

|  |  |  |  |
| --- | --- | --- | --- |
| DEPARTMENT | **MCA** | SEMESTER | **SUMMER SEM. I 2021-2022** |
| COURSE | **PYTHON** | CODE | **ITA-6017** |
| FACULTY | **PROF. RAJRAJESWARI S.** | SLOT | **D1/D2 / L11/L12** |
| STUDENT NAME **:** | **DINESH YOGESH PAREKH** | REG NO | **20MCA1013** |

**Lab Assignment 4.1( 20.7.21) Morning**

1. **Write a program which reads a string from a user and create a list with the set of strings where each string is pair of complete parenthesis cluster.**

**Example:**

**"()()()" ➞ ["()", "()", "()"]**

**"((()))" ➞ ["((()))"]**

**"((()))(())()()(()())" ➞ ["((()))", "(())", "()", "()", "(()())"]**

**"((())())(()(()()))" ➞ ["((())())", "(()(()()))"]**

**CODE:**

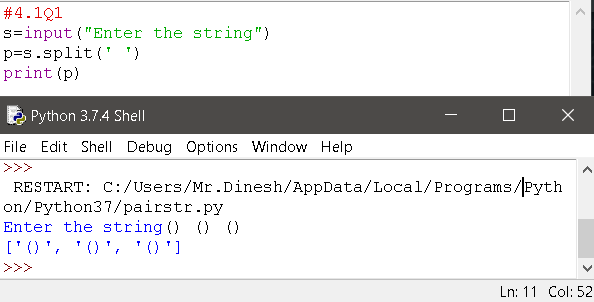
#4.1Q1

s=input("Enter the string")

p=s.split(' ')

print(p)

**OUTPUT:**



1. **A prison can be represented as a list of cells. Each cell contains exactly one prisoner where 1 represents an unlocked cell and a 0 represents a locked cell.**

**Example :**

**[1, 1, 0, 0, 0, 1, 0]**

**Starting from the left most cell. If the cell is unlocked (1) then that prisoner will be made as free and if the cell is locked(0) then he cannot come out. Once the prisoner is made as free all the locked cell become unlocked and all unlocked cell become locked. Once again the process will be continued from where you left in the last iteration ( should not begin from the first).**

**So, if we use the example above:**

**Step1 :**

**[1, 1, 0, 0, 0, 1, 0]**

**# You free the prisoner in the 1st cell. The next iteration will be continued frrmt he second position not from the first.**

**Step2 :**

**[0, 0, 1, 1, 1, 0, 1]**

**# You free the prisoner in the 3rd cell (2nd one locked).**

**Step3 :**

**[1, 1, 0, 0, 0, 1, 0]**

**# You free the prisoner in the 6th cell (3rd, 4th and 5th locked).**

**Step4 :**

**[0, 0, 1, 1, 1, 0, 1]**

**# You free the prisoner in the 7th cell - and you are done!**

**Here, we have set free 4 prisoners in total.**

**CODE:**

#4.Q2

import itertools

def freed\_prisoners(prison):

groups = [(k,list(v)) for k,v in itertools.groupby(prison)]

print(f"Groups: {groups}")

freed = [k for k, \_ in groups]

persons = len(freed) if freed[0] == 1 else 0

print(f"Freed list: {freed}; freed {persons} persons")

return persons

freed\_prisoners([1, 1, 0, 0, 0, 1, 0]) # ➞ 4

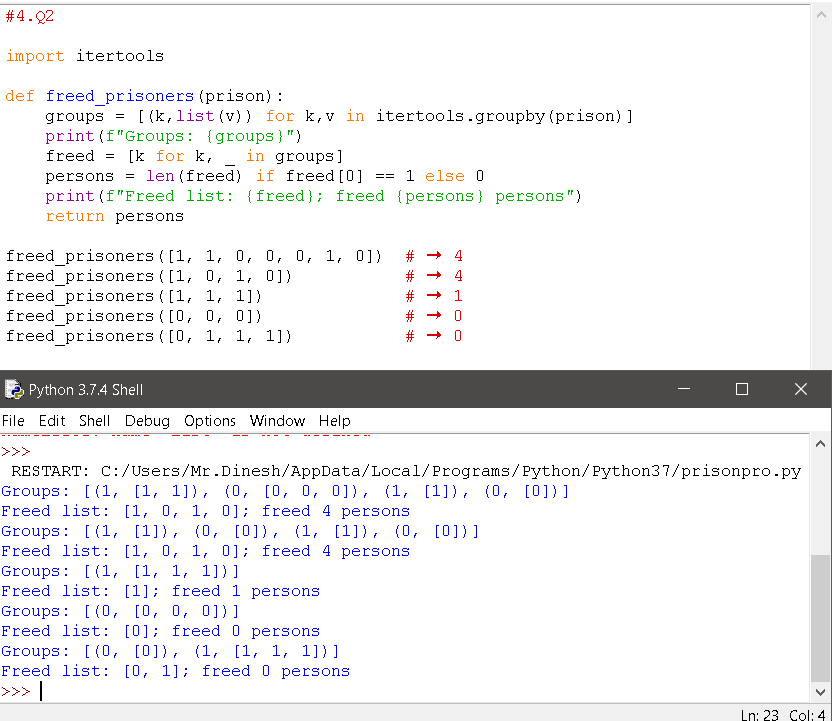
freed\_prisoners([1, 0, 1, 0]) # ➞ 4

freed\_prisoners([1, 1, 1]) # ➞ 1

freed\_prisoners([0, 0, 0]) # ➞ 0

freed\_prisoners([0, 1, 1, 1]) # ➞ 0

**OUTPUT:**



1. **Write a program that takes a list of numbers or strings and prints a list with the items from the original list stored into set of tuples (In the sorted order of the value). Items of the same value should be in the same tuple.**

**Examples**

**[2, 1, 2, 1] ➞ [(2, 2), (1, 1)]**

**[5, 4, 5, 5, 4, 3] ➞ [(5, 5, 5), (4, 4), (3)]**

**["b", "a", "b", "a", "c"] ➞ [("b", "b"), ("a", "a"), ("c")]**

**CODE:**

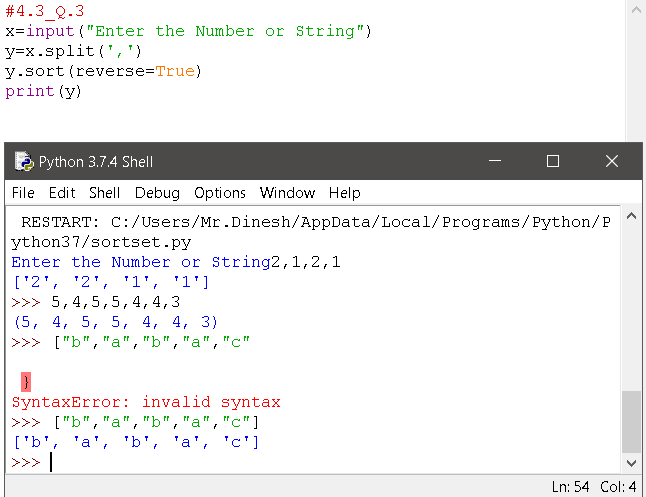
x=input("Enter the Number or String")

y=x.split(',')

y.sort(reverse=True)

print(y)

**OUTPUT:**



***Thank you***