APSSDC

Andhra Pradesh State Skill Development Corporation

(https://apssdc.in)

Day16 Comprehensions, lambda, Iterators, Generators, Map and Filter

Recap

- Regular Expressions
  - re
    - findall
    - sub -> replace
    - subn
    - match

Today Objectives

- Functional Programming/ Comprehensions
- lambda function/ nameless Function/ anoyonomous function
- Iterator
- Generator
- map()
- filter()
- reduce()

Functional Programing

- List Comprehension
- Dictionary Comprehension
- tuple Comprehension
- set comprehension

In [1]:

```
li = []
for i in range(1, 101):
    li.append(i)
```

In [2]:

```
print(li)
```

```
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21,
22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40,
41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59,
60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78,
79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97,
98, 99, 100]
```

In [3]:

```
li = []
for i in range(1, 101):
    if i % 2 == 0 or i % 3 == 0:
        li.append(i)
print(li)
```

```
[2, 3, 4, 6, 8, 9, 10, 12, 14, 15, 16, 18, 20, 21, 22, 24, 26, 27, 28, 30, 3
2, 33, 34, 36, 38, 39, 40, 42, 44, 45, 46, 48, 50, 51, 52, 54, 56, 57, 58, 6
0, 62, 63, 64, 66, 68, 69, 70, 72, 74, 75, 76, 78, 80, 81, 82, 84, 86, 87, 8
8, 90, 92, 93, 94, 96, 98, 99, 100]
```

List Comprehensions

Syntax

```
[ele for IterVar in groupOfElements]
```

In [4]:

```
li = [i for i in range(1, 101)]


print(li)
```

```
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21,
22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40,
41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59,
60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78,
79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97,
98, 99, 100]
```

In [5]:

```
li = [[i] for i in range(1, 101)]


print(li)
```

```
[[1], [2], [3], [4], [5], [6], [7], [8], [9], [10], [11], [12], [13], [14],
[15], [16], [17], [18], [19], [20], [21], [22], [23], [24], [25], [26], [2
7], [28], [29], [30], [31], [32], [33], [34], [35], [36], [37], [38], [39],
[40], [41], [42], [43], [44], [45], [46], [47], [48], [49], [50], [51], [5
2], [53], [54], [55], [56], [57], [58], [59], [60], [61], [62], [63], [64],
[65], [66], [67], [68], [69], [70], [71], [72], [73], [74], [75], [76], [7
7], [78], [79], [80], [81], [82], [83], [84], [85], [86], [87], [88], [89],
[90], [91], [92], [93], [94], [95], [96], [97], [98], [99], [100]]
```

In [6]:

```
li = ['*' for i in range(1, 101)]


print(li)
```

```
['*', '*', '*', '*', '*', '*', '*', '*', '*', '*', '*', '*', '*', '*', '*',
'*', '*', '*', '*', '*', '*', '*', '*', '*', '*', '*', '*', '*', '*', '*',
'*', '*', '*', '*', '*', '*', '*', '*', '*', '*', '*', '*', '*', '*', '*',
'*', '*', '*', '*', '*', '*', '*', '*', '*', '*', '*', '*', '*', '*', '*',
'*', '*', '*', '*', '*', '*', '*', '*', '*', '*', '*', '*', '*', '*', '*',
'*', '*', '*', '*', '*', '*', '*', '*', '*', '*', '*', '*', '*', '*', '*',
'*', '*', '*', '*', '*', '*', '*', '*', '*', '*']
```

In [7]:

```
li = [i for i in range(1, 101) if i % 2 == 0]

print(li)
```

```
[2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 36, 38, 40,
42, 44, 46, 48, 50, 52, 54, 56, 58, 60, 62, 64, 66, 68, 70, 72, 74, 76, 78,
80, 82, 84, 86, 88, 90, 92, 94, 96, 98, 100]
```

In [10]:

```
li = [year for year in range(2000, 3000) if (year % 400 == 0) or (year % 4 == 0 and yea


print(li)
```

```
[2000, 2004, 2008, 2012, 2016, 2020, 2024, 2028, 2032, 2036, 2040, 2044, 204
8, 2052, 2056, 2060, 2064, 2068, 2072, 2076, 2080, 2084, 2088, 2092, 2096, 2
104, 2108, 2112, 2116, 2120, 2124, 2128, 2132, 2136, 2140, 2144, 2148, 2152,
2156, 2160, 2164, 2168, 2172, 2176, 2180, 2184, 2188, 2192, 2196, 2204, 220
8, 2212, 2216, 2220, 2224, 2228, 2232, 2236, 2240, 2244, 2248, 2252, 2256, 2
260, 2264, 2268, 2272, 2276, 2280, 2284, 2288, 2292, 2296, 2304, 2308, 2312,
2316, 2320, 2324, 2328, 2332, 2336, 2340, 2344, 2348, 2352, 2356, 2360, 236
4, 2368, 2372, 2376, 2380, 2384, 2388, 2392, 2396, 2400, 2404, 2408, 2412, 2
416, 2420, 2424, 2428, 2432, 2436, 2440, 2444, 2448, 2452, 2456, 2460, 2464,
2468, 2472, 2476, 2480, 2484, 2488, 2492, 2496, 2504, 2508, 2512, 2516, 252
0, 2524, 2528, 2532, 2536, 2540, 2544, 2548, 2552, 2556, 2560, 2564, 2568, 2
572, 2576, 2580, 2584, 2588, 2592, 2596, 2604, 2608, 2612, 2616, 2620, 2624,
2628, 2632, 2636, 2640, 2644, 2648, 2652, 2656, 2660, 2664, 2668, 2672, 267
6, 2680, 2684, 2688, 2692, 2696, 2704, 2708, 2712, 2716, 2720, 2724, 2728, 2
732, 2736, 2740, 2744, 2748, 2752, 2756, 2760, 2764, 2768, 2772, 2776, 2780,
2784, 2788, 2792, 2796, 2800, 2804, 2808, 2812, 2816, 2820, 2824, 2828, 283
2, 2836, 2840, 2844, 2848, 2852, 2856, 2860, 2864, 2868, 2872, 2876, 2880, 2
884, 2888, 2892, 2896, 2904, 2908, 2912, 2916, 2920, 2924, 2928, 2932, 2936,
2940, 2944, 2948, 2952, 2956, 2960, 2964, 2968, 2972, 2976, 2980, 2984, 298
8, 2992, 2996]
```

In [11]:

```
li = ['even' if i % 2 == 0 else 'odd' for i in range(2000, 3000)]


print(li)
```

```
['even', 'odd', 'even', 'odd', 'even', 'odd', 'even', 'odd', 'even', 'od
d', 'even', 'odd', 'even', 'odd', 'even', 'odd', 'even', 'odd', 'even', 'o
dd', 'even', 'odd', 'even', 'odd', 'even', 'odd', 'even', 'odd', 'even',
'odd', 'even', 'odd', 'even', 'odd', 'even', 'odd', 'even', 'odd', 'even',
'odd', 'even', 'odd', 'even', 'odd', 'even', 'odd', 'even', 'odd', 'even',
'odd', 'even', 'odd', 'even', 'odd', 'even', 'odd', 'even', 'odd', 'even',
'odd', 'even', 'odd', 'even', 'odd', 'even', 'odd', 'even', 'odd', 'even',
'odd', 'even', 'odd', 'even', 'odd', 'even', 'odd', 'even', 'odd', 'even',
'odd', 'even', 'odd', 'even', 'odd', 'even', 'odd', 'even', 'odd', 'even',
'odd', 'even', 'odd', 'even', 'odd', 'even', 'odd', 'even', 'odd', 'even',
'odd', 'even', 'odd', 'even', 'odd', 'even', 'odd', 'even', 'odd', 'even',
'odd', 'even', 'odd', 'even', 'odd', 'even', 'odd', 'even', 'odd', 'even',
'odd', 'even', 'odd', 'even', 'odd', 'even', 'odd', 'even', 'odd', 'even',
'odd', 'even', 'odd', 'even', 'odd', 'even', 'odd', 'even', 'odd', 'even',
'odd', 'even', 'odd', 'even', 'odd', 'even', 'odd', 'even', 'odd', 'even',
'odd', 'even', 'odd', 'even', 'odd', 'even', 'odd', 'even', 'odd', 'even',
'odd', 'even', 'odd', 'even', 'odd', 'even', 'odd', 'even', 'odd', 'even',
'odd', 'even', 'odd', 'even', 'odd', 'even', 'odd', 'even', 'odd', 'even',
'odd', 'even', 'odd', 'even', 'odd', 'even', 'odd', 'even', 'odd', 'even',
```

In [12]:

```
def square(num):
    return num ** 2
```

In [13]:

```
li = [square(i) for i in range(1, 101)]


print(li)
```

```
[1, 4, 9, 16, 25, 36, 49, 64, 81, 100, 121, 144, 169, 196, 225, 256, 289, 32
4, 361, 400, 441, 484, 529, 576, 625, 676, 729, 784, 841, 900, 961, 1024, 10
89, 1156, 1225, 1296, 1369, 1444, 1521, 1600, 1681, 1764, 1849, 1936, 2025,
2116, 2209, 2304, 2401, 2500, 2601, 2704, 2809, 2916, 3025, 3136, 3249, 336
4, 3481, 3600, 3721, 3844, 3969, 4096, 4225, 4356, 4489, 4624, 4761, 4900, 5
041, 5184, 5329, 5476, 5625, 5776, 5929, 6084, 6241, 6400, 6561, 6724, 6889,
7056, 7225, 7396, 7569, 7744, 7921, 8100, 8281, 8464, 8649, 8836, 9025, 921
6, 9409, 9604, 9801, 10000]
```

In [14]:

```
def square(num):
    return cube(num ** 2)


def cube(sq):
    return sq ** 3
```

In [15]:

```
li = [square(i) for i in range(1, 101)]


print(li)
```

```
[1, 64, 729, 4096, 15625, 46656, 117649, 262144, 531441, 1000000, 1771561, 2
985984, 4826809, 7529536, 11390625, 16777216, 24137569, 34012224, 47045881,
64000000, 85766121, 113379904, 148035889, 191102976, 244140625, 308915776, 3
87420489, 481890304, 594823321, 729000000, 887503681, 1073741824, 129146796
9, 1544804416, 1838265625, 2176782336, 2565726409, 3010936384, 3518743761, 4
096000000, 4750104241, 5489031744, 6321363049, 7256313856, 8303765625, 94742
96896, 10779215329, 12230590464, 13841287201, 15625000000, 17596287801, 1977
0609664, 22164361129, 24794911296, 27680640625, 30840979456, 34296447249, 38
068692544, 42180533641, 46656000000, 51520374361, 56800235584, 62523502209,
68719476736, 75418890625, 82653950016, 90458382169, 98867482624, 10791816308
1, 117649000000, 128100283921, 139314069504, 151334226289, 164206490176, 177
978515625, 192699928576, 208422380089, 225199600704, 243087455521, 262144000
000, 282429536481, 304006671424, 326940373369, 351298031616, 377149515625, 4
04567235136, 433626201009, 464404086784, 496981290961, 531441000000, 5678692
52041, 606355001344, 646990183449, 689869781056, 735091890625, 782757789696,
832972004929, 885842380864, 941480149401, 1000000000000]
```

In [16]:

```
def square(num):
    return num ** 2


def cube(sq):
    return sq ** 3

li = [cube(square(i)) for i in range(1, 101)]


print(li)
```

```
[1, 64, 729, 4096, 15625, 46656, 117649, 262144, 531441, 1000000, 1771561, 2
985984, 4826809, 7529536, 11390625, 16777216, 24137569, 34012224, 47045881,
64000000, 85766121, 113379904, 148035889, 191102976, 244140625, 308915776, 3
87420489, 481890304, 594823321, 729000000, 887503681, 1073741824, 129146796
9, 1544804416, 1838265625, 2176782336, 2565726409, 3010936384, 3518743761, 4
096000000, 4750104241, 5489031744, 6321363049, 7256313856, 8303765625, 94742
96896, 10779215329, 12230590464, 13841287201, 15625000000, 17596287801, 1977
0609664, 22164361129, 24794911296, 27680640625, 30840979456, 34296447249, 38
068692544, 42180533641, 46656000000, 51520374361, 56800235584, 62523502209,
68719476736, 75418890625, 82653950016, 90458382169, 98867482624, 10791816308
1, 117649000000, 128100283921, 139314069504, 151334226289, 164206490176, 177
978515625, 192699928576, 208422380089, 225199600704, 243087455521, 262144000
000, 282429536481, 304006671424, 326940373369, 351298031616, 377149515625, 4
04567235136, 433626201009, 464404086784, 496981290961, 531441000000, 5678692
52041, 606355001344, 646990183449, 689869781056, 735091890625, 782757789696,
832972004929, 885842380864, 941480149401, 1000000000000]
```

In [19]:

```
li = [[i for i in range(1, 6)] for j in range(5)]
```

In [20]:

```
li
```

Out[20]:

```
[[1, 2, 3, 4, 5],
 [1, 2, 3, 4, 5],
 [1, 2, 3, 4, 5],
 [1, 2, 3, 4, 5],
 [1, 2, 3, 4, 5]]
```

In [22]:

```python
li = [[i for i in range(1, 6)] for j in range(100)]

li
```

Out[22]:

```
[[1, 2, 3, 4, 5],
 [1, 2, 3, 4, 5],
 [1, 2, 3, 4, 5],
 [1, 2, 3, 4, 5],
 [1, 2, 3, 4, 5],
 [1, 2, 3, 4, 5],
 [1, 2, 3, 4, 5],
 [1, 2, 3, 4, 5],
 [1, 2, 3, 4, 5],
 [1, 2, 3, 4, 5],
 [1, 2, 3, 4, 5],
 [1, 2, 3, 4, 5],
 [1, 2, 3, 4, 5],
 [1, 2, 3, 4, 5],
 [1, 2, 3, 4, 5],
 [1, 2, 3, 4, 5],
 [1, 2, 3, 4, 5],
```

In [23]:

```python
li = [[j for i in range(1, 6)] for j in range(1, 6)]

li
```

Out[23]:

```
[[1, 1, 1, 1, 1],
 [2, 2, 2, 2, 2],
 [3, 3, 3, 3, 3],
 [4, 4, 4, 4, 4],
 [5, 5, 5, 5, 5]]
```

In [24]:

```
li = [[[j for i in range(1, 6)] for j in range(1, 6)] for k in range(5)]

li
```

Out[24]:

```
[[[1, 1, 1, 1, 1],
  [2, 2, 2, 2, 2],
  [3, 3, 3, 3, 3],
  [4, 4, 4, 4, 4],
  [5, 5, 5, 5, 5]],
 [[1, 1, 1, 1, 1],
  [2, 2, 2, 2, 2],
  [3, 3, 3, 3, 3],
  [4, 4, 4, 4, 4],
  [5, 5, 5, 5, 5]],
 [[1, 1, 1, 1, 1],
  [2, 2, 2, 2, 2],
  [3, 3, 3, 3, 3],
  [4, 4, 4, 4, 4],
  [5, 5, 5, 5, 5]],
 [[1, 1, 1, 1, 1],
  [2, 2, 2, 2, 2],
```

In [25]:

```
s = """Python is an interpreted high-level general-purpose programming language. Pythor
Developer: Python Software Foundation
Stable release: 3.9.5 / 3 May 2021; 29 days ago
Preview release: 3.10.0b1 / 3 May 2021; 29 days ago
Typing discipline: Duck, dynamic, strong typing; gradual (since 3.5, but ignored in CPy
First appeared: February 1991; 30 years ago
Paradigm: Multi-paradigm: object-oriented, procedural (imperative), functional, structu
```

In [26]:

```
up = [char for char in s if char.isupper()]
```

In [27]:

```
print(up)
```

```
['P', 'P', 'W', 'D', 'P', 'S', 'F', 'S', 'M', 'P', 'M', 'T', 'D', 'C', 'P',
'F', 'F', 'P', 'M']
```

In [28]:

```
vol = [char for char in s if char in 'aeiou']


print(vol)
```

```
['o', 'i', 'a', 'i', 'e', 'e', 'e', 'i', 'e', 'e', 'e', 'e', 'a', 'u', 'o',
'e', 'o', 'a', 'i', 'a', 'u', 'a', 'e', 'o', 'e', 'i', 'i', 'o', 'o', 'e',
'a', 'i', 'e', 'o', 'e', 'e', 'a', 'a', 'i', 'i', 'i', 'i', 'o', 'a', 'e',
'u', 'e', 'o', 'i', 'i', 'i', 'a', 'i', 'e', 'a', 'i', 'o', 'i', 'i', 'e',
'i', 'a', 'e', 'e', 'o', 'e', 'o', 'o', 'a', 'e', 'o', 'u', 'a', 'i', 'o',
'a', 'e', 'e', 'e', 'a', 'e', 'a', 'a', 'a', 'o', 'e', 'i', 'e', 'e', 'e',
'a', 'e', 'a', 'a', 'a', 'o', 'i', 'i', 'i', 'i', 'e', 'u', 'a', 'i', 'o',
'i', 'a', 'u', 'a', 'i', 'e', 'u', 'i', 'o', 'e', 'i', 'o', 'i', 'a', 'e',
'a', 'e', 'e', 'u', 'a', 'e', 'a', 'a', 'o', 'a', 'a', 'i', 'u', 'i', 'a',
'a', 'i', 'o', 'e', 'o', 'i', 'e', 'e', 'o', 'e', 'u', 'a', 'i', 'e', 'a',
'i', 'e', 'u', 'i', 'o', 'a', 'u', 'u', 'e', 'e', 'e', 'i', 'e']
```

In [29]:

```
vol = [char for char in s if char in 'aeiouAEIOU']

print(vol)
```

```
['o', 'i', 'a', 'i', 'e', 'e', 'e', 'i', 'e', 'e', 'e', 'e', 'a', 'u', 'o',
'e', 'o', 'a', 'i', 'a', 'u', 'a', 'e', 'o', 'e', 'i', 'i', 'o', 'o', 'e',
'a', 'i', 'e', 'o', 'e', 'e', 'a', 'a', 'i', 'i', 'i', 'i', 'o', 'a', 'e',
'u', 'e', 'o', 'i', 'i', 'i', 'a', 'i', 'e', 'a', 'i', 'o', 'i', 'i', 'e',
'i', 'a', 'e', 'e', 'o', 'e', 'o', 'o', 'a', 'e', 'o', 'u', 'a', 'i', 'o',
'a', 'e', 'e', 'e', 'a', 'e', 'a', 'a', 'a', 'o', 'e', 'i', 'e', 'e', 'e',
'a', 'e', 'a', 'a', 'a', 'o', 'i', 'i', 'i', 'i', 'e', 'u', 'a', 'i', 'o',
'i', 'a', 'u', 'a', 'i', 'e', 'u', 'i', 'o', 'e', 'i', 'o', 'i', 'a', 'e',
'a', 'e', 'e', 'u', 'a', 'e', 'a', 'a', 'o', 'a', 'a', 'i', 'u', 'i', 'a',
'a', 'i', 'o', 'e', 'o', 'i', 'e', 'e', 'o', 'e', 'u', 'a', 'i', 'e', 'a',
'i', 'e', 'u', 'i', 'o', 'a', 'u', 'u', 'e', 'e', 'e', 'i', 'e']
```

In [30]:

```
vol = [char for char in s if char.lower() in 'aeiou']

print(vol)
```

```
['o', 'i', 'a', 'i', 'e', 'e', 'e', 'i', 'e', 'e', 'e', 'e', 'a', 'u', 'o',
'e', 'o', 'a', 'i', 'a', 'u', 'a', 'e', 'o', 'e', 'i', 'i', 'o', 'o', 'e',
'a', 'i', 'e', 'o', 'e', 'e', 'a', 'a', 'i', 'i', 'i', 'i', 'o', 'a', 'e',
'u', 'e', 'o', 'i', 'i', 'i', 'a', 'i', 'e', 'a', 'i', 'o', 'i', 'i', 'e',
'i', 'a', 'e', 'e', 'o', 'e', 'o', 'o', 'a', 'e', 'o', 'u', 'a', 'i', 'o',
'a', 'e', 'e', 'e', 'a', 'e', 'a', 'a', 'a', 'o', 'e', 'i', 'e', 'e', 'e',
'a', 'e', 'a', 'a', 'a', 'o', 'i', 'i', 'i', 'i', 'e', 'u', 'a', 'i', 'o',
'i', 'a', 'u', 'a', 'i', 'e', 'u', 'i', 'o', 'e', 'i', 'o', 'i', 'a', 'e',
'a', 'e', 'e', 'u', 'a', 'e', 'a', 'a', 'o', 'a', 'a', 'i', 'u', 'i', 'a',
'a', 'i', 'o', 'e', 'o', 'i', 'e', 'e', 'o', 'e', 'u', 'a', 'i', 'e', 'a',
'i', 'e', 'u', 'i', 'o', 'a', 'u', 'u', 'e', 'e', 'e', 'i', 'e']
```

Dictionary Comprehensions

create a dictionary key as the number and value as the square of the number of 1 - 100

In [31]:

```
di = {}
for num in range(1, 101):
    di[num] = num ** 2

print(di)
```

```
{1: 1, 2: 4, 3: 9, 4: 16, 5: 25, 6: 36, 7: 49, 8: 64, 9: 81, 10: 100, 11: 12
1, 12: 144, 13: 169, 14: 196, 15: 225, 16: 256, 17: 289, 18: 324, 19: 361, 2
0: 400, 21: 441, 22: 484, 23: 529, 24: 576, 25: 625, 26: 676, 27: 729, 28: 7
84, 29: 841, 30: 900, 31: 961, 32: 1024, 33: 1089, 34: 1156, 35: 1225, 36: 1
296, 37: 1369, 38: 1444, 39: 1521, 40: 1600, 41: 1681, 42: 1764, 43: 1849, 4
4: 1936, 45: 2025, 46: 2116, 47: 2209, 48: 2304, 49: 2401, 50: 2500, 51: 260
1, 52: 2704, 53: 2809, 54: 2916, 55: 3025, 56: 3136, 57: 3249, 58: 3364, 59:
3481, 60: 3600, 61: 3721, 62: 3844, 63: 3969, 64: 4096, 65: 4225, 66: 4356,
67: 4489, 68: 4624, 69: 4761, 70: 4900, 71: 5041, 72: 5184, 73: 5329, 74: 54
76, 75: 5625, 76: 5776, 77: 5929, 78: 6084, 79: 6241, 80: 6400, 81: 6561, 8
2: 6724, 83: 6889, 84: 7056, 85: 7225, 86: 7396, 87: 7569, 88: 7744, 89: 792
1, 90: 8100, 91: 8281, 92: 8464, 93: 8649, 94: 8836, 95: 9025, 96: 9216, 97:
9409, 98: 9604, 99: 9801, 100: 10000}
```

In [32]:

```
di = {i: i ** 2 for i in range(1, 101)}

print(di)
```

```
{1: 1, 2: 4, 3: 9, 4: 16, 5: 25, 6: 36, 7: 49, 8: 64, 9: 81, 10: 100, 11: 12
1, 12: 144, 13: 169, 14: 196, 15: 225, 16: 256, 17: 289, 18: 324, 19: 361, 2
0: 400, 21: 441, 22: 484, 23: 529, 24: 576, 25: 625, 26: 676, 27: 729, 28: 7
84, 29: 841, 30: 900, 31: 961, 32: 1024, 33: 1089, 34: 1156, 35: 1225, 36: 1
296, 37: 1369, 38: 1444, 39: 1521, 40: 1600, 41: 1681, 42: 1764, 43: 1849, 4
4: 1936, 45: 2025, 46: 2116, 47: 2209, 48: 2304, 49: 2401, 50: 2500, 51: 260
1, 52: 2704, 53: 2809, 54: 2916, 55: 3025, 56: 3136, 57: 3249, 58: 3364, 59:
3481, 60: 3600, 61: 3721, 62: 3844, 63: 3969, 64: 4096, 65: 4225, 66: 4356,
67: 4489, 68: 4624, 69: 4761, 70: 4900, 71: 5041, 72: 5184, 73: 5329, 74: 54
76, 75: 5625, 76: 5776, 77: 5929, 78: 6084, 79: 6241, 80: 6400, 81: 6561, 8
2: 6724, 83: 6889, 84: 7056, 85: 7225, 86: 7396, 87: 7569, 88: 7744, 89: 792
1, 90: 8100, 91: 8281, 92: 8464, 93: 8649, 94: 8836, 95: 9025, 96: 9216, 97:
9409, 98: 9604, 99: 9801, 100: 10000}
```

In [33]:

```
di = {i: i + 5 for i in range(1, 101)}

print(di)
```

```
{1: 6, 2: 7, 3: 8, 4: 9, 5: 10, 6: 11, 7: 12, 8: 13, 9: 14, 10: 15, 11: 16,
12: 17, 13: 18, 14: 19, 15: 20, 16: 21, 17: 22, 18: 23, 19: 24, 20: 25, 21:
26, 22: 27, 23: 28, 24: 29, 25: 30, 26: 31, 27: 32, 28: 33, 29: 34, 30: 35,
31: 36, 32: 37, 33: 38, 34: 39, 35: 40, 36: 41, 37: 42, 38: 43, 39: 44, 40:
45, 41: 46, 42: 47, 43: 48, 44: 49, 45: 50, 46: 51, 47: 52, 48: 53, 49: 54,
50: 55, 51: 56, 52: 57, 53: 58, 54: 59, 55: 60, 56: 61, 57: 62, 58: 63, 59:
64, 60: 65, 61: 66, 62: 67, 63: 68, 64: 69, 65: 70, 66: 71, 67: 72, 68: 73,
69: 74, 70: 75, 71: 76, 72: 77, 73: 78, 74: 79, 75: 80, 76: 81, 77: 82, 78:
83, 79: 84, 80: 85, 81: 86, 82: 87, 83: 88, 84: 89, 85: 90, 86: 91, 87: 92,
88: 93, 89: 94, 90: 95, 91: 96, 92: 97, 93: 98, 94: 99, 95: 100, 96: 101, 9
7: 102, 98: 103, 99: 104, 100: 105}
```

key as the num and value as the sq of previous number

In [34]:

```
di = {i: (i - 1) ** 2 for i in range(1, 101)}

print(di)
```

```
{1: 0, 2: 1, 3: 4, 4: 9, 5: 16, 6: 25, 7: 36, 8: 49, 9: 64, 10: 81, 11: 100,
12: 121, 13: 144, 14: 169, 15: 196, 16: 225, 17: 256, 18: 289, 19: 324, 20:
361, 21: 400, 22: 441, 23: 484, 24: 529, 25: 576, 26: 625, 27: 676, 28: 729,
29: 784, 30: 841, 31: 900, 32: 961, 33: 1024, 34: 1089, 35: 1156, 36: 1225,
37: 1296, 38: 1369, 39: 1444, 40: 1521, 41: 1600, 42: 1681, 43: 1764, 44: 18
49, 45: 1936, 46: 2025, 47: 2116, 48: 2209, 49: 2304, 50: 2401, 51: 2500, 5
2: 2601, 53: 2704, 54: 2809, 55: 2916, 56: 3025, 57: 3136, 58: 3249, 59: 336
4, 60: 3481, 61: 3600, 62: 3721, 63: 3844, 64: 3969, 65: 4096, 66: 4225, 67:
4356, 68: 4489, 69: 4624, 70: 4761, 71: 4900, 72: 5041, 73: 5184, 74: 5329,
75: 5476, 76: 5625, 77: 5776, 78: 5929, 79: 6084, 80: 6241, 81: 6400, 82: 65
61, 83: 6724, 84: 6889, 85: 7056, 86: 7225, 87: 7396, 88: 7569, 89: 7744, 9
0: 7921, 91: 8100, 92: 8281, 93: 8464, 94: 8649, 95: 8836, 96: 9025, 97: 921
6, 98: 9409, 99: 9604, 100: 9801}
```

In [35]:

```
di = {i: i ** 2 for i in range(1, 101) if i % 2 == 0}

print(di)
```

```
{2: 4, 4: 16, 6: 36, 8: 64, 10: 100, 12: 144, 14: 196, 16: 256, 18: 324, 20:
400, 22: 484, 24: 576, 26: 676, 28: 784, 30: 900, 32: 1024, 34: 1156, 36: 12
96, 38: 1444, 40: 1600, 42: 1764, 44: 1936, 46: 2116, 48: 2304, 50: 2500, 5
2: 2704, 54: 2916, 56: 3136, 58: 3364, 60: 3600, 62: 3844, 64: 4096, 66: 435
6, 68: 4624, 70: 4900, 72: 5184, 74: 5476, 76: 5776, 78: 6084, 80: 6400, 82:
6724, 84: 7056, 86: 7396, 88: 7744, 90: 8100, 92: 8464, 94: 8836, 96: 9216,
98: 9604, 100: 10000}
```

In [37]:

```
di = {i: 'Even' if i % 2 == 0 else 'odd' for i in range(1, 101)}

print(di)
```

```
{1: 'odd', 2: 'Even', 3: 'odd', 4: 'Even', 5: 'odd', 6: 'Even', 7: 'odd', 8:
'Even', 9: 'odd', 10: 'Even', 11: 'odd', 12: 'Even', 13: 'odd', 14: 'Even',
15: 'odd', 16: 'Even', 17: 'odd', 18: 'Even', 19: 'odd', 20: 'Even', 21: 'od
d', 22: 'Even', 23: 'odd', 24: 'Even', 25: 'odd', 26: 'Even', 27: 'odd', 28:
'Even', 29: 'odd', 30: 'Even', 31: 'odd', 32: 'Even', 33: 'odd', 34: 'Even',
35: 'odd', 36: 'Even', 37: 'odd', 38: 'Even', 39: 'odd', 40: 'Even', 41: 'od
d', 42: 'Even', 43: 'odd', 44: 'Even', 45: 'odd', 46: 'Even', 47: 'odd', 48:
'Even', 49: 'odd', 50: 'Even', 51: 'odd', 52: 'Even', 53: 'odd', 54: 'Even',
55: 'odd', 56: 'Even', 57: 'odd', 58: 'Even', 59: 'odd', 60: 'Even', 61: 'od
d', 62: 'Even', 63: 'odd', 64: 'Even', 65: 'odd', 66: 'Even', 67: 'odd', 68:
'Even', 69: 'odd', 70: 'Even', 71: 'odd', 72: 'Even', 73: 'odd', 74: 'Even',
75: 'odd', 76: 'Even', 77: 'odd', 78: 'Even', 79: 'odd', 80: 'Even', 81: 'od
d', 82: 'Even', 83: 'odd', 84: 'Even', 85: 'odd', 86: 'Even', 87: 'odd', 88:
'Even', 89: 'odd', 90: 'Even', 91: 'odd', 92: 'Even', 93: 'odd', 94: 'Even',
95: 'odd', 96: 'Even', 97: 'odd', 98: 'Even', 99: 'odd', 100: 'Even'}
```

{1: [1], 2:[1,2], 3:[1,2,3], 4:[1,2,3,4], 5:[1,2,3,4,5] ..... 100}

In [38]:

```
di = {i: [num for num in range(1, i + 1)] for i in range(1, 101)}

print(di)
```

```
{1: [1], 2: [1, 2], 3: [1, 2, 3], 4: [1, 2, 3, 4], 5: [1, 2, 3, 4, 5], 6:
[1, 2, 3, 4, 5, 6], 7: [1, 2, 3, 4, 5, 6, 7], 8: [1, 2, 3, 4, 5, 6, 7, 8],
9: [1, 2, 3, 4, 5, 6, 7, 8, 9], 10: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10], 11:
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11], 12: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 1
1, 12], 13: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13], 14: [1, 2, 3, 4,
5, 6, 7, 8, 9, 10, 11, 12, 13, 14], 15: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 1
1, 12, 13, 14, 15], 16: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 1
5, 16], 17: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17], 1
8: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18], 19:
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19], 20:
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20], 2
1: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20,
21], 22: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 1
9, 20, 21, 22], 23: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 1
6, 17, 18, 19, 20, 21, 22, 23], 24: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 1
2, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24], 25: [1, 2, 3, 4, 5, 6,
7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25],
26: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 2
0, 21, 22, 23, 24, 25, 26], 27: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 1
```

In [40]:

```
di = {i: ['*' for num in range(1, i + 1)] for i in range(1, 101)}

di
```

Out[40]:

```
{1: ['*'],
 2: ['*', '*'],
 3: ['*', '*', '*'],
 4: ['*', '*', '*', '*'],
 5: ['*', '*', '*', '*', '*'],
 6: ['*', '*', '*', '*', '*', '*'],
 7: ['*', '*', '*', '*', '*', '*', '*'],
 8: ['*', '*', '*', '*', '*', '*', '*', '*'],
 9: ['*', '*', '*', '*', '*', '*', '*', '*', '*'],
 10: ['*', '*', '*', '*', '*', '*', '*', '*', '*', '*'],
 11: ['*', '*', '*', '*', '*', '*', '*', '*', '*', '*', '*'],
 12: ['*', '*', '*', '*', '*', '*', '*', '*', '*', '*', '*', '*'],
 13: ['*', '*', '*', '*', '*', '*', '*', '*', '*', '*', '*', '*', '*'],
 14: ['*', '*', '*', '*', '*', '*', '*', '*', '*', '*', '*', '*', '*',
'*'],
 15: ['*',
  '*',
```

In [42]:

```
100 * (100 + 1)
```

Out[42]:

```
10100
```

In [43]:

```
li = [i for i in range(1, 101)]

di = {i: li[ : i + 1] for i in range(1, 101)}

di
```

Out[43]:

```
{1: [1, 2],
 2: [1, 2, 3],
 3: [1, 2, 3, 4],
 4: [1, 2, 3, 4, 5],
 5: [1, 2, 3, 4, 5, 6],
 6: [1, 2, 3, 4, 5, 6, 7],
 7: [1, 2, 3, 4, 5, 6, 7, 8],
 8: [1, 2, 3, 4, 5, 6, 7, 8, 9],
 9: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10],
 10: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11],
 11: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12],
 12: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13],
 13: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14],
 14: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15],
 15: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16],
 16: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17],
 17: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18],
```

set comprehensions

In [44]:

```
s1 = {char for char in s}
```

In [45]:

```
print(s1)
```

```
{':', 'F', '\n', ',', 'j', 'm', 'S', 'z', 'k', '2', '.', 'M', 'y', '1', 'i',
'g', 's', "'", ';', 't', ' ', 'w', '5', 'n', 'v', 'C', 'b', 'u', 'o', '/',
'W', 'l', '3', 'P', 'T', ')', 'd', 'p', 'r', '-', 'a', '9', 'e', 'f', '(',
'h', 'D', '0', 'c'}
```

In [47]:

```
s1 = {char for char in s if char.isalnum()}

print(s1)
```

```
{'F', 'j', 'm', 'S', 'z', 'k', '2', 'M', 'y', '1', 'i', 'g', 's', 't', 'w',
'5', 'n', 'v', 'C', 'b', 'u', 'o', 'W', 'l', '3', 'P', 'T', 'd', 'p', 'r',
'a', '9', 'e', 'f', 'h', 'D', '0', 'c'}
```

In [48]:

```
t1 = (i for i in range(1, 101))

print(t1)
```

```
<generator object <genexpr> at 0x000001CD60F85120>
```

Nameless Function

In [55]:

```
def funName(arg):
    pass
```

In [56]:

```
funName(6656)
```

In [50]:

```
def funName(arg):

```

```
  File "<ipython-input-50-376a21ecc493>", line 1
    def funName(arg):
                    ^
SyntaxError: unexpected EOF while parsing
```

In [51]:

```
def square(num):
    return num ** 2
```

In [57]:

```
sq = lambda num: num ** 2
```

In [58]:

```
print(square(5))

print(sq(5))
```

```
25
25
```

In [59]:

```
mul = lambda a, b: a ** 2 + b ** 2 + 2 * a * b
```

In [60]:

```
mul(5, 5)
```

Out[60]:

```
100
```

In [61]:

```
for i in range(1, 101):
    print(mul(i, i + 1))
```

```
9
25
49
81
121
169
225
289
361
441
529
625
729
841
961
1089
1225
1369
1521
```

In [62]:

```
sq1 = lambda a: sq(a)
```

In [63]:

```
print(sq1(5))
```

```
25
```

map()

In [65]:

```
inp = input()

print(inp)
```

```
1 2 3 4 5 6 7 8 9 10
1 2 3 4 5 6 7 8 9 10
```

In [66]:

```
inp
```

Out[66]:

```
'1 2 3 4 5 6 7 8 9 10'
```

In [67]:

```
li = inp.split()


print(li)
```

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

In [73]:

```
s = 0
inp = input()
li = inp.split()
for i in li:
    s += int(i)
print(s)
```

```
1 2 3 4 5 6 7 8 9 10
55
```

map()

map(function, seqofData)

It returns map object -> list/tuple/set

In [74]:

```
li = map(int, input().split())

print(li)
```

```
1 2 3 4 5 6 7 8 9 10
<map object at 0x000001CD6062ED60>
```

In [75]:

```
li = list(li)


print(li, sum(li))
```

```
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10] 55
```

In [76]:

```
s = sum(list(map(int, input().split())))

print(li)
```

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

In [77]:

```
print(s)
```

```
55
```

In [78]:

```
li = map(list, input().split())

print(list(li))
```

```
1 2 3 4 5 6 7 8 9 10
[['1'], ['2'], ['3'], ['4'], ['5'], ['6'], ['7'], ['8'], ['9'], ['1', '0']]
```

In [79]:

```
li = map(lambda x: x[0], input().split())

print(list(li))
```

```
11 12 5 6 4 7 8 10 13 4 6 56
['1', '1', '5', '6', '4', '7', '8', '1', '1', '4', '6', '5']
```

In [80]:

```
li = map(lambda x: int(x[0]), input().split())

print(list(li))
```

```
11 12 5 6 4 7 8 10 13 4 6 56
[1, 1, 5, 6, 4, 7, 8, 1, 1, 4, 6, 5]
```

In [83]:

```
li = map(lambda x: x[0], input().split())

print(list(li))
```

```
11 12 5 6 4 7 8 10 13 4 6 56
['1', '1', '5', '6', '4', '7', '8', '1', '1', '4', '6', '5']
```

int object is not callable error

In [84]:

```
for i in 55:
    print(i)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-84-9a6fa4d62144> in <module>
----> 1 for i in 55:
      2     print(i)

TypeError: 'int' object is not iterable
```

In [85]:

```
sum(55)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-85-bd7d1aaf3632> in <module>
----> 1 sum(55)

TypeError: 'int' object is not iterable
```

In [88]:

```
L1= list(map(int, input().split()))
print(L1, sum(L1))
```

```
1 2 3 4 5
[1, 2, 3, 4, 5] 15
```

In [90]:

```
import sys

sys.version
```

Out[90]:

```
'3.8.3 (default, Jul  2 2020, 17:30:36) [MSC v.1916 64 bit (AMD64)]'
```

In [91]:

```
print(55[0])
```

```
<>:1: SyntaxWarning: 'int' object is not subscriptable; perhaps you missed a
comma?
<>:1: SyntaxWarning: 'int' object is not subscriptable; perhaps you missed a
comma?
<ipython-input-91-2cf7ba6c1ef0>:1: SyntaxWarning: 'int' object is not subscr
iptable; perhaps you missed a comma?
  print(55[0])

---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-91-2cf7ba6c1ef0> in <module>
----> 1 print(55[0])

TypeError: 'int' object is not subscriptable
```

filter()

filter(function, iterables)

sum the even numbers from the input

In [93]:

```
fi = filter(lambda x: int(x) % 2 == 0, input().split())
```

```
11 12 5 6 4 7 8 10 13 4 6 56
```

In [94]:

```
print(list(fi))
```

```
['12', '6', '4', '8', '10', '4', '6', '56']
```

In [96]:

```
s = """Python is an interpreted high-level general-purpose programming language. Pythor
Developer: Python Software Foundation
Stable release: 3.9.5 / 3 May 2021; 29 days ago
Preview release: 3.10.0b1 / 3 May 2021; 29 days ago
Typing discipline: Duck, dynamic, strong typing; gradual (since 3.5, but ignored in CPy
First appeared: February 1991; 30 years ago
Paradigm: Multi-paradigm: object-oriented, procedural (imperative), functional, structu
```

In [97]:

```
fi = list(filter(lambda x: x.isupper(), s))

print(fi)
```

```
['P', 'P', 'W', 'D', 'P', 'S', 'F', 'S', 'M', 'P', 'M', 'T', 'D', 'C', 'P',
'F', 'F', 'P', 'M']
```

In [103]:

```
sq = lambda num: num ** 2
```

In [104]:

```
sq(5)
```

Out[104]:

```
25
```

In [105]:

```
sq = lambda num: [num ** 2, num ** 3, num ** 4]
```

In [107]:

```
sq(5)
```

Out[107]:

```
[25, 125, 625]
```

In [113]:

```
1 print(list(filter(lambda x: x.islower(), s)))
```

```
['y', 't', 'h', 'o', 'n', 'i', 's', 'a', 'n', 'i', 'n', 't', 'e', 'r', 'p',
'r', 'e', 't', 'e', 'd', 'h', 'i', 'g', 'h', 'l', 'e', 'v', 'e', 'l', 'g',
'e', 'n', 'e', 'r', 'a', 'l', 'p', 'u', 'r', 'p', 'o', 's', 'e', 'p', 'r',
'o', 'g', 'r', 'a', 'm', 'm', 'i', 'n', 'g', 'l', 'a', 'n', 'g', 'u', 'a',
'g', 'e', 'y', 't', 'h', 'o', 'n', 's', 'd', 'e', 's', 'i', 'g', 'n', 'p',
'h', 'i', 'l', 'o', 's', 'o', 'p', 'h', 'y', 'e', 'm', 'p', 'h', 'a', 's',
'i', 'z', 'e', 's', 'c', 'o', 'd', 'e', 'r', 'e', 'a', 'd', 'a', 'b', 'i',
'l', 'i', 't', 'y', 'w', 'i', 't', 'h', 'i', 't', 's', 'n', 'o', 't', 'a',
'b', 'l', 'e', 'u', 's', 'e', 'o', 'f', 's', 'i', 'g', 'n', 'i', 'f', 'i',
'c', 'a', 'n', 't', 'i', 'n', 'd', 'e', 'n', 't', 'a', 't', 'i', 'o', 'n',
'i', 'k', 'i', 'p', 'e', 'd', 'i', 'a', 'e', 'v', 'e', 'l', 'o', 'p', 'e',
'r', 'y', 't', 'h', 'o', 'n', 'o', 'f', 't', 'w', 'a', 'r', 'e', 'o', 'u',
'n', 'd', 'a', 't', 'i', 'o', 'n', 't', 'a', 'b', 'l', 'e', 'r', 'e', 'l',
'e', 'a', 's', 'e', 'a', 'y', 'd', 'a', 'y', 's', 'a', 'g', 'o', 'r', 'e',
'v', 'i', 'e', 'w', 'r', 'e', 'l', 'e', 'a', 's', 'e', 'b', 'a', 'y', 'd',
'a', 'y', 's', 'a', 'g', 'o', 'y', 'p', 'i', 'n', 'g', 'd', 'i', 's', 'c',
'i', 'p', 'l', 'i', 'n', 'e', 'u', 'c', 'k', 'd', 'y', 'n', 'a', 'm', 'i',
'c', 's', 't', 'r', 'o', 'n', 'g', 't', 'y', 'p', 'i', 'n', 'g', 'g', 'r',
'a', 'd', 'u', 'a', 'l', 's', 'i', 'n', 'c', 'e', 'b', 'u', 't', 'i', 'g',
'n', 'o', 'r', 'e', 'd', 'i', 'n', 'y', 't', 'h', 'o', 'n', 'i', 'r', 's',
't', 'a', 'p', 'p', 'e', 'a', 'r', 'e', 'd', 'e', 'b', 'r', 'u', 'a', 'r',
'y', 'y', 'e', 'a', 'r', 's', 'a', 'g', 'o', 'a', 'r', 'a', 'd', 'i', 'g',
'm', 'u', 'l', 't', 'i', 'p', 'a', 'r', 'a', 'd', 'i', 'g', 'm', 'o', 'b',
'j', 'e', 'c', 't', 'o', 'r', 'i', 'e', 'n', 't', 'e', 'd', 'p', 'r', 'o',
'c', 'e', 'd', 'u', 'r', 'a', 'l', 'i', 'm', 'p', 'e', 'r', 'a', 't', 'i',
'v', 'e', 'f', 'u', 'n', 'c', 't', 'i', 'o', 'n', 'a', 'l', 's', 't', 'r',
'u', 'c', 't', 'u', 'r', 'e', 'd', 'r', 'e', 'f', 'l', 'e', 'c', 't', 'i',
'v', 'e']
```

In [112]:

```
1 print(list(map(lambda x: x.islower(), s)))
```

```
[False, True, True, True, True, True, False, True, True, False, True, Tru
e, False, True, True, True, True, True, True, True, True, True, True, Tru
e, False, True, True, True, True, False, True, True, True, True, True, Fal
se, True, True, True, True, True, True, True, False, True, True, True, Tru
e, True, True, True, False, True, True, True, True, True, True, True, Tru
e, True, True, True, False, True, True, True, True, True, True, True, Tru
e, False, False, False, True, True, True, True, True, False, True, False,
True, True, True, True, True, True, True, False, True, True, True, True, T
rue, True, True, True, True, False, True, True, True, True, True, True, Tr
ue, True, True, True, False, True, True, True, True, False, True, True, Tr
ue, True, True, True, True, True, True, True, True, False, True, True, Tru
e, True, False, True, True, True, False, True, True, True, True, True, Tru
e, True, False, True, True, True, False, True, True, False, True, True, Tr
ue, True, True, True, True, True, True, True, True, False, True, True, Tru
e, True, True, True, True, True, True, True, True, False, False, False, Tr
ue, True, True, True, True, True, True, True, False, False, True, True, Tr
ue, True, True, True, True, True, False, False, False, True, True, True, T
rue, True, False, False, True, True, True, True, True, True, True, False,
False. True. True. True. True. True. True. True. True. True. False. False.
```

In [111]:

```
1 list(filter(lambda x: x.islower(), input()))
```

dbvdbvSDBFKJSDBVKdnfkjvbdk

Out[111]:

['d', 'b', 'v', 'd', 'b', 'v', 'd', 'n', 'f', 'k', 'j', 'v', 'b', 'd', 'k']

In [ ]:

```
1
```