Algorithms and Data Structures for Data Science
lab_recursion

CS 277

March 3, 2023

Brad Solomon

UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN

Department of Computer Science

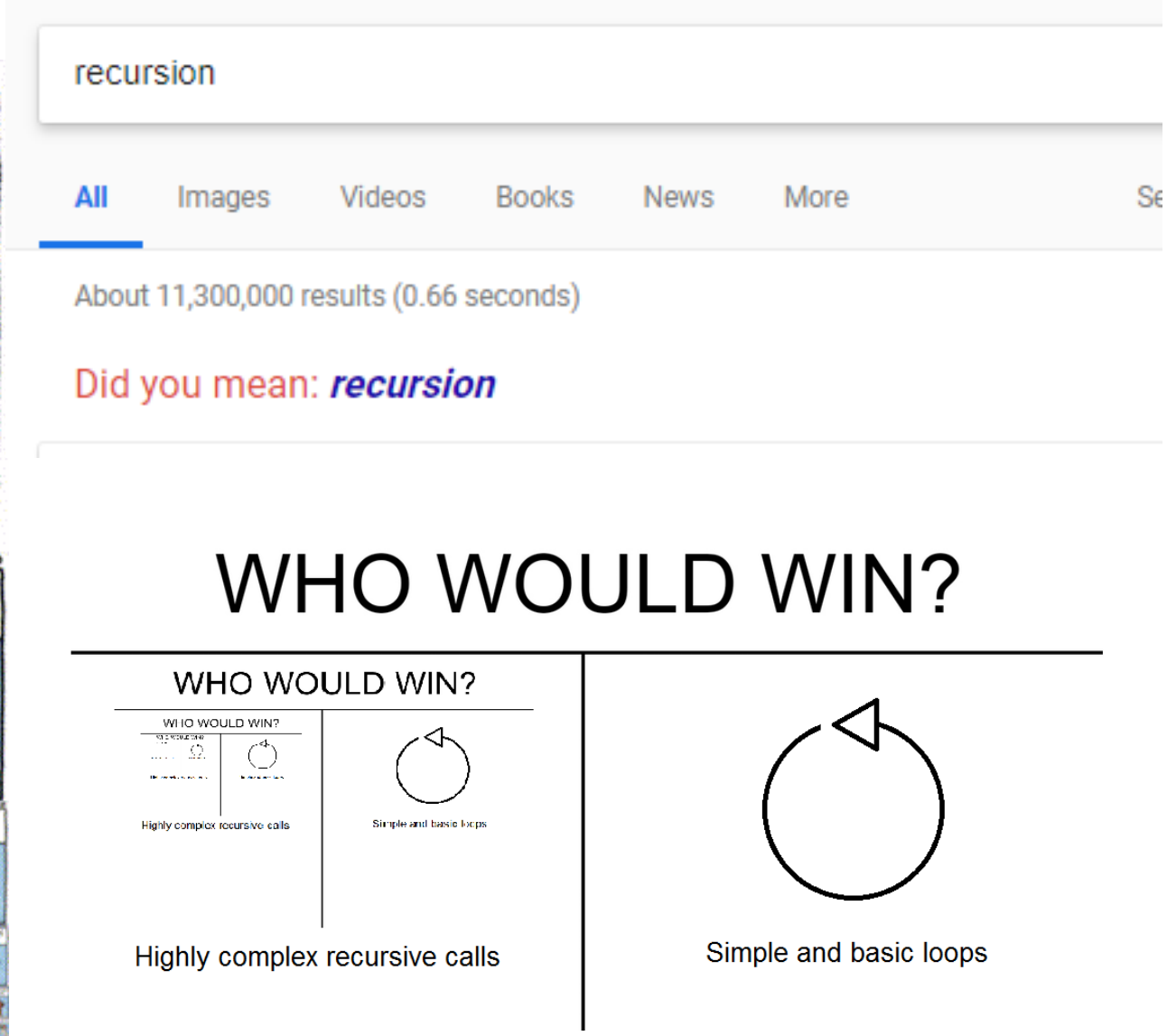
Learning Objectives

Review fundamentals of recursion

Implement recursive functions to handle a variety of tasks

Recursion


SAFELY ENDANGERED
SWEET JESUS, POOH! THAT'S NOT HONEY
YOU'RE EATING RECURSION


recursion
All
Images
Videos
Books
News
More
About 11,300,000 results (0.66 seconds)
Did you mean: recursion
WHO WOULD WIN?
WHO WOULD WIN?
Highly complex recursive calls
Simple and basic loops
Highly complex recursive calls
Simple and basic loops

Recursion

The success or failure of this lab (and the time it takes you) depends on your ability to answer the following:

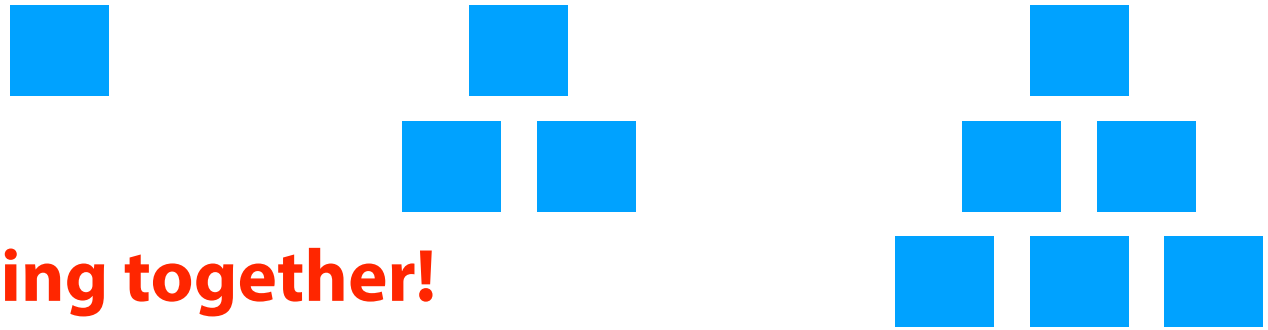
Base Case: What is the smallest sub-problem? What is the trivial solution?

Recursive Step: How can I reduce my problem to an easier one?

Combining: How can I build my solution from recursive pieces?

Start on your own but ask for help if you can't answer these questions!

Triangle



Lets practice by discussing together!

Given the height of a triangle, how many total blocks were used to make it?

Base Case:

Recursive Step:

Combination Step:

Each exercise a fun new twist!

Sum of Digits:

Triangle:

Palindrome:

List Partitioning:

Recursive List Partitioning

Using all elements in a list, can we make two lists which have equal sums?

Base Case:

Recursive List Partitioning

Using all elements in a list, can we make two lists which have equal sums?

Recursive Step:

Recursive List Partitioning

Using all elements in a list, can we make two lists which have equal sums?

(New) Base Case:

Recursive List Partitioning

Using all elements in a list, can we make two lists which have equal sums?

Combination Step:

Recursive List Partitioning

Using all elements in a list, can we make two lists which have equal sums?

Recursive Helper Function

```
1  def can_partition(number_list):
2      return False
3
4
5
6
7
8
9  def partition_helper(number_list, leftList, rightList):
10     pass
11
12
```

Using all elements in a list, can we make two lists which have equal sums?

Input

```
[4, 3, 1]                    ([], [])

[3, 1]        ([4], [])                    ([], [4])

[1]  ([3, 4], [])  ([4], [3])      ([3], [4])    ([], [3, 4])

[]
([1, 3, 4], [])  ([1, 4], [3])   ([1, 3], [4])  ([1], [3, 4])
([3, 4], [1])    ([4], [1, 3])   ([3], [1, 4])  ([], [1, 3, 4])
```