Case study:
Interface Design

Chapter 4

The turtle module

? Module, filename issues

- Allows to create images using turtle graphics
- Comes with standard Python 3 installation (no need for extra download)

```
import turtle
bob = turtle.Turtle()
print(bob)
turtle.mainloop()
```

Method vs. Function

- Same role, but different syntax

Ex:

```
print(bob)
bob.fd(100)
```

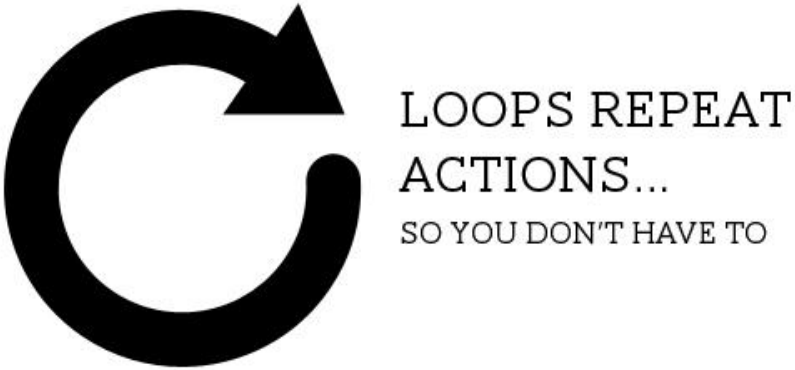
- Subject vs. Object
- You are asking *bob* to move forward

Simple Repetition

Ex:

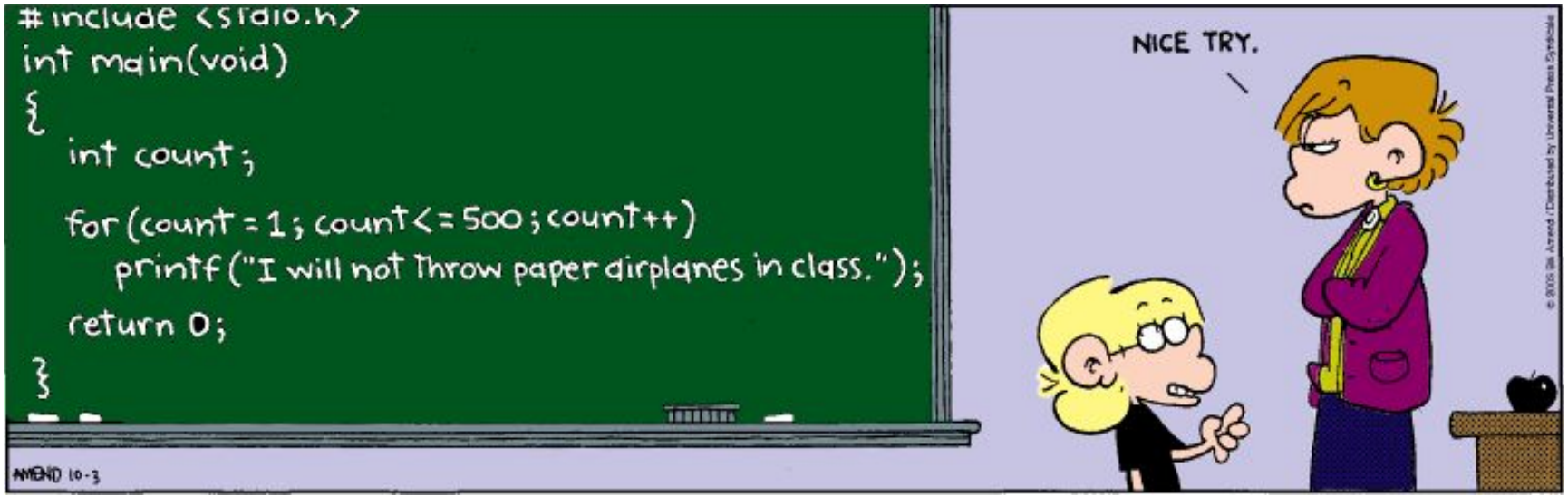
```
for i in range(4):

    print('Salem!')
```



- Syntax is similar to function definition
- Statement(s) in the body executed/repeated *n* times (loop)
- *range()* function

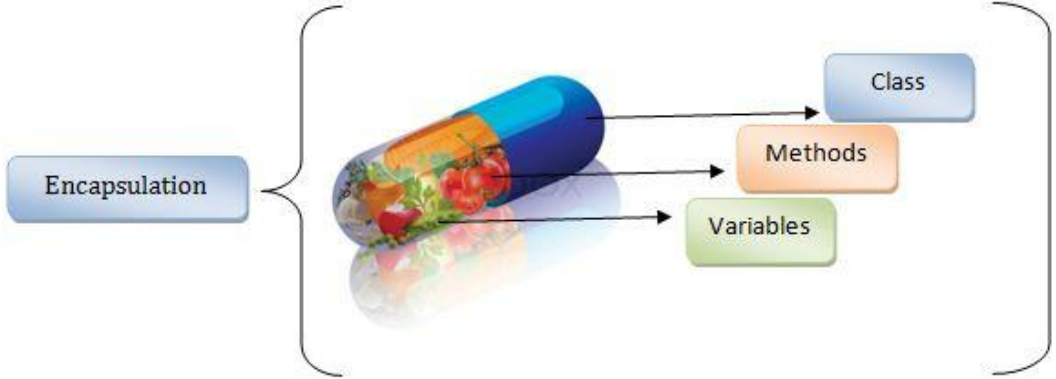
Loops in Action



Encapsulation

```
def square(t):
   for i in range(4):
     t.fd(100)
     t.lt(90)

square(bob)
```



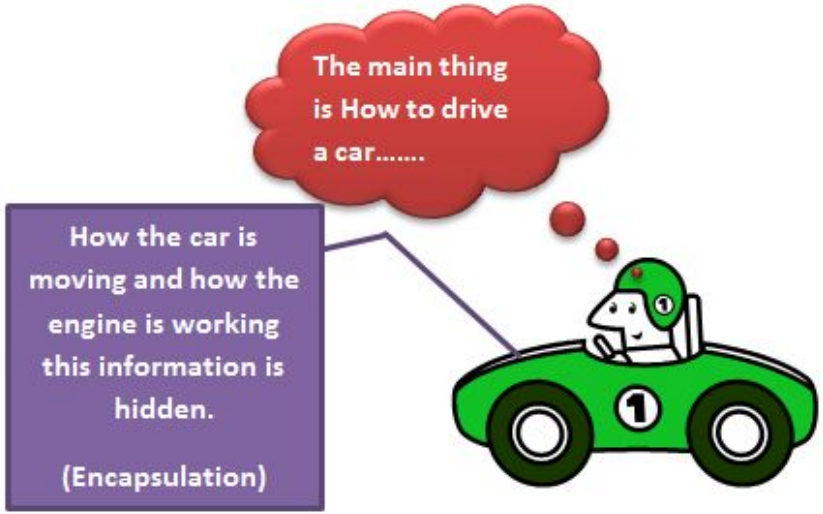
- Wrapping up a piece of code in a function

Encapsulation Benefits

- Naming a code (documentation)
- Code re-use
- Information hiding
- ...

Ex:

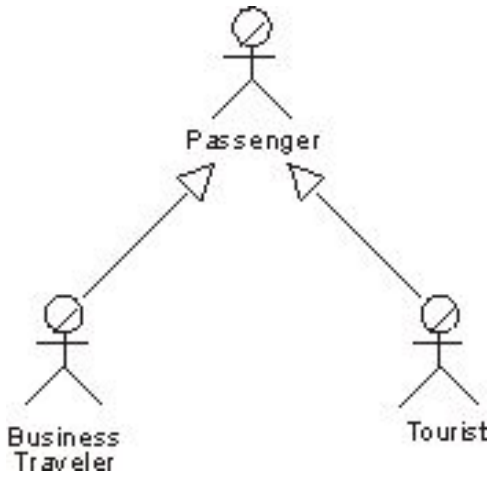
```
alice = turtle.Turtle()
square(alice)
```



Generalization

```
def square(t, length):
    for i in range(4):
        t.fd(length)
        t.lt(90)

square(bob, 100)
```



- Adding a parameter and making the function more general

```
polygon(bob, 7, 70) # even more generalized
```

Keyword Arguments

Ex:

```
def polygon(t, n, length):
    angle = 360 / n
    for i in range(n):
        t.fd(length)
        t.lt(angle)

polygon(bob, 7, 70)
polygon(bob, n=7, length=70)
```

- Parameter names as keywords (not Python keywords)

? Order of arguments