This course examines basic concepts underlying the design of modern programming languages: types, control structures, abstraction mechanisms, inheritance, and constructs for programming. This course will include programming assignments in several languages.

**Imperative vs Functional Programming**

Imperative programming is a list of step-by-step instructions for the program to follow in order to execute. The programmer tells the computer exactly how to solve the problem. In functional programming, the programmer defines functions that are very similar to mathematical functions, defining what is computed, not how. For example, adding 1 to each element of a list or array is very different in a functional vs imperative language.

**Some languages used**

**Imperative:** Ada, C

**Object-Oriented:** Java, C++

**Functional:** Haskell, Lisp

**Scripting:** Bash, Perl

**Logic-based:** Prolog

**IMPERATIVE (C)**

```c
int x = 0;
while( x < arraySize ){
    array[x] = array[x] + 1;
    x = x + 1;
}
```

**FUNCTIONAL (Haskell)**

```haskell
map (1+) [LIST]
```