Functional Programming

PRESENTATION BY MICHAEL WOLLOWSKI

Key Programming Languages and their Inception Dates

FORTRAN	1957
LISP	1958
ALGOL	1958/60
SIMULA	1962
PROLOG	1972

Inner Product, von Neuman Style

```
public static void main (String [] args) {
        int[] a = {1,2,3};
        int[] b = {6,5,4};
        int sum = 0;
        for (int i = 0; i < a.length; i++) {
            sum += a[i] * b[i];
        }
        System.out.println(sum);
    }
}
```

Inner Product, Functional Style

```
;;; Look Ma, no variables (OK, so we have ;;;
parameters. Sue me!)
(define (innerProduct a b)
  (apply +
         (map (lambda (x)
                (* (car x) (cdr x)))
              (map (lambda (x y)
                     (cons x y)) v w))))

(define v `(1 2 3))

(define w `(6 5 4))
```

Functional Style

Consists of functional forms that combine existing functions to form new ones.

Von Neuman vs. Functional

Statements operate on an invisible state	Operates on arguments; no hidden state
Not hierarchical	Hierarchical; built from simpler functions
Must mentally execute it to understand it	Look at structure in order to understand it; no need to mentally execute it
Computes word-at-a-time by assignment and variables	Operates on conceptual units

Functional Programming Systems

An FP system comprises the following:

- A set of objects
- A set of functions
- Function application
- Functional forms, which are used to combine existing functions or objects to form new functions
- Definitions

Factorial

```
(define (fac n)
 (if (= n 1)
     1
     (* n (fac (- n 1)))))
```

```
public static int fac(int n){
    int fac = 1;
    for (int i = 1; i <= n; i++)
      fac *= i;
    return fac;
    }
```

Fibonacci

```
(define (fib n)
  (if (<= n 2)
      1
      (+ (fib (- n 1)) (fib (- n 2)))))
```

```
public static int fib(int n){
   int prev = 1;
   int current = 1;
   int temp = 0;
   for (int i = 2; i < n; i++) {
     temp = prev + current;
     prev = current;
     current = temp;
   }
   return current;
}
```