Python 101

Lec02

Ifs and Loops

thoum

August 5, 2019

Possible Projects

1. Beating sugang.snu.ac.kr's CAPTCHA
   - Digit recognition with kNN
2. Music Recognition with kNN
3. JoonggoNara Notifier

If

```
if some_boolean:
    pass #do something
```

If examples

```
score = int(input())

if 90 <= score and score <= 100:
    print('A')
# chained comparison (a little bit slower*)
elif 80 <= score <= 89:
    print('B')
elif 70 <= score <= 79:
    print('C')
elif 60 <= score <= 69:
    print('D')
else: # when every value above evals to False
    print('F')
```

*why?

More examples

```
# Membership
pengram =  "The quick brown fox jumps over the
    lazy dog"
if 'e' in pengram:
    print("YES")

# even?
if 33 % 2 == 0:
    print("EVEN")

if [1,2,3]:
    print("Non-zero is True")
if []:
    print("Empty is Null")
if '':
    print("NULL is False")
```

and more examples

```
ERROR_MSG = "lst is empty"

#'ask for permission' pattern
if len(lst) != 0:
    print(lst[0])
else:
    print(ERROR_MSG)

if lst:
    print(lst[0])
else:
    print(ERROR_MSG)
```

For Loops

~~Repeat N times~~

Iterate members of a sequence.

For Loops, C style

```c
#include <stdio.h>

int main()
{
  int i = 0;
  for (i = 0; i < 10; i++) {
    printf("%d", i);
  }
  return 0;
}
```

```
for value in iterable:
    pass
```

For Loops

```
for i in range(6):
    print("hello!")

nums = [1, 2, 3, 4, 5, 6]

# use len(nums) instead of 6
# access list elements with their index
for i in range(len(nums)):
    print(nums[i])
    nums[i] += 1

# get the values
for e in nums:
    print(e)
```

For Loops Cont'

```
for i in range(6):
    print("hello!")

nums = [1, 2, 3, 4, 5, 6]

# use len(nums) instead of 6
# access list elements with their index
for i in range(len(nums)):
    print(nums[i])
    nums[i] += 1

# get the values
for e in nums:
    print(e)
```

For Loops Nested

```
matrix = [[1,2,3], [4,5,6], [7,8,9]]

#double for-loop
for row in matrix:
    # prints each column
    # print(*row)
    for e in row:
        print(e, end=' ')
    print()

#triple
for i in range(1000):
    for j in range(1000):
        for k in range(1000):
            print(i, j, k)
```

While Loops

Repeat *while* condition(boolean) is satisfied.

```
while condition:
    pass
```

```
while True:
    print("Forever")

while False:
    print("Can I print?")

# Realistic use case
while True:
    s = input("Please Enter 'hi': ")
    if s == 'hi':
        break
```

Practice

Output: print 99dan from 1 to N. Hint:

Practice with stars!

Input: N
Print:

```
*
**
***

   *
  **
***

   *
  ***
*****
```

Hint:

Using for loops, create list of even numbers in [0, 10]

```
lst = []
for n in range(0, 11):
    if n % 2 == 0:
        lst.append(n)

# or

lst = []
for n in range(0, 11, 2):
    lst.append(n)
```

Comprehension

The easy way (and faster!)

```
lst = [n for n in range(0, 11, 2)]
# we can use if as well
lst = [n for n in range(11) if n % 2 == 0]
```

Performance Comparsion

time python3 test1.py

```
for _ in range(1000):
    lst = []
    for n in range(0, 100000):
        if n % 2 == 0:
            lst.append(n)
```

real: 12.744s

Performance Comparsion

```
for _ in range(1000):
    lst = []
    for n in range(0, 100000, 2):
        lst.append(n)
```

real: 5.019s

Performance Comparsion

```
for _ in range(1000):
    lst = [n for n in range(100000) if n%2==0]
```

real: 7.085s

Performance Comparsion

```
for _ in range(1000):
    lst = [n for n in range(0, 100000, 2)]
```

real: 1.356s

Comprehension

new_list = [expression *for* element *in* iterable (*if* condition)]

```
new_list = []
for element in iterable:
    if condition:
        new_list.append(expression)
```

Comprehension

```
lst = [n for n in range(10)]
lst == list(range(10)) # True

# getting input of 5 lines
# convention: we use '_' when we don't need the
    element (_=0,1,2,3,4)
input_nums = [int(input()) for _ in range(5)]
input_nums = [int(n) for n in input().split()]

lst = [n**2 for n in range(10)]
lst = [[0] * 10 for _ in range(10)]
upper = [c for c in 'AbcD' if c.isupper()]

# try to avoid this. We want to avoid side effects
lst = [print("HELLO") for _ in range(10)]
```

Generators

Lazy version of comprehension $[x \text{ for } x \text{ in...}] \rightarrow (x \text{ for } x \text{ in...})$

Performance Comparsion

```
for _ in range(1000):
    gen = (n for n in range(0, 100000, 2))
```

real: 0.053s

Fast, but has limited usage.

Practice

Mismatched ()s (*e.g.* "(()", ")(()") are painful when we are coding. Write a program that prints 'yes' when the ()s match, and 'no' when they do not match.

```
())()
no
(())()
yes
```