AI showcase: NewsBlaster

http://newsblaster.cs.columbia.edu/

Practice Using Variables

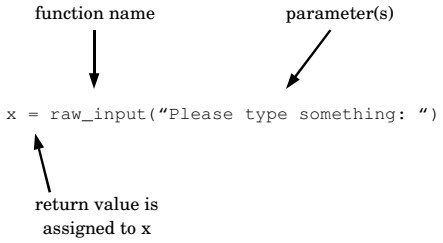
1) A coffee shop sells coffee at $10.50 a pound plus the cost of shipping. Each order ships for $0.86 per pound + $1.50 fixed cost for overhead. Write a program that calculates the cost of an order. (I.e., ask the user to type in how many pounds he wants, then calculate the cost of this order.

2) Write a program that determines the distance to a lightning strike based on the time elapsed between the flash and the sound of thunder. The speed of sound is approximately 1100 ft/sec and 1 mile is 5280 ft.

3) Write a program that calculates the cost per square inch of a circular pizza, given its diameter and price. To get the value of pi, import the `math` module (write `import math` at the top of your file). This module defines a name `math.pi` that refers to the value of pi.

Functions

Function calls



Practice Using Functions

A number guessing game:

> Both the computer and the user choose a number between 0 and 100. The higher number wins.

Implement this game. That is: write a program that randomly chooses a number between 0 and 100, then asks the user for a number between 0 and 100, and then prints out the higher number together with a statement that this is the winning number.

Hint: The library/module `random` provides a function `randint` that generates a random number between an upper and a lower bound. Check the module's documentation to find out how to use it.

Defining Functions

What's needed:

- name
- parameters (how many?, their names, maybe their types)
- body (the algorithm)
- return value (if there is one)

Defining Functions - example

- name: **avg**
- parameters: **x, y**
- body: **res = (x + y) / 2**
- return value: **res**

Practice Defining Functions

Write down specifications for the following functions. Use English (not Python) to specify the algorithm for the body of the function.

- a function that converts celsius to fahrenheit (Given a temperature in celsius, you have to multiply it by 9/5 and then add 32.)
- a function that echoes what the user types in, i.e., it reads in a string from the user and then prints the same string onto the screen
- a functions that sums up all integers up to a given integer

Practice Defining Functions in Python

```
def <name>(<parameter names>):
   <statements>
   return <expression>
```

optional

Example:

```
def avg (x, y):
   res = (x + y) / 2
   return res
```


Practice Defining Functions in Python

- Remember last week's program that printed a triangle of x's? Write a function that takes some character (string) as parameter and then prints a triangle using that character. For example, if the function is called with "t" as parameter, it should print

```
    t
   ttt
  ttttt
 ttttttt
```

- Write a function that computes the area of a circle given its radius.

Functions can call Functions

```
def happy():
    print "Happy Birthday to you!"

def sing(person):
    happy()
    happy()
    print "Happy Birthday, dear " + person + "."
    happy()

def main():
    sing("Fred")
    print
    sing("Lucy")
    print
    sing("Elmer")

main()
```


More Practice

Redo the pizza exercise:

> Write a program that calculates the cost per square inch of a circular pizza, given its diameter and price.

Use two functions – one to compute the area of a pizza, and one to compute the cost per square inch.

Why Functions?

- avoid duplication of code
  - less writing
  - easier to maintain
- breaking problems into manageable chunks
- hide implementation details

Variable Scope

The scope of a variable: the area of a program where this variable may be referenced (where this variable is visible).

Example:

```
me = "Kristina"

def sing(person):
    happy()
    happy()
    print "Happy Birthday, dear " + person + "."
    happy()

sing(me)
```


Variable Scope

The scope of a variable: the area of a program where this variable may be referenced (where this variable is visible).

Example:

scope of me

scope of person

```
me = "Kristina"

def sing(person):
    happy()
    happy()
    print "Happy Birthday, dear " + person + "."
    happy()

sing(me)
```


Scope Mysteries

On the course web site, you find 5 programs called scope1.py, ..., scope5.py. Download them, look at them, and run them. Explain for each one (in terms of variable assignment and scope), why the print statements print what they print.

control structures: if-statements

if *some condition is true*
then do *this*
else do *that*

if *the number input by the user is greater than the number randomly generated by the computer*
then *print out that the user has won*
else *print out that the computer has won*

if *there is a wall to the north and there is no wall to the west*
then *go west*
else *go south*

if-statements in Python

simple:

```
if <condition> :
   <statements>
```

two-way:

```
if <condition> :
   <statements>
else :
   <statements>
```

multi-way:

```
if <condition> :
   <statements>
elif <condition> :
   <statements>
elif <condition> :
...
else :
   <statements>
```

Conditions

Conditions are expressions that evaluate to `True` or `False`, i.e., expressions that create an object of type **boolean.**

Some comparison operators:

`==`	equal to	work for numbers and strings!
`!=`	not equal to	
`>`	greater than	
`<`	less than	
`>=`	greater or equal to	
`<=`	less than or equal to	

Examples of if-statements in Python

simple:

```
if x=="bye" :
   print "Bye, bye!"
   print "Nice talking to
                      you."
```

two-way:

```
if x > y :
   return x
else :
   return y
```

multi-way:

```
if x > y :
   print "You win!"
elif x < y :
   print "I win!"
else :
   print "It's a draw!"
```


More complex conditions

if it *rains* **or** *snows* **and** I **don't** *have an umbrella* ...

Boolean operators: `and, or, not`

```
if x>y

if x>y and y>z

if not(x>y and y>z)

if not(x>y and y>z) or x<z
```


Boolean algebra: Complex Conditions

and

True and True => True
True and False => False
False and True => False
False and False => False

or

True or True => True
True or False => True
False or True => True
False or False => False

not

not True => False
not False => True

Practice using if-statements in Python

- Implement a function that finds the greatest of three numbers. Don't use the built-in `max` function.
- Many companies pay time-and-a-half for any hours worked above 40 in a given week. Write a function that takes the number of hours worked and the hourly rate and calculates the total wages for the week.
- A person is eligible to be a US senator if they are at least 30 years old and have been a US citizen for at least 9 years. To be a US representative these numbers are 25 and 7, respectively. Write a program that asks for a person's age and years of citizenship as input and outputs their eligibility for the Senat and House.


```
import math

def floatRgb(mag, cmin, cmax):
       try:
              x = float(mag-cmin)/float(cmax-cmin)
       except:
              x = 0.5
       blue = min((max((4*(0.75-x), 0.)), 1.))
       red  = min((max((4*(x-0.25), 0.)), 1.))
       green= min((max((4*math.fabs(x-0.5)-1., 0.)), 1.))
       return (red, green, blue)

def strRgb(mag, cmin, cmax):
       red, green, blue = floatRgb(mag, cmin, cmax)
       return "#%02x%02x%02x" % (red*255, green*255, blue*255)

def rgb(mag, cmin, cmax):
       red, green, blue = floatRgb(mag, cmin, cmax)
       return (int(red*255), int(green*255), int(blue*255))

def htmlRgb(mag, cmin, cmax):
       return "#%02x%02x%02x"%rgb(mag, cmin, cmax)
```