Lecture 15: Simulation and program design

Instructor: Michael Szell
Oct 25, 2023

Welcome back! Today we start "Program Design"

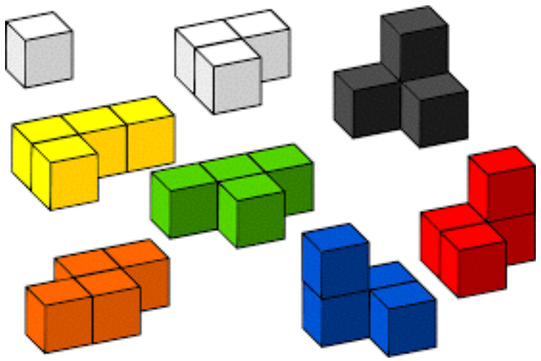
Lecture	Instructor	Time	Date	Day	Place	Topic
1	A	10:00-12:00	Aug 30	Wed	Aud 2	Introduction
2	A	10:00-12:00	Sep 1	Fri	Aud 2	
3	A	10:00-12:00	Sep 6	Wed	Aud 2	
4	A	10:00-12:00	Sep 8	Fri	Aud 2	
5	A	10:00-12:00	Sep 13	Wed	Aud 2	Python Crash Course
6	A	10:00-12:00	Sep 15	Fri	Aud 2	
7	A	10:00-12:00	Sep 20	Wed	Aud 2	
8	A	10:00-12:00	Sep 22	Fri	Aud 2	
9	A	10:00-12:00	Sep 27	Wed	Aud 2	
10	A	10:00-12:00	Sep 29	Fri	Aud 2	
11	M	10:00-12:00	Oct 4	Wed	Aud 2	
12	M	10:00-12:00	Oct 6	Fri	Aud 2	Exploratory Data Analysis
13	M	10:00-12:00	Oct 11	Wed	Aud 2	
14	M	10:00-12:00	Oct 13	Fri	Aud 2	
Holiday week		Holiday week				
15	M	10:00-12:00	Oct 25	Wed	Aud 2	
16	M	10:00-12:00	Oct 27	Fri	Aud 2	Program Design
17	M	10:00-12:00	Nov 1	Wed	Aud 2	
18	M	10:00-12:00	Nov 3	Fri	Aud 2	
19	M	10:00-12:00	Nov 8	Wed	Aud 2	
20	M	10:00-12:00	Nov 10	Fri	Aud 2	Networks & Skewed Data
21	M	10:00-12:00	Nov 15	Wed	Aud 2	
22	M	10:00-12:00	Nov 17	Fri	Aud 2	
23	M	10:00-12:00	Nov 22	Wed	Aud 2	
24	M	10:00-12:00	Nov 24	Fri	Aud 2	Data Science in Practice
25	M	10:00-12:00	Nov 29	Wed	Aud 2	
26	M	10:00-12:00	Dec 1	Fri	Aud 2	
27	M	No lecture	Dec 6	Wed	-	Mock Exam
28	M	10:00-12:00	Dec 8	Fri	Aud 2	
		09:00-13:00	Jan 3	Wed	TBD	Final Exam

We are here

Today you will learn about good program design

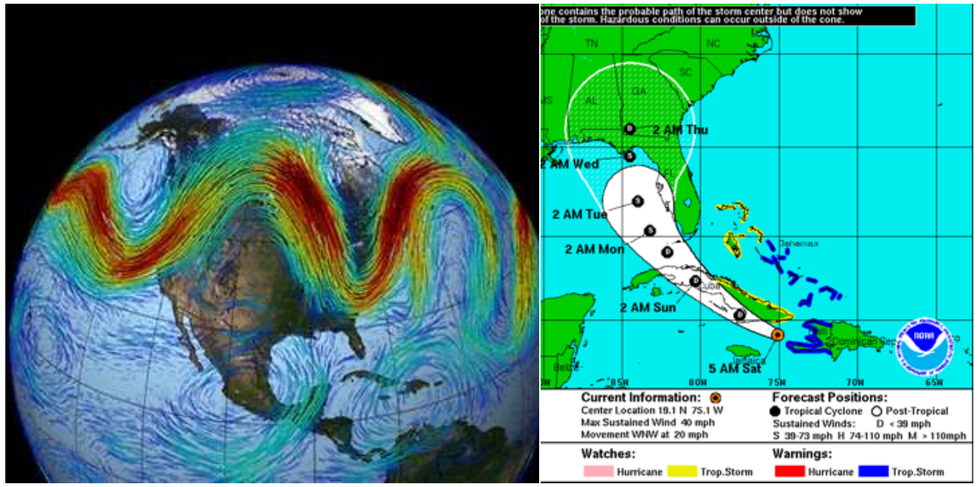
How to use random numbers

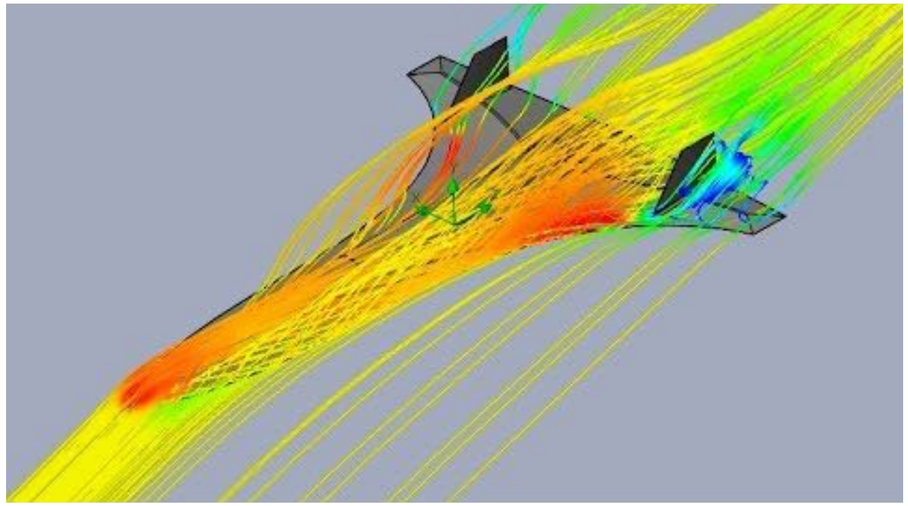
Unit testing

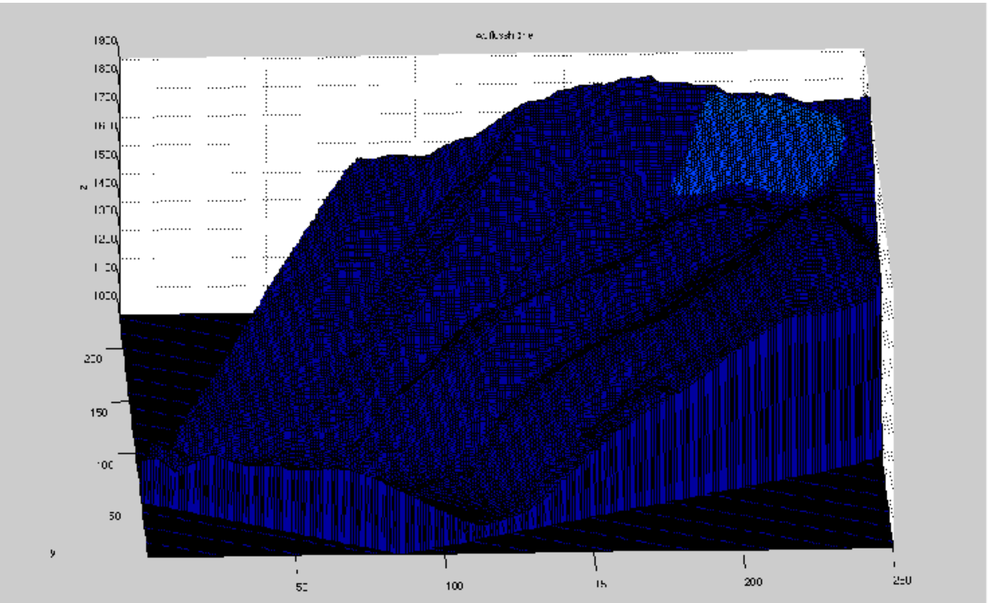


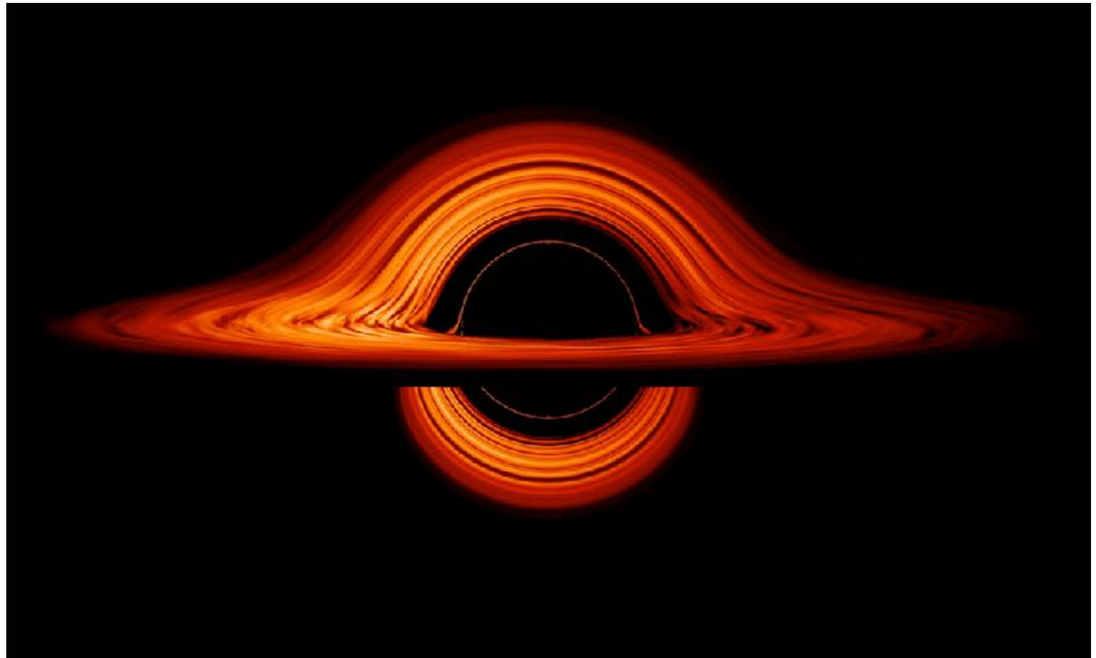
Top down design of raquetball simulation

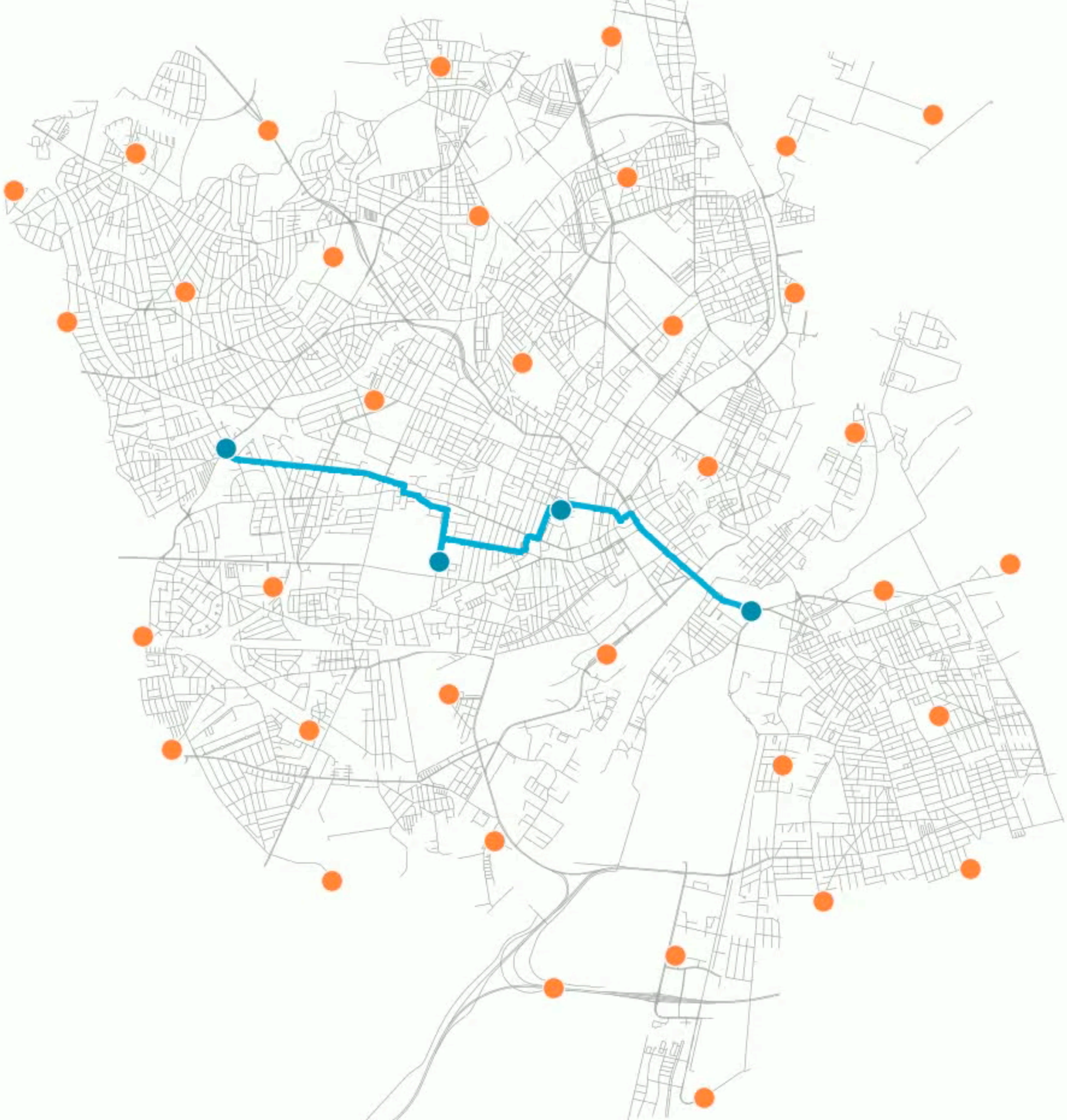
Simulation can give you an otherwise unobtainable understanding of real-world problems







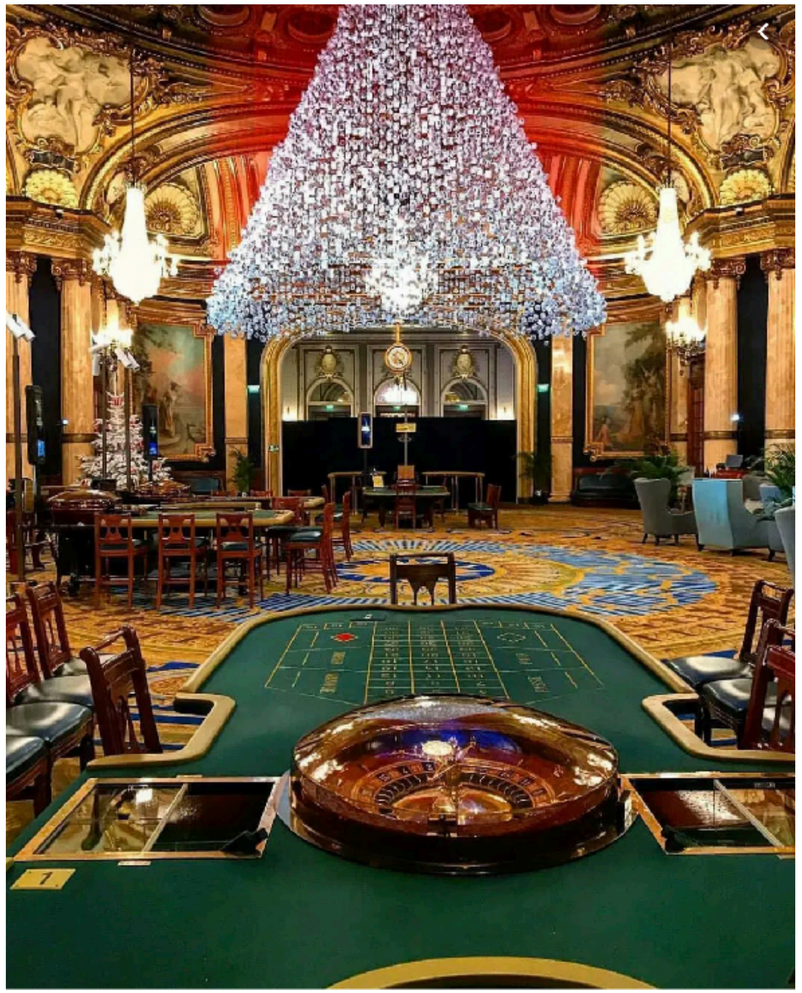




Simulation can create virtual worlds for entertainment or marketing


E-FORCE
RENOVART
ICODER
GEARBOX
geep
COSTA RICA
NUTRI SNACKS
Rumba
OLYMPIC CHANNEL
LONG 4
MART 0

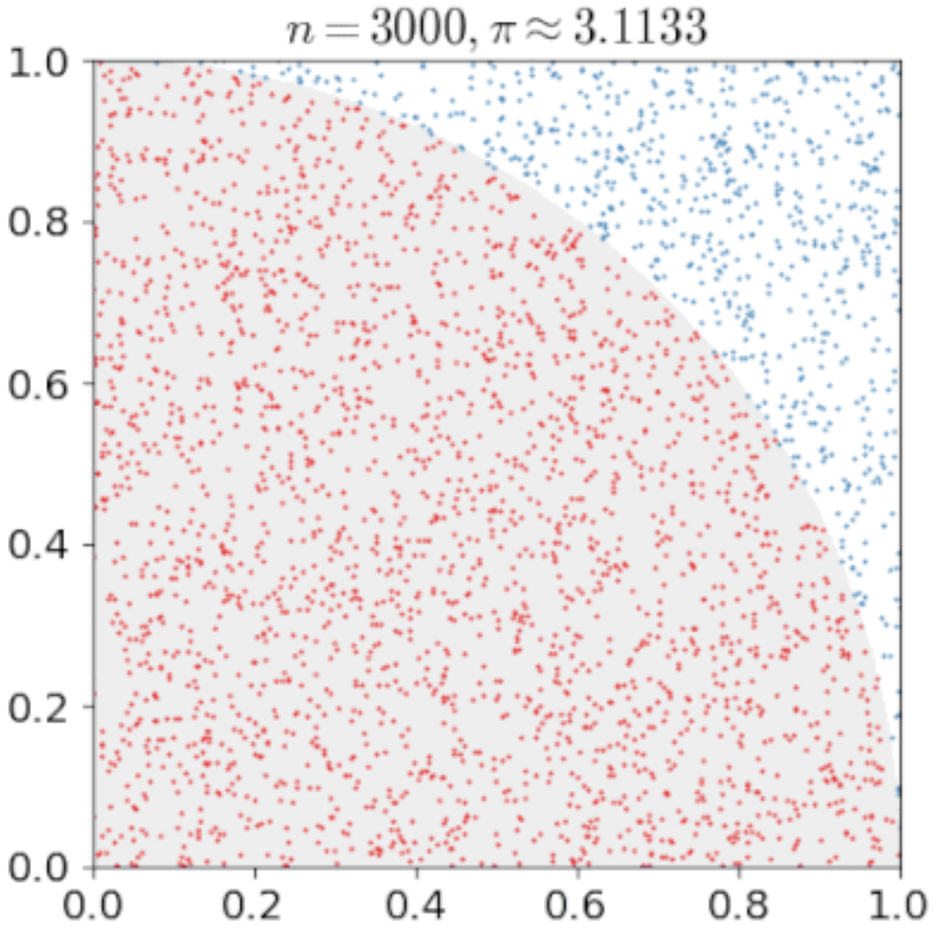
To simulate uncertain events, we generate random numbers



Monte Carlo methods are computational algorithms that rely on repeated random sampling to obtain numerical results.

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

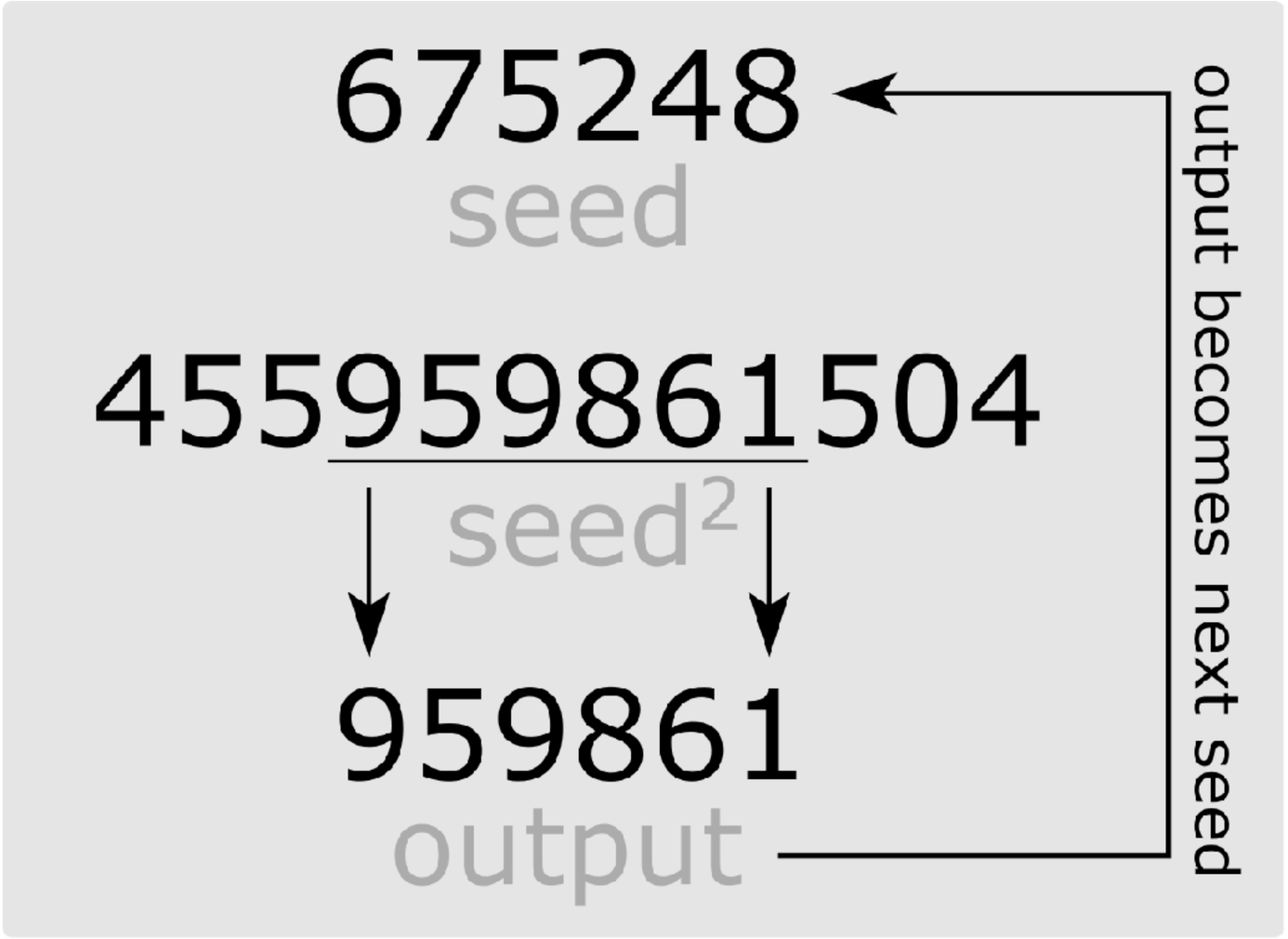
To simulate uncertain events, we generate random numbers



Monte Carlo methods are computational algorithms that rely on repeated random sampling to obtain numerical results.

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

Pseudorandom numbers are often good enough



https://en.wikipedia.org/wiki/Middle-square_method

If you need better randomness, measure nature

Lava lamps

https://www.cloudflare.com/learning/ssl/lava-lamp-encryption/

Atmospheric noise

random.org

Dice-O-Matic

https://www.youtube.com/watch?v=7n8LNxGbZbs

Use random to generate random numbers in Python

```
>>> from random import randrange
>>> randrange(1,6)
A number from [1,2,3,4,5]
```

random.randrange for integers

Use random to generate random numbers in Python

```
>>> from random import random
>>> random()
A number from [0,1)
```

random.random for floats

Use random() to decide with a given probability

```
prob = 0.7

if random() < prob:
    # 70% this happens
else:
    # 30% this happens
```

Use choice() to select a random element

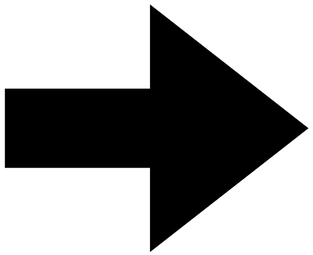
```
>>> from random import choice
>>> list = [100,200,300,400,500,600]

>>> random_item = choice(list)
A random item from the list
```

The seed makes your simulation reproducible

```
from random import *
seed(10)
print(random())
print(random())
print(random())

seed(10)
print(random())
print(random())
print(random())
```



```
0.5714025946899135
0.4288890546751146
0.5780913011344704

0.5714025946899135
0.4288890546751146
0.5780913011344704
```

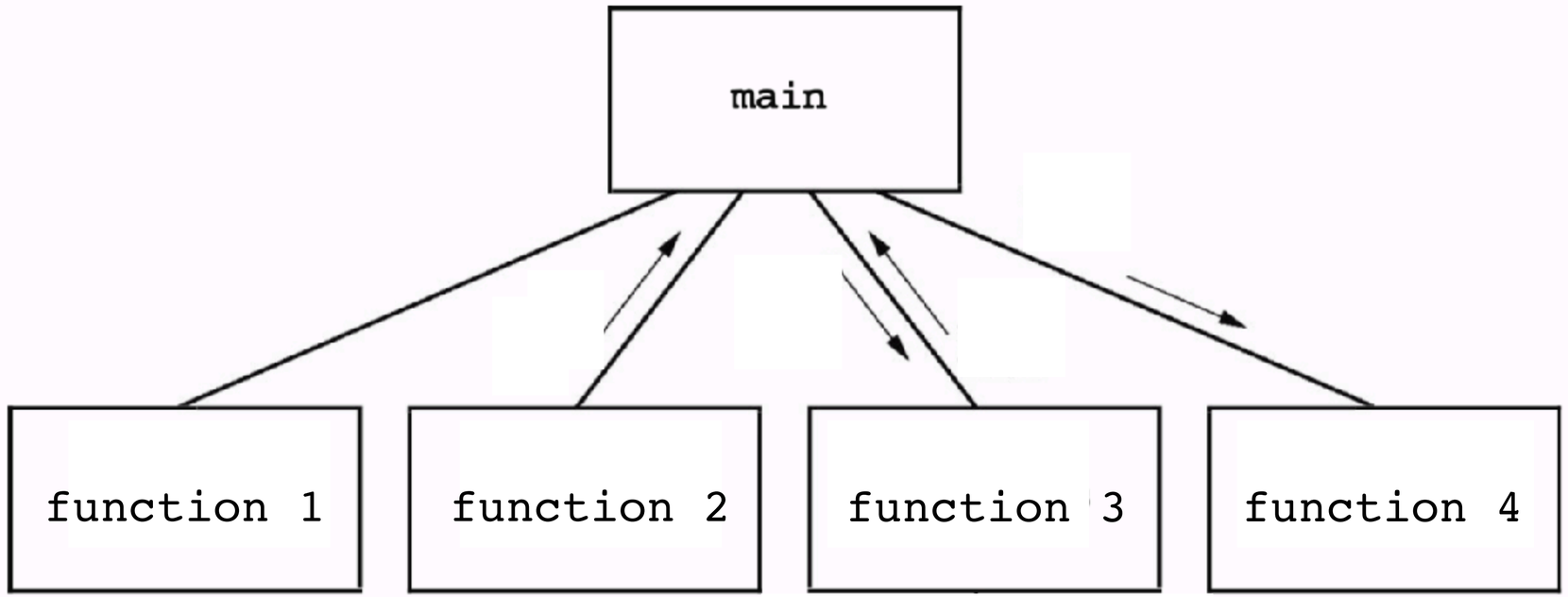
Many more: shuffle, sample, etc.

https://pynative.com/python-random-module/

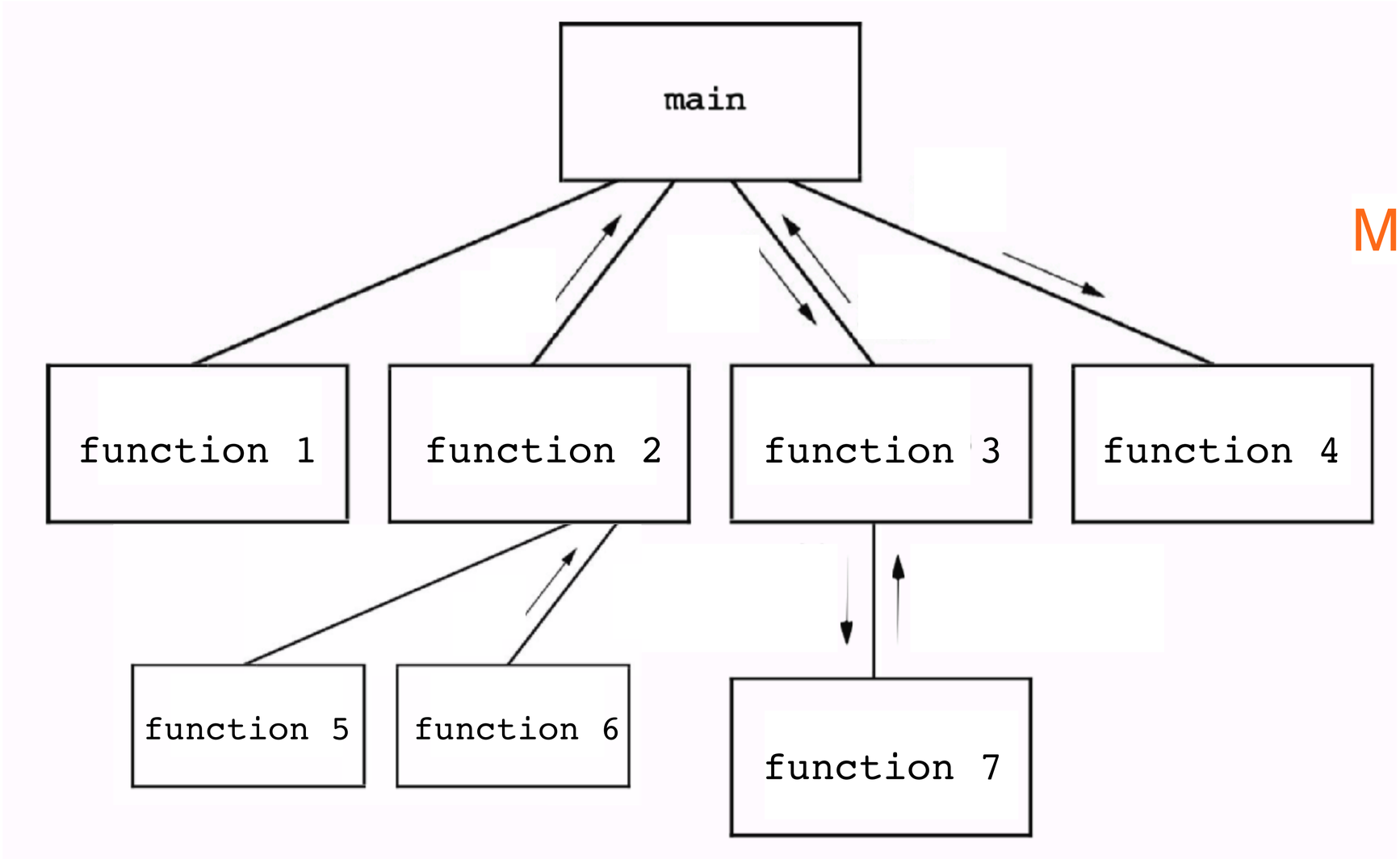

E-FORCE
RENOVART
ICODER
GEARBOX
geep
COSTA RICA
NUTRI SNACKS
Rumba
OLYMPIC CHANNEL
LONG 4
MART 0

Top-down development

Top-down is a programming style that starts with the big picture



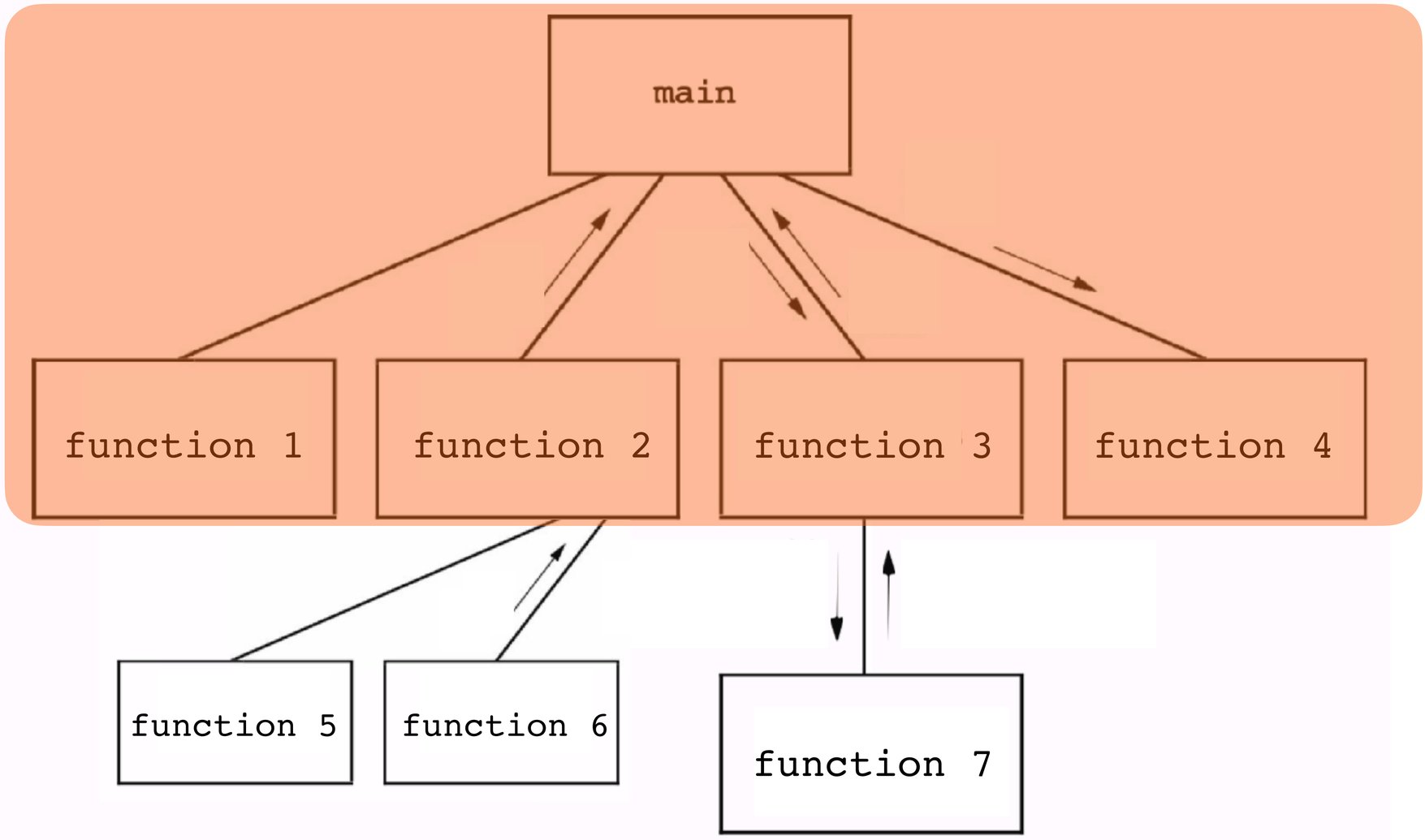
Top-down is a programming style that starts with the big picture, then divides the system into smaller pieces



Structure chart

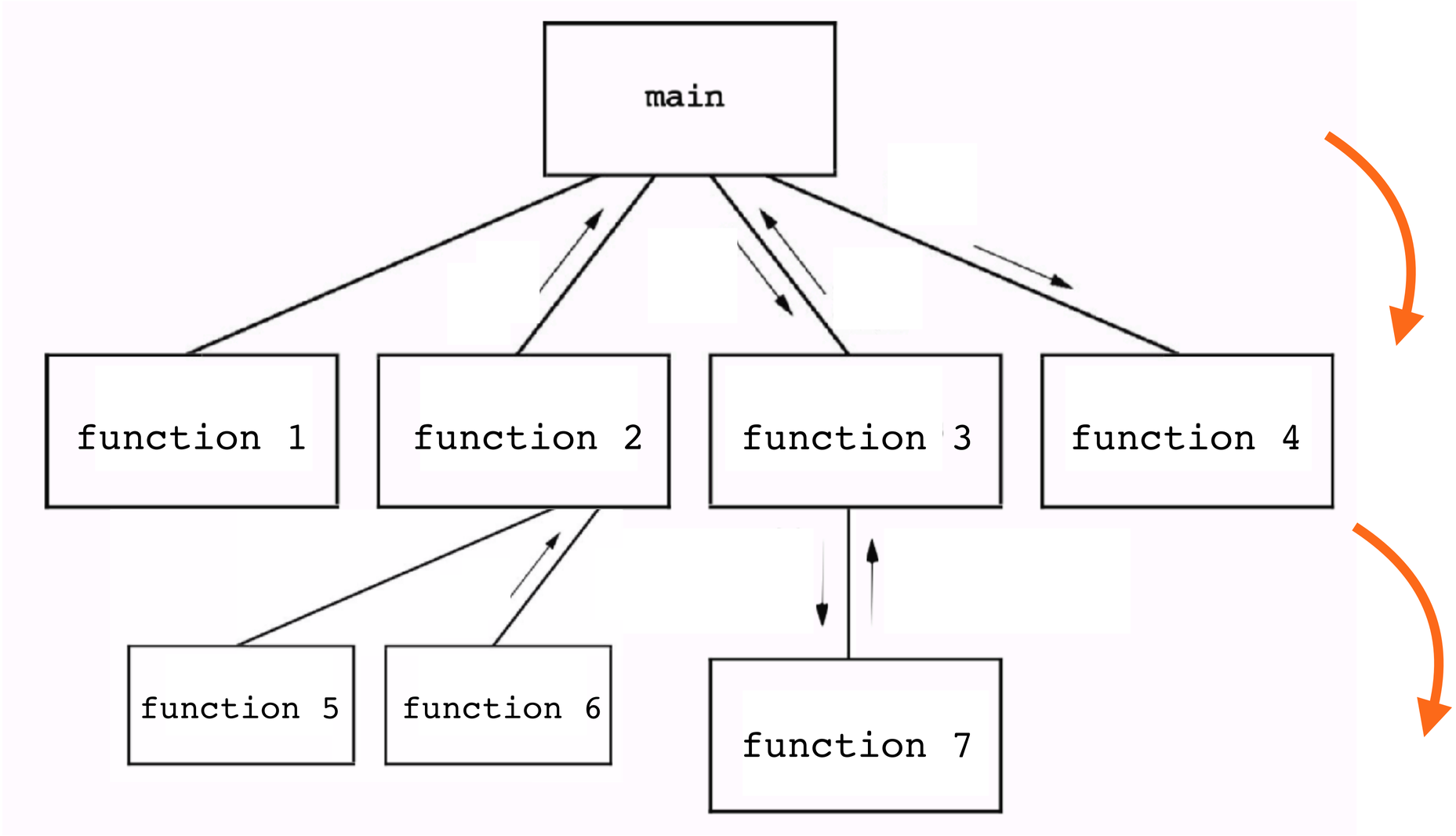
Module hierarchy chart

Top-down is a programming style that starts with the big picture, then divides the system into smaller pieces



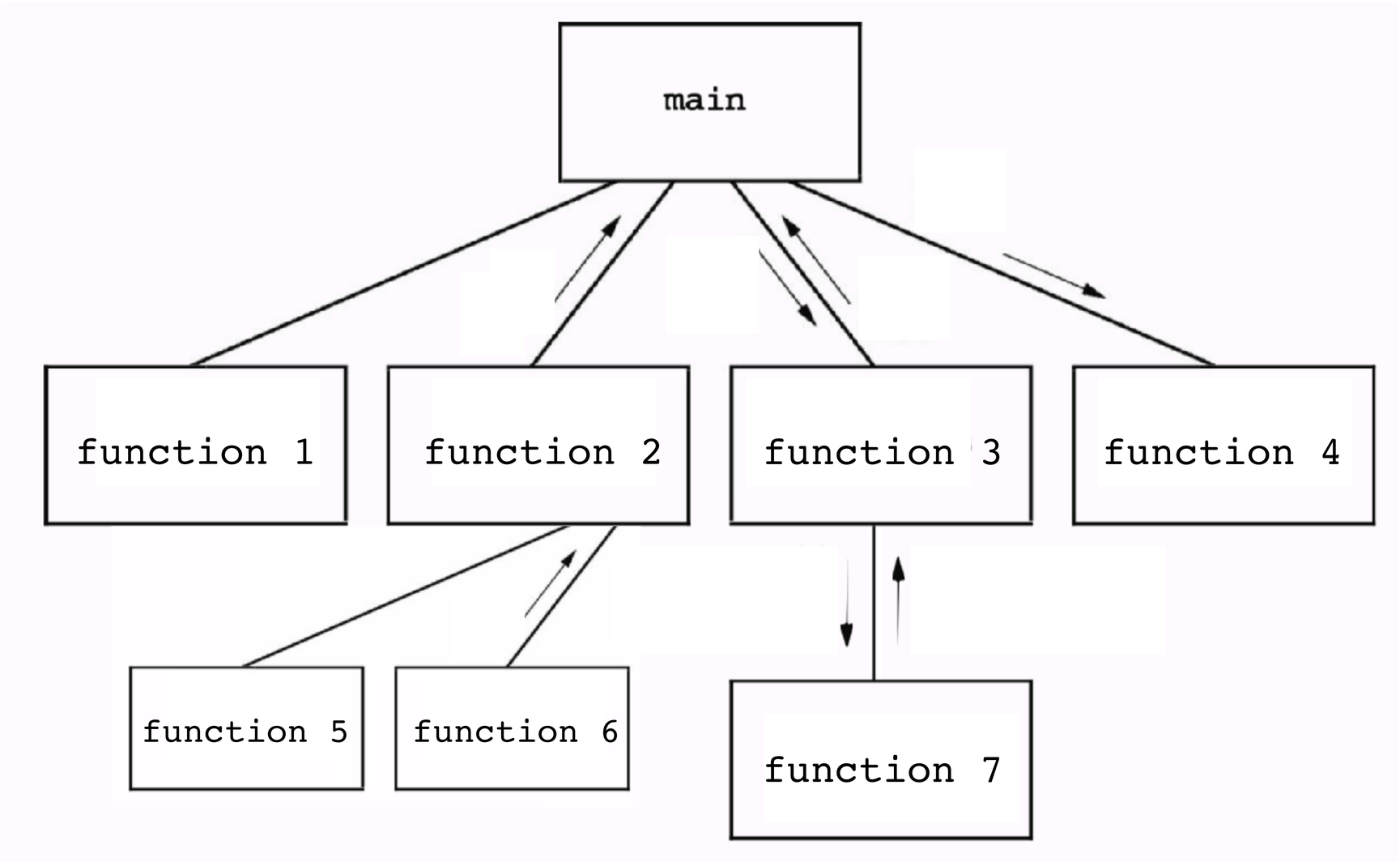
Problem decomposition

Top-down is a programming style that starts with the big picture, then divides the system into smaller pieces

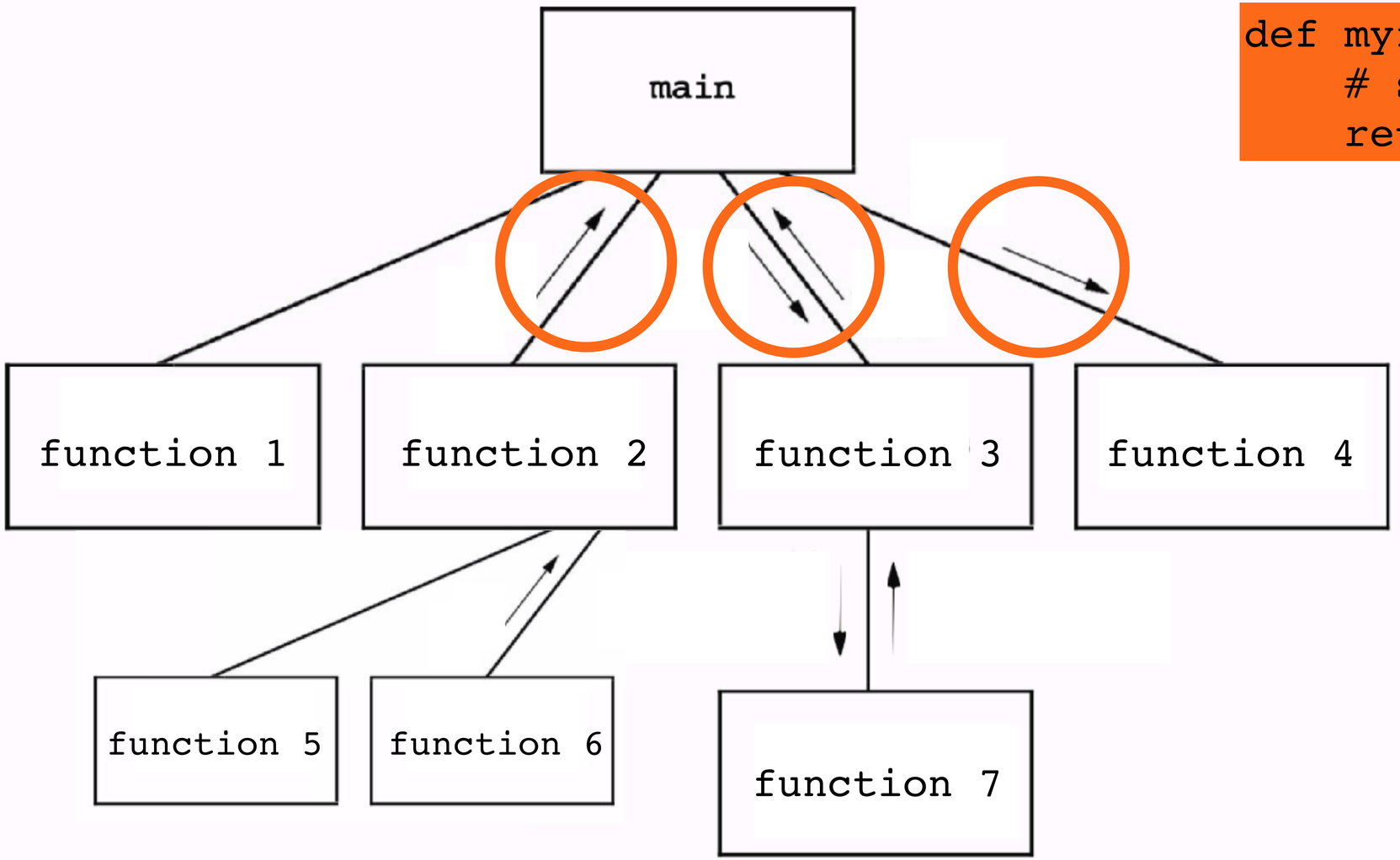


Stepwise refinement

Identifying the main characteristics of modules while ignoring other details is called abstraction

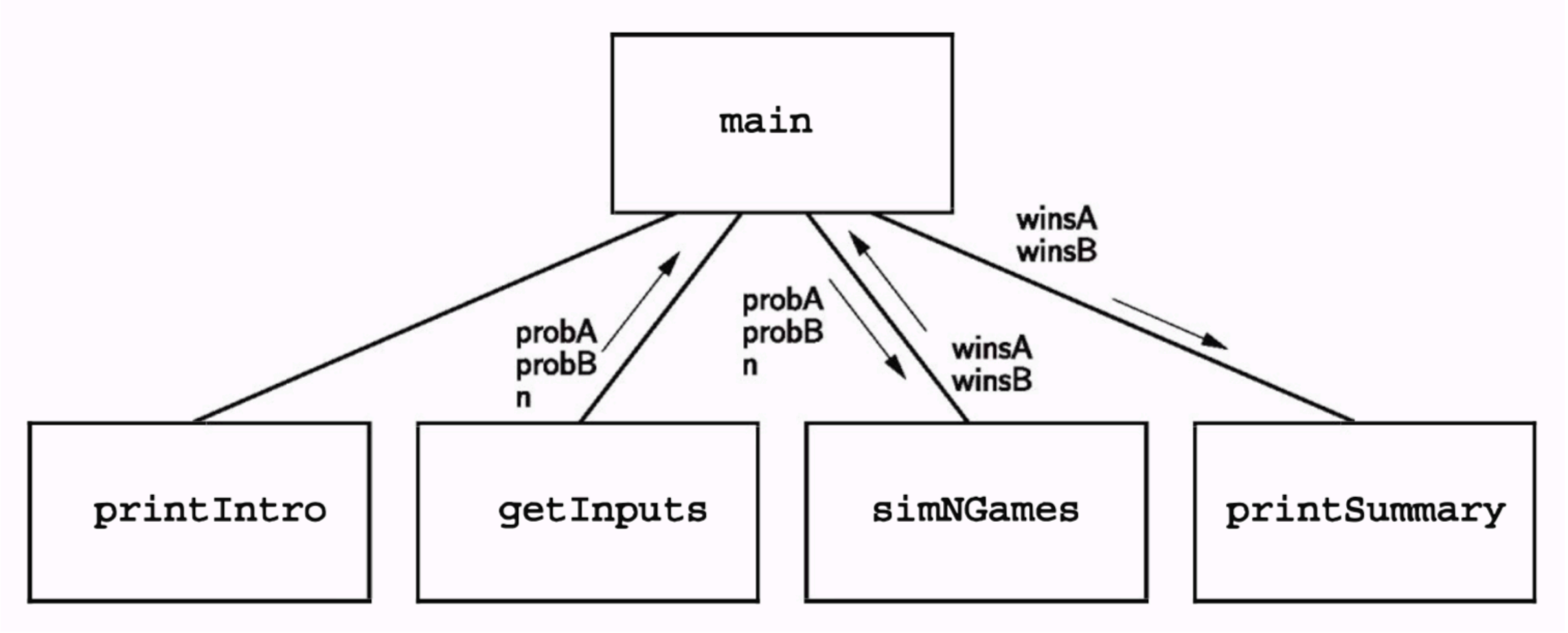


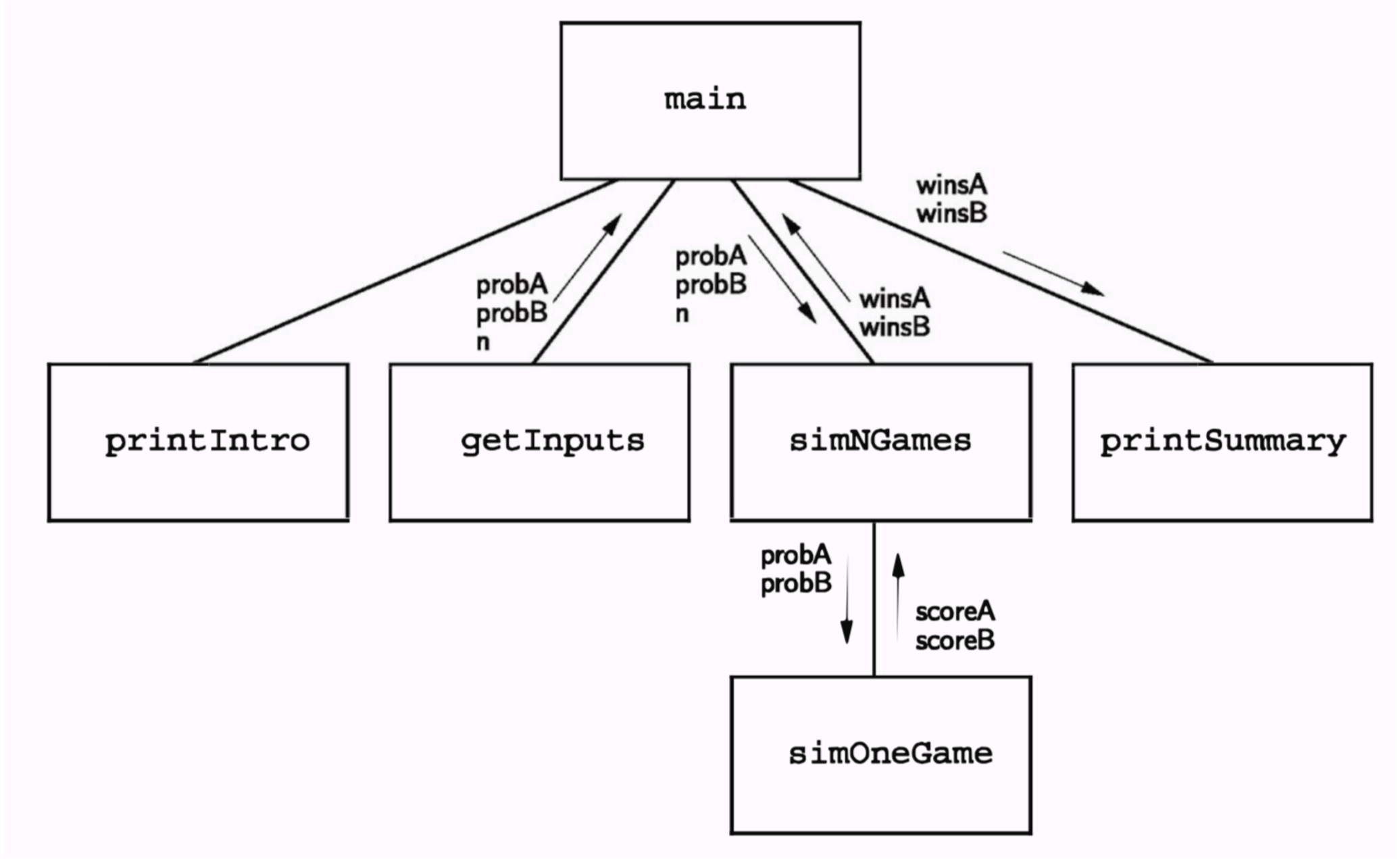
Top-down is a programming style that starts with the big picture, then divides the system into smaller pieces

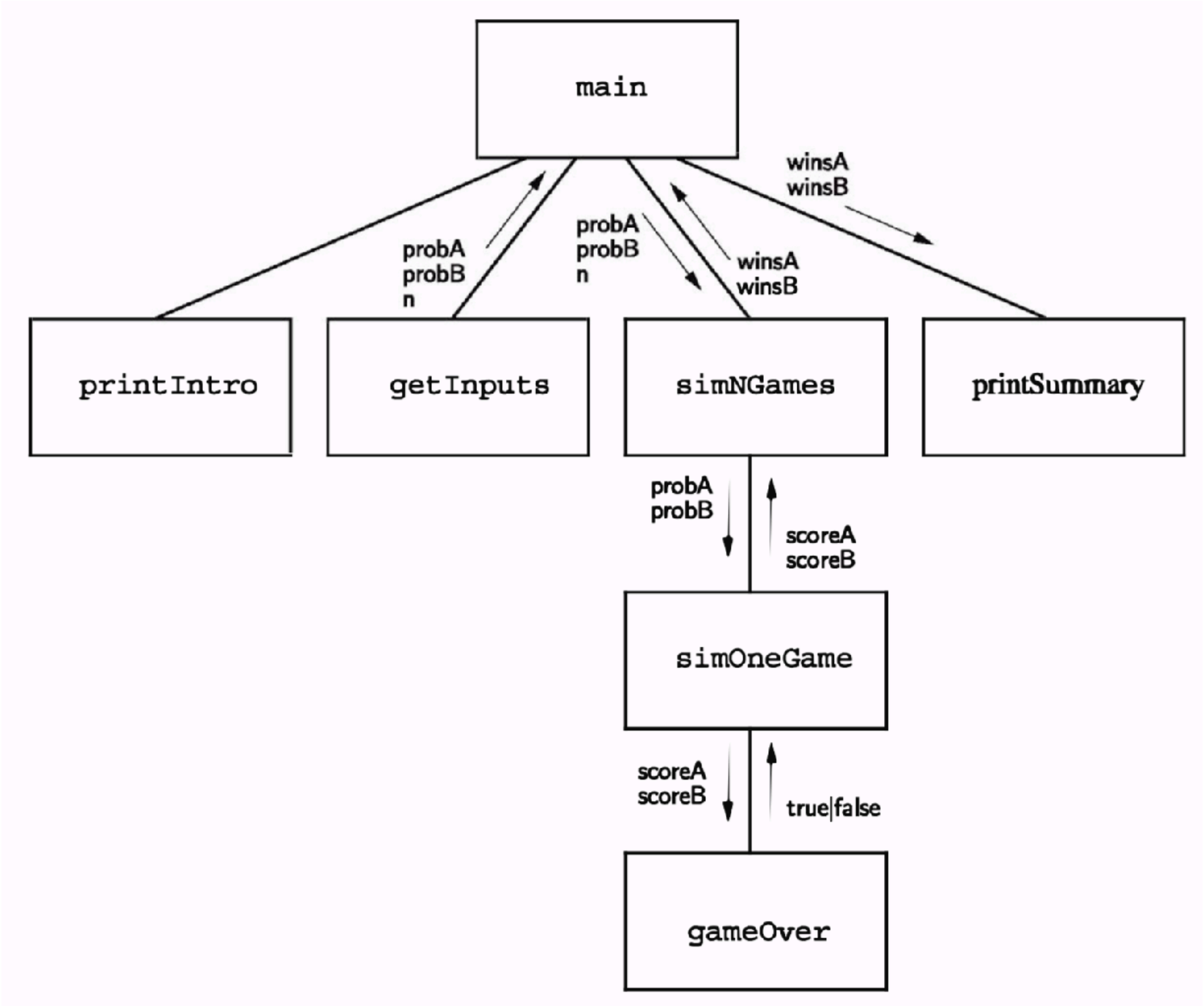



```
def myfunc(input1, input2):
    # stuff happening
    return output1, output2
```

Interface


main
printIntro
getInputs
simNGames
printSummary
probA
probB
n
probA
probB
n
winsA
winsB
winsA
winsB


main
printIntro
getInputs
simNGames
printSummary
simOneGame
probA
probB
n
probA
probB
n
winsA
winsB
winsA
winsB
probA
probB
scoreA
scoreB


main
printIntro
getInputs
simNGames
printSummary
simOneGame
gameOver
probA
probB
n
probA
probB
n
winsA
winsB
winsA
winsB
probA
probB
scoreA
scoreB
scoreA
scoreB
true|false

Top-down development encourages planning and understanding of the whole system

Top-down development encourages planning and understanding of the whole system

Top-down development delays testing of the functional units of a system until significant design is complete

Top-down

Thesis Title

Introduction

- Research question
- Motivation
- Literature research

Method

- Tools
- Data

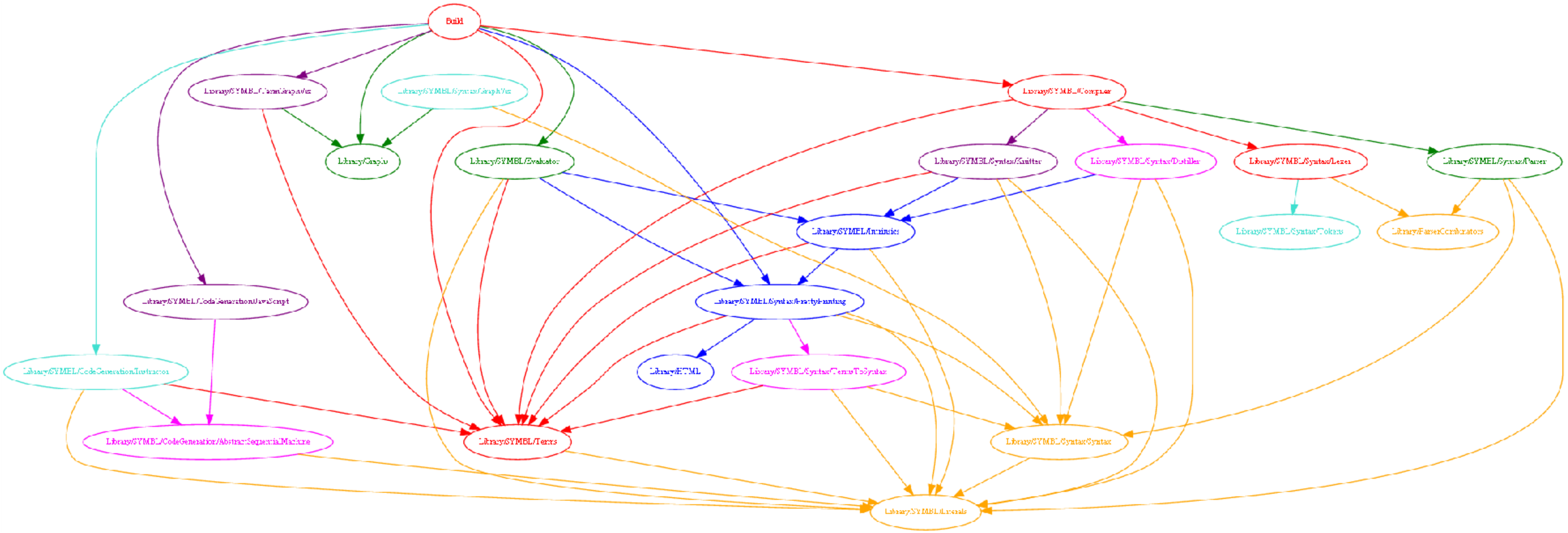
Results

Discussion

- Limitations
- Future work

Conclusion

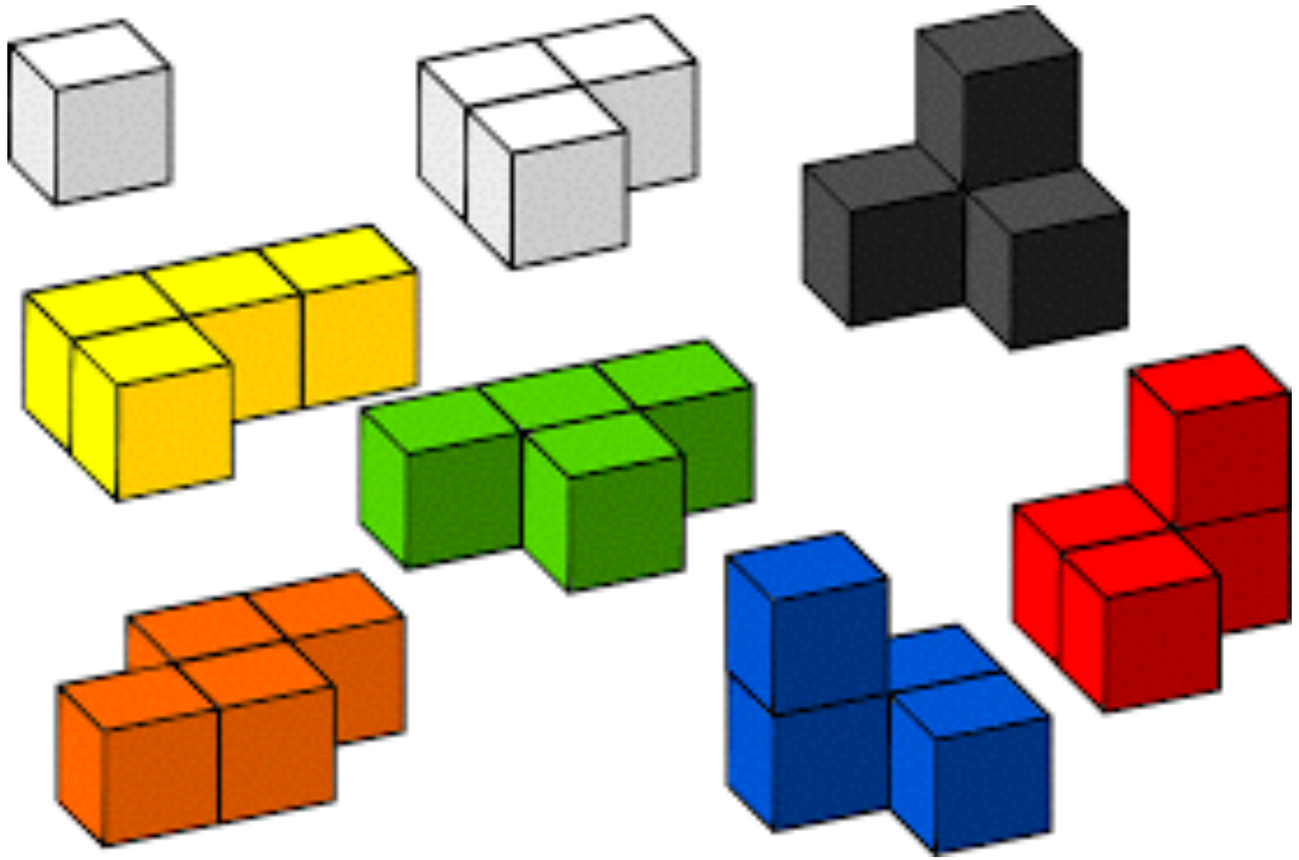
Overzealous separation into modules can make navigation through the code more difficult



Overzealous separation into modules can make navigation through the code more difficult

Bottom-up development

Bottom-up emphasizes early coding and unit testing



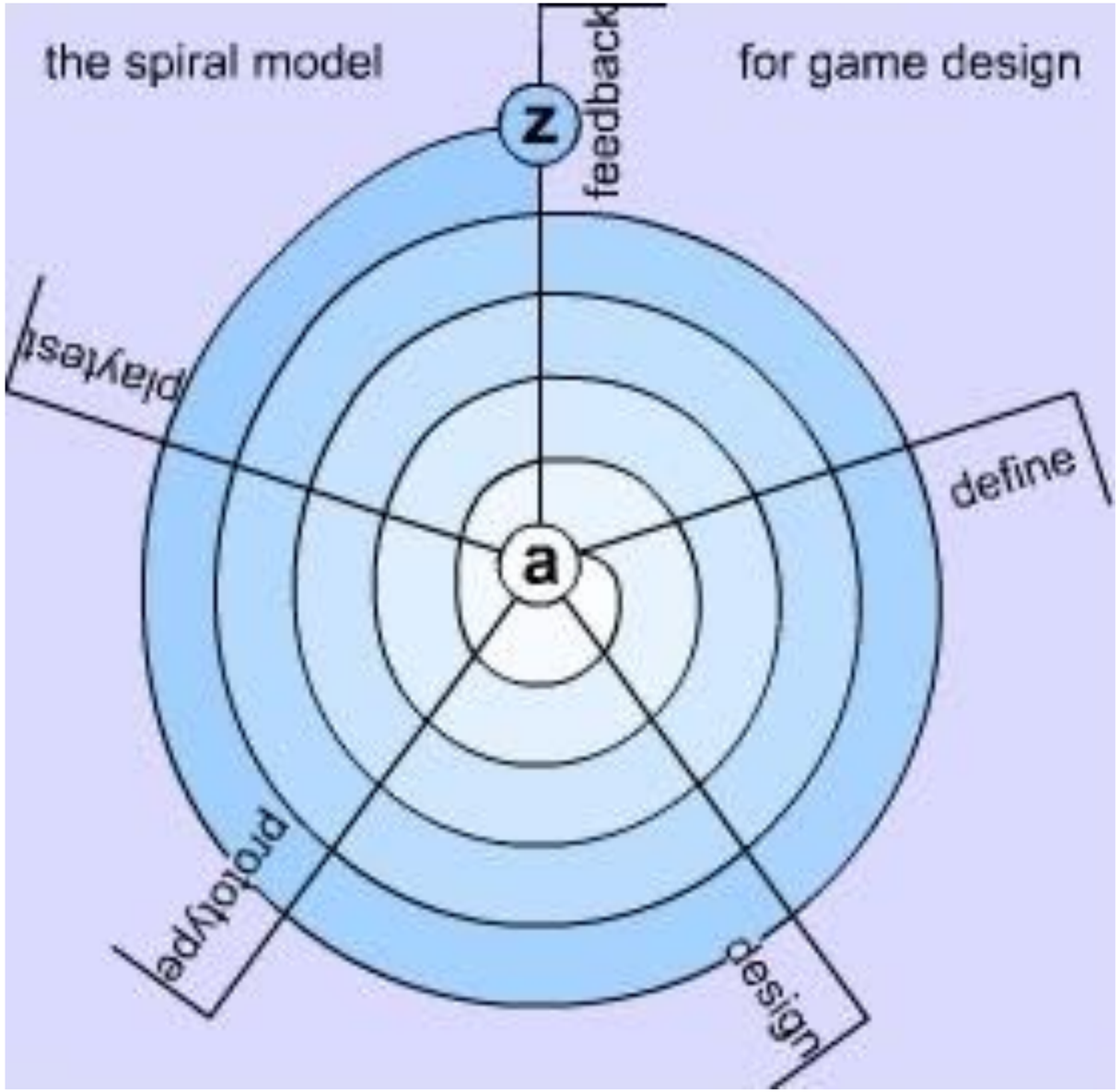
Unit testing is a level of software testing where individual units of a software are tested.

Bottom-up can lead easier to Spaghetti code

Protoyping starts with a simple version, then adds features

Useful if you get stuck, if you need early feedback, or if the specification is too complicated or unclear

Protoyping often follows a spiral development



Bottom-up

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ac feugiat sed lectus vestibulum mattis. Elementum curabitur vitae nunc sed velit dignissim sodales. Morbi tristique senectus et netus et malesuada. Magna eget est lorem ipsum dolor. Duis at consectetur lorem donec massa. Viverra nibh cras pulvinar mattis nunc. Ullamcorper sit amet risus nullam eget felis eget nunc. Metus aliquam eleifend mi in nulla posuere sollicitudin aliquam. Rhoncus urna neque viverra justo nec ultrices dui sapien eget. Turpis egestas integer eget aliquet nibh praesent tristique. Vitae aliquet nec ullamcorper sit amet risus nullam. Amet consectetur adipiscing elit pellentesque habitant. Purus gravida quis blandit turpis cursus in hac. Id diam vel quam elementum pulvinar etiam non quam. Amet mauris commodo quis imperdiet massa tincidunt nunc pulvinar. Pharetra pharetra massa massa ultricies mi quis hendrerit dolor magna. In massa tempor nec feugiat nisl pretium. Purus viverra accumsan in nisl nisi scelerisque. Morbi quis commodo odio aenean sed adipiscing diam. Vitae turpis massa sed elementum tempus egestas sed sed risus. At ultrices mi tempus imperdiet nulla malesuada pellentesque elit. Augue ut lectus arcu bibendum at varius vel. Lectus magna fringilla urna porttitor rhoncus dolor.

Top-down

Thesis Title

Introduction

- Research question
- Motivation
- Literature research

Method

- Tools
- Data

Results

Discussion

- Limitations
- Future work

Conclusion

Jupyter

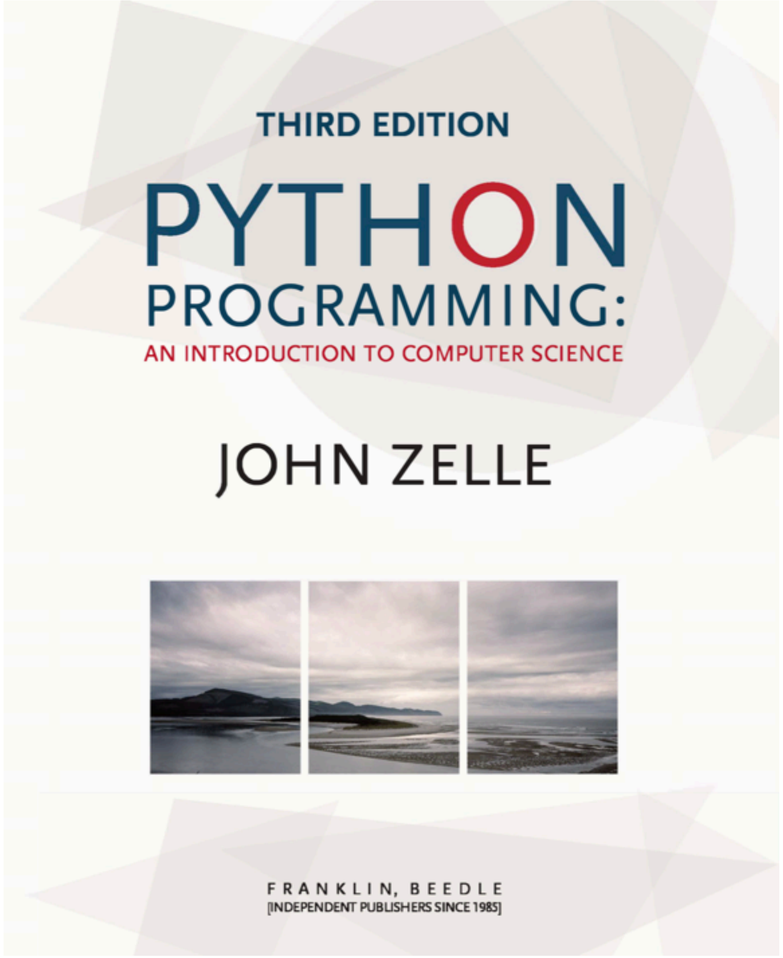
Unit testing in Python

https://www.youtube.com/watch?v=1Lfv5tUGsn8

Validating the output against a known response is called assertion.

https://realpython.com/python-testing/

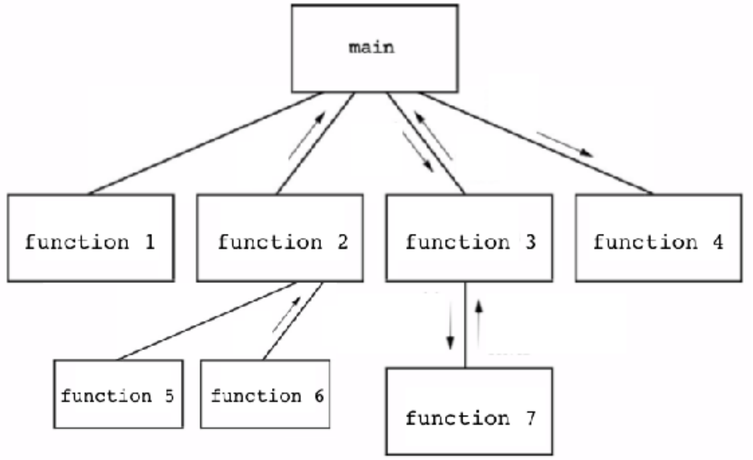
Sources and further materials for today's class



Chapter 9

What you learned today

Top-down programming



Using random numbers for simulation and program flow

Unit testing