algograf

Projekt 2: Zagubiony w grobowcu

W ramach projektu należy napisać program w języku Python rozwiązujący poniższe zadanie. Następnie program należy wysłać do oceny poprzez następujący kurs na platformie UPEL:

- nazwa: Algorytmy grafowe 2023/2024 - projekty
- hasło: galgo2023

Projekt zostanie oceniony w następujący sposób:

- osoby z najlepszymi programami (10 najszybszych) otrzymają +1.5 do oceny końcowej
- osoby z dobrymi programami otrzymują +1.0 do oceny końcowej
- osoby z programami niezbyt dobrymi (rozwiązującymi mało testów) otrzymują +0.5 do oceny końcowej
- osoby, które nie nadeślą programu lub ich program nie będzie działał otrzymują +0.0 do oceny końcowej

Termin nadsyłania rozwiązań: **15 lutego 2024, 23:59**

Treść zadania

Książe Bitkacy nie szczędzi czasu i zawartości królewskiego skarbca na swoją pasję - archeologię. Tym razem niemałym kosztem udało mu się wejść w posiadanie ekscytującego znaleziska.

W ręce księcia trafił papirus stanowiący relację z ostatnich dni życia Al-Gorima, rabusia grobowców ze starożytnego orientalnego państwa Grafaara. Al-Gorim postanowił zbadać miejsce spoczynku wysokiej rangi oficjela. Grafaarskie grobowce zorganizowane były jako zestaw komnat połączonych korytarzami, z jednym wejściem. Ze względów religijnych, od wejścia do każdej komnaty prowadziła dokładnie jedna droga. W pewnym momencie eksploracji Al-Gorim stracił poczucie orientacji, i postanowił od tej pory oznaczać swoją drogę, a także sporządzać notatki opisujące trasę, którą się przemieszcza. Za każdym razem, gdy wchodził w nowy, niezbadany jeszcze korytarz, zapisywał na papirusie znak `+`. Gdy szedł korytarzem, który już wcześnej przeszedł w kierunku przeciwnym do pierwszego przejścia (tj. gdy się nim cofał), oznaczał to jako `^`. Gdy natomiast ponownie wkraczał w zbadany już korytarz w kierunku zgodnym z pierwszym przejściem, zapisywał liczbę oznaczającą ile innych korytarzy wychodzących z danej komnaty zbadał, zanim pierwszy raz wszedł do tego (w momencie gdy rozpoczął swoje zapiski, nie pamiętał z którego korytarza przyszedł do obecnej komnaty, ale jako że jakiś korytarz prowadzący do tej komnaty musiał już przejść, przyjął za ilość korytarzy już zbadanych

1). Niestety, ostatnie zapiski sugerują, że mimo starannych notatek nie udało się znaleźć wyjścia z grobowca, nim zabrakło sił i zapasów wody.

Książe Bitkacy zapragnął odnaleźć miejsce wiecznego spoczynku Al-Gorima i podążając jego śladami zwiedzić opisany w papirusie grobowiec. Problem w tym, że na przestrzeni kilku tysiącleci, pod piaskami pustyń Grafaary tego typu grobowców powstało wiele. Dzięki pracy archeologów i nowoczesnym technikom obrazowania, dysponujemy dokładnymi informacjami dotyczącymi rozkładu pomieszczeń we wszystkich odnalezionych grobowcach. Twoim zadaniem jest odpowiedzieć na pytanie, które z nich dopuszczają trasę opisaną w papirusie.

Dane wejściowe

Do zaimplementowanej funkcji przekazane zostaną następujące argumenty:

- *N* - ilość komnat w grobowcu
- numer komnaty zawierającej wyjście na zewnątrz
- lista korytarzy, zawierająca krotki postaci *(a, b)*, gdzie *a*, *b* to numery komnat
- string zawierający oddzielone pojedynczymi spacjami notatki opisujące drogę

Komnaty są indeksowane od *1*, tj. mają indeksy *1*, *2*, ..., *N*.

Przykładowe wywołanie może wyglądać następująco:

```
solve(6, 1, [(1, 2), (2, 3), (2, 4), (1, 5), (5, 6)], "+ ^ + ^ +")
```

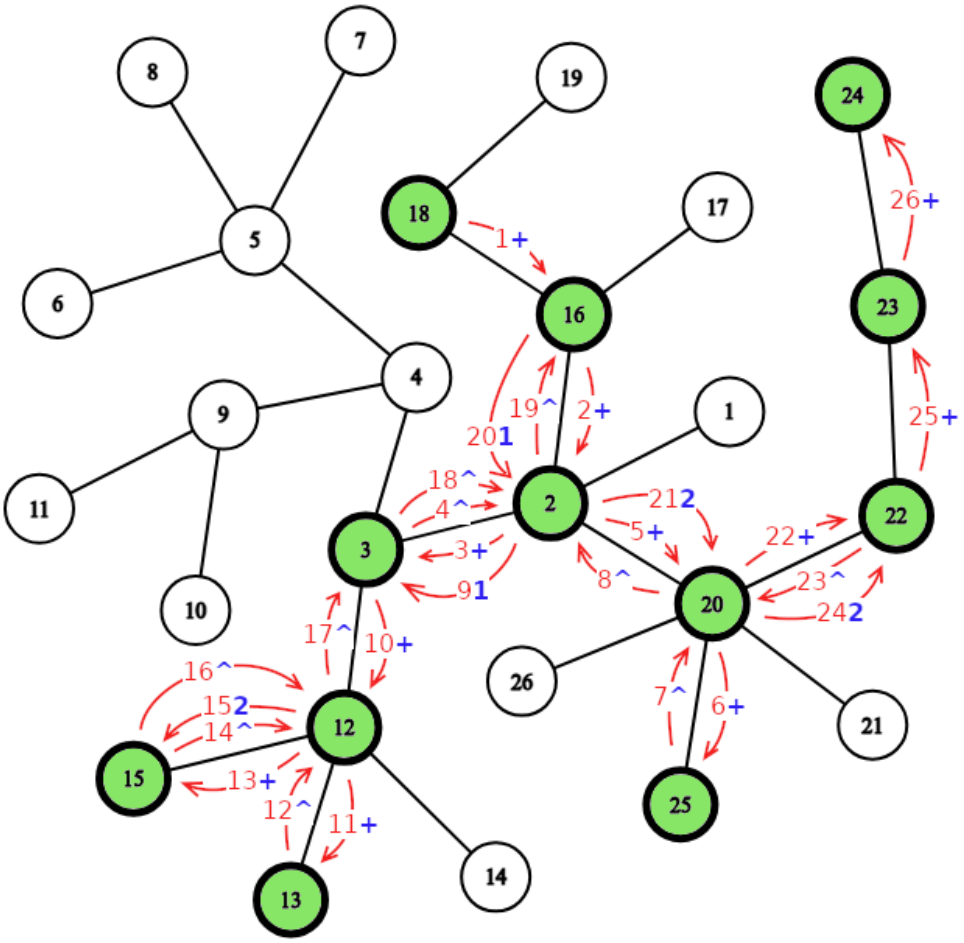
Funkcja *solve* powinna zwracać wartość logiczną `True` lub `False`, w zależności od tego, czy opisana ostatnim argumentem trasa jest możliwa do zrealizowania w grobowcu opisanym pierwszymi trzema argumentami.

Dla powyższych danych nie jest to możliwe - trasa musi rozpoczynać się w komnacie z której wychodzą co najmniej 3 korytarze, ale jedyna taka komnata w tym grobowcu ma dokładnie 3 korytarze, z których jeden prowadzi do komnaty wejściowej, co stoi w sprzeczności z tym, że Al-Gorim nie zdołał wydostać się z grobowca.

Przykład

Rozważmy następujący grobowiec z *1* jako komnatą wejściową oraz trasę opisaną następującym ciągiem:

```
+ + + ^ + + ^ ^ 1 + + ^ + ^ 2 ^ ^ ^ ^ 1 2 + ^ 2 + +
```



Dla takich danych grobowiec pasuje istotnie do opisanej trasy. Przykładowa jej realizacja zaznaczona jest ponumerowanymi strzałkami na powyższym obrazku.

Instrukcja

Infrastruktura do projektu dostępna jest w formie archiwum z plikami źródłowymi w języku Python (link na dole). Szkielet rozwiązania znajduje się w pliku *example.py* - importuje on funkcję `runtests` z modułu `data` i uruchamia ją, podając swoją funkcję rozwiązującą jako argument. Przesyłane rozwiązania powinny mieć postać analogicznego pliku. Przetestować rozwiązanie można uruchamiając ów plik, np.

```
python3 example.py
```

Na wyjście standardowe wypisane zostaną informacje o rezultatach poszczególnych testów, a także podsumowanie z ilością testów zakończonych sukcesem i przybliżonym łącznym czasie obliczeń.

Warunki techniczne

- Program powinien być napisany w języku Python i działać z wersją 3.12.1

- Program nie może wykorzystywać zewnętrznych bibliotek (biblioteka standardowa jest dopuszczalna)

Pliki

- project2.zip