VirginiaTech
Invent the Future®

Programming 2

ME 2984

“Debugging is twice as hard as writing the code in the first place. Therefore, if you write the code as cleverly as possible, you are, by definition, not smart enough to debug it.”
— Brian W. Kernighan

QUICK REVIEW

- Basic statements in Python
  - Commands
  - Variable assignments
  - Data Types
  - Boolean Logic
  - Control Flow
    - If/else
    - For
    - While

VirginiaTech
Invent the Future®

WATCH YOUR TYPES

- Python doesn't convert types until something quacks
- Numbers are the primary pain point here
  - 3 / 4 = ?
- Functions in Python have a similar issue
- If unsure, Python offers the type function to ask what type something is

VirginiaTech
Invent the Future®

BUILDING BLOCKS

- Software problem solving requires decomposing problems into solvable pieces
  - Conversely, software is about building up systems from small components
- Building up layers relies on previous layers

OBJECTS AND CLASSES

- Defining custom types using the metaphor of objects
- Classes define types of objects, containing
  - *Attributes* - properties, data
  - *Methods* - actions, functions
- Door
  - Width, Height, IsOpen,
  - State(), Open(), Close()

VirginiaTech
Invent the Future®

MAKING OBJECTS

- Objects provide one of the first building blocks
- Important properties is how these guide thinking
- Interacting with objects
  - What can be done? What can’t?
  - Defining rules of interactions
- Encapsulation
  - Black box interactions
  - Focusing on interfaces, not internal details

VirginiaTech
Invent the Future®

CLASS RELATIONSHIPS

- Consider the proposed Door class
- How do these relationships work out?
  - Revolving, Security, Emergency Exit
    - Some shared functionality, some unique properties
- Languages support defining and using these relationships as *inheritance*

VirginiaTech
Invent the Future®

INHERITANCE

- Base class holds common attributes and methods
  - Defining other classes as derived classes provides base class features
- Derived classes can define additional attributes or methods
  - Can also *replace* methods with more specialized methods

VirginiaTech
Invent the Future®

POLYMORPHISM

- Flipping the coin on inheritance yields polymorphism
- Imagine writing code that interacts with doors
  - Desire to open any sort of door
  - Don’t want to write code for every type of door
- Write code that interacts with generic doors
  - Language handles routing calls to correct type of door

COMPOSITION

- Imagine defining a class for a room
  - What's a room's relationship to a door?
- Not an *Is-A* relationship
- Creates a *Has-A* relationship
- Building up more complex objects from simpler ones

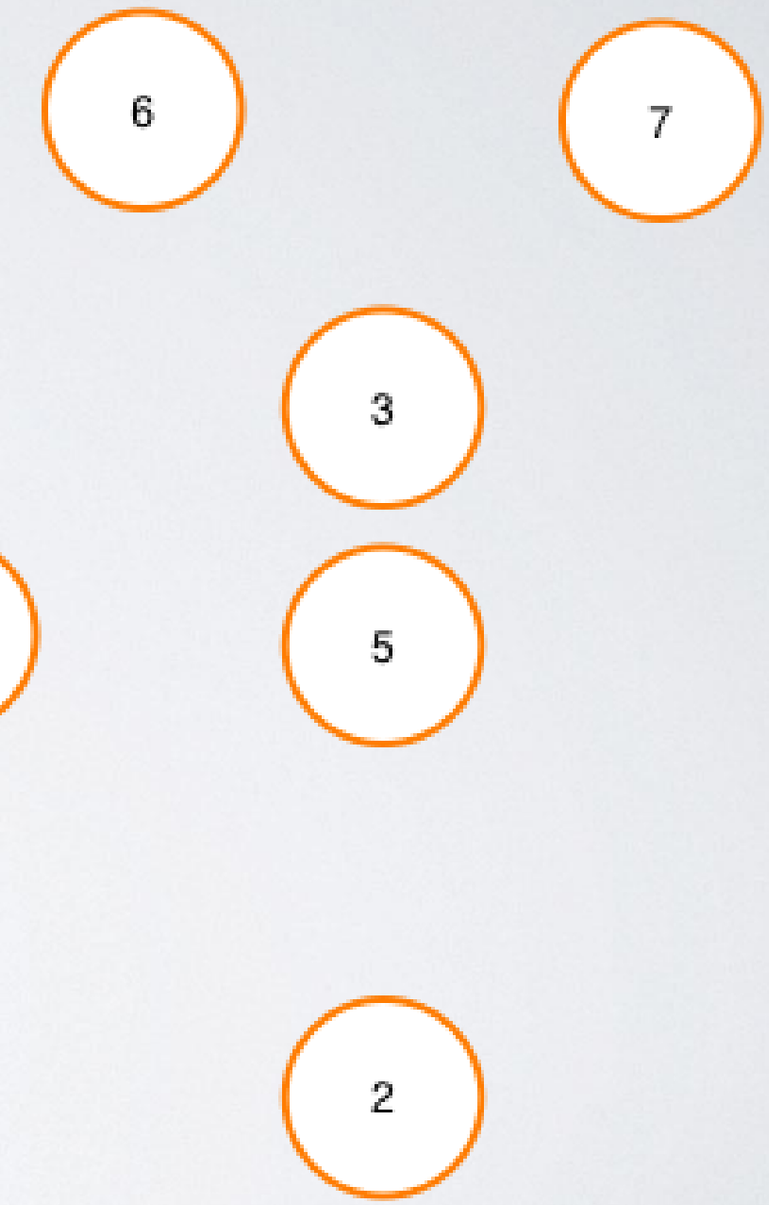
VirginiaTech
Invent the Future®

ORGANIZATION

- Classes provide an intuitive metaphor for organizing code
- Class organization has huge impact on understanding and re-using code

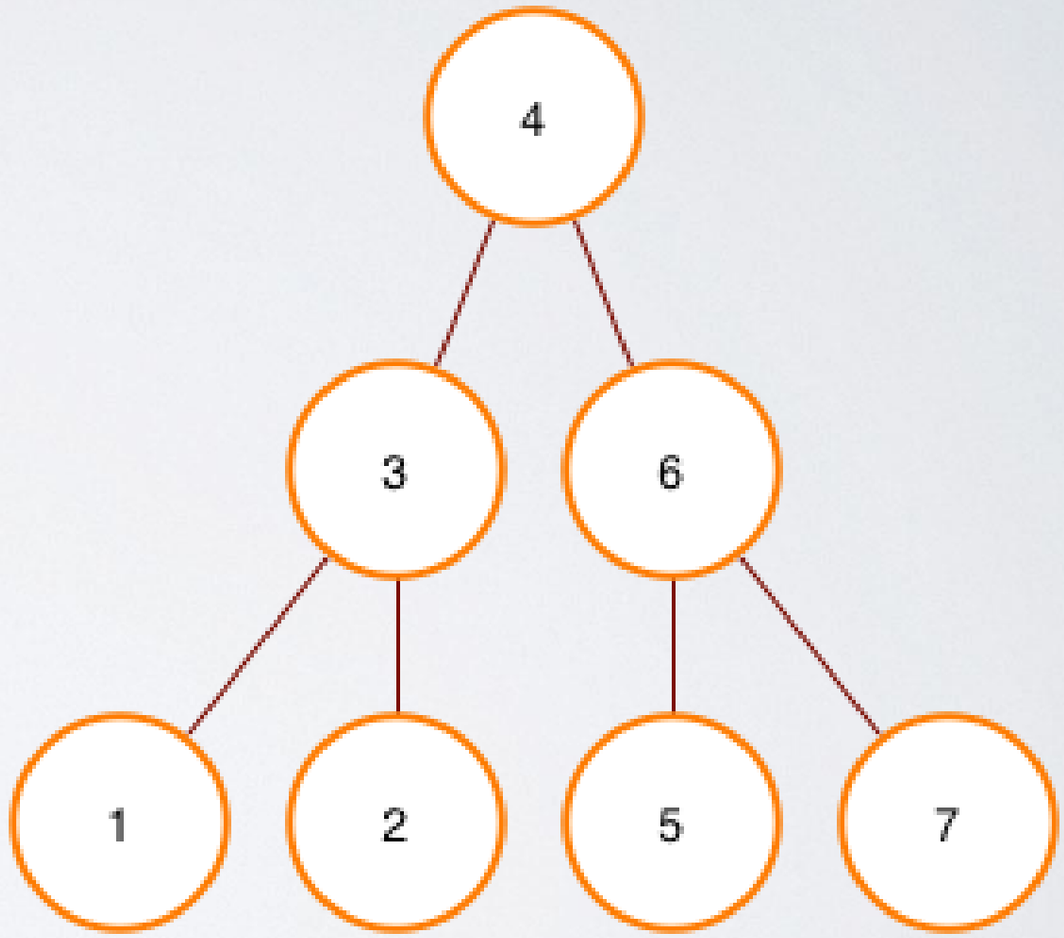
DATA STRUCTURES

- Classes enable one method of organizing data
  - Structures are orthogonal
- Being well organized has benefits



DATA STRUCTURES

- Classes enable one method of organizing data
  - Structures are orthogonal
- Being well organized has benefits
- Imagine dealing with a thousands/millions of numbers



RELEVANT DATA STRUCTURES

- Containers
  - Structures which store many instances of some object
    - Array, Matrix
- Sorted/Sequenced Structures
  - Structures which encode some order over elements
  - Trees, Grid Maps, Queues
- Maps

VirginiaTech
Invent the Future®

PACKAGING CODE

- Written useful code, we want to encourage reuse by other people
- Need some standard method for providing that code in a usable manner
  - And some method to pulling that code in!
- Luckily, Python comes Batteries Included

PYTHON MODULES

- Python provides a massive suite of tools in the standard library
  - Not part of the language
  - But always on-call
- It really is that easy

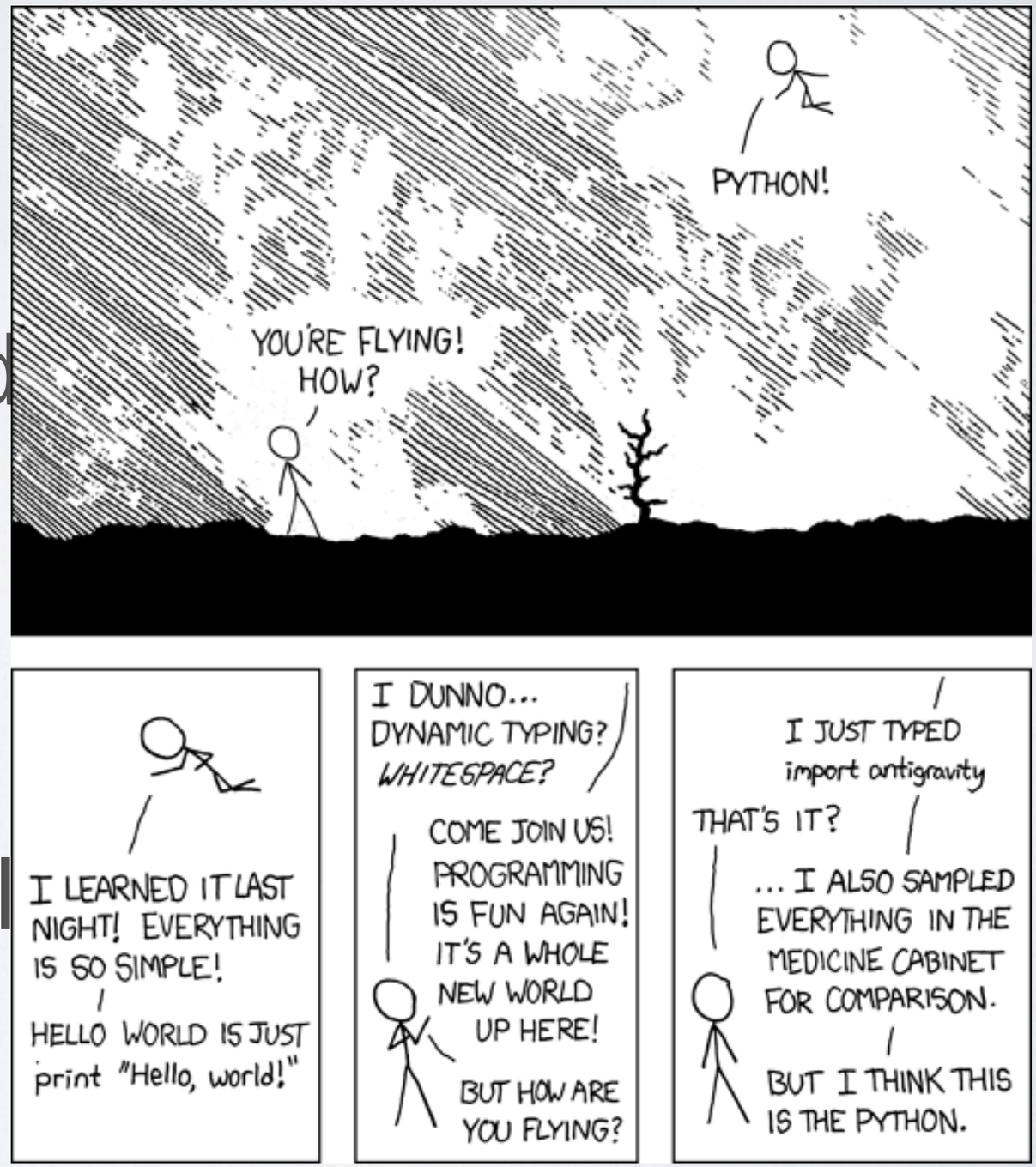


Image Credit: XKCD

VirginiaTech
Invent the Future®

OPENING PACKAGES

- How do you understand a new package?
  - dir()
  - help()
  - Experimentation
  - Documentation
- How do you find packages?
  - Debian package search
  - Standard Library
  - PyPI

VirginiaTech
Invent the Future®

LET'S GO FLYING!

VirginiaTech
Invent the Future®

CREATING PACKAGES

- If you can write Python functions in a file, you're basically done
  - Official documentation covers the details
- There's got to be more, right?
  - Yes, but it extends beyond packages

VirginiaTech
Invent the Future®

DOCUMENTATION

"Always code as if the guy who ends up maintaining your code will be a violent psychopath who knows where you live." - Rick Osborne

VirginiaTech
Invent the Future®

DOCUMENTING CODE

- Documentation is a key part of coding
  - Helps reason through code
  - Don’t slap it in last minute
- Document *Why*, not *What*
- Communicate information relevant to strangers
  - Future You is one of those strangers
- Good documentation is a invaluable
  - In more than just code

VirginiaTech
Invent the Future®

TESTING CODE

- How do you know some stranger's code is going to work?
- How about if your own code works?
  - I mean, *really* works
- Reading code isn't enough
  - Although reviewing code with a colleague can help

VirginiaTech
Invent the Future®

BUILD AND TEST

- Imagine a function readSensorAndUpdateMap()
  - Does what it says in the name
  - What's the typical development story?
  - The typical testing story?
  - Did we cover all the cases?

VirginiaTech
Invent the Future®

FAILED TO TEST

- How did we miss the errors?
- Is there a better way to think about testing?
- Develop tests first
  - Test to prove it fails
  - Run those tests every time you change the code
- How to be good and lazy

VirginiaTech
Invent the Future®

UNIT TESTING

- Software has been broken down into small chunks to implement and re-use throughout the code
- Write software which exercises those units of code and checks the results of the operations for correctness
  - Once the tests are written in software, computer can execute them whenever we want
- Immediate snapshot of library status!

REGRESSION TESTING

- Testing is both forward and backwards looking
  - Did my change fix anything?
  - Worse yet, did it break anything?
- Know the impact of a change

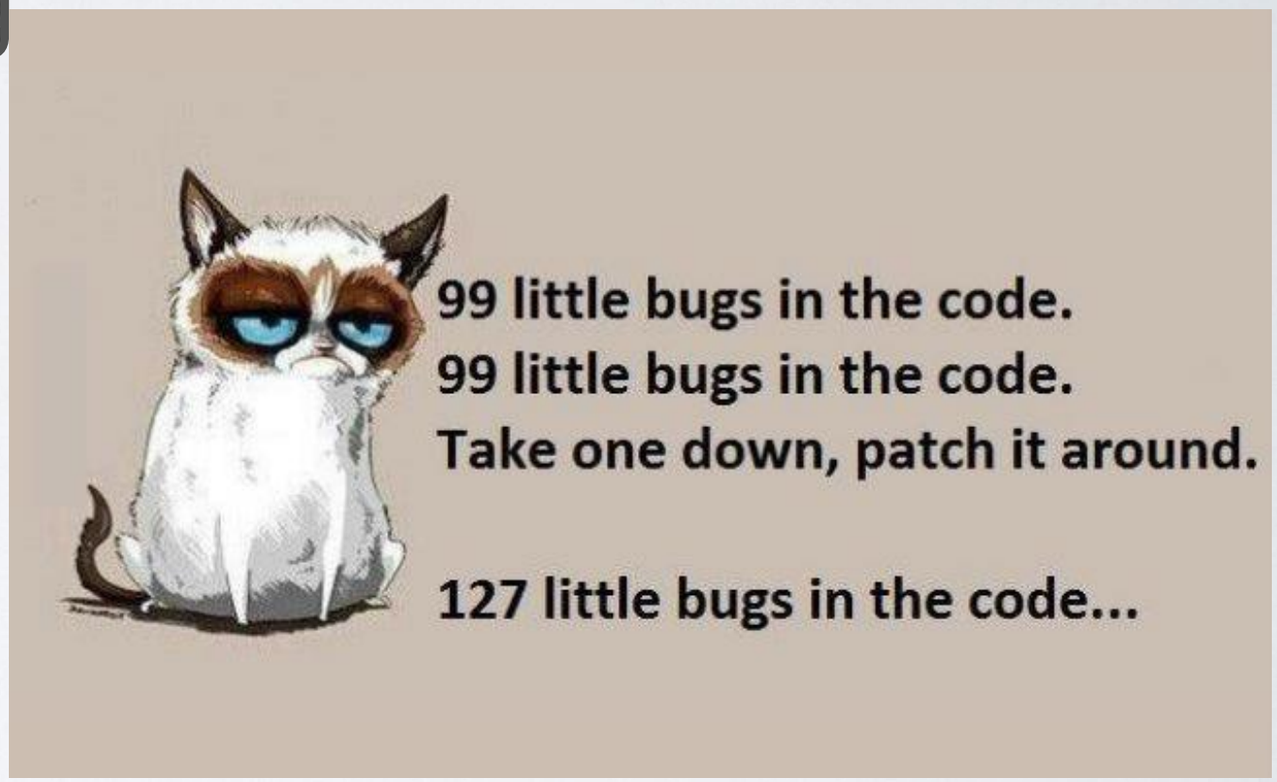


Image Credit: WeKnowMemes

VirginiaTech
Invent the Future®

INTEGRATION TESTING

- Connecting all the pieces isn't easy
  - Individual pieces may be correct
- Testing the integration of components is crucial
  - Not only in terms of component correctness, but correct specification
- Integrate early, integrate often
  - Effort is non-linear in time since last integration

VirginiaTech
Invent the Future®

SYSTEM TESTING

- Does the system do what is intended?
- Covering the functional requirements as a whole
- WARNING: *Really weird issues ahead*
- Understand what's been put together
  - How does it work
  - When does it fail
  - How does it fail