algograf

Projekt 2: Szpiedzy

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 2022/2023 - projekty
- hasło: galgo2223

Projekt zostanie oceniony w następujący sposób:

- osoby z najlepszymi programami (15 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: **14 lutego 2023, 23:59**

Treść zadania

Jakkolwiek od niefortunnej, przerwanej pełnym wzajemnej nieufności i niechęci rozejmem wojny z Księstwem Qubicji minęło już parę lat, relacje pozostają napięte, a konflikt nie tyle się zakończył, co przybrał jedynie nieco bardziej subtelną formę. Jako zwierzchnik królewskiego wywiadu, Twoim zadaniem jest bronić Króla Bajtycjana i jego poddanych przed niewidzialnymi zagrożeniami ze strony agentów Qubicji. W tym celu zneutralizować należy odkrytą przed paroma tygodniami siatkę szpiegowską.

Jakkolwiek dysponujemy pełnymi informacjami odnośnie szpiegów oraz kanałów informacyjnych pomiędzy nimi, zamiast pozbyć się ich wszystkich, podjęta została decyzja o przeniknięciu w ich szeregi, by w razie możliwości zdobyć cenne informacje o wrogim księstwie. W tym celu konieczne będzie wyeliminowanie części wrogich agentów i zastąpienie ich własnymi. Każdy agent może porozumiewać się bezpośrednio z pewnym podzbiorem pozostałych agentów (relacja ta jest symetryczna). By uzyskać maksymalną ilość informacji, chcemy by z każdej pary bezpośrednio porozumiewających się ze sobą wrogich agentów co najmniej jeden został wymieniony na sojusznika. Jako że operacja jest delikatna i wymaga dobrej koordynacji, do jej sukcesu wymagane jest, by wszyscy podstawieni agenci Qubicji mogli komunikować się ze sobą bezpośrednio za pomocą istniejących kanałów. Ponadto, by nie budzić czujności wroga, chcemy spełnić obydwa warunki podmieniając możliwie jak najmniejszą liczbę agentów.

Dane wejściowe

Do zaimplementowanej funkcji przekazane zostaną następujące argumenty:

- *N* - ilość szpiegów

- lista zawierająca pary szpiegów, którzy mogą się ze sobą bezpośrednio kontaktować Szpiedzy są indeksowani od *1*, tj. mają indeksy *1*, *2*, ..., *N*

Przykładowe wywołanie może wyglądać następująco:

```
solve[10, [
    (1,2), (1, 3), (1, 4), (1, 5),
    (2, 3), (2, 4), (2, 5),
    (3, 4), (3, 5), (4, 5),
    (6, 1), (6, 2), (6, 5),
    (7, 2), (8, 2), (8, 3),
    (9, 4), (10, 2), (10, 3)
])
```

Takie wywołanie oznacza, że mamy 10 szpiegów i zadane w liście kanały komunikacyjne pomiędzy nimi. Zaimplementowana funkcja powinna zwracać liczbę całkowitą - minimalną ilość szpiegów, których należy wymienić aby mieli dostęp do wszystkich kanałów komunikacyjnych i mogli wszyscy komunikować się ze sobą bezpośrednio. Dla powyższego wywołania powinna zwrócić 5, jako że *1, 2, 3, 4, 5* to najmniejszy podzbiór szpiegów o takiej własności.

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 3.8.10
- Program nie może wykorzystywać zewnętrznych bibliotek (biblioteka standardowa jest dopuszczalna)

Pliki

- project2.zip