CMPSCI 220 Programming Methodology

10: Introduction to Functional Programming

Based on Head First Design Patterns

What is functional programming?

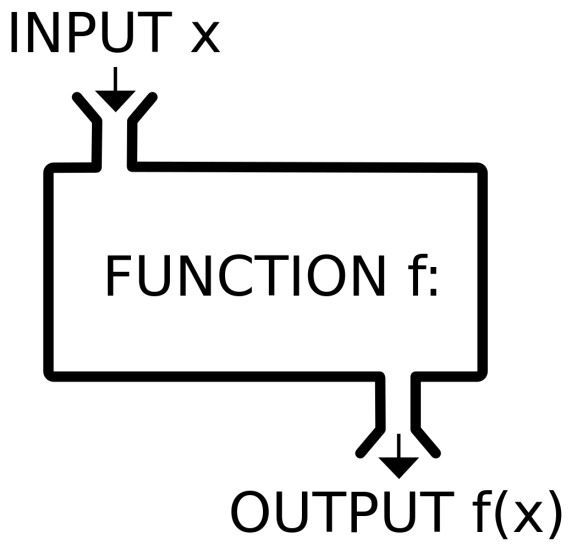
Functional Programming (FP)

The construction of programs using only
pure functions

What is a pure function?

What is a pure function?

A *pure function* is a function that…



has no side effects!

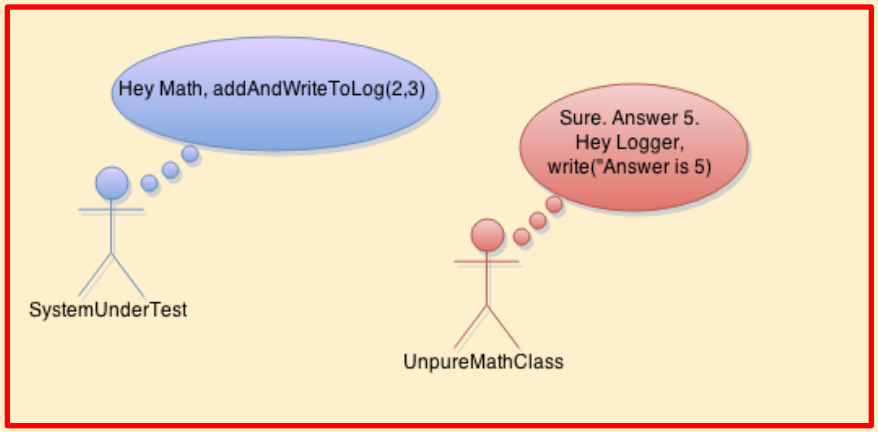
What is a side effect?

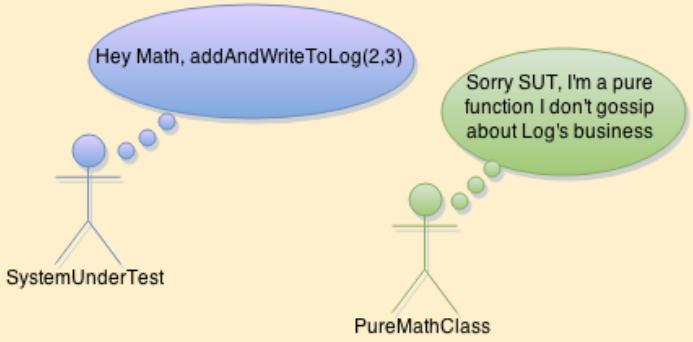
What is a side effect?

A function has a side effect if it does something other than simply returning a result.

- Modifying a variable
- Modifying a data structure in place
- Setting a field on an object
- Throwing an exception or halting with an error
- Printing to the console or reading user input
- Reading from or writing to a file
- Drawing on the screen

Pure Functions

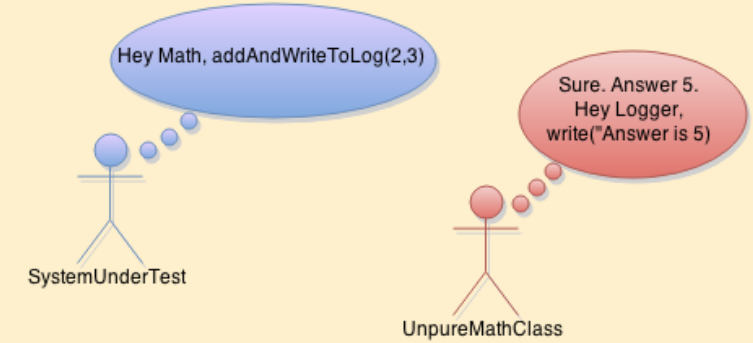


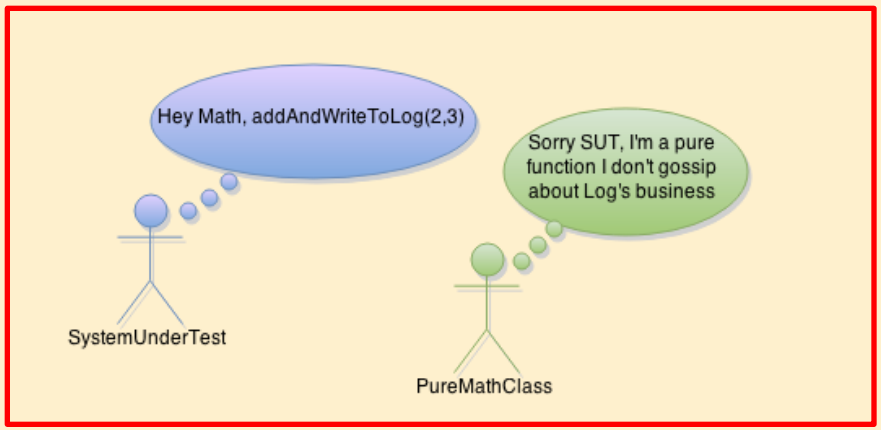


We must respect the pure functions for not gossiping around the corners of the System.

blog.adrianbolboaca.ro

Pure Functions





We must respect the pure functions for not gossiping around the corners of the System.

blog.adrianbolboaca.ro

Programs without side effects?

Consider what programming would be like without the ability to do these things…

Programs without side effects?

Consider what programming would be like without the ability to do these things…

Or, with significant restrictions on when and how these actions can occur…

Programs without side effects?

Consider what programming would be like without the ability to do these things…

Or, with significant restrictions on when and how these actions can occur…

Programs without side effects?

It may be difficult to

imagine…

Programs without side effects?

How is it even possible to write useful programs at all?

Programs without side effects?

How is it even possible to write useful programs at all?

If we can’t reassign variables,

Programs without side effects?

How is it even possible to write useful programs at all?

If we can't reassign variables,

- How would write simple programs with loops?

Programs without side effects?

How is it even possible to write useful programs at all?

If we can't reassign variables,

- How would write simple programs with loops?
- Handle data that changes?

Programs without side effects?

How is it even possible to write useful programs at all?

If we can't reassign variables,

- How would write simple programs with loops?
- Handle data that changes?
- Handle errors/exceptions?
- Performing IO?

The answer…

Functional programming is a restriction on *how* we write programs…

The answer…

Functional programming is a restriction on *how* we write programs…

But, not on *what* programs we can express.

The answer…

Functional programming is a restriction on *how* we write programs…

But, not on *what* programs we can express.

FP is a programming paradigm – a pattern in how you write programs without side effects.

What does FP give us?

FP is *tremendously* beneficial because of the increase in *modularity*.

Pure functions are easier to…

- Test
- Reuse
- Parallelize
- Generalize
- Reason About

What does FP give us?

FP is *tremendously* beneficial because of the increase in *modularity*.

Pure functions are easier to…

- Test
- Reuse
- Parallelize
- Generalize
- Reason About

} Much less prone to bugs!

A program with side effects

```
1 class Cafe {
2   def buyCoffee(cc: CreditCard): Coffee = {
3     val cup = new Coffee()
4     cc.charge(cup.price)
5     cup
6   }
7 }
```

Where is the side effect?

a) line 3
b) line 4
c) line 5
d) none of these

A program with side effects

```
1 class Cafe {
2   def buyCoffee(cc: CreditCard): Coffee = {
3     val cup = new Coffee()
4     cc.charge(cup.price)
5     cup
6   }
7 }
```

Where is the side effect?

a) line 3
b) line 4
c) line 5
d) none of these

A program with side effects

```
1 class Cafe {
2   def buyCoffee(cc: CreditCard): Coffee = {
3     val cup = new Coffee()
4     cc.charge(cup.price)
5     cup
6   }
7 }
```

Where is the side effect?

As a result of this side effect, the code is difficult to test.

Why?

A program that is better?

```
1 class Cafe {
2   def buyCoffee(cc: CreditCard, p: Payments): Coffee = {
3     val cup = new Coffee()
4     p.charge(cc, cup.price)
5     cup
6   }
7 }
```

This is a little better…Why?

But, we still have issues…

A program that is better?

```
1 class Cafe {
2   def buyCoffee(cc: CreditCard, p: Payments): Coffee = {
3     val cup = new Coffee()
4     p.charge(cc, cup.price)
5     cup
6   }
7 }
```

We could create a “mock” implementation, but we are likely to modify some internal state. Additionally, creating an interface just so we can test a function is *overkill*.

A program that is better?

```
1 class Cafe {
2   def buyCoffee(cc: CreditCard, p: Payments): Coffee = {
3     val cup = new Coffee()
4     p.charge(cc, cup.price)
5     cup
6   }
7 }
```

Furthermore, it is difficult to reuse **buyCoffee**.

Why?

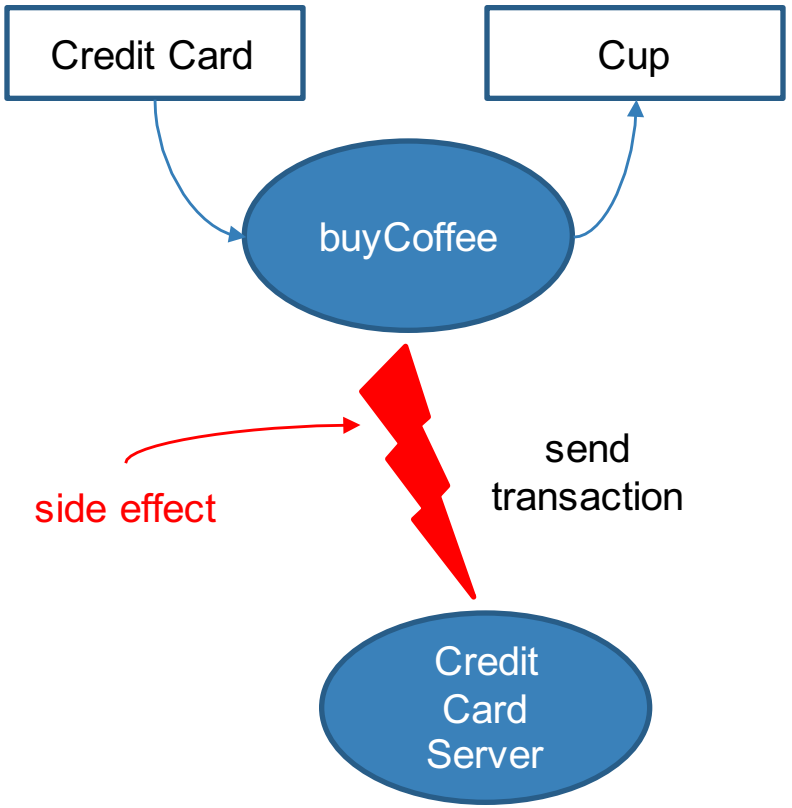
Imagine a customer orders 12 cups of coffee.
How might we reuse buyCoffee?

A program that is better?

```
1 class Cafe {
2   def buyCoffee(cc: CreditCard, p: Payments): Coffee = {
3     val cup = new Coffee()
4     p.charge(cc, cup.price)
5     cup
6   }
7 }
```

How can we fix this?

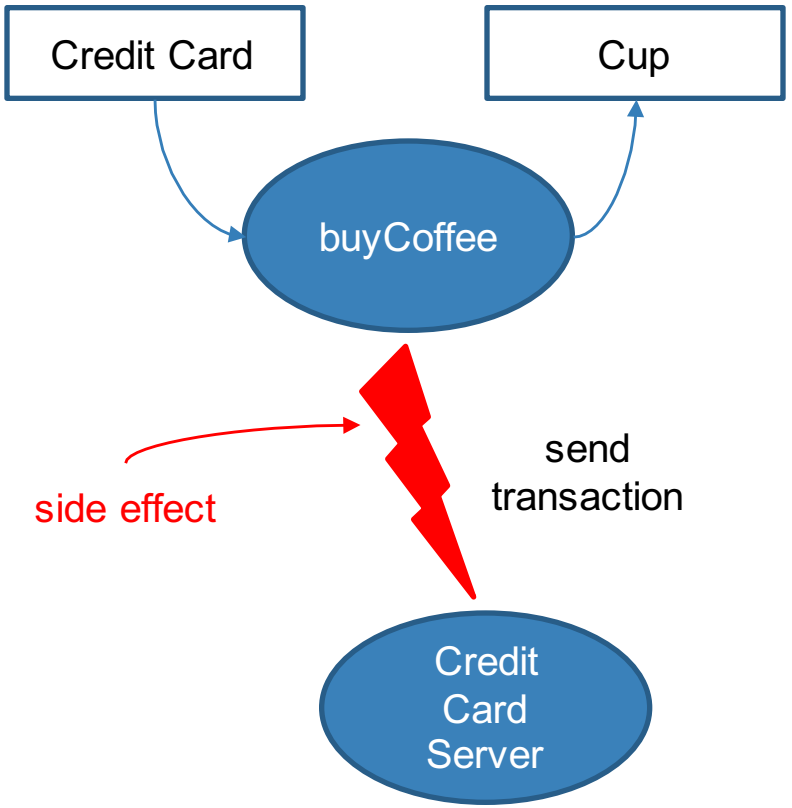
A program that is better?

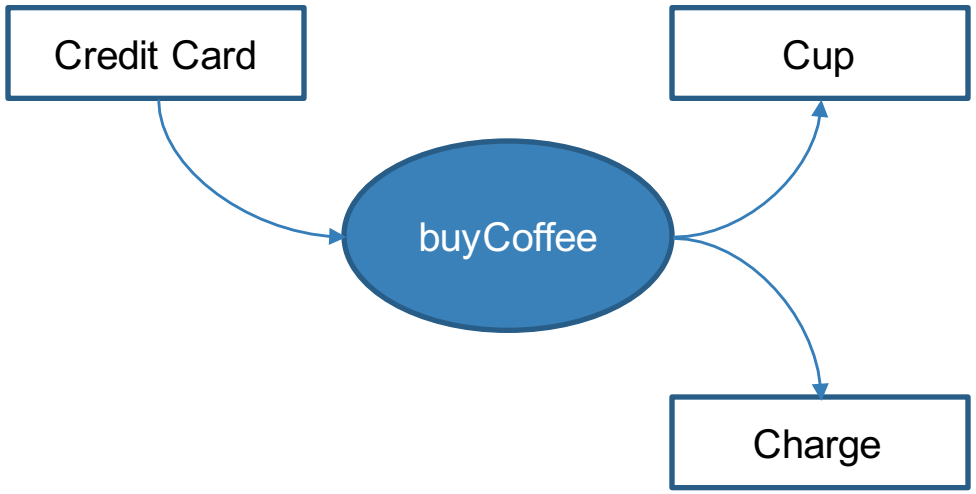


Can't test buyCoffee without credit card server.

Can't combine two transactions into one.

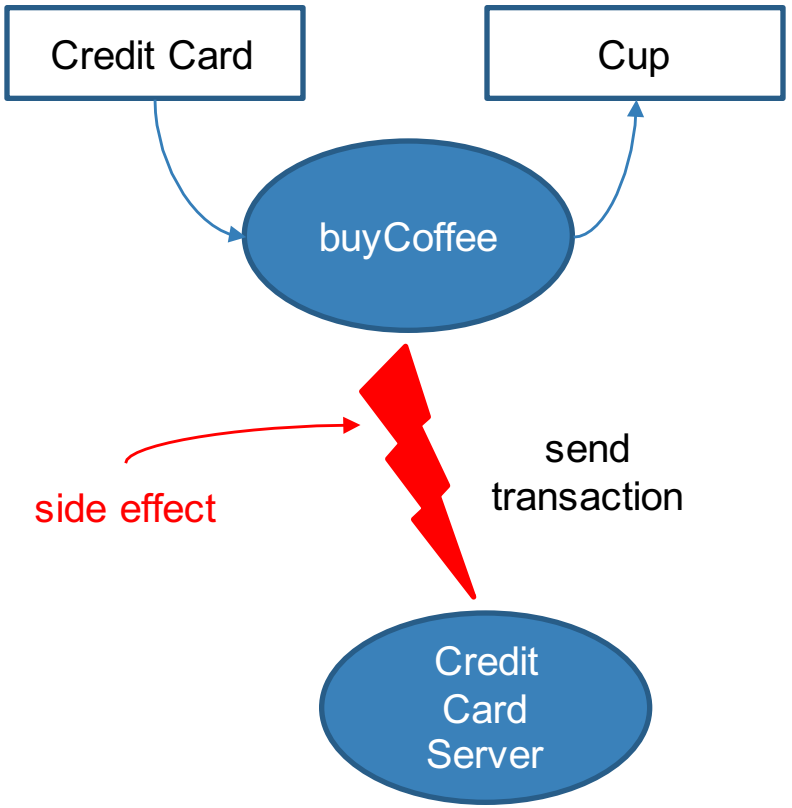
A program that is better?

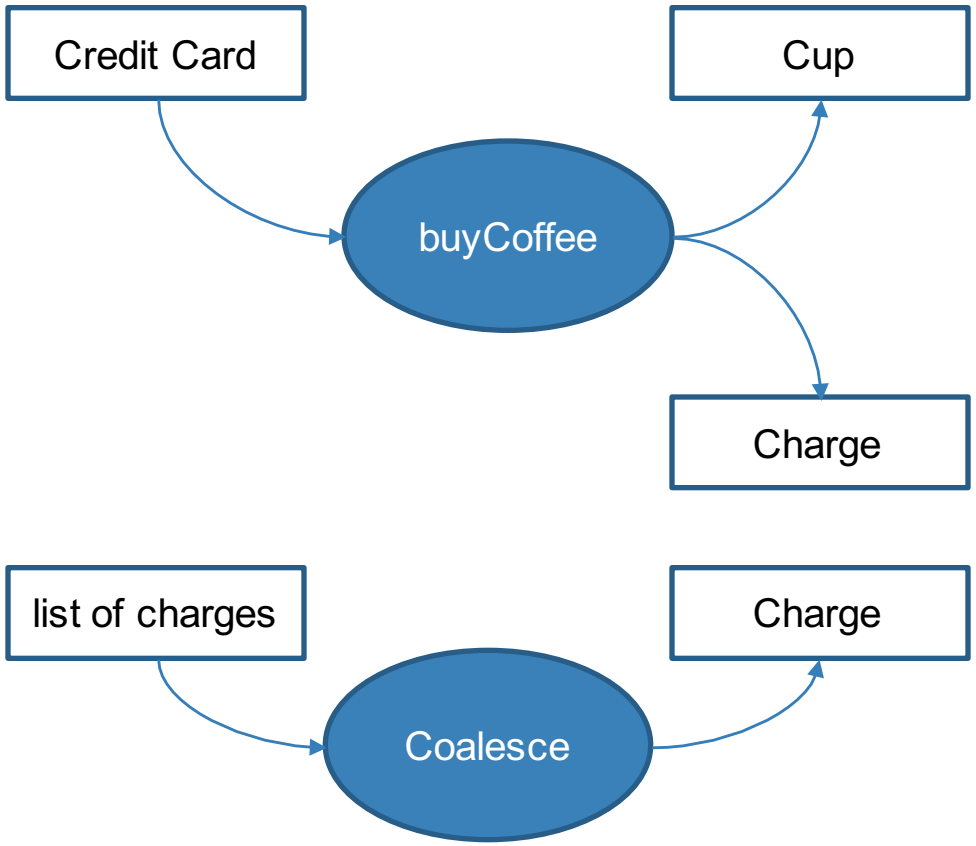




If buyCoffee returns a charge object instead of performing a side effect, then…

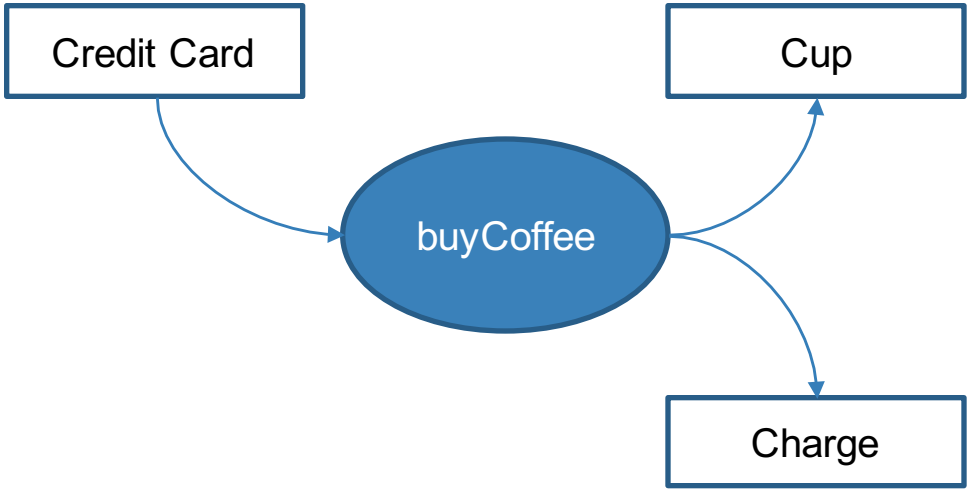
A program that is better?





The *functional* solution is to **eliminate** side effects

buyCoffee *returns the charge as a value* in addition to returning the Coffee.



A functional solution.

```
class Cafe {
  def buyCoffee(cc: CreditCard): (Coffee, Charge) = {
    val cup = new Coffee()
    (cup, Charge(cc, cup.price))
  }
}
```

buyCoffee now returns a pair of a Coffee and a Charge.

A functional solution.

```
class Cafe {
  def buyCoffee(cc: CreditCard): (Coffee, Charge) = {
    val cup = new Coffee()
    (cup, Charge(cc, cup.price))
  }
}
```

buyCoffee now returns a pair of a Coffee and a Charge.

Here we have separated the concern of creating a charge from the processing or interpretation of that charge. 0 side effects.

A functional solution.

```
class Cafe {
  def buyCoffee(cc: CreditCard): (Coffee, Charge) = {
    val cup = new Coffee()
    (cup, Charge(cc, cup.price))
  }
}
```

How can we reuse this version to easily purchase multiple coffees with a single transaction?

What is a Charge?

A Charge Class

```
case class Charge(cc: CreditCard, amount: Double) {
  def combine(other: Charge): Charge =
    if (cc == other.cc) Charge(cc, amount + other.amount)
  else
    throw new Exception("Can't combine different cards")
}
```

What is a **case** class?

A case class defines an algebraic data type.
It combines a class and a companion object into one.
Even better – it allows *pattern matching*.

Buying multiple cups…

```
class Cafe {
  def buyCoffee(cc: CreditCard): (Coffee, Charge) = …

  def buyCoffees(cc: CreditCard, n: Int): (List[Coffee], Charge) = {

    val purchases: List[(Coffee, Charge)] = List.fill(n)(buyCoffee(cc))

    val (coffees, charges) = purchases.unzip

    (coffees, charges.reduce((c1,c2) => c1.combine(c2)))
  }

}
```