Tamrinhaye safeye 385

1. بله لیست پایتون میتواند نوع های متفاوت داشته باشد مثلا میتوانیم یک رشته یک عدد و حتی یک لیست دیگر را در یک لیست ذخیره کنیم.

2.(یعنی اخرین مورد در لیست).(اندیس -n1 از انتهای لیست خوانده میشود)

3. lst=[45,-3,16,8]

4.

A) lst[0]

b) lst[3]

c) 10

d) 29

e) -4

f) 29

g) 10

h) illegal [0:3]

5.

a) 3

b) 5

c) 1

d) 5

e) 5

f) 2

g) 0

h) 3

6. Len( )

7.[ ]

8.

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

b) [20,1,-34]

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

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

e) [40, -8]

f) [20, 1, -34]

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

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

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

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

k) true

l) false

m) 8

9.

a)lst[5:5] = [12,14,16,18,20]

b) lst[0:0] = [-10,-8,-6,-4,-2,0]

c) lst [1:1] = [3]

lst[3:3] = [5]

lst[5:5] = [7]

d) lst[3:3] = ["a","b","c"]

e) lst[:]

f) lst[0:5] =[]

g) lst[0:5] = [10,8,6,4,2] or lst[::-1]

h) lst[3:5] = []

i) lst[0:2] = []

j) lst[1:4] = []

k) lst[0:1] = []

lst[3:4] = []

10.

A) [8, 8, 8, 8]

b) [2, 7, 2, 7, 2, 7, 2, 7, 2, 7, 2, 7]

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

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

e) [1, 2, 4, 2, 1, 2, 4, 2, 1, 2, 4, 2]

11.

a) [3, 5, 7, 9]

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

c) [12, 15, 18]

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)]

e) [(0, 0), (0, 2), (1, 1), (1, 3), (2, 0), (2, 2)]

12.

a) print([x**2 for x in range(1,6)])

b) print([x/4 for x in range(1,7)])

c) print([(x, y) for x in ["a","b"] for y in range(3)])

13.

X In lst

x not in lst

14. یک روش داده لیست است که عناصر لیست را در محل معکوس می کند. این روش به جای ایجاد لیست جدید, لیست اصلی را اصلاح می کند.

15.

```
  lst=[3,-3,5,2,2,-1]
sum_positive = 0
for x in lst:
    if x>=0:
        sum_positive+=x
print("sum_positive:",sum_positive)
```

16.

```
lst=[3,5,4,-1,0]
even_count=0
for x in lst:
    if x%2== 0:
        even_count += 1
print("even_count:",even_count)
```

18.

```
def next_number(lst):
    lst.sort()
    print(lst)
    i=0
    x=lst[0]
    print(i," ",x)
    while i>0:
        if i<x:
            break
        else:
            i+=1
    return i
list=[1,7,5,10]
print(next_number(list))
```

20.

```
m = [[1 for _ in range(9)] for _ in range(6)]
print(m)
 m[2][4]=0
print(m)
```

21.

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

```
lst1=[1,2,3,4,5]
lst2=[6,7,8,9,10]
result_lst_1=lst1 + lst2


lst3= list (range (1 , 11))
print(lst)


lst4= []
for i in range (1 , 11 , 1):
  lst4 += [i]

lst5 = [x for x in range(1 , 11)]
```

23.

```
import numpy as np


def checkRows(board):
    for row in board:
        if len(set(row)) == 1:
            return row[0]
```

```
    return 0


def checkDiagonals(board):
    if len(set([board[i][i] for i in range(len(board))])) == 1:
        return board[0][0]
    if len(set([board[i][len(board)-i-1] for i in range(len(board))])) ==
1:
        return board[0][len(board)-1]
    return 0


def checkWin(board):
    #transposition to check rows, then columns
    for newBoard in [board, np.transpose(board)]:
        result = checkRows(newBoard)
        if result:
            return result
    return checkDiagonals(board)




a = [['X', 'A', 'X'],
     ['A', 'X', 'A'],
     ['A', 'X', 'A']]
```

```
print(checkWin(a))
```