BÀI KIỂM TRA GIỮA KỲ - TTNT2024 - B

Thời gian: 60 phút

I. Quy định nộp bài

- Sinh viên phải có mặt tại lớp khi làm bài và nộp bài,
- Sinh viên phải nộp một file duy nhất, **ID.py**, chứa các hàm được yêu cầu. Mọi câu lệnh khác để kiểm thử / debug phải được thực hiện trong hàm main.

Lưu ý: Thay thế ID bằng mã số sinh viên của bạn.

II. Quy tắc chấm điểm

- Bài nộp của bạn sẽ được biên dịch bằng câu lệnh terminal sau:

```
>> python id.py -V:3.14
```

- Nếu có bất kỳ hàm nào chưa có lời giải, phần thân hàm phải được để trống (`pass`),
- KHÔNG ĐƯỢC thay đổi nguyên mẫu (prototypes) của các hàm/class được cung cấp trong đề,
- Không chấm điểm ý tưởng, chỉ chấm các chương trình chạy được,
- Các bài nộp sau đây sẽ bị đánh dấu 0 điểm nếu:
    - Vi phạm quy định,
    - Thay đổi nguyên mẫu của các hàm/class,
    - Bài nộp tương tự nhau,
    - Bài nộp không thể biên dịch/không chạy được,
    - Tạo ra vòng lặp vô hạn.

(Xem các trang tiếp theo để xem nội dung đề thi)

Content

Lưu ý: Chỉ các thư viện sau đây được phép sử dụng trong bài làm:

```
import json
from typing import List, Tuple
from collections import deque
```

Cho file dữ liệu `filename` chứa thông tin các Chuyến bay. Thông tin của mỗi chuyến bay được trình bày trên một dòng riêng biệt, với định dạng sau:.

```
{"From,To": ["Flight", "business, economy", "hours, minutes"]}
{"Myanmar,South Sudan": ["Comac 90", "11 business, 84 economy", "9 hours, 25 minutes"]}
{"Faeroe Islands,Ethiopia": ["Comac C919", "92 business, 98 economy", "9 hours"]}
```

Hãy thực hiện các yêu cầu sau:

A. (1) Viết hàm nhận file đã cho làm đầu vào và trả về một dictionary các chuyến bay, giữ nguyên thứ tự mà các chuyến bay xuất hiện trong file.

```
  def readfile(filename: str) -> dict:
```

B. (1) Với dictionary Flights chứa thông tin chuyến bay, hãy tạo class FGraph đại diện cho một đồ thị vô hướng có trọng số G(V, E, W), trong đó đỉnh V là các quốc gia, cạnh E kết nối các quốc gia có chung chuyến bay, và trọng số W bao gồm thông tin chung của chuyến bay và giá vé hạng phổ thông của các chuyến bay đó.

Giá vé hạng phổ thông được tính theo công thức:

> $ticket_{economy} = \frac{Ct*time+Cc*seat}{3*business+economy}$
> trong đó Ct là chi phí nguyên liệu mỗi phút $= \$275$, Cc là chi phí cố định cho mỗi chỗ ngồi $= \$35$, seat là tổng số chỗ ngồi bao gồm cả hạng thương gia và phổ thông

C. (3) Xây dựng một đồ thị vô hướng có trọng số, FGraph, bao gồm các chuyến bay có số hiệu đầy đủ (số hiệu nằm trong tên chuyến bay, và được xem là đầy đủ khi chứa ít nhất 1 chữ số và ít nhất 1 dấu gạch '-') từ dictionary Flights. Sau khi có đồ thị, viết hàm tìm tất cả các quốc gia có chuyến bay trực tiếp từ một quốc gia cho trước.

```
def __init__(self, Flights: dict) -> None
def findDirectFlights(self, country_name: str) -> List[str]
```

D. (3) Với đồ thị được tạo từ câu C và tên 1 quốc gia bất kì, viết hàm tìm thêm bốn quốc gia khác mà cùng với quốc gia đã cho, tạo thành một đồ thị hai phía (bipartite graph).

```
def find4Bipartite(G: FGraph, country_name: str) -> List[str]
```

E.[Counting method] (2) Viết hàm đếm số cây nhị phân tìm kiếm khác nhau có thể tạo từ n nút có giá trị từ 1 đến n. Yêu cầu: dùng đệ quy hoặc quy hoạch động.

```
def findBST(n: int) -> int
```