Mid-class Review

Dr. Baldassano

chrisb@princeton.edu

Yu's Elite Education

Using variables

- Almost every program needs to keep track of information
- We want to be able to apply the same operation to different pieces of information
- A *variable* is a name we give to a piece of data
- Assign a variable using the assignment operator

  x = ‘World’

Variable operations

- We can change the value of a variable by assigning to it again

```
x = 10
x = 15
y = x+5
x = x+5
x = x+5
```

Variable Types

- Every variable is of a certain “type”
- In python types usually get determined automatically
- Calling type(varname) will give a variable’s type
- Some operations (like +) mean different things for different types

Questions

What is a function?

- A function is like a mini-program: it takes some information, and performs some action
- Variables passed into a function are called *arguments*
- Functions in python are called like:

```
functionName(argument1, argument2)
```

Two types of functions

- **Void** function: simply performs some action
  - print(‘This is a void function’)
- **Value-returning** function: performs some processing, then “returns” a value
  - x = input(‘This function returns a string: ’)
  - y = type(x)

Function syntax

```
def functionName(arguments):
    statement
    statement
    return variable     # if a value-returning function
```

Local variables

- Variables created or changed inside a function (including its names for the arguments) are *local* to the function don’t affect main program
- The part of a program where a variable lives is called its *scope*

Multiple function arguments

- Many functions take more than 1 argument
- Order matters!
- Some arguments may be optional
- Can override order of arguments by naming them

Questions

The if statement

- Python syntax:

```
if condition:
      Statement
      Statement
```

- First line is keyword `if` followed by condition
  - The condition can be true or false
  - If it is true the block statements are executed, otherwise block statements are skipped

Boolean Expressions

- The condition of an if statement is a “Boolean expression” that should have a value of either True or False
- Examples:
  - Function that returns True or False:

    ```
    if IsPrime(x):
    ```

  - Relational operator:

    ```
    if x > y:
    ```

Relational Operators

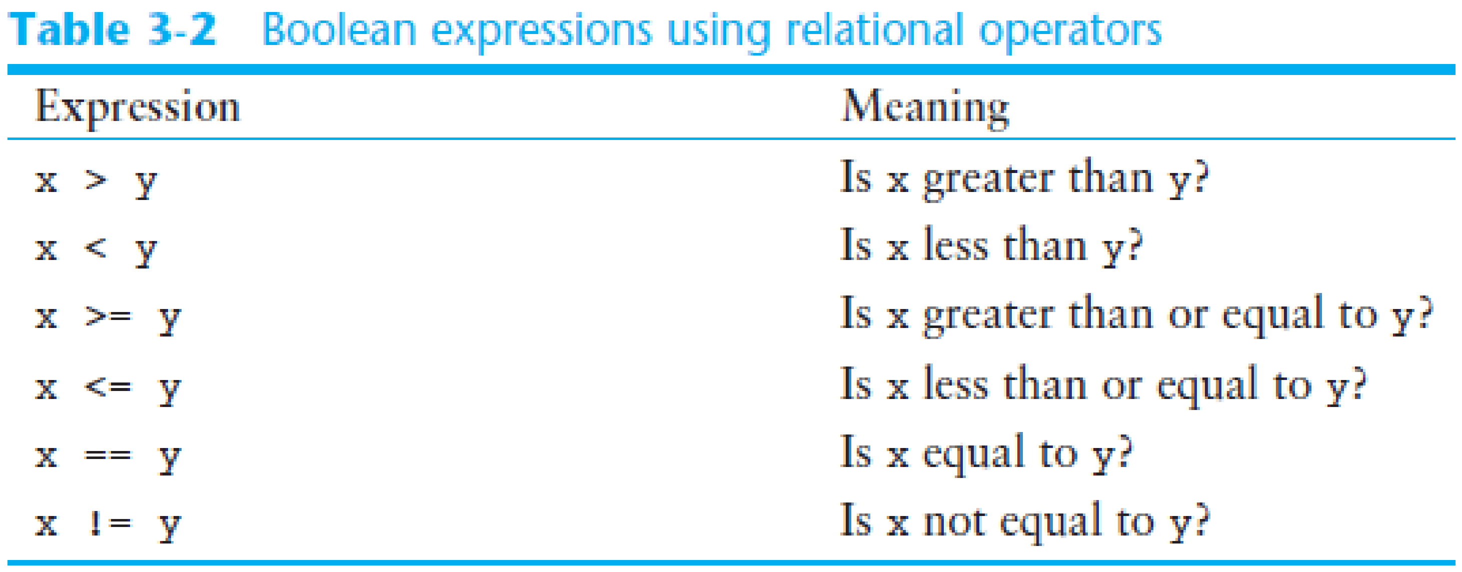

Table 3-2 Boolean expressions using relational operators

Expression	Meaning
`x > y`	Is x greater than y?
`x < y`	Is x less than y?
`x >= y`	Is x greater than or equal to y?
`x <= y`	Is x less than or equal to y?
`x == y`	Is x equal to y?
`x != y`	Is x not equal to y?

Logical Operators

- not: reverses the boolean value of what comes after it
  - `if not IsPrime(x):`
- and: true only if both sides are true
  - `if x > 5 and x < 10:`
- or: true if either side is true
  - `if x < 4 or x > 15:`

if - elif

```
if year == 2015:
   print('This year')
elif year == 2014:
    print('Last year')
else:
    print('A while back')
```

Questions

The `while` Loop: a Condition-Controlled Loop

- `while` loop: while condition is true, do something
  - Condition tested for true or false value
  - Statements repeated as long as condition is true
  - General format:

```
while condition:
    statements
```

The for loop

```
for x in range(1, 11):
    print(x)
```

- for [variable] in range([start], [stop]):
- Last number in loop is ONE LESS than stop

Questions

Using lists

- Creating a list: varname = [element, ...]
- Accessing a list:
  - varname[i] = element i (starting from 0)
  - negative i counts from the end
  - varname[i:j] = elements i up to j (not including element j)
- Can also create list of repeated elements using * operator: `list = [True] * 10`

List length function

```
scores = [9, 8.5, 4, 10]
print(len(scores))
for index in range(len(scores)):
    print(scores[index])
```

Building a list

- The append function adds a value to the end of the list

```
L = []
for n in range(2,11,2):
    L.append(n)
```

Strings: like read-only lists of characters

```
date = 'October 6th'
print(date[:7])
print(date[-3:])
print(date[:3] + ' ' + date[-3:])
```

Questions

Homework: Compute square root of number

- If r is the square root of X, then

  r = X/r
- Can find r by starting with a guess, then keep averaging r and X/r

X = 10

r = 1

r = (r + X/r)/2 = 5.5

r = (r + X/r)/2 = 3.659…

…