Tutoriat 3 – Liste
- rezolvări –

1. Se citesc de la tastatură două liste, `l1` și `l2`, reprezentând mulțimi (elementele nu se repetă). Determinați reuniunea și intersecția lor.

```python
#V1 - interclasare
l1 = [ int(x) for x in input().split() ]
l2 = [ int(x) for x in input().split() ]
l1.sort()
l2.sort()
i = 0
j = 0
reuniune = l1[:] # facem o copie, nu o referinta - se poate si cu l1.copy()
intersectie = []
while i < len(l1) and j < len(l2):
    if l1[i] == l2[j]:
        intersectie.append(l1[i])
        i += 1
        j += 1
    elif l1[i] < l2[j]:
        i += 1
    else:
        j += 1
for i in l2:
    if i not in reuniune:
        reuniune.append(i)
print( reuniune )
print( intersectie )

#V2 - comprehensiune
l1 = [ int(x) for x in input().split() ]
l2 = [ int(x) for x in input().split() ]
reuniune = l1 + [ x for x in l2 if x not in l1 ]
intersectie = [ x for x in reuniune if x in l1 and x in l2 ]
print( reuniune )
print( intersectie )
```

2. Generați mulțimea numerelor prime mai mici decât `n` (citit de la tastatură), folosind Ciurul lui Eratostene.

```python
n = int( input() )
prime = [1] * ( n + 1 )
prime[0] = 0
prime[1] = 0
for i in range( 2, n + 1 ):
    for multiplu in range( 2 * i, n + 1, i ):
        prime[ multiplu ] = 0
prime = [ i for i in range( 2, n + 1 ) if prime[i] == 1 ]
print( " ".join( [ str(x) for x in prime ] ) )
```

3. Se citesc de la tastatură numere naturale pe o singură linie, separate prin spații. Creați o listă ce conține toți divizorii, în ordinea apariției numerelor. Exemplu: `[4, 7, 12, 3] => [1, 2, 4, 1, 7, 1, 2, 3, 4, 6, 12, 1, 3]`.

```
v = [ int(x) for x in input().split() ]
divizori = []
for n in v:
    divn = [ x for x in range( 1, n + 1 ) if n % x == 0 ]
    divizori.extend( divn )
    # [2, 3, 4].append([5, 6]) => [2, 3, 4, [5, 6]]
    # [2, 3, 4].extend([5, 6]) => [2, 3, 4, 5, 6]
print( divizori )
```

4. Se citește de la tastatură o listă de numere. Adăugați după fiecare element impar dublul său, iar după fiecare element par, jumătatea sa. Exemplu: `[1, 4, 7, 2, 3] => [1, 2, 4, 2, 7, 14, 2, 1, 3, 6]`.

```
v = [ int(x) for x in input().split() ]
final = []
for n in v:
    final.append( n )
    if n % 2 == 0:
        final.append( n // 2 )
    else:
        final.append( n * 2 )
print( final )
```

5. Se citește un număr natural `n` și apoi o matrice pătratică de dimensiune `n`, cu elemente numere naturale. Afișați elementele de pe diagonala principală și pe cele de pe diagonala secundară, prin comprehensiune.

```
'''
1 2 3
4 5 6   => [ [1, 2, 3], [4, 5, 6], [7, 8, 9] ]
7 8 9
'''
n = int( input() )
mat = []
for i in range( n ):
    mat.append( [ int(x) for x in input().split() ] )
# diagonala principala i = j
dp = [ mat[i][i] for i in range(n) ]
# diagonala secundara i + j = n - 1 => j = n - i - 1
ds = [ mat[i][n - i - 1] for i in range(n) ]
print( f"Elementele de pe diagonala principala sunt: {dp}" )
print( f"Elementele de pe diagonala secundara sunt: {ds}" )
```

6. Se citesc două liste, `l1` și `l2`. Să se genereze produsul cartezian al celor două liste. Exemplu: `l1 = [3, 5, 8], l2 = [6, 2] => [(3, 6), (3, 2), (5, 6), (5, 2), (8, 6), (8, 2)]`.

```
l1 = [ int(x) for x in input().split() ]
l2 = [ int(x) for x in input().split() ]
produsCartezian = [ (x, y) for x in l1 for y in l2 ]
print( produsCartezian )
```

7. Se citește un număr natural `k` și o listă de numere naturale. Afișați câte numere sunt divizibile cu `k`, folosind comprehensiune.

```
k = int( input() )
v = [ int(x) for x in input().split() ]
divizibil = [ 1 if x % k == 0 else 0 for x in v ]
print( sum( divizibil ) )
```

8. Se citește o listă de numere naturale. Sortați lista în funcție de numărul de divizori.

```
v = [ int(x) for x in input().split() ]
v.sort( key = lambda x: len( [ d for d in range( 1, x + 1 ) if x % d == 0 ] ) )
print( v )
```