Imperative Programming

x = 1;
y = 2;
z = x + y;
x = x + 12;
y = x + 2;
Procedural Programming

function calculate(int x, int y) {
    x = x + 12;
    y = y + 8;
    return x * y;
}
Before Objects

CHARACTER*8  NAME
INTEGER   AGE
Before Objects

    CHARACTER*8 CSNAME
    INTEGER CSAGE
Before Objects

C This is Customer Data

CHARACTER*8 CSNAME

INTEGER CSAGE
Records or Structs

01 CUSTOMER-RECORD
  05 NAME PIC A(10)
  05 AGE 999
Records or Structs

```c
struct customer {
    char* name;
    int    age;
    ...
}
```
Records or Structs

```c
struct customer {
    char* name;
    int age;
    ...
}

void updateAddress(struct customer cust)
void printInfo(struct customer cust) ...
```
class Customer {
    private String name;
    private int age;
    ...
    public void updateAddress() ...
    public void printInfo() ...
}
Logic Programming: Rules and Facts

• x is the grandparent of z if x is the parent of y, and y is the parent of z.
• x is an ancestor of y if x is the parent of y.
• x is the ancestor of z if y is the parent of z and x is an ancestor of y.

• Mary is the parent of Alice.
• John is the parent of Mary.
• Elizabeth is the parent of John.
Logic Programming: Questions

• Is Mary the grandparent of John?
• Is Elizabeth an ancestor of Alice?
Functions as First-Class Objects

sort(studentList);

sort(studentList, sortByLastName);
sort(studentList, sortByYearInSchool);

sortByYearInSchool = new Function(st1, st2) {
    return st1.year > st2.year;
};
Functions as First-Class Objects

currStudent = new Student("Jasmine");

increment = new Function(x) {
    x = x + 1;
};
No Side Effect vs. Side Effect

function increment(x) {
    return x + 1;
}

x = 1;

function increment() {
    x = x + 1;
}