CSC445 – Modern Programming Languages

Lecture 1

Types of Languages

Programming Skill

- Programming is a skill that is becoming increasingly sought after in the job market. Having at least a basic understanding of how software functions is helpful for anyone who interacts with technology.

- With a background in programming, you can get a job coding, designing software, data architecture, or creating intuitive user interfaces.

Major Types of Languages

- Procedural
- Functional
- Object Oriented
- Scripting

Procedural Languages

- A procedural language follows a sequence of statements or commands in order to achieve a desired output.
- Each series of steps is called a procedure, and a program written in one of these languages will have one or more procedures within it. Common examples of procedural languages include:
    - C and C++
    - Java
    - Pascal
    - BASIC

Functional Languages

- Rather than focusing on the execution of statements, functional languages focus on the output of mathematical functions and evaluations.
- Each function–a reusable module of code–performs a specific task and returns a result. The result will vary depending on what data you input into the function. Some popular functional programming languages include:
    - Scala
    - Erlang
    - F#

Object Oriented Language

- This type of language treats a program as a group of objects composed of data and program elements, known as attributes and methods. Objects can be reused within a program or in other programs. This makes it a popular language type for complex programs, as code is easier to reuse and scale. Some common object-oriented programming (OOP) languages include:
  - Java
  - Python
  - PHP
  - C++

Scripting Languages

- Programmers use scripting languages to automate repetitive tasks, manage dynamic web content, or support processes in larger applications. Some common scripting languages include:
  - PHP
  - Ruby
  - Python
  - bash
  - Perl
  - Node.js

Other Ways of Classifying Programing Languages

Font-end vs Back-end

- The Font End languages are primary concerned with the UI,
- The font-end deals with the text, colours, buttons, images and navigation that the user will face while navigating through applications or websites.
  - HTML
  - CSS
  - JavaScript
  - React

Font-end vs Back-end

- The Back-End languages with storage and manipulation of the server side of software.
- The user does not directly come into contact with this is the part of the software but supports their experience behind the scenes.
- This includes data architecture, scripting, and communication between applications and underlying databases.
  - Node, Python, Ruby, Java, C#, JavaScript

Hight Level vs Low Level

- Low Level Languages are machine friendly, which makes them highly efficient in terms of memory usage but difficult to understand.
- Hight Level Languages, on the other hand, are less memory efficient but much more human friendly. This makes them easier to write, understand, maintain, and debug.

Interpreted vs Compiled

- With interpreted languages the code goes through a program called interpreter, which reads and executes the code line by line.
- This tends to make these languages more flexible and platform independent.
  - Python
  - JavaScript
  - PHP
  - Ruby

Interpreted vs Compiled

- Compiled languages go through a build step where the entire program is converted into machine code.
- This makes it faster to execute, but it also means that you have to compile or "build" the program again anytime you need to make a change.
  - C, C++, C#
  - Java
  - Rust