**NAME –** Siddharth Ladha

**REG. NO –** RA1811003010554

**EX – 1**

**TOY PROBLEM**

PROBLEM -

You are given the array paths, where paths[i] = [cityAi, cityBi] means there exists a direct path going from cityAi to cityBi. *Return the destination city, that is, the city without any path outgoing to another city.*

It is guaranteed that the graph of paths forms a line without any loop, therefore, there will be exactly one destination city.

SOLUTION –

class Solution:

def destCity(self, paths) -> str:

dic ={}

for i in range(len(paths)):

if paths[i][0] not in dic:

dic[i]= paths[i][0]

for j in range(len(paths)):

if paths[j][1] not in dic.values():

return paths[j][1]

**Example 1:**

**Input:** paths = [["B","C"],["D","B"],["C","A"]]

**Output:** "A"

**Explanation:** All possible trips are:

"D" -> "B" -> "C" -> "A".

"B" -> "C" -> "A".

"C" -> "A".

"A".

Clearly the destination city is "A".

**1.**What is a toy problem?

**Ans :** A toy problem is a problem that doesn't have all the complexity of a real-world engineering problem. It could be a simplified or shallow version of a more difficult and intricate problem or class of problems.

Solving a small toy problem can be beneficial in the context of learning a new field or skill, as an exercise, or as a way to eventually work your way into the larger, real-world problem.

**2.**How many approaches do you have for solving the toy problem which you have taken?

**Ans :** There are multiple approaches, the above one is using dictionary in python 3. We have used for loop to solve the above situation. The same problem can also be solved using O(n) approach.

**3.**Map your toy problem to any real life application.

**Ans :** This situation can be used in navigational Instruments such as GPS tracking system, maps etc. to reach a suitable destination.

***GITHUB*** - <https://github.com/Shristi-98/Design-Parking-System>