Python

Assignment 4

[4]:

Exercise 1: Create a function with a default argument

**def** show\_emp(name,salary=9000): print("employee name : "+name) print("Salary is : "+str(salary))

show\_emp("Ben", 12000) show\_emp("Jessa")

[23]:

employee name : Ben Salary is : 12000 employee name : Jessa Salary is : 9000

Exercise 2: Create an inner function to calculate the addition in the following way Create an outer function that will accept two parameters, a and b Create an inner function inside an outer function that will calculate the addition of a and b At last, an outer function will add 5 into addition and return it

**def** addnum(a,b):

**def** add():

tot=a+b add()

**return** tot+5 sum=addnum(5,10) print(sum)

[26]:

20

Exercise 3: Generate a Python list of all the even numbers between 4 to 30

**def** even\_num(): evn=[]

**for** x **in** range(4, 31):

**if** x % 2 == 0: evn.append(x)

print(evn)

even\_num()

[11]:

[4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30]

Exercise 4: Lambda Function to Check if value is in a List Given a list, the task is to write a Python program to check if the value exists in the list or not using the lambda function. Input : L

= [1, 2, 3, 4, 5] element = 4 Output : Element is Present in the list

Input : L = [1, 2, 3, 4, 5] element = 8 Output : Element is NOT Present in the list

L = [1, 2, 3, 4, 5]

elem =5

x = **lambda** L, elem: **True if** elem **in** L **else False**

**if** (x(L,elem)):

print ("Element is present")

**else**:

print ("Element is not present")

*# element=8*

elem=8

x = **lambda** L, elem: **True if** elem **in** L **else False**

**if** (x(L,elem)):

print ("Element is present")

**else**:

print ("Element is not present")

[12]:

Element is present Element is not present

Exercise 5: Sort list of tuples with their sum Sort the points based on their sum of elements in the tuples

points = [(1, 2), (5, 3), (0, 7), (3, 1)]

points = [(1, 2), (5, 3), (0, 7), (3, 1)]

print("The sorted list of tuple is :**\n**") print(sorted(points, key = **lambda** x:x[0] + x[1]))

[13]:

The sorted list of tuple is :

[(1, 2), (3, 1), (0, 7), (5, 3)]

Exercise 6 : Write a python function, which will find all such numbers between 1000 and 3000 (both included) such that each digit of the number is an even number. Return the results as a list

**def** evn\_num(): evnlist=[]

**for** x **in** range(1000, 3001):

**if** x % 2 == 0:

evnlist.append(x)

**return** evnlist ans=evn\_num() print(ans)

[1000, 1002, 1004, 1006, 1008, 1010, 1012, 1014, 1016, 1018, 1020, 1022, 1024,

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1026, | 1028, | 1030, | 1032, | 1034, | 1036, | 1038, | 1040, | 1042, | 1044, | 1046, | 1048, | 1050, |
| 1052, | 1054, | 1056, | 1058, | 1060, | 1062, | 1064, | 1066, | 1068, | 1070, | 1072, | 1074, | 1076, |
| 1078, | 1080, | 1082, | 1084, | 1086, | 1088, | 1090, | 1092, | 1094, | 1096, | 1098, | 1100, | 1102, |
| 1104, | 1106, | 1108, | 1110, | 1112, | 1114, | 1116, | 1118, | 1120, | 1122, | 1124, | 1126, | 1128, |
| 1130, | 1132, | 1134, | 1136, | 1138, | 1140, | 1142, | 1144, | 1146, | 1148, | 1150, | 1152, | 1154, |
| 1156, | 1158, | 1160, | 1162, | 1164, | 1166, | 1168, | 1170, | 1172, | 1174, | 1176, | 1178, | 1180, |
| 1182, | 1184, | 1186, | 1188, | 1190, | 1192, | 1194, | 1196, | 1198, | 1200, | 1202, | 1204, | 1206, |
| 1208, | 1210, | 1212, | 1214, | 1216, | 1218, | 1220, | 1222, | 1224, | 1226, | 1228, | 1230, | 1232, |
| 1234, | 1236, | 1238, | 1240, | 1242, | 1244, | 1246, | 1248, | 1250, | 1252, | 1254, | 1256, | 1258, |
| 1260, | 1262, | 1264, | 1266, | 1268, | 1270, | 1272, | 1274, | 1276, | 1278, | 1280, | 1282, | 1284, |
| 1286, | 1288, | 1290, | 1292, | 1294, | 1296, | 1298, | 1300, | 1302, | 1304, | 1306, | 1308, | 1310, |
| 1312, | 1314, | 1316, | 1318, | 1320, | 1322, | 1324, | 1326, | 1328, | 1330, | 1332, | 1334, | 1336, |
| 1338, | 1340, | 1342, | 1344, | 1346, | 1348, | 1350, | 1352, | 1354, | 1356, | 1358, | 1360, | 1362, |
| 1364, | 1366, | 1368, | 1370, | 1372, | 1374, | 1376, | 1378, | 1380, | 1382, | 1384, | 1386, | 1388, |
| 1390, | 1392, | 1394, | 1396, | 1398, | 1400, | 1402, | 1404, | 1406, | 1408, | 1410, | 1412, | 1414, |
| 1416, | 1418, | 1420, | 1422, | 1424, | 1426, | 1428, | 1430, | 1432, | 1434, | 1436, | 1438, | 1440, |
| 1442, | 1444, | 1446, | 1448, | 1450, | 1452, | 1454, | 1456, | 1458, | 1460, | 1462, | 1464, | 1466, |
| 1468, | 1470, | 1472, | 1474, | 1476, | 1478, | 1480, | 1482, | 1484, | 1486, | 1488, | 1490, | 1492, |
| 1494, | 1496, | 1498, | 1500, | 1502, | 1504, | 1506, | 1508, | 1510, | 1512, | 1514, | 1516, | 1518, |
| 1520, | 1522, | 1524, | 1526, | 1528, | 1530, | 1532, | 1534, | 1536, | 1538, | 1540, | 1542, | 1544, |
| 1546, | 1548, | 1550, | 1552, | 1554, | 1556, | 1558, | 1560, | 1562, | 1564, | 1566, | 1568, | 1570, |
| 1572, | 1574, | 1576, | 1578, | 1580, | 1582, | 1584, | 1586, | 1588, | 1590, | 1592, | 1594, | 1596, |
| 1598, | 1600, | 1602, | 1604, | 1606, | 1608, | 1610, | 1612, | 1614, | 1616, | 1618, | 1620, | 1622, |
| 1624, | 1626, | 1628, | 1630, | 1632, | 1634, | 1636, | 1638, | 1640, | 1642, | 1644, | 1646, | 1648, |
| 1650, | 1652, | 1654, | 1656, | 1658, | 1660, | 1662, | 1664, | 1666, | 1668, | 1670, | 1672, | 1674, |
| 1676, | 1678, | 1680, | 1682, | 1684, | 1686, | 1688, | 1690, | 1692, | 1694, | 1696, | 1698, | 1700, |
| 1702, | 1704, | 1706, | 1708, | 1710, | 1712, | 1714, | 1716, | 1718, | 1720, | 1722, | 1724, | 1726, |
| 1728, | 1730, | 1732, | 1734, | 1736, | 1738, | 1740, | 1742, | 1744, | 1746, | 1748, | 1750, | 1752, |
| 1754, | 1756, | 1758, | 1760, | 1762, | 1764, | 1766, | 1768, | 1770, | 1772, | 1774, | 1776, | 1778, |
| 1780, | 1782, | 1784, | 1786, | 1788, | 1790, | 1792, | 1794, | 1796, | 1798, | 1800, | 1802, | 1804, |
| 1806, | 1808, | 1810, | 1812, | 1814, | 1816, | 1818, | 1820, | 1822, | 1824, | 1826, | 1828, | 1830, |
| 1832, | 1834, | 1836, | 1838, | 1840, | 1842, | 1844, | 1846, | 1848, | 1850, | 1852, | 1854, | 1856, |
| 1858, | 1860, | 1862, | 1864, | 1866, | 1868, | 1870, | 1872, | 1874, | 1876, | 1878, | 1880, | 1882, |
| 1884, | 1886, | 1888, | 1890, | 1892, | 1894, | 1896, | 1898, | 1900, | 1902, | 1904, | 1906, | 1908, |
| 1910, | 1912, | 1914, | 1916, | 1918, | 1920, | 1922, | 1924, | 1926, | 1928, | 1930, | 1932, | 1934, |
| 1936, | 1938, | 1940, | 1942, | 1944, | 1946, | 1948, | 1950, | 1952, | 1954, | 1956, | 1958, | 1960, |
| 1962, | 1964, | 1966, | 1968, | 1970, | 1972, | 1974, | 1976, | 1978, | 1980, | 1982, | 1984, | 1986, |
| 1988, | 1990, | 1992, | 1994, | 1996, | 1998, | 2000, | 2002, | 2004, | 2006, | 2008, | 2010, | 2012, |
| 2014, | 2016, | 2018, | 2020, | 2022, | 2024, | 2026, | 2028, | 2030, | 2032, | 2034, | 2036, | 2038, |
| 2040, | 2042, | 2044, | 2046, | 2048, | 2050, | 2052, | 2054, | 2056, | 2058, | 2060, | 2062, | 2064, |
| 2066, | 2068, | 2070, | 2072, | 2074, | 2076, | 2078, | 2080, | 2082, | 2084, | 2086, | 2088, | 2090, |

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 2092, | 2094, | 2096, | 2098, | 2100, | 2102, | 2104, | 2106, | 2108, | 2110, | 2112, | 2114, | 2116, |
| 2118, | 2120, | 2122, | 2124, | 2126, | 2128, | 2130, | 2132, | 2134, | 2136, | 2138, | 2140, | 2142, |
| 2144, | 2146, | 2148, | 2150, | 2152, | 2154, | 2156, | 2158, | 2160, | 2162, | 2164, | 2166, | 2168, |
| 2170, | 2172, | 2174, | 2176, | 2178, | 2180, | 2182, | 2184, | 2186, | 2188, | 2190, | 2192, | 2194, |
| 2196, | 2198, | 2200, | 2202, | 2204, | 2206, | 2208, | 2210, | 2212, | 2214, | 2216, | 2218, | 2220, |
| 2222, | 2224, | 2226, | 2228, | 2230, | 2232, | 2234, | 2236, | 2238, | 2240, | 2242, | 2244, | 2246, |
| 2248, | 2250, | 2252, | 2254, | 2256, | 2258, | 2260, | 2262, | 2264, | 2266, | 2268, | 2270, | 2272, |
| 2274, | 2276, | 2278, | 2280, | 2282, | 2284, | 2286, | 2288, | 2290, | 2292, | 2294, | 2296, | 2298, |
| 2300, | 2302, | 2304, | 2306, | 2308, | 2310, | 2312, | 2314, | 2316, | 2318, | 2320, | 2322, | 2324, |
| 2326, | 2328, | 2330, | 2332, | 2334, | 2336, | 2338, | 2340, | 2342, | 2344, | 2346, | 2348, | 2350, |
| 2352, | 2354, | 2356, | 2358, | 2360, | 2362, | 2364, | 2366, | 2368, | 2370, | 2372, | 2374, | 2376, |
| 2378, | 2380, | 2382, | 2384, | 2386, | 2388, | 2390, | 2392, | 2394, | 2396, | 2398, | 2400, | 2402, |
| 2404, | 2406, | 2408, | 2410, | 2412, | 2414, | 2416, | 2418, | 2420, | 2422, | 2424, | 2426, | 2428, |
| 2430, | 2432, | 2434, | 2436, | 2438, | 2440, | 2442, | 2444, | 2446, | 2448, | 2450, | 2452, | 2454, |
| 2456, | 2458, | 2460, | 2462, | 2464, | 2466, | 2468, | 2470, | 2472, | 2474, | 2476, | 2478, | 2480, |
| 2482, | 2484, | 2486, | 2488, | 2490, | 2492, | 2494, | 2496, | 2498, | 2500, | 2502, | 2504, | 2506, |
| 2508, | 2510, | 2512, | 2514, | 2516, | 2518, | 2520, | 2522, | 2524, | 2526, | 2528, | 2530, | 2532, |
| 2534, | 2536, | 2538, | 2540, | 2542, | 2544, | 2546, | 2548, | 2550, | 2552, | 2554, | 2556, | 2558, |
| 2560, | 2562, | 2564, | 2566, | 2568, | 2570, | 2572, | 2574, | 2576, | 2578, | 2580, | 2582, | 2584, |
| 2586, | 2588, | 2590, | 2592, | 2594, | 2596, | 2598, | 2600, | 2602, | 2604, | 2606, | 2608, | 2610, |
| 2612, | 2614, | 2616, | 2618, | 2620, | 2622, | 2624, | 2626, | 2628, | 2630, | 2632, | 2634, | 2636, |
| 2638, | 2640, | 2642, | 2644, | 2646, | 2648, | 2650, | 2652, | 2654, | 2656, | 2658, | 2660, | 2662, |
| 2664, | 2666, | 2668, | 2670, | 2672, | 2674, | 2676, | 2678, | 2680, | 2682, | 2684, | 2686, | 2688, |
| 2690, | 2692, | 2694, | 2696, | 2698, | 2700, | 2702, | 2704, | 2706, | 2708, | 2710, | 2712, | 2714, |
| 2716, | 2718, | 2720, | 2722, | 2724, | 2726, | 2728, | 2730, | 2732, | 2734, | 2736, | 2738, | 2740, |
| 2742, | 2744, | 2746, | 2748, | 2750, | 2752, | 2754, | 2756, | 2758, | 2760, | 2762, | 2764, | 2766, |
| 2768, | 2770, | 2772, | 2774, | 2776, | 2778, | 2780, | 2782, | 2784, | 2786, | 2788, | 2790, | 2792, |
| 2794, | 2796, | 2798, | 2800, | 2802, | 2804, | 2806, | 2808, | 2810, | 2812, | 2814, | 2816, | 2818, |
| 2820, | 2822, | 2824, | 2826, | 2828, | 2830, | 2832, | 2834, | 2836, | 2838, | 2840, | 2842, | 2844, |
| 2846, | 2848, | 2850, | 2852, | 2854, | 2856, | 2858, | 2860, | 2862, | 2864, | 2866, | 2868, | 2870, |
| 2872, | 2874, | 2876, | 2878, | 2880, | 2882, | 2884, | 2886, | 2888, | 2890, | 2892, | 2894, | 2896, |
| 2898, | 2900, | 2902, | 2904, | 2906, | 2908, | 2910, | 2912, | 2914, | 2916, | 2918, | 2920, | 2922, |
| 2924, | 2926, | 2928, | 2930, | 2932, | 2934, | 2936, | 2938, | 2940, | 2942, | 2944, | 2946, | 2948, |
| 2950, | 2952, | 2954, | 2956, | 2958, | 2960, | 2962, | 2964, | 2966, | 2968, | 2970, | 2972, | 2974, |
| 2976, | 2978, | 2980, | 2982, | 2984, | 2986, | 2988, | 2990, | 2992, | 2994, | 2996, | 2998, | 3000] |

[5]:

**def** cnt():

x=input("enter a string : ") l=n=0

**for** a **in** x:

**if** a.isalpha(): l=l+1

**elif** a.isdigit(): n=n+1

Exercise 7 : Write a python function that accepts a sentence and calculate and return the number of letters and digits. Suppose the following input is supplied to the program: hello world! 123 Then, the output should be: LETTERS 10 DIGITS 3

**else**:

**pass** print("Letters :"+str(l)) print("Numbers : "+str(n))

cnt()

[ ]:

enter a string : hello world! 123 Letters :10

Numbers : 3

Exercise 8 MAP:

Write a Python program to convert all the characters into uppercase **and**␣

𝗌lowercase **and** eliminate duplicate

letters **from a** given sequence.Use the map() function

[6]:

**def** case\_func(st):

**return** str(st).upper(), str(st).lower()

chr = {'a', 'b', 'c', 'f', 'a', 'G', 'o', 'U', 'W'}

res = map(case\_func, chr) print(set(res))

[8]:

{('A', 'a'), ('B', 'b'), ('F', 'f'), ('C', 'c'), ('W', 'w'), ('G', 'g'), ('O', 'o'), ('U', 'u')}

Exercise 9 MAP: Write a Python program to add two given lists and find the difference between them. Use the map() function

ls1 = [9,8,7]

ls2 = [4,5,6]

diff = list(map(**lambda** x, y: x - y, ls1, ls2)) print("The difference between the two lists is:", diff)

[13]:

The difference between the two lists is: [5, 3, 1]

Write a Python program to filter the height and weight of students, which are stored in a dictionary using lambda. Original Dictionary: {‘Cierra Vega’: (6.2, 71), ‘Alden Cantrell’: (5.9, 65), ‘Kierra

Gentry’: (6.0, 68), ‘Pierre Cox’: (5.8, 66)} Height> 6ft and Weight> 70kg: {‘Cierra Vega’: (6.2,

71)}

dic={'Cierra Vega': (6.2, 71), 'Alden Cantrell': (5.9, 65), 'Kierra Gentry': (6.

𝗌0, 68), 'Pierre Cox': (5.8, 66)}

**def** fil(dic):

res = dict(filter(**lambda** x: (x[1][0], x[1][1]) > (6.0, 70), dic.items()))

**return** res

print(fil(dic))

{'Cierra Vega': (6.2, 71)}

[14]:

Exercise 10 Filter: Write a Python program to remove all elements from a given list present in another list using lambda. Original lists: list1: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10] list2: [2, 4, 6, 8] Remove

all elements from ‘list1’ present in ’list2: [1, 3, 5, 7, 9, 10]

list1=[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

list2=[2, 4, 6, 8]

**def** rem(list1, list2):

newls = list(filter(**lambda** x: x **not in** list2, list1))

**return** newls print(rem(list1, list2))

[3]:

[1, 3, 5, 7, 9, 10]

Exercise 11 Reduce:

Write a Python program to calculate the product of a given list of numbers using lambda. list1: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10] Product of the said list numbers: 3628800

**import functools**

list1=[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

print(functools.reduce(**lambda** a, b: a\*b, list1))

[4]:

3628800

Exercise 12 Reduce:

Write a Python program to multiply all the numbers in a given list using lambda. Original list: [4, 3, 2, 2, -1, 18] multiply all the numbers of the said list: -864

**import functools**

l1=[4, 3, 2, 2, -1, 18]

print(functools.reduce(**lambda** a, b: a\*b, l1))

[6]:

-864

Exercise 13 Reduce:

Write a Python program to calculate the average value of the numbers in a given tuple of tuples using lambda. Original Tuple: ((10, 10, 10), (30, 45, 56), (81, 80, 39), (1, 2, 3)) Average value of the numbers of the said tuple of tuples: (30.5, 34.25, 27.0)

**from functools import** reduce

t1=((10, 10, 10), (30, 45, 56), (81, 80, 39), (1, 2, 3))

result = tuple(reduce(**lambda** x, y: map(sum, zip(x,y)), t1))

print(f"Average value of tuples:**\n{**tuple(map(**lambda** x: x/len(t1), result))**}**")

Average value of tuples: (30.5, 34.25, 27.0)

[2]:

Exercise 13:

Write a Python program to sort a given mixed list of integers and strings using lambda. Numbers must be sorted before strings. Original list: [19, ‘red’, 12, ‘green’, ‘blue’, 10, ‘white’, ‘green’, 1] Sort the said mixed list of integers and strings: [1, 10, 12, 19, ‘blue’, ‘green’, ‘green’, ‘red’, ‘white’]

li1=[19, 'red', 12, 'green', 'blue', 10, 'white', 'green', 1]

sortlist = **lambda** li1: sorted(li1, key=**lambda** x: (isinstance(x, str), x)) print(sortlist(li1))

[3]:

[1, 10, 12, 19, 'blue', 'green', 'green', 'red', 'white']

Exercise 14: Write a Python program to count the occurrences of items in a given list using lambda. Original list: [3, 4, 5, 8, 0, 3, 8, 5, 0, 3, 1, 5, 2, 3, 4, 2]

mylist=[3, 4, 5, 8, 0, 3, 8, 5, 0, 3, 1, 5, 2, 3, 4, 2]

result = dict(map(**lambda** el : (el, list(mylist).count(el)), mylist)) print(result)

[4]:

{3: 4, 4: 2, 5: 3, 8: 2, 0: 2, 1: 1, 2: 2}

Write a Python program to remove None values from a given list using the lambda function. Original list: [12, 0, None, 23, None, -55, 234, 89, None, 0, 6, -12] Remove None value from the

said list: [12, 0, 23, -55, 234, 89, 0, 6, -12]

nwlist=[12, 0, **None**, 23, **None**, -55, 234, 89, **None**, 0, 6, -12] rst = filter(**lambda** v: v **is not None**, nwlist) print(list(rst))

[12, 0, 23, -55, 234, 89, 0, 6, -12]

[ ]: