Week 4

Strings, `if/else`, `return`, user input

Special thanks to Scott Shawcroft, Ryan Tucker, and Paul Beck for their work on these slides.

Math commands

```
from math import *
```

Function name	Description
`abs(`**value**`)`	absolute value
`ceil(`**value**`)`	rounds up
`cos(`**value**`)`	cosine, in radians
`degrees(`**value**`)`	convert radians to degrees
`floor(`**value**`)`	rounds down
`log(`**value**`, `**base**`)`	logarithm in any base
`log10(`**value**`)`	logarithm, base 10
`max(`**value1**`, `**value2, ...**`)`	larger of two (or more) values
`min(`**value1**`, `**value2, ...**`)`	smaller of two (or more) values
`radians(`**value**`)`	convert degrees to radians
`round(`**value**`)`	nearest whole number
`sin(`**value**`)`	sine, in radians
`sqrt(`**value**`)`	square root
`tan(`**value**`)`	tangent

Constant	Description
`e`	2.7182818...
`pi`	3.1415926...

Strings

index	0	1	2	3	4	5	6	7
or	-8	-7	-6	-5	-4	-3	-2	-1
character	P	.		D	i	d	d	y

- Accessing character(s):

  variable [**index**]

  variable [**index1** : **index2**]

  - index2 exclusive
  - index1 or index2 can be omitted (end of string)

```
>>> name = "P. Diddy"
>>> name[0]
'P'
>>> name[7]
'y'
>>> name[-1]
'y'
>>> name[3:6]
'Did'
>>> name[3:]
'Diddy'
>>> name[:-2]
'P. Did'
```

String Methods

Java	Python
`length`	`len(`**str**`)`
`startsWith, endsWith`	`startswith, endswith`
`toLowerCase, toUpperCase`	`upper, lower, isupper, islower, capitalize, swapcase`
`indexOf`	`find`
`trim`	`strip`

```
>>> name = "Martin Douglas Stepp"
>>> name.upper()
'MARTIN DOUGLAS STEPP'
>>> name.lower().startswith("martin")
True
>>> len(name)
20
```

for Loops and Strings

- A for loop can examine each character in a string in order.

for **name** in **string**:
    statements

```
>>> for c in "booyah":
...     print c
...
b
o
o
y
a
h
```

input

- `input` : Reads a number from the user's keyboard.
  - You can store the result of `input` into a variable.
  - Example:

```
age = input("How old are you? ")
print "Your age is", age
print "You have", 65 - age, "years til retirement"
```

Output:

```
How old are you? 53
Your age is 53
You have 12 years til retirement
```

`raw_input`

`raw_input` : Reads a string from the user's keyboard.

– reads and returns an entire line of input

```
>>> name = raw_input("Howdy. What's yer name?")
Howdy. What's yer name? Paris Hilton

>>> name
'Paris Hilton'
```

Exercise

- Write a program that reads two employees' hours and displays each's total and the overall total.
  - Cap each day at 8 hours.

```
Employee 1: How many days? 3
Hours? 6
Hours? 12
Hours? 5
Employee 1's total hours = 19 (6.33 / day)

Employee 2: How many days? 2
Hours? 11
Hours? 6
Employee 2's total hours = 14 (7.00 / day)

Total hours for both = 33
```

Formatting Text

"**format string**" `%` (**parameter**, **parameter**, **...**)

- *Placeholders* insert formatted values into a string:

– `%d`	an integer
– `%f`	a real number
– `%s`	a string
– `%8d`	an integer, 8 characters wide, right-aligned
– `%08d`	an integer, 8 characters wide, padding with 0s
– `%-8d`	an integer, 8 characters wide, left-aligned
– `%12f`	a real number, 12 characters wide
– `%.4f`	a real number, 4 characters after decimal
– `%6.2f`	a real number, 6 total characters wide, 2 after decimal

```
>>> x = 3; y = 3.14159; z = "hello"
>>> print "%-8s, %04d is close to %.3f" % (z, x, y)
hello   , 0003 is close to 3.142
```

if

`if` **condition** `:`
statements

– Example:

```
gpa = input("What is your GPA? ")
if gpa > 2.0:
    print "Your application is accepted."
```

if/else

```
if condition:
    statements
elif condition:
    statements
else:
    statements
```

– Example:

```
gpa = input("What is your GPA? ")
if gpa > 3.5:
    print "You have qualified for the honor roll."
elif gpa > 2.0:
    print "Welcome to Mars University!"
else:
    print "Your application is denied."
```

Logical Operators

Operator	Meaning	Example	Result
`==`	equals	`1 + 1 == 2`	`True`
`!=`	does not equal	`3.2 != 2.5`	`True`
`<`	less than	`10 < 5`	`False`
`>`	greater than	`10 > 5`	`True`
`<=`	less than or equal to	`126 <= 100`	`False`
`>=`	greater than or equal to	`5.0 >= 5.0`	`True`

Operator	Example	Result
`and`	`(2 == 3) and (-1 < 5)`	`False`
`or`	`(2 == 3) or  (-1 < 5)`	`True`
`not`	`not (2 == 3)`	`True`

python™

String Comparison

- Can also use logical operators on strings!
- `"text" in str` as abbreviation for `str.find("text") != -1`

```
>>> def get_access(password):
...     if password == "one two three four five":
...             print "Access granted."
...     elif "one two three four" in password:
...             print "Oh, you were close!"
...
>>> get_access("one two three four six")
Oh, you were close!
>>> get_access("one two three four five")
Access granted.
```

Returning Values

```
def name(parameters):
    statements
    ...
    return value
```

```
>>> def ftoc(temp):
...     tempc = 5.0 / 9.0 * (temp - 32)↰
...     return tempc

>>> ftoc(98.6)↰
37.0
```

Exercise

- Write a program that encrypts a secret message by rotating the letters of the message.
  - e.g. `"Attack!"` when rotated by 1 becomes `"buubdl!"`

```
Encrypt or Decrypt? (E/D) E
What is the message? Attack!
How many rotations? 1
Here's the ciphertext: buubdl!

Encrypt or Decrypt? (E/D) D
What is the message? hal
How many rotations? -1
Here's the plaintext: ibm
```

Strings and Integers

- `ord(`**text**`)` - Converts a string into a number.
  - `ord("a")` is 97
  - `ord("b")` is 98
  - Uses standard mappings such as *ASCII* and *Unicode*.

- `chr(`**number**`)` - Converts a number into a string.
  - `chr(97)` is `"a"`
  - `chr(99)` is `"c"`