Administtrivia

• Homework 3 assigned today
  – *It’s about twice as many questions*
REVIEW
s=“74.125.21.147”
i=s.find(“.”)
x=s[i+1:i+3]
x=x*2

What is the value of x?
a) “125125”
b) 250
c) “1212”
d) 24
WRITING FUNCTIONS
Defining functions

• We **define** a function by typing:
  1. the keyword **def**
  2. the name of our function
  3. a pair of parentheses
  4. a **block** of code
def greetings():
    print("Hola!")
    print("Bonjour!")
    print("Ni hao!")
    print("Hello!")
    print("Shalom!")
    print("Guten tag!")
    print ("Konnichiwa!")
    print ("As-salamu alaykum!")
Block

• A section of code grouped together
• Begins with a colon :
• Contents of the block are *indented*
  – “Tabbed in”

def hello():
    print(“hello”)

Scope

• Variables declared *inside* a block are independent of variables *outside* the block.

• Variables inside a block *do not exist* outside the block.

• Blocks are their own little world!

• Blocks are *isolated* from the rest of our code.
a=5
def fun():
a=3
b=4
a=a+b
fun()
print(a)
a=5
def fun():
a=3
b=4
a=a+b
fun()
print(a)
Return

• Our function can return a value (output).
• We use the keyword return.

```python
def three():
    return 3

def hello():
    return 0
    print("hello")
```
• Return immediately exits the function.
Parameters

• Our function can take *input* (arguments) as well.
• Parameters are variables declared in function header.

```python
def print_message(message):
    print(message)
```
• Multiple parameter are separated by commas.
def fun(a):
    return a+2

x=fun(2)*fun(3)

What is the value of x?

a) 9
b) 4
c) 16
d) None of the above.
def fun(m):
    return m.title().swapcase()

x = fun("abb") + fun("acab")

What is the value of x?

a) "AbbAcab"
b) "aBBaCAB"
c) "abbacab"
d) "ABBACAB"
def fun(a,b):
    c=((a+" ")*len(b)).title()

x=fun("ab","caa")

What is the value of x?

a) "ab ab ab ab"
b) "Ab Ab Ab Ab"
c) "AB AB AB AB"
d) None of the above.
def fun(a,b):
    c=((a+" ")*len(b)).title()
    return c

x=fun("ab","caa")

What is the value of x?

a) "ab ab ab ab"
b) "Ab Ab Ab Ab"
c) "AB AB AB AB"
BOOLEAN TYPE
Booleans

- A type with only **two values**:  
  - **True** and **False**
- Used to represent **logic**
- We’ll use them to **make decisions**.
- Based on **Boolean algebra**
- Operators for Boolean type:
  - and, or, not
# Logical operators

<table>
<thead>
<tr>
<th>and</th>
<th>True</th>
<th>False</th>
</tr>
</thead>
<tbody>
<tr>
<td>True</td>
<td>True</td>
<td>False</td>
</tr>
<tr>
<td>False</td>
<td>False</td>
<td>False</td>
</tr>
</tbody>
</table>

- **True when BOTH inputs are True**

<table>
<thead>
<tr>
<th>or</th>
<th>True</th>
<th>False</th>
</tr>
</thead>
<tbody>
<tr>
<td>True</td>
<td>True</td>
<td>True</td>
</tr>
<tr>
<td>False</td>
<td>True</td>
<td>False</td>
</tr>
</tbody>
</table>

- **True if ONE input is True**
### Logical operators

<table>
<thead>
<tr>
<th>not</th>
<th>result</th>
</tr>
</thead>
<tbody>
<tr>
<td>True</td>
<td>False</td>
</tr>
<tr>
<td>False</td>
<td>True</td>
</tr>
</tbody>
</table>

Inverts the input
def fun():
    return True and False

x = fun() and not (True or False)

What is the value of x?
a) True  
b) False
Comparison

• Operators that produce Boolean output
  <  less than
  <= less than or equal <=
  >  greater than
  >= greater than or equal
  == equal
  != not equal
a=5
b=3

\[ x = (a < 5) \text{ and } ((b \leq 5) \text{ or } (a \neq b)) \]

What is the value of \( x \)?
a) True
b) False
a="HAWKEYE"
b="IRON MAN"

x=a<b and a[1]!=b[-2]

What is the value of x?
a) True
b) False
def fun(a, b):
    return a < b

a = 3
b = 4
x = fun(b, a)

What is the value of x?

a) True
b) False
CONDITIONAL EXECUTION
Conditional Execution

• Make decisions in our program
• Change program behavior
  – Based on a Boolean value
• Change the control flow
If statement

• We create an *if statement* by typing:
  1. the keyword *if*
  2. a Boolean expression
  3. a *block* of code
print("Welcome to my program.")
input=input("Are you nice?")
if input=="Yes":
    print("Hello, friend!")
Alternative Execution

• Make decisions in our program
• Change program behavior
• Change the *control flow*
• Execute one block OR another block
If… else statement

• We create an \textit{if… else statement} with:
  1. the keyword \textit{if}
  2. a Boolean expression
  3. a \textit{block} of code
  4. the keyword \textit{else}
  5. another \textit{block} of code
print("Welcome to my program.")
input=raw_input("Are you nice?")
if input=="Yes":
    print("Hello, friend!")
else:
    print("HEY! BE NICE!")