


The Beauty and Joy of Computing

Lecture #2 Functions


Chip Design Savings from Functional Language?

A new functional language (you'll learn what that means today) "makes it possible a chip design transformation at completely zero-cost". Huge potential cost-savings!
"Functional Programming is not part of the standard curriculum".
It is alive and well in Snap!, yay!

www.utwente.nl/en/news-events/1/2015/1/167544/massive-chip-design-savings-to-be-realized




UC Berkeley "The Beauty and Joy of Computing" : Functions (2)



When Do You Learn Things in CS10?

- Lecture**
 - Computing in the News + Discussion
 - Big ideas, Inspiring Introductions, Demos
 - NOT THE CODING DETAILS**
- Lab, Homework, Projects**
 - Coding, Collaboration, Deep Learning
- Reading**
 - Context, Impact of Computing, Current Events
- Discussion**
 - Clarify week's material, Unplugged Activities

UC Berkeley "The Beauty and Joy of Computing" : Functions (2)




Office Hours and Discussions

- You can go to as many office hours and discussions as you wish!**
 - You're not just limited to your TA
- Please check the schedule on cs10.org, as that will have the currently correct times.**
- My OH – right after lecture on Tues (5-6pm), before lecture on Thurs (3-4pm)**

UC Berkeley "The Beauty and Joy of Computing" : Functions (3)



Function Basics




Abstraction: Generalization

REVIEW

- You are going to learn to write functions, like in math class:**


$$y = \text{plus5}(n)$$

- plus5 is the function
- n is the input, a number
- It returns a single value, here a number 5 more than the input.




plus5
Function machine
(Simply Scheme, Harvey)

UC Berkeley "The Beauty and Joy of Computing" : Functions (5)




Functions in 2nd Grade Math Curricula!



UC Berkeley "The Beauty and Joy of Computing" : Functions (6)

Function Definition

- Functions take in 0 or more inputs, return exactly 1 output
- The same inputs MUST yield same outputs.
 - Output function of input only
- Other rules of functions
 - No state (prior history)
 - No mutation (no variables get modified)
 - No side effects (nothing else happens)



Function Metaphor
(CS Illustrated, Katrina Yim)

UC Berkeley "The Beauty and Joy of Computing" : Functions (7)

Which is NOT a function?

- pick random to
- $x < y$
- length of
- sqrt of
- true

UC Berkeley "The Beauty and Joy of Computing" : Functions (8)

Data Types

Domain & Range

Basic Data Types (You'll make more)

Sentences	<ul style="list-style-type: none"> Words separated by N spaces, $N \geq 0$ E.g., CS 10 is great
Word	<ul style="list-style-type: none"> Length ≥ 1, no spaces E.g., Cal, 42, CS10
Character	<ul style="list-style-type: none"> Length = 1 E.g., A, 3, #
Letter	<ul style="list-style-type: none"> A-Z, a-z only E.g., h

UC Berkeley "The Beauty and Joy of Computing" : Functions (10)

Domain and Range (from Math)


Domain	Range
<ul style="list-style-type: none"> The "class" of input a function accepts 	<ul style="list-style-type: none"> All the possible return values of a function
Examples <ul style="list-style-type: none"> Sqrt of <ul style="list-style-type: none"> Non-negative numbers Length of <ul style="list-style-type: none"> Sentence, word, number $_ < _$ <ul style="list-style-type: none"> Sentence, word, number $_ \text{ and } _$ <ul style="list-style-type: none"> Boolean 	Examples <ul style="list-style-type: none"> Sqrt of <ul style="list-style-type: none"> Non-negative numbers Length of <ul style="list-style-type: none"> Non-negative integer $_ < _$ <ul style="list-style-type: none"> Boolean (true or false) $_ \text{ and } _$ <ul style="list-style-type: none"> Boolean (true or false)

UC Berkeley "The Beauty and Joy of Computing" : Functions (11)

Types of Blocks in Snap!

Procedures, Subroutines

- Command**
 - No outputs, meant for side-effects
 - Not a function...
- Reporter (Function)**
 - Any type of output
- Predicate (Function)**
 - Boolean output
 - (true or false)



UC Berkeley "The Beauty and Joy of Computing" : Functions (12)

Domain, Range of...

letter of

Domain	Range
a) Integer ≥ 1 , Number	Digit
b) Integer ≥ 1 , Word	Letter
c) Integer ≥ 1 , Word	Character
d) Integer ≥ 1 , Sentence	Letter
e) Integer ≥ 1 , Sentence	Character

UC Berkeley "The Beauty and Joy of Computing" : Functions (3)

Why Should You Use Functions?

Why Use Functions? (1/3)

UC Berkeley "The Beauty and Joy of Computing" : Functions (5)

Why Use Functions? (2/3)

- They allow for generalization of code!
- The building blocks of our programs
- They can be composed together to make even more magnificent things.
- Breaking big problems down into smaller ones is functional decomposition

UC Berkeley "The Beauty and Joy of Computing" : Functions (6)

Why Use Functions? (3/3)

en.wikipedia.org/wiki/Functional_programming

- If a function only depends on the information it gets as input, then nothing else can affect the output
 - It can run on any computer and get the same answer.
- This makes it easy to parallelize computation.
 - Functional programming is a great model for writing software that runs on multiple systems at the same time.

UC Berkeley "The Beauty and Joy of Computing" : Functions (7)

Quick Preview: Recursion

Recursion is a technique for defining functions that use themselves to complete their own definition.

This is one of our Big Ideas!

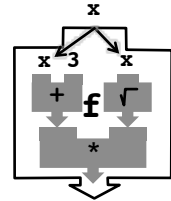
UC Berkeley "The Beauty and Joy of Computing" : Functions (8)

Functions Demo!



Functions Summary

- **Abstraction: Generalization** $f(x) = (x+3) * \sqrt{x}$
- **Computation is the evaluation of functions**
 - Plugging pipes together
 - Function: ≥ 0 inputs, 1 output
 - Functions can be input!
- **Features**
 - No state
 - E.g., variable assignments
 - No mutation
 - E.g., changing variable values
 - No side effects
 - E.g., nothing else happens



UC Berkeley "The Beauty and Joy of Computing" : Functions (20)

