parts to abstraction:**
  1. **Detail removal**
  2. **Generalization**

Detail removal : The act or process of leaving out of consideration one or more properties of a complex object so as to attend to others.

Generalization : The process of formulating general concepts by abstracting common properties of instances.

Genetic Programming

- One way of using abstraction:
  - Randomly pick several concrete instances of some generic set
  - Keep the one that does best at something
  - Randomly pick several concrete instances near the one we kept
  - Keep the one that does best at something
  - …
- You'll get to see this on one of the lab worksheet activities

Interface

An interface is how **different components communicate**.

For example, when you use a TV remote, you interact with buttons that send signals to the TV. You don’t need to understand how the electronics work inside.

Similarly, in programming, interfaces allow different parts of a system to work together without needing to understand each other’s internal code

Turing Completeness & Computation

- **What does it mean for a language to be Turing Complete?**

- **Why is this important for real-world applications?**

Turing Completeness & Computation

- **What does it mean for a language to be Turing Complete?**
  - Able to simulate any computational process given enough time and memory.

- **Why is this important for real-world applications?**
  - Ensures that a language can solve complex problems.

Applications of Programming Languages

Where do different languages shine?

- **C:** System programming, embedded systems.
- **Java:** Business applications, large-scale systems.
- **Python:** Data science, automation, “glue” between advanced packages (like AI, math).

How Do We Choose a Programming Language?

- **Key Factors:**
  1. **Performance Needs** – When is speed a priority?
  2. **Platform Compatibility** – Where will the software run?
  3. **Community & Libraries** – How does support impact development?
  4. **Ease of Learning & Use** – How does a language's syntax affect adoption?