Object Magic

Introduction to Computer Programming

Dr. Paul Vrbik

November 27, 2018

Underscores

How underscores are used in python. (List not comprehensive.)

1. As anonymous variables like in `for _ in [1, 2, 3]` or `x, _, z = (1, 2, 3)`.
2. For giving special meaning to function and names.
   2.1 `__names__` are for Python's magic methods like `__init__()`.

Magic Methods

Initializer

Runs when the object is instantiated (i.e. created).

```
class point():
    def __init__(self):
        #Initializes the objects class-scoped variables
        self.x = 0
        self.y = 0
```

```
class point():
    def __init__(self, x: int, y: int):
        #Initializes the class-scope variables using input
        self.x = x
        self.y = y
```

Magic Methods

String Representation

The string representation says what to print when printing the object. The default is

```
    <__main__.point object at 0x10ef30b70>
```

```
>>> class Point():
...     def __str__(self) -> str:
...         return "(x, y)".format(x: self.x, y: self.y)
>>> p = Point(2, 3)
>>> print(p)
(2, 3)
```

Magic Methods

Representation

The representation of an object is what Python prints when displaying it on console.

```
    <__main__.point object at 0x10ef30b70>
```

```
>>> class Point():
...     def __repr__(self) -> str:
...         return "({}, {})".format(self.x, self.y)
>>> p = Point(2, 3)
>>> p
(2, 3)
```

Magic Methods

Add

Add instructs Python on how to add two objects together.

```
>>> class Point():
...     def __add__(self, other) -> str:
...         return (self.x + self.y, other.x + other.y)
>>> p = Point(2, 3)
>>> p
(2, 3)
```

Magic Methods

Binary Operator	Magic Method
`+`	`__add__`
`-`	`__sub__`
`*`	`__mul__`
`**`	`__pow__`
`//`	`__floordiv__`
`/`	`__truediv__`

Magic Methods

Unary Operator	Magic Method
`-`	`__neg__`
`abs`	`__abs__`
`~`	`__invert__`

Magic Methods

Comparison	Magic Method
`<`	`__gt__`
`<=`	`__le__`
`==`	`__eq__`
`!=`	`__ne__`
`>`	`__gt__`
`>=`	`__ge__`

Question

Queue A queue is a useful data structure that implements the following:

1. `queue(x)` – person `x` joins the queue.
2. `deqeue` – returns person at front of queue and removes this person from queue.

Implement `queue` are notation for the

Question

Currency Implement a class for working with currencies in Python.

Implement the `__repr__` and `__add__` magic methods.

Question

Write a class for working with fractions.

Next Time

1. More sophisticated examples.