# Announcements

HW3 due on 02/21 at 7:59 am

Lab 4 due on Friday 02/09 at 11:59 pm

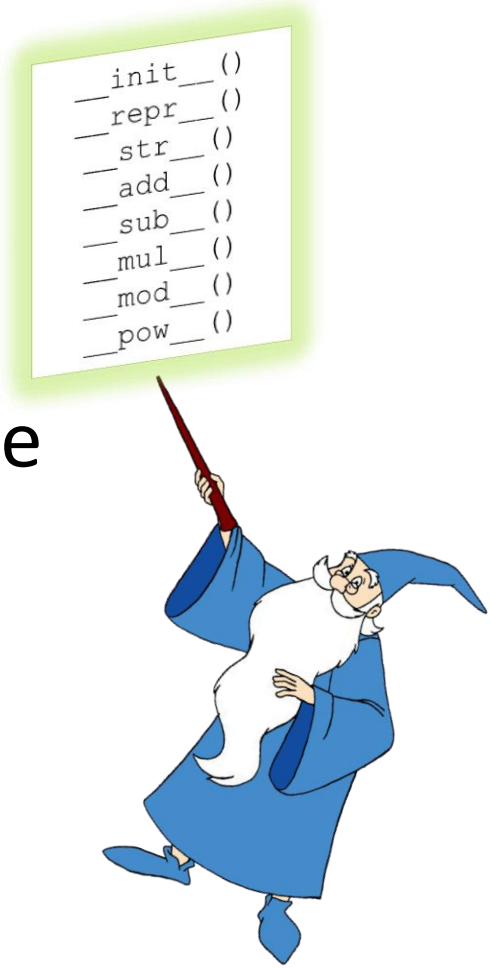
UNIX Lab 2 and Quiz 2 due on 02/14 at 11:59 pm

Exam 1 grades up! Answers are now available

# On Monday…

We talked about Magic methods:

- Also known as special methods
- Magic methods have double underscores at the beginning and the end
- You do not have to invoke them directly. The invocation is realized behind the scenes



# Magic Methods and Operator Overloading

- We have already seen how to overload the __init__ method so that we can customize it to initialize our class.

```
class Dog:
        number_of_legs=0

        def __init__(self, name, breed, size, age):
                self.name = name
                self.breed = breed
                self.size = size
                self.age = age
```

- We can also overload other special methods

# Magic Methods and Operator Overloading

For example, the purpose of the __str__ method is to output a useful string representation of our object. But by default if we use the *print* function on a Dog object (which will call the __str__ method), all that we will get is the class name and an ID.

```
>>> harry=Dog('Harry','Chihuahua','Small',4)
>>> print(harry)
<__main__.Dog object at 0x000001B1EFD76208>
```

That's not very useful!

# Magic Methods and Operator Overloading

```
class Dog:
        number_of_legs=0

        def __init__(self, name, breed, size, age):
                self.name = name
                self.breed = breed
                self.size = size
                self.age = age

        def eat(self):
                return(self.name + " is eating")

        def sleep(self):
                return(self.name + " is sleeping")

        def sit(self):
                return(self.name + " is sitting")

        def roll_over(self):
                return(self.name + " is rolling over")

        def run(self):
                return(self.name + " is running")

        def __str__(self):
                return "{} is a {} year old {}!".format(self.name, self.age, self.breed)
```

```
>>> harry=Dog('Harry','Chihuahua','Small',4)
>>> print(harry)
Harry is a 4 year old Chihuahua!
```

# Hands on!

From Module 8, download the script customer.py

Write a custom __str__ method which shows name and balance of the object in the following format:

```
>>jeff = Customer(“Jeff Brown”,”08/16/1985”,”123-456-7890”,1500)
>>print(jeff)
Jeff Brown has 1500 USD in this bank
```

# Magic Methods and Operator Overloading

- We can overload operators for the purposes of our own classes

- There is a special (or a "magic") method for every operator sign
  - "+" operator is the __add__ method
  - "-" operator is the __sub__ method

https://docs.python.org/3.6/library/operator.html

```
>>> duke=Dog('Duke','German Shepherd','Large',2)
>>> lupin=Dog('Lupin','Chihuahua','Small',3)
>>> duke+lupin
```

Python calls duke.__add__(lupin)

```
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: unsupported operand type(s) for +: 'Dog' and
'Dog'
```

# Magic Methods and Operator Overloading

```
class Dog:
        number_of_legs=0

        def __init__(self, name, breed, size, age):
                self.name = name
                self.breed = breed
                self.size = size
                self.age = age

        def eat(self):
                return(self.name + " is eating")

        def sleep(self):
                 return(self.name + " is sleeping")

        def sit(self):
                 return(self.name + " is sitting")

        def roll_over(self):
                 return(self.name + " is rolling over")

        def run(self):
                 return(self.name + " is running")

        def __add__(self,other):
                return(self.name + " and "+ other.name +" are friends")
```

```
>>> duke+lupin
Duke and Lupin are friends
```

# Hands on!

Script: customer.py

Write a custom __add__ method which will combine the balances for two objects in the following format:

```
>>jeff = Customer(“Jeff Brown”,”08/16/1985”,”123-456-7890”,1500)
>>liz = Customer(“Lizeth Brown”,”05/12/1989”,”113-456-7820”,1000)
>>jeff+liz
2500 in combined funds
```

# Submission

Submit your script customer.py to the 02/07 Lecture assignment

Due date: Friday 02/09, 7:59 am