Home Work № 9

CSCI B505 Applied Algorithms 11/15/2022

Instructions

1. Under any circumstances the deadline will NOT be extended.
2. Code should be written in python language in a Jupyter notebook .
3. Unless mentioned in the questions, feel free to use built-in python functions
4. Upload Jupyter notebook and html file of the Jupyter notebook in canvas. No other forms submissions is considered as valid submission. Make sure to have the output of the required cells of the jupyter notebook and its html version before making submission.
5. Plagiarism is unacceptable and we have ways to find it. So do not do it.
6. There are four problems. Each problem is worth 25 points.
7. The code should readable with variables named meaningfully.
8. Write test cases wherever required so that they cover all scenarios.
9. students are expected to use the function signature given in the question or they'll get a penalty.

Problem 1

Alice is fond of painting with blue and red colors.She was given a graph with vertices and edges and her task is to paint the vertices with favorite two colors such that adjacent vertices do not have the same color.
Write a program which takes the graph as adjacency matrix and returns true if Alice succeeds else false

```
Example:

    adj_matrix = [[0,1,1,0], [1,0,0,1], [1,0,0,1],[0, 1, 1, 0]]
    output = True
```

Following the below given method signature

```
1
2 def can_color(adj_matrix):
3     ## logic here
4     pass
```

Problem 2

You are given a list of connections between train stations, a source station number and a destination station number. Your job is to determine if all the routes from source station end at the destination station. Return true if all the outgoing routes from source end at destination station.
Input: n: int Number of stations
connections: List[List[int]] list of pairs of connected stations
begin: int Source station number
end: int Destination station number .
Expected Time complexity <= $O(n^2)$

```
Example 1:

n = 3, connections = [[0,1],[0,2]], begin = 0, end = 2
Ans: false
There is no outgoing route from station 1. hence starting from begin(0) and
ending at station 1 will get us stuck at station 1.


Example 2:

n = 4, connections = [[0,1],[0,3],[1,2],[2,1]], begin = 0, end = 3
Ans: false
Only outgoing route from station 2 is towards station 1 and
from station 1 is towards station 2. Hence we will get stuck indefinitely between these
two stations if we end up at station 1 from begin (0).


Example 3:

n = 4, connections = [[0,1],[0,2],[1,3],[2,3]], begin = 0, end = 3
Ans: true
```

Use following function signature:

```
1
2 def all_routes_to_dest(n: int, connections: List[List[int]],
3              begin: int, end: int) -> bool:
4     ########logic#######
5
6     ####################
7     return ans
```

Problem 3

You are given an 1D array, Find if there is a cycle in the given array.Assume that moving forward from the last element puts you on the first element and moving backwards from the first element puts you on the last element. So, assumption is it is circular array.

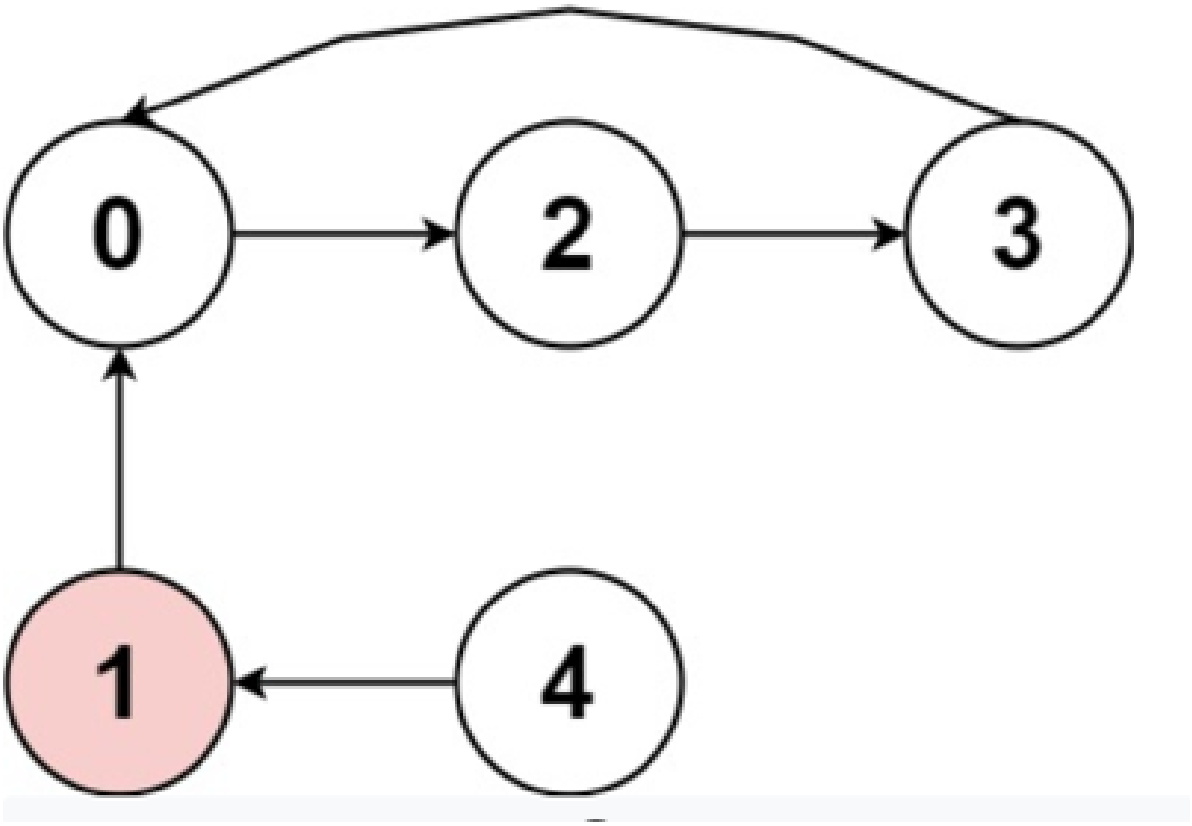
```
    if arr[i] > 0, move arr[i] steps forward
    if arr[i] < 0, move arr[i] steps backward
    if arr[i] = 0, stay idle

    Example 1:

    Input array - [2, -1, 1, 2, 2]

    Output- True (Cycle Exists)

Explanation:
We can see the cycle 0 --> 2 --> 3 --> 0 --> ..., and all
its nodes are white (jumping in the same direction).
```




```
def checkCycle(arr):
  ### Logic ###
  #############
  return True/False
```

Problem 4

You are given a list of estimated connection costs between 'n' airports. Your job is to find the minimum cost to connect these airports such that there is at least one path(direct or indirect) between any two airports. If it is not possible to connect all airports in a way described above, return -1. You can assume that airport names are always 1,2,3,4,....,n in the input.
Input: n : integer = total number of airports
costs: list of lists. $costs[i] = [airport\_a, airport\_b, cost\_ab] = cost\_ab$ is the connection cost between given two airports($airport\_a \& airport\_b$). All numbers are integers.
Expected time complexity = $Llog(n)$ where L = length of costs.

```
## This is how the usage looks like

Example1 :
```

```
n = 3, costs = [[1,2,4],[1,3,9],[2,3,7]]
output: 11
Airport 1 can be connected to 2 and 2 can be connected to 3. This way all airports
are connected (1 and 3 are indirectly connected through 2)
with minimum cost of connections 4+7 = 11.

Example 2:

n = 4, costs = [[1,2,3],[3,4,4]]
output = -1
It is not possible to connect
all airports in this case.
```

use the following function signature:

```python
def get_min_cost(n: int, costs: List[List[int]]) -> int:
    ####### your logic ###

    ######################
    return mincost
```