1) بله لیست ها در پایتون میتوانند object های مختلف از class های مختلف را نگهداری کنند.
2) اگه به یک لیست اندیس منفی بدهیم، درواقع شمارش از آخر لیست حساب میشود و درواقع میتوانیم به اندیس های آخر دسترسی داشته باشیم.

```
3)   lst = [45, -3, 16, 8]
4.a) lst[0]
4.b) lst[-1]
4.c) 10
4.d) 29
4.e) -4
4.f) 29
4.g) 10
```

4.h) خیر، اینکار باعث ایجاد TypeError میشود، چون اندکس ها باید از نوع integers باشند.

```
5.a) 3
5.b) 5
5.c) 1
5.d) 2
5.e) 5
5.f) 2
5.g) 0
5.h) 3
```

```
6) len(lst)
```

```
7) []
```

```
8.a) [20, 1, -34, 40, -8, 60, 1, 3]

8.b) [20, 1, -34]

8.c) [-8, 60, 1, 3]

8.d) [-8, 60, 1, 3]

8.e) [40, -8]

8.f) [20, 1, -34, 40, -8]

8.g) [-8, 60, 1, 3]

8.h) [20, 1, -34, 40, -8, 60, 1, 3]

8.i) [20, 1, -34, 40]

8.j) [1, -34, 40, -8]

8.k) True
```

```
8.k) False

8.m) 8
```

9)

Original List	Target List	m	n
[2, 4, 6, 8, 10]	[2, 4, 6, 8, 10, 12, 14, 16, 18, 20]	5	nothing
[2, 4, 6, 8, 10]	[-10, -8, -6, -4, -2, 0, 2, 4, 6, 8, 10]	0	6
[2, 4, 6, 8, 10]	[2, 3, 4, 5, 6, 7, 8, 10]	1	6
[2, 4, 6, 8, 10]	[2, 4, 6, 'a', 'b', 'c', 8, 10]	-2	-2
[2, 4, 6, 8, 10]	[2, 4, 6, 8, 10]	nothing	nothing
[2, 4, 6, 8, 10]	[]	0	0
[2, 4, 6, 8, 10]	[10, 8, 6, 4, 2]	0	5
[2, 4, 6, 8, 10]	[2, 4, 6]	3	6
[2, 4, 6, 8, 10]	[6, 8, 10]	0	2
[2, 4, 6, 8, 10]	[2, 10]	1	-1
[2, 4, 6, 8, 10]	[4, 6, 8]	nothing	nothing

```
10.a) [8, 8, 8, 8]

10.b) [2, 7, 2, 7, 2, 7, 2, 7, 2, 7, 2, 7]

10.c) [1, 2, 3, 'a', 'b', 'c', 'd']

10.d) [1, 2, 1, 2, 1, 2, 4, 2]

10.e) [1, 2, 4, 2, 1, 2, 4, 2, 1, 2, 4, 2]
```

```
11.a) [3, 5, 7, 9]

11.b) [50, 60, 70, 80, 90]

11.c) [12, 15, 18]

11.d) [(0, 0), (0, 1), (0, 2), (0, 3), (1, 0), (1, 1), (1, 2), (1, 3), (2, 0), (2, 1), (2, 2), (2, 3)]

11.e) [(0, 0), (0, 2), (1, 1), (1, 3), (2, 0), (2, 2)]
```

```
12.a) [x * x for x in range(5)]

12.b) [x * 0.25 for x in range(1, 7)]

12.c) [(x, y) for x in ['a', 'b'] for y in range(3)]
```

```
13) x in lst
```

14) این دستور یک لیست را از ایندکس آخر به اول مرتب میکند

```
15) def sum_positive(a):

    sum = 0
    if a:
        for i in a:
            if i > 0:
                sum += i
    else:
        return 0
    return sum
```

```
16) def count_evens(lst):

    num_of_even_number = 0
    if lst:
        for i in lst:
            if i % 2 == 0:
                num_of_even_number += 1
    else:
        return 0

    return num_of_even_number
```

```
17) def print_big_enough(lst, x):

    for i in lst:
        if i >= x:
            print(i)
```

```
18) def next_number(lst):

    positive_int = 1
    while 1:
        if positive_int not in lst:
            return positive_int
        positive_int += 1
```

```python
def reverse(lst):

    reversed_list = [0 for i in range(len(lst))]

    for i in range(len(lst)):
        reversed_list[len(lst) - i - 1] = lst[i]

    return reversed_list
```

```python
def Q20():

    m = [[1 for i in range(9)] for j in range(6)]

    for i in range(6):
        for j in range(9):
            print(m[i][j], end='  ')
        print('\n')
    print('\n\n')
    m[2][4] = 0

    for i in range(6):
        for j in range(9):
            print(m[i][j], end='  ')
        print('\n')
```

```python
# 1st way

lst = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

# 2nd way
lst = [x for x in range(1, 11)]

# 3rd way
lst = list(range(1, 11))

# 4th way
lst = list()
for i in range(1,11):
    lst.append(i)

# 5th way
lst = [0] * 10
for i in range(1, 11):
    lst[i-1] = i
```

22)
```python
def Q22(m):

    new_mat = [[0] * len(m)] * len(m)
    for i in range(len(m)):
        for j in range(len(m[0])):
            new_mat[j][i] = m[i][j]
    flag = 0

    for i in range(len(m)):
        for j in range(len(m)):
            if m[i] == new_mat[j]:
                flag = 1

    if flag:
        return True
    else:
        return False
```

23)
```python
def check_winner(m):

    new_mat = [[0] * len(m)] * len(m)
    for i in range(len(m)):
        for j in range(len(m[0])):
            new_mat[j][i] = m[i][j]
    for i in m:
        if i[0] == i[1] == i[2] == 'X':
            return 'X'
        elif i[0] == i[1] == i[2] == 'O':
            return 'O'
    for i in new_mat:
        if i[0] == i[1] == i[2] == 'X':
            return 'X'
        elif i[0] == i[1] == i[2] == 'O':
            return 'O'
    if m[0][0] == m[1][1] == m[2][2] == 'X':
        return 'X'
    elif m[0][0] == m[1][1] == m[2][2] == 'O':
        return 'O'
    if m[0][2] == m[1][1] == m[2][0] == 'X':
        return 'X'
    elif m[0][2] == m[1][1] == m[2][0] == 'X':
        return 'O'
    return ' '
```