Compound data types1 (tuple and list)

Ilhoon Shin

tuple

- An ordered sequence of elements (similar with a list)
  - a collection of values that are wrapped with parenthesis, ().
    - (1, 2, 3)
    - (1, ‘aa’)
    - (2, 4)
  - We can identify each value by an order, i.e., an index.
    - a = (1, 2, 3)
    - print(a)
    - print(a[0])
    - print(a[1])
  - Elements can have different types.
    - (1, ‘aa’)
  - Immutable
    - a[0] = 4 (X)

operations

- Indexing (same with a string)
  - a = (0, 1, 2)
  - print(a[0])
- slicing (same with a string)
  - a[start:stop:step]
  - print(a[0:2:1])
- concatenating (same with a string)
  - b = a + (3, 4)
- get a size
  - len()

operations

- lab

```
a = (0, 1, ‘aa’, 3)
print(a[2])
print(a[0:3:2])
b = a + (‘bb’, 5)
print(b)
print(len(b))
```

list

- An ordered sequence of elements (similar with a tuple)
  - a collection of values that are wrapped with blackets, [].
    - [1, 2, 3]
    - [1, ‘aa’]
    - [2]
    - []
  - We can identify each value by an order, i.e., an index.
    - a = [1, 2, 3]
    - print(a)
    - print(a[0])
    - print(a[1])
  - Elements can have different types.
    - [1, ‘aa’]

list

- Differences from a tuple
  - Mutable
    - a[0] = 4 (O)
  - Its size is not fixed. (big advantage from a programmer's perspective)
    - We can expand or shrink a list (append, remove, insert, …)
    - easy to write a list managing code (Note that an array is static in C language.)

manipulating list1

- Indexing
  - a = [0, 1, 2]
  - print(a[0])
- slicing
  - a[start:stop:step]
  - print(a(0, 1, 1))
- concatenating
  - b = a + [3, 4]
- get a size
  - len()
- get a sum
  - sum()

manipulating list1

- lab

```
a = [0, 1, 2]
print(a[0])
a[0] = -1
print(a)
print(a[0:2:1])
b = a + [3, 4]
print(b)
print(len(b))
print(sum(b))
```

manipulating list2

- append
  - b.append(5) # append() gets a scalar
- extend
  - b.extend([6, 7]) # extend() gets a list
- insert
  - b.insert(2, 3) # insert 3 into index 2
- remove
  - b.remove(6) # remove the first element whose value is 6
  - b.pop() # remove and return the last element
  - b.pop(2) # remove and return the element whose index is 2
- move
  - we can implement move by using pop() and insert().

manipulating list2

- lab

```
b = [0, 1, 2, 3, 4]
b.append(5)
print(b)
b.extend([6, 7])
print(b)
b.insert(0, -1)
print(b)
b.remove(1)
print(b)
k = b.pop()
print(k, b)
k = b.pop(0)
print(k, b)
```

manipulating list3

- sorting
  - list = [3, 7, 1, 2]
  - list.sort() # list = [1, 2, 3, 7]
  - list2 = sorted(list) # list = [3, 7, 1, 2]
    # list2 = [1, 2, 3, 7]
  - list.reverse() # list = [7, 3, 2, 1]

copy a list

- l1 = [1, 2, 3]
- l2 = l1
  - Both l1 and l2 point to the same object, [1, 2, 3]
- lab

```
l1 = [1, 2, 3]
l2 = l1
l2.append(4)
print(l1, l2)
```

copy a list

- To copy a list, use slicing or list()
  - l2 = l1[::] # l1[0:len(l1):1]
    # l1 and l2 point to the different objects.
  - l3 = list(l2) # l2 and l3 point to the different objects.
- lab

```
l1 = [1, 2, 3]
l2 = l1[0:3:1]                    # l2 = l1[:] or l2 = l1[::]
l2.append(4)
l3 = list(l2)
l3.append(5)
print(l1, l2, l3)
```

convert a list to a string and vice versa

- from string to list
  - list(s) : returns a list with every character from s an element in the list
  - lab

```
l = list('abc')
print(l)
```

convert a list to a string and vice versa

- from string to list (split)
  - s.split(delimiter)
    - By default, the delimiter is space.
  - lab

```
s = ‘a,b,c,d’
l = s.split(‘,’)
print(l)
s2 = ‘a b c d’
l2 = s2.split()
print(l2)
```

convert a list to a string and vice versa

- from list to string (join)
  - s.join(list)
    - Each element of a list should be a string.
    - s is inserted between each element
  - lab

```
l = [‘a’, ‘b’, ‘c’]
s1 = ‘’.join(l)
s2 = ‘_’.join(l)
print(s1)
print(s2)
```

Multidimensional List

- Each element of a list can be another list.
- Lab

```
a = [[1, 2, 3], [4, 5]]
print(a)
a.append(6)
print(a)
```

example1

- Get 5 integers from a user, store them in a list, and print the sum.

example1

```
numbers = []
for count in range(5):
        num = int(input('input a number'))
        numbers.append(num)


sum = 0
for num in numbers:
        sum += num
print(sum)
```

example1

```
def get_sum(numbers) :
        sum = 0
        for num in numbers:
                sum += num
        return sum

numbers = []
for count in range(5):
        num = int(input('input a number'))
        numbers.append(num)

print(get_sum(numbers))
```

example2

- Get 5 integers from a user, store them in a list, and print the sorted list in the ascending order.

example2

```
numbers = []
for count in range(5):
        num = int(input('input a number'))
        numbers.append(num)

numbers.sort()
print(numbers)
```

exercise1

- Get a positive integer from a user and store all even numbers that are less than the integer in a list. Print the list.
  - if a user types 5, then [2, 4] should be printed.

exercise2

- Get 5 integers from a user, store them in a list, and print the maximum number.

exercise3

- Get 5 integers from a user and print the list sorted in descending order. (Don’t use reverse()).