Intro to Computer Science

Previous

- Sorting
- Backtracking

Next

- Objects

Readings	
Gaddis	• Chapter 10

Journey through programming

- Started with sequential programming
  - Top-down collection of statements
  - Run one after the other
- Introduced control structures
  - Repeat certain actions (loops)
  - Decide when to execute certain actions (if-then)
  - Group certain actions (functions)
- Usage of functions known as *procedural* programming

Task philosophies

Procedural programming

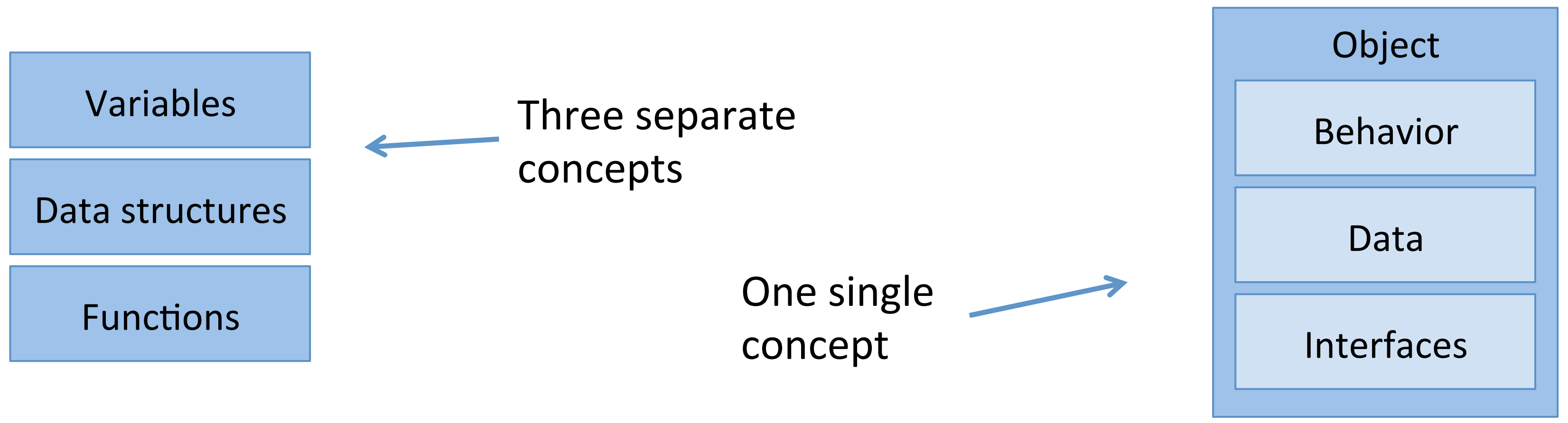
Breaks down tasks using

- Variables
- Data structures
- Subroutines

Object-oriented programming

Breaks down tasks using

- Behavior
- Data
- Interfaces

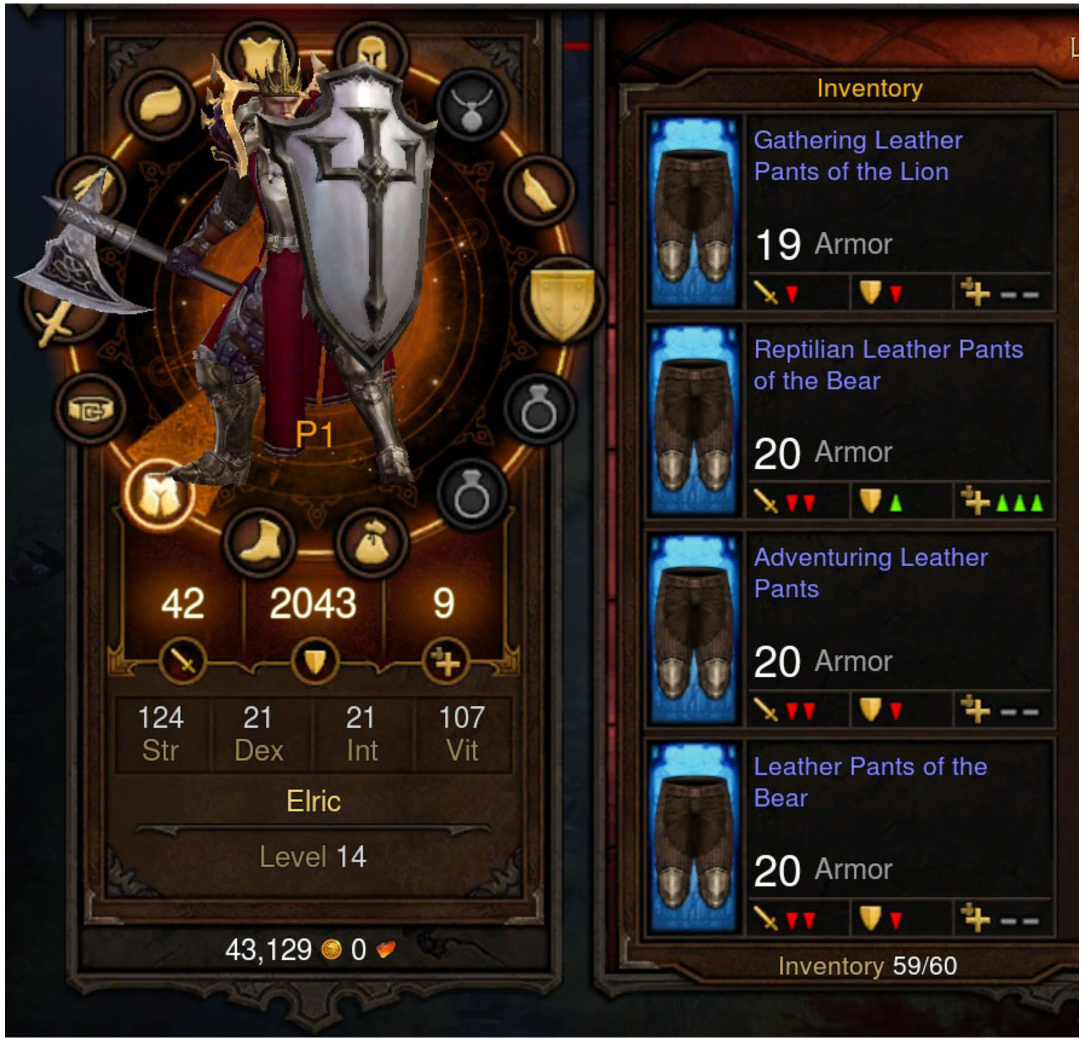


Re-thinking our data types

- Notice that our data types were storage mechanisms *and* sets of rules around how to use them
  - **Integers** have to be numbers
  - Addition of two **strings** is a new string
  - How a for-loop behaves with a **list**
- These are effectively objects!
- And in fact, all data-types in Python are objects

Motivation

RPG with characters and attributes



Inventory
Gathering Leather Pants of the Lion
19 Armor
Reptilian Leather Pants of the Bear
20 Armor
Adventuring Leather Pants
20 Armor
Leather Pants of the Bear
20 Armor
Inventory 59/60
P1
42
2043
9
124 Str
21 Dex
21 Int
107 Vit
Elric
Level 14
43,129
0

https://inventwithpython.com/blog/2014/12/02/why-is-object-oriented-programming-useful-with-a-role-playing-game-example/

Syntax

keyword “class”

- Class name can be whatever you want (standard naming rules apply)
- Customary to use “CamelCase”

```
class ClassName:
    block
```

Colon and block: usual suspects

What goes in the block?

Methods and attributes!

Recall:

“Objects contain data, along with *functions* that operate using that *data*”

Technically:

“Objects contain *methods* and *attributes*”

A function that is inside an object

A variable that is inside an object

✓ Clarity

– “The print function”

– “A string method”

Methods

- Utilize parameters and attributes
- Can return values
- First parameter is special
  - Reference to the current object
  - Definitions will have one more parameter than is used by caller
  - Traditionally called "self"
- When calling other methods, defaults to *global scope*

Looks like any other method… almost

```
class Singer:
   def note(self):
      return ‘a minor’
```

- Standard class definition
- Standard function definition
  - That’s in the class definition!
  - Whose first parameter is `self`

```
s = Singer()
print(s.note())
```

- Instantiate the class
- Call the `note` method within the instance

Instantiation

- Primary method of assigning a class to a variable is by calling the class name
- Known as *instantiation*
  - Creating an "instance" of the class
- Vernacular:
  - The definition is a *class*
  - The instantiation is an *object*

The class becomes the instance

Class

- Type of “thing” with certain characteristics that are shared by all things of that type
- Value of those characteristics not necessarily specified

Instance

- Particular thing that belongs to a class
- Has those characteristics specified

Class: singer, style

Instance: Justin Bieber, terrible

Instantiation

```
class Singer:
    …
```

Define the class ‘Singer’

```
s = Singer()
```

- s is now an instantiation of Singer
- s is an object

Some methods are special

- Some methods are not called explicitly
  - Used internally by Python and various Python functions
- These methods control
  - How your object is printed
  - How your object is iterated
  - How your object is ordered
  - How your object is copied
  - …
  - *What happens when you're object is instantiated*

Instantiation (again)

You call

```
s = Singer()
```

Python does

```
class Singer:
    def __init__(self):
        return
```

What to notice:

1. Function name is special
    - Cannot change this!
2. Double underscore syntax
3. Obligatory `self` parameter

Instantiation (again)

You call

```
s = Singer('Bieber')
print(s.name)
```

Python does

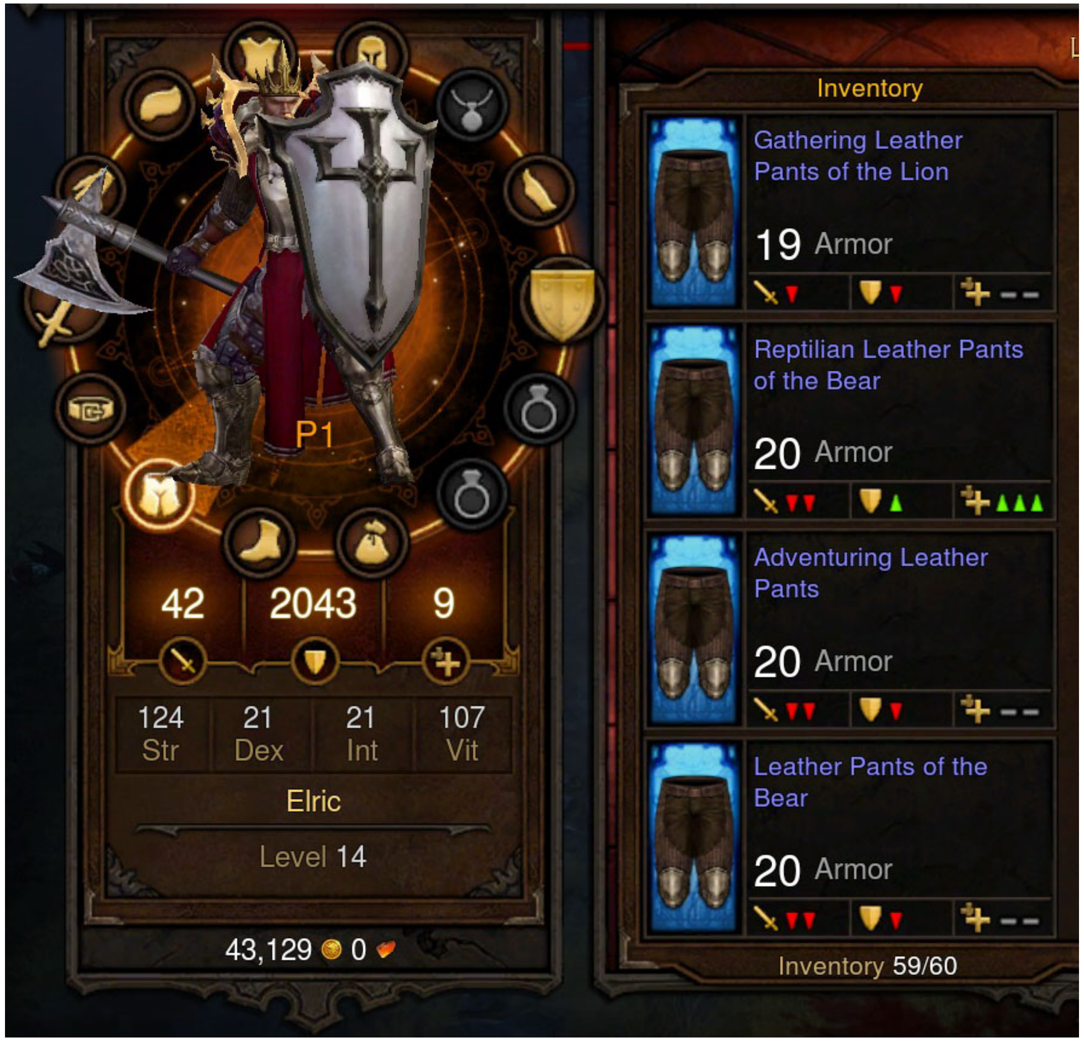
```
class Singer:
    def __init__(self, name):
        self.name = name
```

- `name` now becomes a bound variable in `Singer`
- Since the assignment takes place in the constructor, all instances will have their own distinct copy

➢ This is the other (*best practice*) method of establishing attributes

Motivation

RPG with characters and attributes


P1
42
2043
9
124 Str
21 Dex
21 Int
107 Vit
Elric
Level 14
43,129 0
Inventory
Gathering Leather Pants of the Lion
19 Armor
Reptilian Leather Pants of the Bear
20 Armor
Adventuring Leather Pants
20 Armor
Leather Pants of the Bear
20 Armor
Inventory 59/60