CS1001.py

Recitation 1 - Work in Progress

28.2-4.3.2013

Python general comments

1. Course site at http://tau-cs1001-py.wikidot.com
2. Programming language -> Interpreter -> Machine language
3. IDLE (editor + interpreter), see site for installation instructions
4. Interactive mode vs. Script mode
5. Python version 3.2

Variables, types

- int - integers: . . . , -3, -2, -1, 0, 1, 2, 3, . . .

```
x=5
y=-3
x, type(x)
y, type(y)
```

```
(-3, builtins.int)
```

- float - floating point numbers, decimal point fractions: -3.2, 1.5, 1e-8, 3.2e5

```
x=5.0
y=-3.2
z=2.2e6
x, type(x)
z, type(z)
```

```
(2200000.0, builtins.float)
```

- str - character strings, text: “intro2CS”, ‘python’

```
x = "intro2CS"
y = 'i love python'
x, type(x)
```

```
('intro2CS', builtins.str)
```

```
type(4), type(4.0), type("4")
```

```
(builtins.int, builtins.float, builtins.str)
```

- bool - boolean values: True and False **TODO**

Operators

Addition:

```
4 + 5
```

```
9
```

```
x = 5
4 + x
```

```
9
```

```
x = 4.0 + 5
x, type(x)
```

```
(9.0, builtins.float)
```

Subtraction:

```
x - 3
```

```
6.0
```

Multiplication:

```
x * 3
```

```
27.0
```

Division - float and integral with / and //:

```
10 / 3, 10//3
```

```
(3.3333333333333335, 3)
```

Power:

```
2 ** 3, 2 ** 3.0, 3 ** 2
```

```
(8, 8.0, 9)
```

String concatenation using +:

```
"Hello" + "World"
```

```
'HelloWorld'
```

String duplication using *:

```
"Bye" * 2
```

```
'ByeBye'
```

Strings vs. numbers:

```
4 + 5
```

```
9
```

```
"4" + "5"
```

```
'45'
```

```
"4" + 5
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-22-f945f8c7e111> in <module>()
----> 1 "4" + 5

TypeError: Can't convert 'int' object to str implicitly
```

Comparisons - **TODO**

Logical operatos and - **TODO**

Conversions

Use the functions `int()`, `float()`, and `str()` to convert between types (we will talk about *functions* next time):

```
int("6")
```

```
6
```

```
float("1.25")
```

```
1.25
```

```
str(4)
```

```
'4'
```

```
int("a")
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-26-91097a4105a2> in <module>()
----> 1 int("a")

ValueError: invalid literal for int() with base 10: 'a'
```

```
course = "intro" + str(2) + "cs"
course
```

```
'intro2cs'
```

Flow control

Conditional statements

The `if` condition formula - replace conditions and statements with meaningful code:

```
if *condition*:
    *statement*
    *statement*
    ...
```

```
elif *condition*: # 0 or more elif clauses
    *statement*
    *statement*
    ...
else:             # optional
    *statement*
    *statement*
```

Example:

```
today = "Sunday"
strike = "No"
my_recitation = "Monday"

if today == "Sunday":
    print("Shvizut Yom Alef")
    if strike == "Y":
        print("Stay home")
    else:
        print("Lecture in intro to CS!")
elif today == "Wednesday":
    print("Another lecture in intro to CS!")
elif today == my_recitation:
    print("Go to recitation!")
elif today == "Monday" or today == "Tuesday" or today == "Thursday" or \
                    today == "Friday" or today == "Saturday":
    print("no intro to CS")
else:
    print("Not a day")
```

```
Shvizut Yom Alef
Lecture in intro to CS!
```

Loops

While

```
while *condition*:
    *statement*
    *statement*
```

Example - count how many times 0 appears in an integer number:

```
num = 2**100
print(num)
count = 0

while num>0:   #what if we changed to >=0?
    if num % 10 == 0:
        count = count + 1
    num = num // 10

print(count)
```

```
1267650600228229401496703205376
6
```

For

```
for *variable* in *iterable*:
    *statement*
    *statement*
```

Example - solve the same problem with a `str` type instead of `int`:

```
num = 2**100
count = 0
for digit in str(num):
    if digit == "0":
        count = count + 1

print(count)
```

```
6
```

Builtin solution

```
num = 2**100
count = str.count(str(num), "0")

print(count)
```

```
6
```

Efficiency We can measure which solution is faster:

```
%%timeit
num = 2**100
count = 0
while num>0:   #what if we changed to >=0?
    if num % 10 == 0:
        count = count + 1
    num = num // 10

10000 loops, best of 3: 35.1 us per loop
```

```
%%timeit
num = 2**100
count = 0
for digit in str(num):
    if digit == "0":
        count = count + 1

100000 loops, best of 3: 6.01 us per loop
```

```
%%timeit
num = 2**100
count = str.count(str(num), "0")

1000000 loops, best of 3: 1.07 us per loop
```

The builtin solution is 4 times faster than the `for` solution which is 3 times faster than the `while` solution.

Other notes

- The `while` solution will not work for `num <= 0`
- The `while` solution will not work for non-numerals (e.g, `num = "Cola 0 is awesome!"`)
- The builtin solution is implemented with C and that is why it is faster

Fin

This notebook is part of the Extended introduction to computer science course at Tel-Aviv University.

The notebook was written using Python 3.2 and IPython 0.13.1.

The code is available at https://raw.github.com/yoavram/CS1001.py/master/recitation1.ipynb.

The notebook can be viewed online at http://nbviewer.ipython.org/urls/raw.github.com/yoavram/CS1001.py/master/recitation1.ipynb.

The notebooks is also available as a PDF at https://github.com/yoavram/CS1001.py/blob/master/recitation1.pdf?raw=true.