Programming Languages

Higher-Order Functions

WashU CSE 4205
Prof. Dennis Cosgrove
#06: Tue, Sep 16, 2025

Office Hours starting to fill in

One Week Extension on All Exercises

S&Q: Higher-order functions seem polymorphic to me. I associate polymorphism with OOP. Should I associate higher-order functions with OOP or is that an FP thing.

V V V

Polymorphism (biology)

from the Greek for many shapes

the ability of different things to take on multiple forms or exist in different states

Light-morph jaguar

Dark-morph or melanistic jaguar (about 6% of the South American population)

Polymorphism (computer science)

note: a term with many forms. (you can't make this stuff up.)

- Ad hoc polymorphism: defines a common interface for an arbitrary set of individually specified types.
- Parametric polymorphism: when one or more types are not specified by name but by abstract symbols that can represent any type.
- Subtyping (also called subtype polymorphism or inclusion polymorphism): when a name denotes instances of many different classes related by some common superclass.

Polymorphism (computer science)

generics
SML: 'a list
Java: List<E>

note: a term with many forms. (you can't make this stuff up.)

- Ad hoc polymorphism: defines a common interface for an arbitrary set of individually specified types.
- Parametric polymorphism: when one or more types are not specified by name but by abstract symbols that can represent any type.
- Subtyping (also called subtype polymorphism or inclusion polymorphism): when a name denotes instances of many different classes related by some common superclass.

S&Q: Are there other programming flavors besides OOP and FP? Are OOP and FP the two main ones?

paradigms

- imperative
  - procedural (C)
  - object-based (wikipedia calls it object-oriented)
    - object-oriented (java) (wikipedia calls it Class-based object-oriented)
    - prototype-based (javascript)
- declarative
  - functional (SML)
  - logic
- constraint (prolog)
- others

S&Q: What's the difference between first-class functions and higher order functions?

- first-class function: function that can be used like any other value
- higher-order: a function that takes or returns other functions

S&Q: What is the relationship of anonymous functions to lambdas?

they are synonyms

S&Q: I still don't understand the difference between using function binding and anonymous functions. Why/when would anonymous functions be more beneficial?

- less noise

S&Q: To me, when you use anonymous functions you in some sense sacrifice readability in the name of conciseness. Do you have any thoughts on this?

there is a balance

- when is the lambda it too terse?
- when is the named function too much noise?

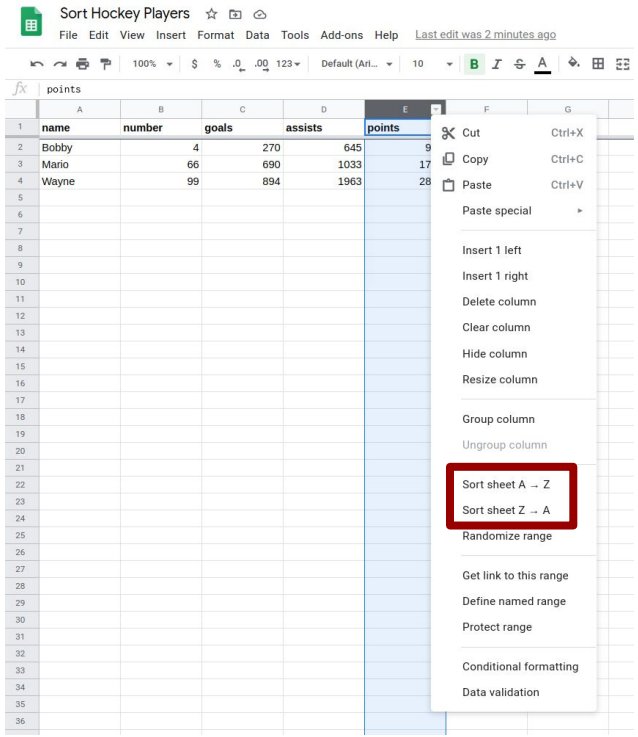
```
i = i + 1;
i += 1;
++i;
i++;
```

S&Q: Is it a good habit to take functions as arguments? I think it is confusing for both the coder and the other programmers who see the code.

- things generally seem confusing when you first see them
- map, filter, fold, find are very useful tools
- we seek code reuse (I often do this to a fault)

S&Q: can you maybe show us some more examples of higher-order and anonymous functions in class?

Sort Hockey Players



Sort

```
class Player {
    private final String name;
    private final int number;
    private final int goals;
    private final int assists;
    public Player(String name, int number, int goals, int assists) {
        super();
        this.name = name;
        this.number = number;
        this.goals = goals;
        this.assists = assists;
    }
    public String getName() {
        return name;
    }
    public int getNumber() {
        return number;
    }
    public int getGoals() {
        return goals;
    }
    public int getAssists() {
        return assists;
    }
    public int getPoints() {
        return getGoals() + getAssists();
    }
    @Override
    public String toString() {
        return "Player [name=" + name + ", number=" + number + ", goals=" + goals + ", assists="
+ assists + "]";
    }
}
```

```
Player mario = new Player("Mario", 66, 690, 1033);
Player wayne = new Player("Wayne", 99, 894, 1963);
Player bobby = new Player("Bobby", 4, 270, 645);
List<Player> players =
Arrays.asList(bobby, mario,
wayne);
Collections.sort(players, ???);
```

Sort Client

```
Player mario = new Player("Mario", 66, 690, 1033);
Player wayne = new Player("Wayne", 99, 894, 1963);
Player bobby = new Player("Bobby", 4, 270, 645);
List<Player> players = Arrays.asList(bobby, mario, wayne);

sortByAssists(players, true);

System.out.println();
System.out.println("sorted by assists low to high");
System.out.println("============================");
for (Player player : players) {
   System.out.println(player);
}
```

Why Not just implement Comparable<Player>???

even for Strings

A-Z	Z-A
case sensitive	case insensitive

When We Build Sort We Want It To Be Generally Useful

especially when we put a ton of effort into it:

- parallel merge sort
- dual pivot quicksort
- insertion sort

S&Q: What's the point of the Map function? Can you give an example of when I'd be used?

MapReduce

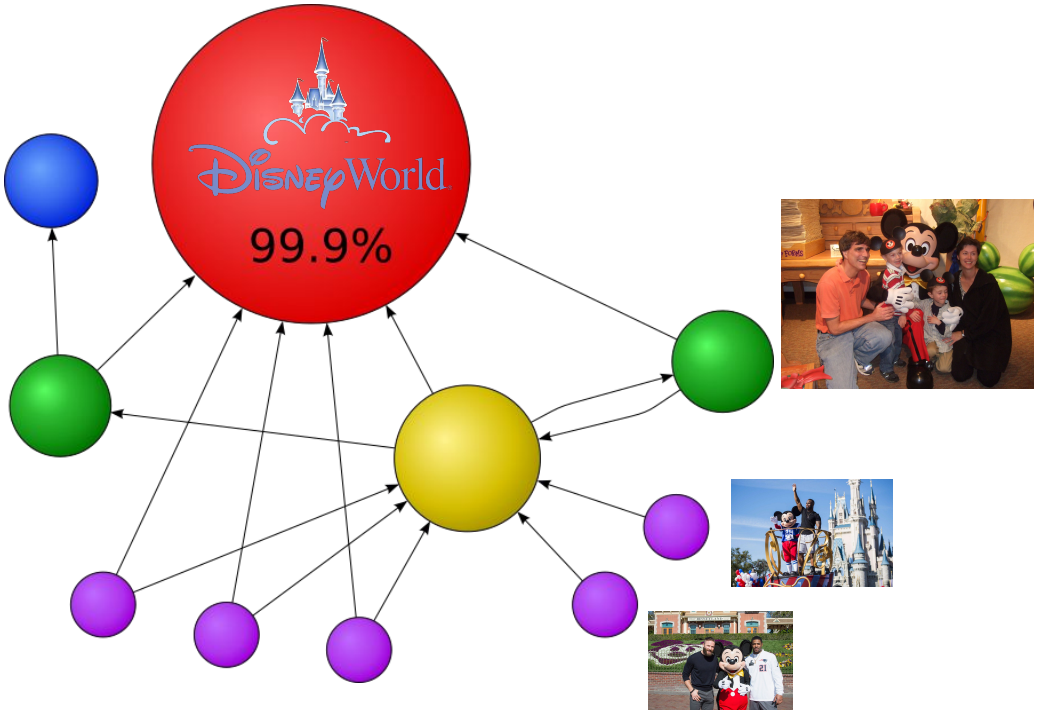
transforming your data can be powerful

https://en.wikipedia.org/wiki/PageRank

PageRank works by counting the number and quality of links to a page to determine a rough estimate of how important the website is. The underlying assumption is that more important websites are likely to receive more links from other websites.

*note: named after Larry Page not web page

Google: “Disney World”



S&Q: can you do parallel programming in sml? would you be able to teach 231 in sml (the anonymous functions are reminiscent of that course)? (note: multiple students asked this)

Yes and no

- https://www.cs.cmu.edu/~guyb/
- http://www.cs.cmu.edu/~15210/

aside: Haskell (a descendant of SML) has automatic and explicit parallelism

*** what do you need to have a race condition?

S&Q: member Dan mentioned all function have exactly one parameter. What would be the case when a function takes a variable and another function as input? Will this hold true?

that would be a tuple that pattern matching makes seem like multiple parameters

S&Q: Is it common across languages that lambdas have the limitation where they can't be recursive? I've never run into a situation where I was writing a lambda and wanted it to be recursive... is there a reason for this? Is it rare to want recursion in a place where you have a lambda?

V V V

S&Q: It was mentioned that anonymous functions can't be recursive, even when used in a val binding. Why is this? It seems like `val foo = fn () => foo()` should work just like `fun foo () = foo()`.

out of scope

S&Q: Are there any languages that could support recursive anonymous functions?

- you could add a keyword

*** what would be the verb equivalent for this or self?

Some candidates

generic verb: am or be

what you are doing: again or recurse

S&Q: Java doesn't have true first order functions. It has lambdas which take the place of annonymous classes, and you use a lambda to make an instance of an annonymous class that implements a functional interface. But in practice, is there any difference?

S&Q: Are there any languages that don't support passing in functions as arguments? If so, what is an example of one?

- Java

there has been reflection since the beginning, but that is not the same (you lose type checking, for starters)

JDK8 added lambdas and method references (which blurs the lines),
but you still have to declare a parameter or variable declaration as an interface with one abstract method

S&Q: What are the drawbacks of allowing functions to be passed in and bound as values?

*** Why did the designers of Java remove first class functions?

S&Q: What are the drawbacks of allowing functions to be passed in and bound as values?

*** Why did the designers of Java remove first class functions?

Some thoughts:

- Did they overdo it when cleaning up a hoarder's house???
- Fell in love with nouns so much that they started hating verbs???
- Prevent mistakes?
  - the SML parser will frustratingly grab the function as your value instead of the result of calling the function (not a problem for Lisp/Racket)

my_length to_squares(xs)

S&Q: Is functional programming always at odds with object oriented style code?

I know JavaScript is a weird middleground language, but you can do functional styled things in js. You can pass functions in as parameters and do similar function bindings.

- no
- they do tend to break down problems from a different angle
  (we will cover this more later in this course)

S&Q: Is encapsulation something that is valued in functional programming languages? It seems that closures may violate that, but it could also be due to my misunderstanding of closures/functional programming

- encapsulation (specifically) is an OOP term
- however, the value of limiting access crosses all boundaries

closures gather the values of what a function can access, which is different (we will discuss modules later)

ToSquares with HOF

```
fun to_squares(xs : int list) : int list =
    raise Fail "NotYetImplemented"
```

Sum Distances Using HOF

```
fun sum_distances_to_origin(xys) =
    raise Fail "NotYetImplemented"
```

S&Q: Is there a similar predefined function for filter in java?

Stream

- map
- filter

HOF:

```
public interface Stream<T>
extends BaseStream<T,Stream<T>>
```

A sequence of elements supporting sequential and parallel aggregate

```
int sum = widgets.stream()
                 .filter(w -> w.getColor() == RED)
                 .mapToInt(w -> w.getWeight())
                 .sum();
```

S&Q: What are some other hall of fame higher order functions?

- fold (a.k.a. reduce)
- find

S&Q: I was confused by Dr. Grossman's explanation of the type t1 -> t2 -> t3 -> t4 and how it is equivalent to t1 -> (t2 -> (t3 -> t4)). Could you give a more concrete example?