



Python Programming

Лекц 5

Багш Ж.Золжаргал
Х.Хулан

Өнөөдөр



- Өгөгдлийн төрлүүд: int, float, bool, string
- Нийлмэл өгөгдлийн төрөл
 - tuples
 - lists
- Alias-ийн тухай
- Mutability тухай
- Cloning тухай

TUPLES

- Элемэнтүүдийн дараалал, элемэнтийн төрөл хольж болно
- Элемэнтийн утгыг өөрчилж болохгүй, **immutable**
- Хаалтанд дүрсэлнэ

`te = ()` *empty tuple*

`t = (2, "mit", 3)`

`t[0]` → evaluates to 2

`(2, "mit", 3) + (5, 6)` → evaluates to `(2, "mit", 3, 5, 6)`

`t[1:2]` → slice tuple, evaluates to `("mit",)`

`t[1:3]` → slice tuple, evaluates to `("mit", 3)`

`len(t)` → evaluates to 3




`t[1] = 4` → gives error, can't modify object

*extra comma
means a tuple
with one element*

TUPLES



- Хувьсагчийн утгыг солиход тохиромжтой (**swap**)

<code>x = y</code>		<code>temp = x</code>		<code>(x, y) = (y, x)</code>
<code>y = x</code>		<code>x = y</code>		
		<code>y = temp</code>		

- Функцийн хувьд нэгээс илүү утга буцаахад ашигладаг

```
def quotient_and_remainder(x, y):
```

```
    q = x // y
```

```
    r = x % y
```







```
    return (q, r)
```

*integer
division*

```
(quot, rem) = quotient_and_remainder(4, 5)
```

MANIPULATING TUPLES



aTuple: (( ), ( ), ( ))

ints *strings*

- Tuple дээр давтаж болно

```
def get_data(aTuple):
```

```
    nums = ()
```

```
    words = ()
```

```
    for t in aTuple:
```

```
        nums = nums + (t[0],)
```

```
        if t[1] not in words:
```

```
            words = words + (t[1],)
```

```
    min_n = min(nums)
```

```
    max_n = max(nums)
```

```
    unique_words = len(words)
```

```
    return (min_n, max_n, unique_words)
```

empty tuple

singleton tuple

nums (

words (? ? ?)

*if not already in words
i.e. unique strings from aTuple*

LISTS



- Индексээр хандах боломжтой мэдээллийн дараалсан дараалал
- Жагсаалтыг дөрвөлжин хаалтаар тэмдэглэнэ, []
- Жагсаалтын агуулах өгөгдлүүд нь
 - Ихэвчлэн нэгэн төрлийн (integer)
 - Холилдсон төрөл байж болно (түгээмэл биш)
- Жагсаалтын элементийг өөрчлөх боломжтой

INDICES AND ORDERING



`a_list = []` *empty list*

`L = [2, 'a', 4, [1, 2]]`

`len(L)` → evaluates to 4

`L[0]` → evaluates to 2

`L[2]+1` → evaluates to 5

`L[3]` → evaluates to `[1, 2]`, another list!

`L[4]` → gives an error

`i = 2`

`L[i-1]` → evaluates to 'a' since `L[1]='a'` above

CHANGING ELEMENTS

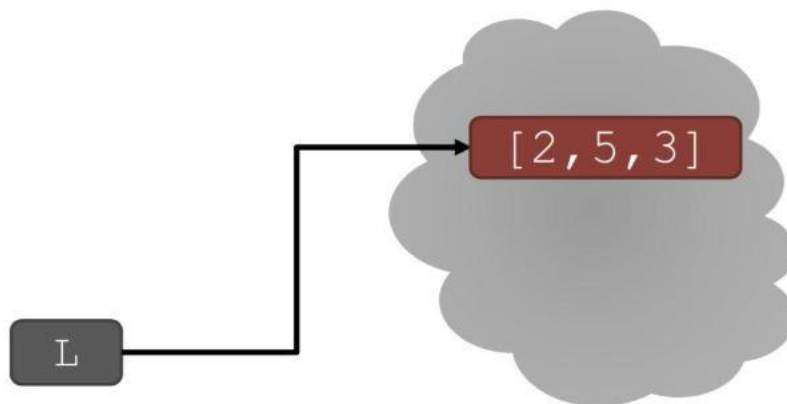


- Жагсаалтыг **өөрчлөх** боломжтой
- Индекс дэх элементэд оноож өгсөнөөр утгыг өөрчилдөг.

$L = [2, 1, 3]$

$L[1] = 5$

- Одоо L нь $[2, 5, 3]$, Энэ нь өмнөх L тэй ижилхэн объект



ITERATING OVER A LIST



- Жагсаалтын элементүүдийн нийлбэр олох

```
total = 0
for i in range(len(L)):
    total += L[i]
print total
```

```
total = 0
for i in L:
    total += i
print total
```

- Мэдэгдэл
 - Жагсаалтын элементүүд нь $0 - \text{len}(L) - 1$ хооронд индекслэгдсэн байдаг.
 - `range(n)` нь 0 -оос $n-1$ утгууд авна

OPERATIONS ON LISTS - ADD



- Жагсаалтын төгсгөлд **элемент** нэмэх

- `L.append(element)`

- Жагсаалтыг **өөрчлөх**

- `L = [2, 1, 3]`

- `L.append(5)` \rightarrow L is now `[2, 1, 3, 5]`



- Энэ цэг юу вэ?
 - Жагсаалт нь паятоны объект, Паятонд бүх зүйл объект
 - Объект нь өгөгдөлтэй
 - Объект нь функцтэй
 - Мэдээлэлд хандах `object_name.do_something()`

OPERATIONS ON LISTS - ADD



- Жагсаалтуудыг залгах, + үйлдэл нь шинэ жагсаалт үүсгэнэ
- Жагсаалтыг өөрчлөх
 - `L.extend(some_list)`

`L1 = [2, 1, 3]`

`L2 = [4, 5, 6]`

`L3 = L1 + L2`

→ `L3` is `[2, 1, 3, 4, 5, 6]`
`L1`, `L2` unchanged

`L1.extend([0, 6])`

→ mutated `L1` to `[2, 1, 3, 0, 6]`

OPERATIONS ON LISTS - REMOVE

- Заасан индекстэй элемент устгах
 - `del(L[index])`
- Жагсаалтын сүүлийн элемент устгах, устсан элементийг буцаана
 - `L.pop()`
- Заасан элементийг устгах `L.remove(element)`
 - Элементийг хайгаад устгана
 - Хэрэв олон байвал эхнийхийг, үгүй бол алдаа

all these
operations
mutate
the list

```
L = [2, 1, 3, 6, 3, 7, 0] # do below in order
L.remove(2) → mutates L = [1, 3, 6, 3, 7, 0]
L.remove(3) → mutates L = [1, 6, 3, 7, 0]
del(L[1])   → mutates L = [1, 3, 7, 0]
L.pop()     → returns 0 and mutates L = [1, 3, 7]
```

CONVERT LISTS TO STRINGS AND BACK



- Тэмдэгт мөрийг жагсаалт руу хувиргах
 - `list(s)` - `L` урттай, `s` тэмдэгт мөрийн тэмдэгтүүдээс тогтсон жагсаалт буцаана
- `s.split()` - тэмдэгт мөрийг салгах, параметргүй үед хоосон зайгаар салгана
- `"".join(L)` - жагсаалтын тэмдэгтүүдийг string рүү хөрвүүлнэ

```
s = "I<3 cs"
```

```
list(s)
```

```
s.split('<')
```

```
L = ['a', 'b', 'c']
```

```
' '.join(L)
```

```
'_'.join(L)
```

→ `s` is a string

→ returns `['I', '<', '3', ' ', 'c', 's']`

→ returns `['I', '3 cs']`

→ `L` is a list

→ returns `"abc"`

→ returns `"a_b_c"`

OTHER LIST OPERATIONS



- `sort()` ба `sorted()`
- `reverse()`
- болон дэлгэрэнгүйг
 - <https://docs.python.org/3/tutorial/datastructure.html>

-
`L = [9, 6, 0, 3]`

`sorted(L)` → returns sorted list, does **not mutate** L

`L.sort()` → **mutates** L = [0, 3, 6, 9]

`L.reverse()` → **mutates** L = [9, 6, 3, 0]

AN ANALOGY



- Хүний атрибут
 - Singer, rich
- Түүнийг олон нэрээр нь мэднэ
- Бүх хоч нь нэг хүнийг илэрхийлнэ
 - Нэг хочинд атрибут нэмэх

Justin Bieber

singer

rich

troublemaker

- Бүх хоч нь ижилхэн хүн болон ижилхан атрибут

The Bieb

singer

rich

troublemaker

JBeebs

singer

rich

troublemaker

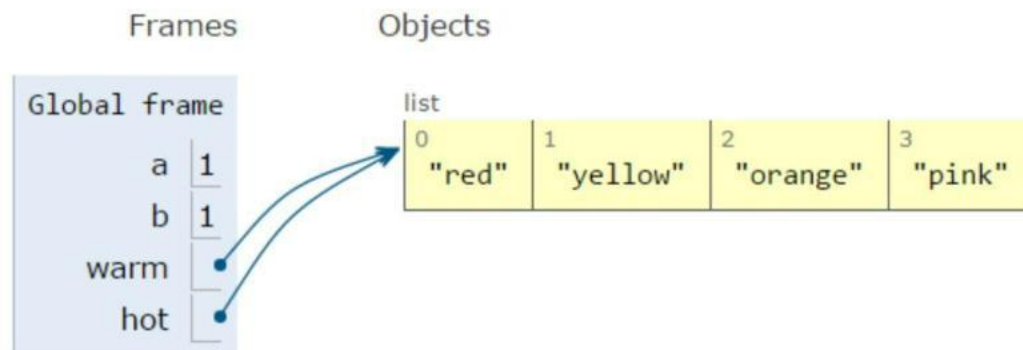
ALIASES



- Hot нь warm-ийн өөр нэршил - нэг өөрчлөлт нь нөгөөхөд нь ч адил
- `append()` - нөлөө үзүүлнэ

```
1 a = 1
2 b = a
3 print(a)
4 print(b)
5
6 warm = ['red', 'yellow', 'orange']
7 hot = warm
8 hot.append('pink')
9 print(hot)
10 print(warm)
```

```
1
1
['red', 'yellow', 'orange', 'pink']
['red', 'yellow', 'orange', 'pink']
```



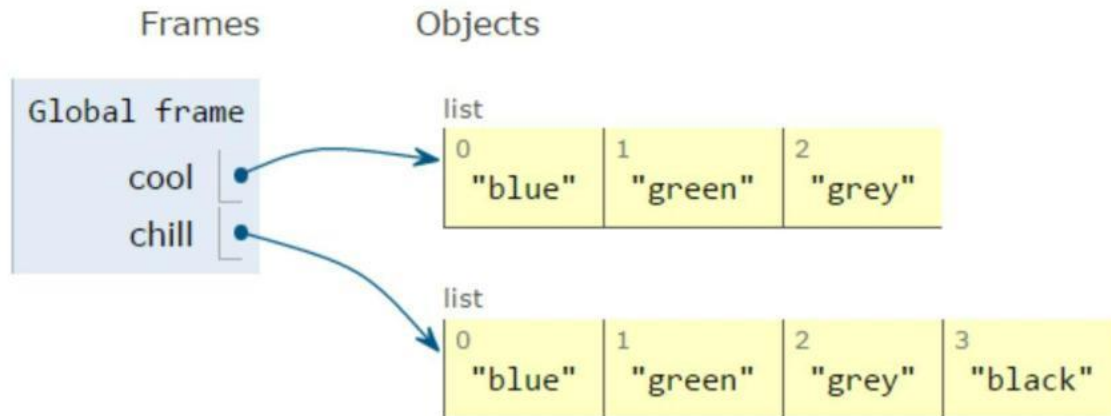
CLONING A LIST



- Бүх элементийг хуулж шинэ жагсаалт үүсгэх
 - `Chill = cool[:]`

```
1 cool = ['blue', 'green', 'grey']
2 chill = cool[:]
3 chill.append('black')
4 print(chill)
5 print(cool)
```

```
['blue', 'green', 'grey', 'black']
['blue', 'green', 'grey']
```

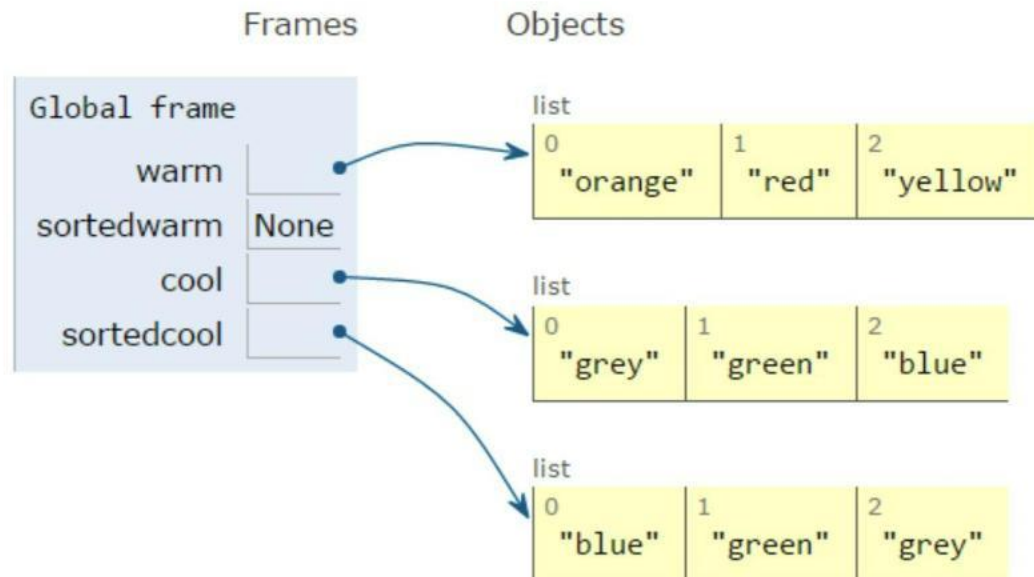


SORTING LIST



- `sort()` - дуудахад жагсаалт өөрчлөгдөж, үрдүн буцаахгүй
- `sorted()` - дуудахад жагсаалт өөрчлөгдөхгүй, өөрчлөгдсөн утга буцна.

```
1 warm = ['red', 'yellow', 'orange']
2 sortedwarm = warm.sort()
3 print(warm)
4 print(sortedwarm)
5
6 cool = ['grey', 'green', 'blue']
7 sortedcool = sorted(cool)
8 print(cool)
9 print(sortedcool)
```

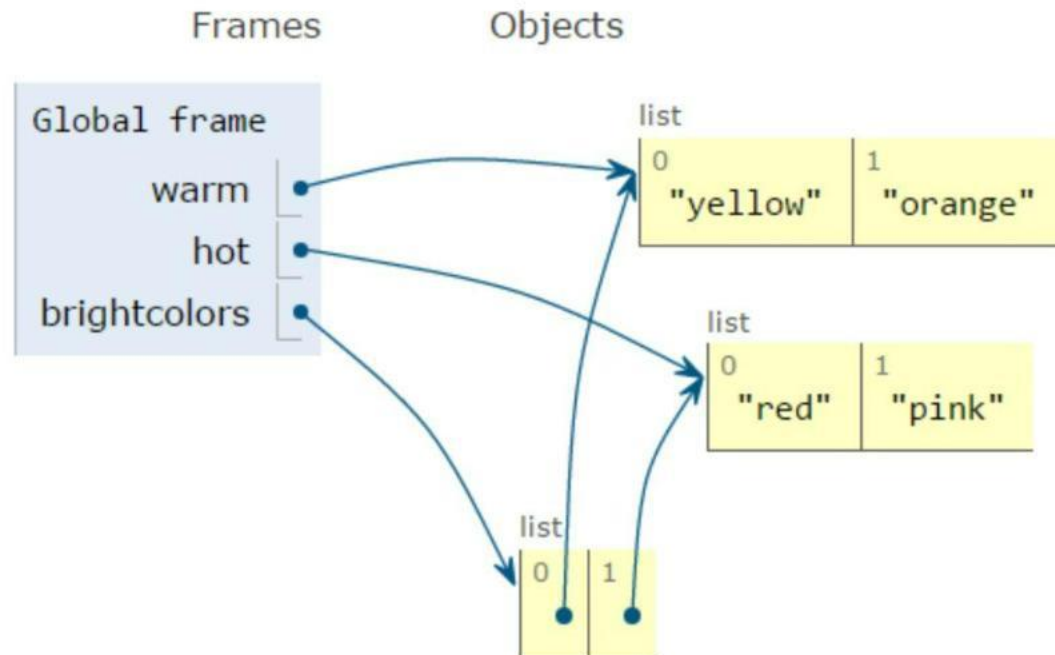


LIST OF LIST OF LIST OF ...

- **Nested** жагсаалт байж болно


```
[['yellow', 'orange'], ['red']]  
['red', 'pink']  
[['yellow', 'orange'], ['red', 'pink']]
```

```
1 warm = ['yellow', 'orange']  
2 hot = ['red']  
3 brightcolors = [warm]  
4 brightcolors.append(hot)  
5 print(brightcolors)  
6 hot.append('pink')  
7 print(hot)  
8 print(brightcolors)
```




MUTATION AND ITERATION

- Давталын явцад жагсаалтыг өөрчлөхөөс зайлсхий



```
def remove_dups(L1, L2):  
    for e in L1:  
        if e in L2:  
            L1.remove(e)
```

```
L1 = [1, 2, 3, 4]  
L2 = [1, 2, 5, 6]  
remove_dups(L1, L2)
```



```
def remove_dups(L1, L2):  
    L1_copy = L1[:]  
    for e in L1_copy:  
        if e in L2:  
            L1.remove(e)
```

clone list first, note
that L1_copy = L1
... NOT clone

- L1 нь [2, 3, 4] болохоос [3, 4] биш. Яагаад?
 - Паятон давталтанд дотоод индекс ашигладаг
 - Жагсаалтын урт өөрчлөгдөхөд, Паятон нь тоолуураа шинэчлэхгүй
 - Давталт хэзээ ч 2 дахь утга авахгүй