Python Basics!

loops, iteration

CS101 Lecture #7

2016-09-14

Administrivia

Administrivia

- Homework #3 is due Friday Sep. 16.
- Midterm #1 will be Monday Oct. 3. (evening)

Warmup Quiz

Question #1

```
a = 1
def fun(a,b):
    return a + b
a = fun( a,a ) + a
```

What is the final value of `a`?

A 2

B 3

C 4

Question #1 (Worked)

```
a = 1
def fun(c,b):
    return c + b
a = fun( a,a ) + a
```

Question #2

```
x = 10
if ((x/2) < 5) or ((x%3) == 1):
    x = x + 2
if (x != 10) or ((x**2) <= 144):
    x = x * 2
```

What is the final value of x?

A `10`
B `12`
C `20`
D `24`

Question #3

```
def fun(x):
    if x and x:
        return not x
    return x or x

x = fun(True) or fun(False)
```

What is the final value of x?

A `True`

B `False`

Question #4

```
def fun(a,b):
    if len(a)+len(b)>5:
        return (a+b)[0:5]
  return (b+a)+str(len(a))

x = fun(”abc”,”def”) + fun(”gh”,”ij”)
```

What is the final value of x?

A `’abcdefijgh4’`

B `’defabcghij4’`

C `’abcdeijgh’`

D None of the above.

Question #5

The following code should increment x if the hundreds place contains a zero:

```
def fun(x):
    if x < 100 or ???:
        return x+1
    return x
```

What should replace the `???` to complete the code?

A `x.string(3) == '0'`

B `str(x)[-3] == '0'`

C `((x/100) % 10) == 0`

D None of the above.

Loops

Loops

- We frequently need to process each value in a set of values.

Loops

- We frequently need to process each value in a set of values.
- Two kinds: `while` and `for`

Example: *while* Loop

```
number = 10
while number > 0:
    print(number)
    number = number - 1
print('Blast off!')
```

Defining loops: *while*

- A `while` loop has only:
  - the keyword `while`
  - a logical comparison (`bool`-valued result)
  - a **block** of code

Example

The following code should increment x if the hundreds place contains a zero:

```
x = 3
while x > 0:
    print(”Hello”)
x -= 1
```

How many times is ’Hello’ printed?

A zero
B once
C twice
D thrice
E four times

Exercise

Write a program for a user to create a new password. The program should accept a password attempt from the user and check it with the function `validate_password`. If the password is valid, the program ends. If the password is invalid, the program asks for a new attempt, repeating until the user enters a valid password.

Solution

```
pwd = input(”Enter a password:  ”)
while not validate_password(pwd):
    pwd = input(”INVALID!  Try again:  ”)
print(”Your password is valid.”)
```

Infinite loops

- Make sure that your code always has a way to end!

Infinite loops

- Make sure that your code always has a way to end!

```
while True:
    print('Hello!')
```

Infinite loops

- Make sure that your code always has a way to end!

```
while True:
    print('Hello!')
```

- Use `Ctrl+C` to break free.

Accumulator pattern

- Design patterns are common structures we encounter in writing code.

Accumulator pattern

- Design patterns are common structures we encounter in writing code.
- The accumulator pattern uses an accumulator variable to track a result inside of a loop:

Accumulator pattern

- Design patterns are common structures we encounter in writing code.
- The accumulator pattern uses an accumulator variable to track a result inside of a loop:

```
i = 0
sum = 0
while i <= 4:
    sum += i
    i += 1
```

Example

```
i = 0
sum = 0
while i <= 4:
    sum += i
    i += 1
```

What is the value of `sum`?

A `6`
B `10`
C `15`
D None of the above.

Example

```
i = 0
sum = 0
while i < 7:
    if (i % 2) == 1:
        sum += i
        i += 1
```

What is the value of `sum`?

A `9`
B `12`
C `16`
D `21`

Exercise

Write a function to sum all of the digits in a number.

Exercise

Write a function to sum all of the digits in a number. I.e.,

$$12145 \rightarrow 1 + 2 + 1 + 4 + 5 \rightarrow 13$$

Solution (*while*)

```
def sum_digits( n ):
    s = str( n )
    i = 0
    result = 0
    while i < len( s ):
        result = result + int( s[i] )
        i = i + 1
    return result
```

Example: Loop

```
for number in 1,2,3,4,5:
    print(number)
```

Defining loops: *for*

- A `for` loop requires:
  - the keyword `for`
  - a loop variable
  - the keyword `in`
  - a set of values
  - a **block** of code
- `for` loops iterate over iterable types one at a time.

Example

```
s = 'abcdefg'
t = ''
for c in s:
    t = c + t
```

What is the value of `t`?

A `'abcdefg'`

B `'gfedcba'`

C `'a'`

D `'g'`

Exercise

Write a function to sum all of the digits in a number. I.e.,

$$12145 \rightarrow 1 + 2 + 1 + 4 + 5 \rightarrow 13$$

Solution (*for*)

```
def sum_digits( n ):
    result = 0
    for letter in str( n ):
        result += int( letter )
    return result
```

Reminders

Reminders

- Homework #3 is due Friday Sep. 16.
- Midterm #1 will be Monday Oct. 3. (evening)