Problem 1: string comparison

Hint: the following is True: "" < "0" < "9" < "A" < "Z" < "a" < "z"

Circle the expressions that are True:

"a" < "z"	"ax" < "ay"	"abc" < "abCd"
"a" < "Z"	"x2" < "x1"	"zero" < "999"
"x" < "x"	"abcX" < "abcY"	"10" < "999"
"0" < "x"	"abcX" < "aBcY"	"1000" < "999"
"1" < "0"	"abc" < "abcd"	"88888888888" < "9"

Problem 2: string ~~functions~~ methods

Functions: upper, lower, strip, rstrip, lstrip, format, startswith, endswith, find.

Expression:	Value (put in quotes):
"dog".upper()	"DOG"
"Dog".lower()	"dog"
" paint ".strip()	"paint"
" paint ".rstrip()	" paint"
"val: {}%".format(99)	
"{} {}".format("X", "Y")	

Expression:	Value
"abcd".startswith("ab")	True
"abcd".endswith("bc")	False
"abcd".find("a")	0
"abcd".find("c")	2
"abcd".find("B")	-1
"Python".find("th")	2

Problem 3: sequence indexing

Assume **msg** is "Hello" and **x** is "num= 13". Some expressions cause an error.

Expression	Result
"abc"[0]	a
"abc"[2]	c
"abc"[-1]	c

Expression	Result
msg[4]	o
msg[5]	ERROR
msg[len(msg)]	!!

Expression	Result
x[len(x) - 1]	3
x[3]	=
x[1] + x[2]	um

0	1	2	3	4
H	e	l	l	o

x[7-1] = x[6]

msg: 0 1 2 3 4 — H e l l o

x: 0 1 2 3 4 5 6 — n u m = 1 3

Problem 4: sequence slicing

Assume **msg** and **x** are as before, and **p** is "= ".

Expression	Result
"abcde"[0:2]	
"abcde"[2:6]	cde
"abcde"[2:9]	

Expression	Result
msg[:2]	He
msg[2:]	
msg[-2:]	

Expression	Result
x[:msg.find('=')]	num
x[msg.find(' ')+1:]	
x[msg.find(p)+len(p):]	

x[:3] = "num"

Problem 5: for loop over sequence

What does the following code print?

```
msg = "301"
A = ""
B = ""

for character in msg:
    print(msg)
    A = A + character + "."
    B = character + B
```

msg: 0 1 2 — 3 0 1

A: "3." "3.0." "3.0.1."

B: "3" "03" "103"

What is in A afterwards?

"1" + "3" + "."

What is in B afterwards?

Problem 6: for loop over range

What does this code print?

```
s = "PYTHON"
for i in range(len(s)):
    print(s[:i+1])
```

6

i: 0 1 2 3 4 5

A = "3." + "0" + "."

B = "0" + "3" = "03"

P

PY

s[:1]

s[:2] . . . s[:6]

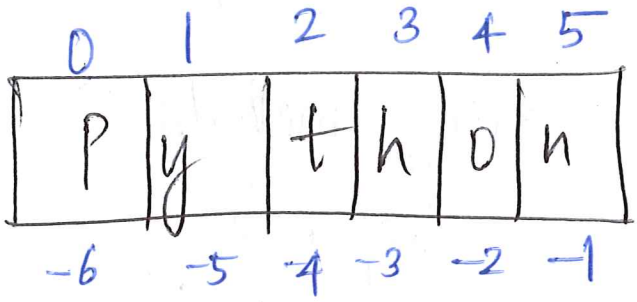
Lecture - 10

Projects

Review

Strings

msg = "Python"

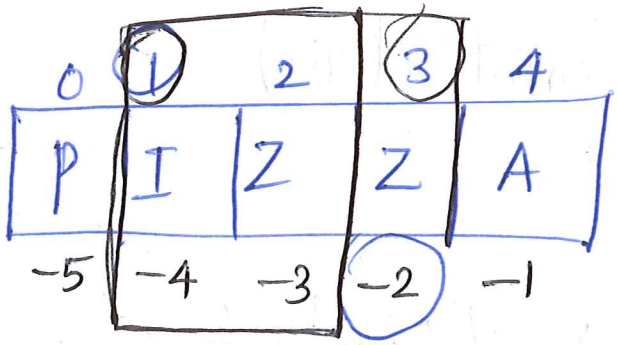


1. Indexing

msg[2] ⟹ 't'

msg[-4] ⟹ 't'

2. Slicing



S = "PIZZA"

S[1:3] ⟹ 'IZ'

S[:2] ⟹ 'PI'

S[2:] ⟹ 'ZZA'

[1, 3)

included

excluded

3. len(s)

4. Immutable

~~s[0] = 'x'~~

5. String methods

upper()
isdigit()
find()

---

for elem in s:
    print(elem)

Comparison

'a' < 'b' < 'c' ... < 'z'

'A' < 'B' < 'C' ... < 'Z'

'0' < '1' < '2' ... < '9'

③

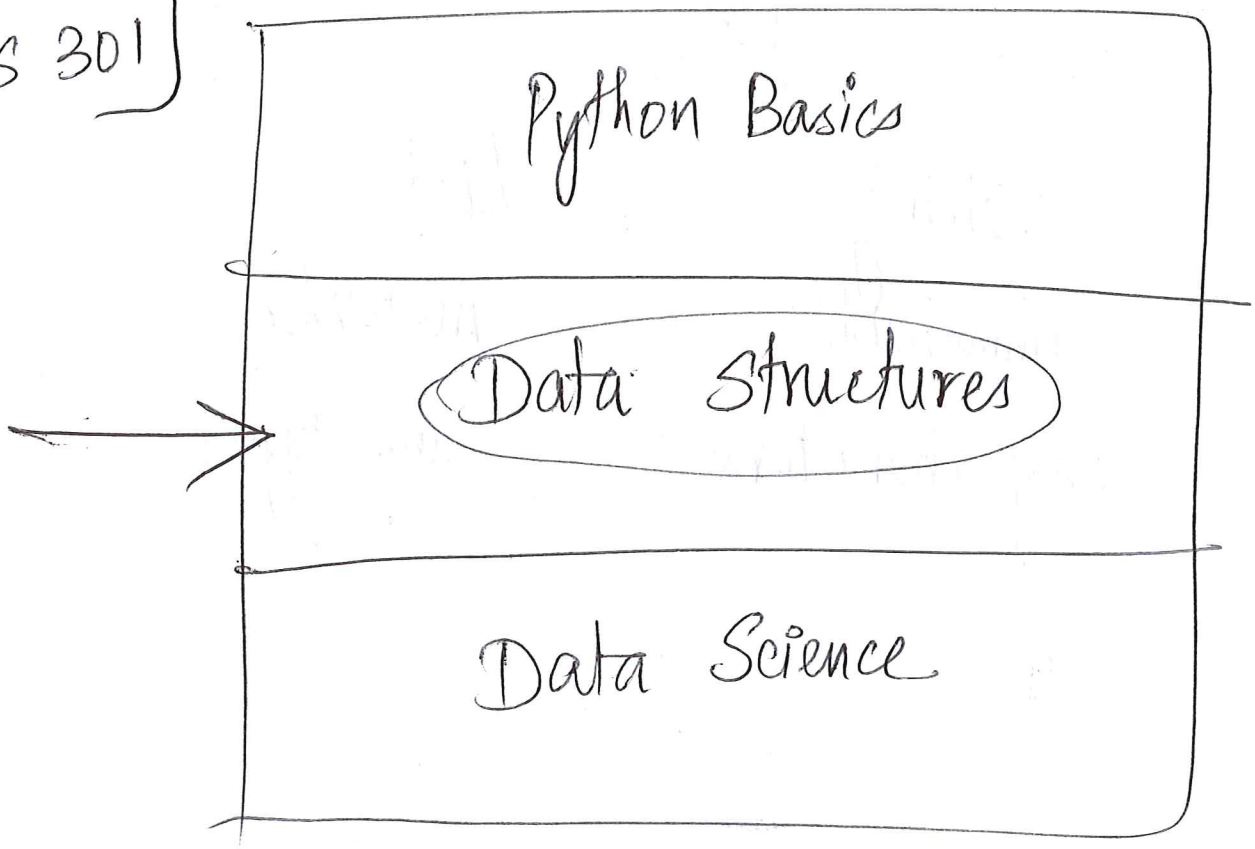
'A' < 'a'

'0' < 'A'

'abc' < 'ade'

'0' < '1' ... < '9' < 'A' < ... < 'Z' < 'a' ... < 'z'

[CS 301]



List

index: 0 1 2 3 4

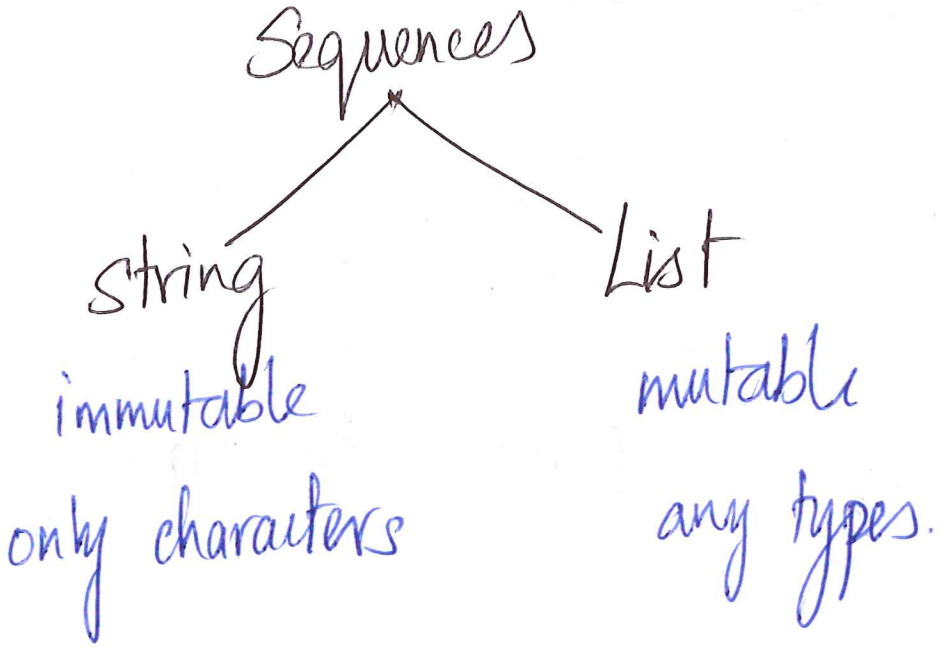
scores = [100, 95, 71, 82, 42]

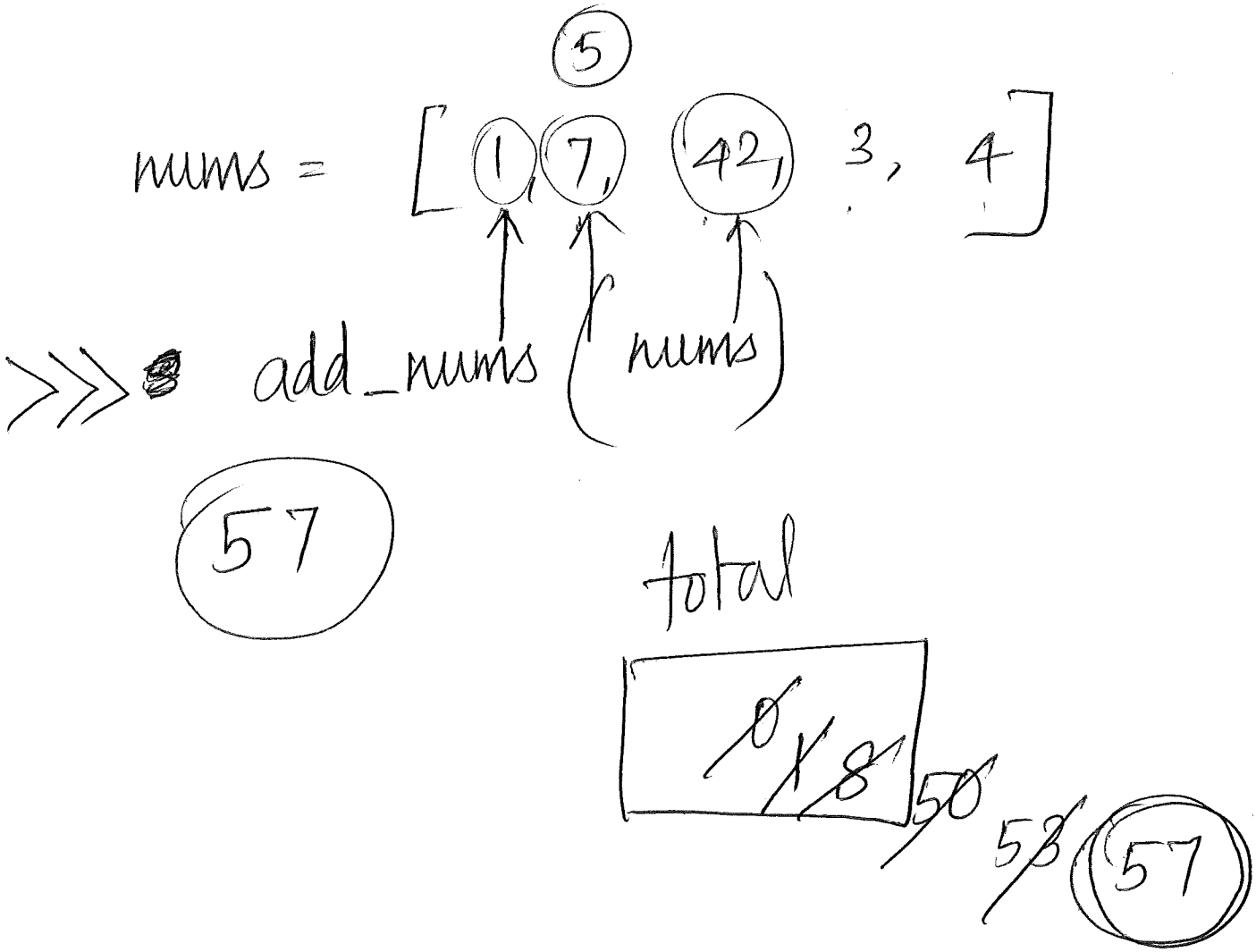
index: -5 -4 -3 -2 -1

List Sequence of values (any type)

string - sequence of characters.

Lists are mutable!




5
nums = [1, 7, 42, 3, 4]
>>> add_nums (nums)
57
total
0
8
50
58
57