Functional Programming

- Why do we have it?
- What can you do with it?
- Why?

1. Clarity - write and maintain easier
    - BUT: program is a series of commands to be executed
    - loop - repeat the series
    - functions - shorthand for other series of commands
    - every solution must be a long thin string of commands
    - not every problem is most clearly solved this way
    - e.g.

```
int b = g(z);
int a = f(b) + 1;
...
int z - f(b) + 1;
```

    - is a == z? MAYBE
    - lacks referential transparency
        - references by identifiers (or names) to values are obvious
        - if you write the same expression twice you get the same value
2. Performance - we want our programs to run faster
    - to be more easily optimizable (compiler should do heavy lifting)

3 main types of programming:

- Imperative
- Functional
- Logical

C++ has functions and you can limit yourself to a functional style but it is awkward. It is better to learn the functional way of thinking

ex. Google used functional programming idea MAP-REDUCE to speed up query processing

Imperative

```
basic unit: statement S1 S2...
glue: S1; S2; S3;
```

relies on:

- variables with state, modified via assignment

Functional

```
basic unit: function F1 F2...
glue: F1(F2(x), F3(y, z), w)
```

relies on:

- functional evaluation providing partial order on computation
- referential transparency
- functional forms - functions that take other functions as arguments, or return other functions as results
  give up:
- assignment statements (no variables with state; variables have values but the values don't change)
- side effects - when a statement affects machine state either by assignment, 1/0, etc.

Q. Are there cases where F2 and F3 cannot be computer in parallel?
A. In a functional language no, in C/C++/FORTRAN/Java, yes (unless F2 and F3 are independent)

Basic Properties of OCaml

- compile time (static) type checking
  - types are checked before the program starts running
  - like: C/C++, Java...
  - unlike: python, sh...
- goal: reliability
- you don't need to write down types all the time
- less of a need to worry about storage management: no free or del, there is a garbage collector