

### ***Franchise Warung Ikan Magetan***



(Source: [Generated with DALL-E 3](#))

#### **Deskripsi**

Setelah mengadakan kompetisi tim yang *viral* ke seluruh penjuru tanah air, Sofita kini sudah memiliki beberapa pemegang hak *franchise* untuk bisnisnya “*Warung Ikan Magetan*”. Namun, rekan-rekan bisnis Sofita tersebar ke seluruh Indonesia dan Sofita wajib memantau setiap *franchise*-nya secara rutin untuk memastikan usahanya berjalan dengan lancar. Oleh karena itu, Sofita kali ini membeli rumah-rumah di berbagai kota di Indonesia untuk mengawasi “*Warung Ikan Magetan*” yang dipegang rekan-rekan bisnisnya.

Kota-kota yang ada di Indonesia akan memiliki **id unik** sebagai penanda setiap kota tersebut, **dimulai dari id 1**. Sofita memiliki beberapa rumah yang tersebar di beberapa kota. Kota-kota yang ada tersebut dihubungkan oleh berbagai jalan yang memastikan **seluruh kotanya terhubung** satu dengan yang lainnya. Setiap jalan pastinya memiliki jarak masing-masing. Sofita ingin membuat pengalaman berpindah kotanya menjadi efisien dan efektif karena Sofita akan banyak berpindah-pindah kota sehingga dia butuh bantuan agar tidak terlalu lelah saat mencapai rumahnya yang ada di kota lain. Oleh karena itu, Sofita meminta tolong untuk membuat program yang menjawab beberapa hal berikut:

### 1. R [ENERGI]

Sofita hanya bisa menempuh satu perjalanan antar kota dengan maksimal <ENERGI>. Setiap 1 ENERGI memungkinkan dia menempuh jarak sejauh 1 satuan. Sofita akan mengisi penuh <ENERGI> kembali ketika ia sudah mencapai suatu kota. **Cetak jumlah kota maksimum yang dapat dikunjungi oleh Sofita.** Jika Sofita tidak bisa mengunjungi kota manapun dari lokasinya saat ini, maka **cetak -1**. Kota yang ditempati Sofita di awal **tidak termasuk** dalam jumlah kota yang dapat dikunjungi.

### 2. F [TUJUAN]

Sofita sudah tahu rekan bisnis mana yang harus dikunjungi terlebih dahulu. Sofita ingin mengunjungi rekan bisnis yang berada di kota dengan id < TUJUAN > secara langsung. Namun, Sofita harus mengetahui jaraknya terlebih dahulu agar ia bisa menyiapkan bahan bakar yang cukup sehingga tidak kehabisan di tengah jalan. Oleh karena itu, **cetak jarak terpendek yang dapat ditempuh Sofita dari posisi kotanya saat ini ke kota dengan id < TUJUAN >**.

### 3. M [ID] [PASSWORD]

Sofita berpindah ke salah satu rumahnya di kota dengan id < ID >. Setelah ia berpindah, ia membutuhkan cara termudah untuk membuka *password* rumahnya yang super ketat. Ia ingin membuka rumahnya yang dikunci dengan kata sandi berupa bilangan *PASSWORD*. Pada awalnya, Sofita mengunci rumahnya dengan *password* inisial 0000. Sofita harus mencari kombinasi angka dari sistem yang sudah ada untuk menghasilkan *password* <PASSWORD>. Aturan pembuatan pembentukan *password* adalah sebagai berikut:

- Setiap digit dari angka pertama dijumlahkan dengan digit angka kedua, lalu hasilnya diambil digit terakhir saja.

Contoh: Jika *password* inisial adalah 8932 dan angka yang digunakan adalah 3174, maka

$$8 + 3 = 11 \rightarrow \text{ambil } 1$$

$$9 + 1 = 10 \rightarrow \text{ambil } 0$$

$$3 + 7 = 10 \rightarrow \text{ambil } 0$$

$$2 + 4 = 6 \rightarrow \text{ambil } 6$$

$$\text{Sehingga } 8932 + 3174 = 1006$$

- Angka yang digunakan oleh Sofita untuk membentuk *password* dapat digunakan berulang kali.

Jika Sofita berhasil mencari kombinasi untuk menghasilkan <PASSWORD> dengan angka yang sudah ada di sistem, **cetak jumlah minimum kombinasi cara yang dapat dilakukan untuk membuka kunci rumahnya**. Selain itu, jika Sofita berhasil membuka *password*-nya, maka kunci inisial akan berubah menjadi <PASSWORD>. Apabila *password* rumahnya tidak bisa dibuka, kunci inisial akan tetap sama seperti sebelumnya dan **cetak -1**. Meskipun Sofita tidak dapat membuka *password* rumahnya, dia **tetap berhasil berpindah**.

#### 4. J [ID]

Sofita berandai-andai apakah dia bisa menemukan jarak terpendek yang menyambungkan seluruh kota yang ada di Indonesia. Oleh karena itu, Sofita memulai imajinasinya dengan menghubungkan seluruh kota dimulai dari kota dengan id *ID*. Namun, Sofita ingin **seluruh jalan yang terhubung langsung dengan kota tersebut juga diperhitungkan** dalam perhitungan jarak. Maka dari itu, **cetak total jarak terpendek yang menyambungkan seluruh kota di Indonesia dengan memperhitungkan seluruh jalan yang terhubung langsung dengan kota *ID***.

#### Format Masukan

- Baris pertama berisi dua bilangan, yaitu *V* dan *E*, yang dipisahkan dengan sebuah spasi. *V* menyatakan jumlah kota yang ada, sedangkan *E* menyatakan jumlah seluruh jalan yang ada.
- *V* baris selanjutnya berisi 3 bilangan, yaitu  $V_i$ ,  $V_j$ ,  $L_i$ , di mana ketiga bilangan tersebut dipisahkan oleh spasi di antara bilangan.  $V_i$  dan  $V_j$  adalah dua id kota yang tersambung oleh jalan sepanjang  $L_i$ .
- Baris selanjutnya berisi bilangan *P* yang merupakan banyaknya angka yang bisa digunakan Sofita untuk membuat *password*.
- *P* baris selanjutnya berisi bilangan  $P_i$  yang merupakan angka yang bisa digunakan oleh Sofita untuk membentuk *password*.
- Baris selanjutnya berisi bilangan *Q* yang merupakan jumlah banyaknya aktivitas yang akan dilakukan.
- *Q* baris selanjutnya berisi aktivitas yang sesuai dengan format aktivitas yang telah dijabarkan sebelumnya.

#### Format Keluaran

- Keluaran dari perintah R
  - Keluarkan **jumlah kota maksimum yang dapat dikunjungi** oleh Sofita.
  - Keluarkan **-1** jika tidak ada kota yang dapat dikunjungi dengan <ENERGI>.
- Keluaran dari perintah F
  - Keluarkan **nilai jarak terpendek yang dapat ditempuh** Sofita dari posisi kotanya saat ini ke kota dengan id <TUJUAN>.
- Keluaran dari perintah M

- Keluarkan jumlah minimum kombinasi cara yang dapat dilakukan untuk menghasilkan nilai <PASSWORD>.
  - Keluarkan -1 jika tidak ada kombinasi angka yang dapat menghasilkan nilai <PASSWORD>.
- Keluaran dari perintah J
  - Keluarkan total jarak terpendek yang menyambungkan seluruh kota dari kota <ID> dengan memperhitungkan seluruh jalan yang terhubung langsung dengan kota <ID>.

#### Batasan

- $2 \leq V \leq 5000$
- $V \leq E \leq \min(10\,000, V(V-1)/2)$

#### Batasan Kueri Aktivitas

- Banyak Query R dijamin tidak melebihi 1000
- Banyak Query F dijamin tidak melebihi 1000
- Banyak Query J dijamin tidak melebihi 100
- Banyak Query M dijamin tidak melebihi 100
- Query M maksimal hanya terdiri dari 5 ID Unik.

#### Contoh Masukan 1

```

6 7
1 2 3
1 3 5
2 3 2
2 4 4
3 5 6
4 5 1
5 6 2
4
1234
5678
4321
8765
5
R 5
F 6
M 4 2468
J 5

```

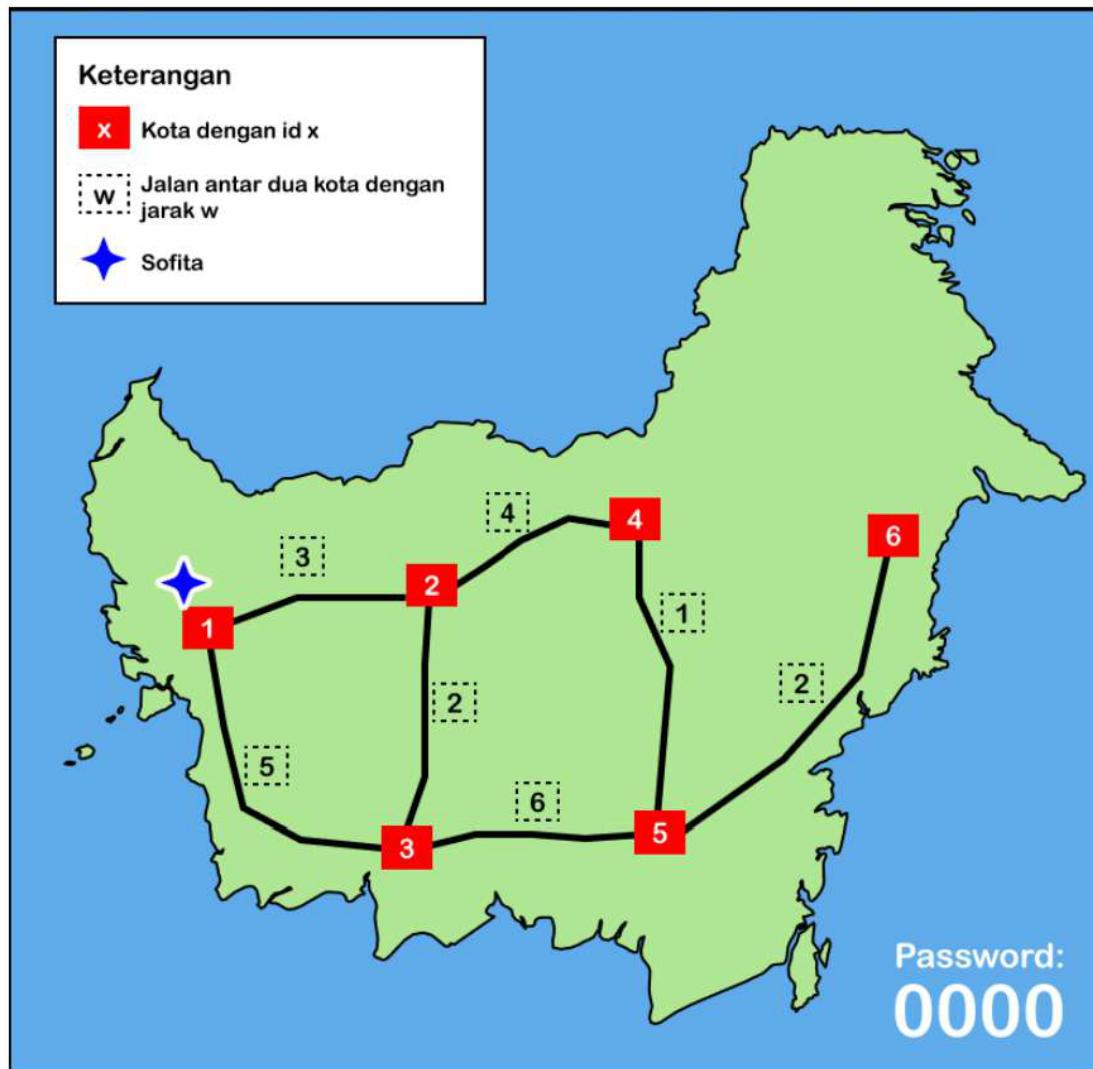
M 2 4826

**Contoh Keluaran 1**

5  
10  
2  
14  
2

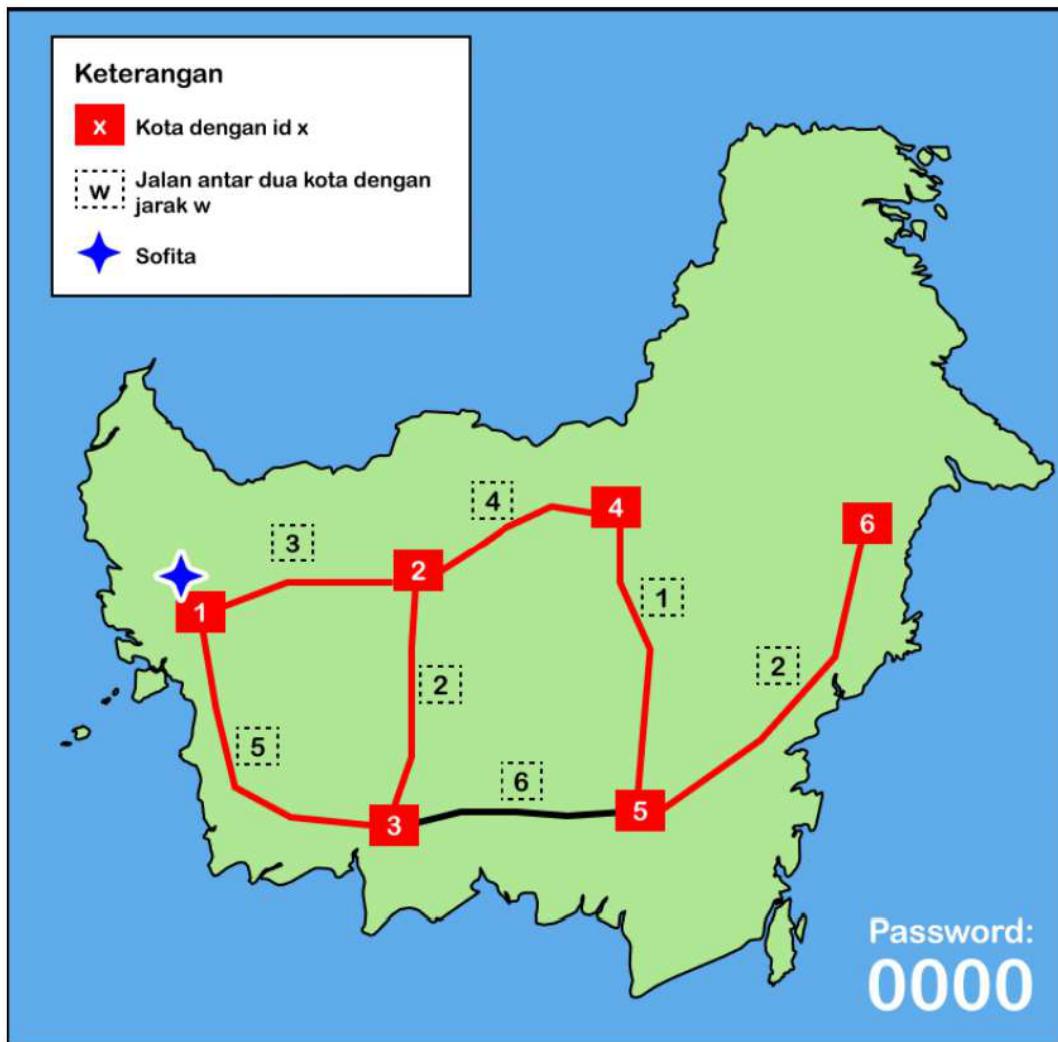
## Penjelasan Contoh 1

### 1. Kondisi awal



Sofita berada di rumahnya yang terletak di kota dengan id 1 dan dengan *password* inisial 0000.

## 2. R 5



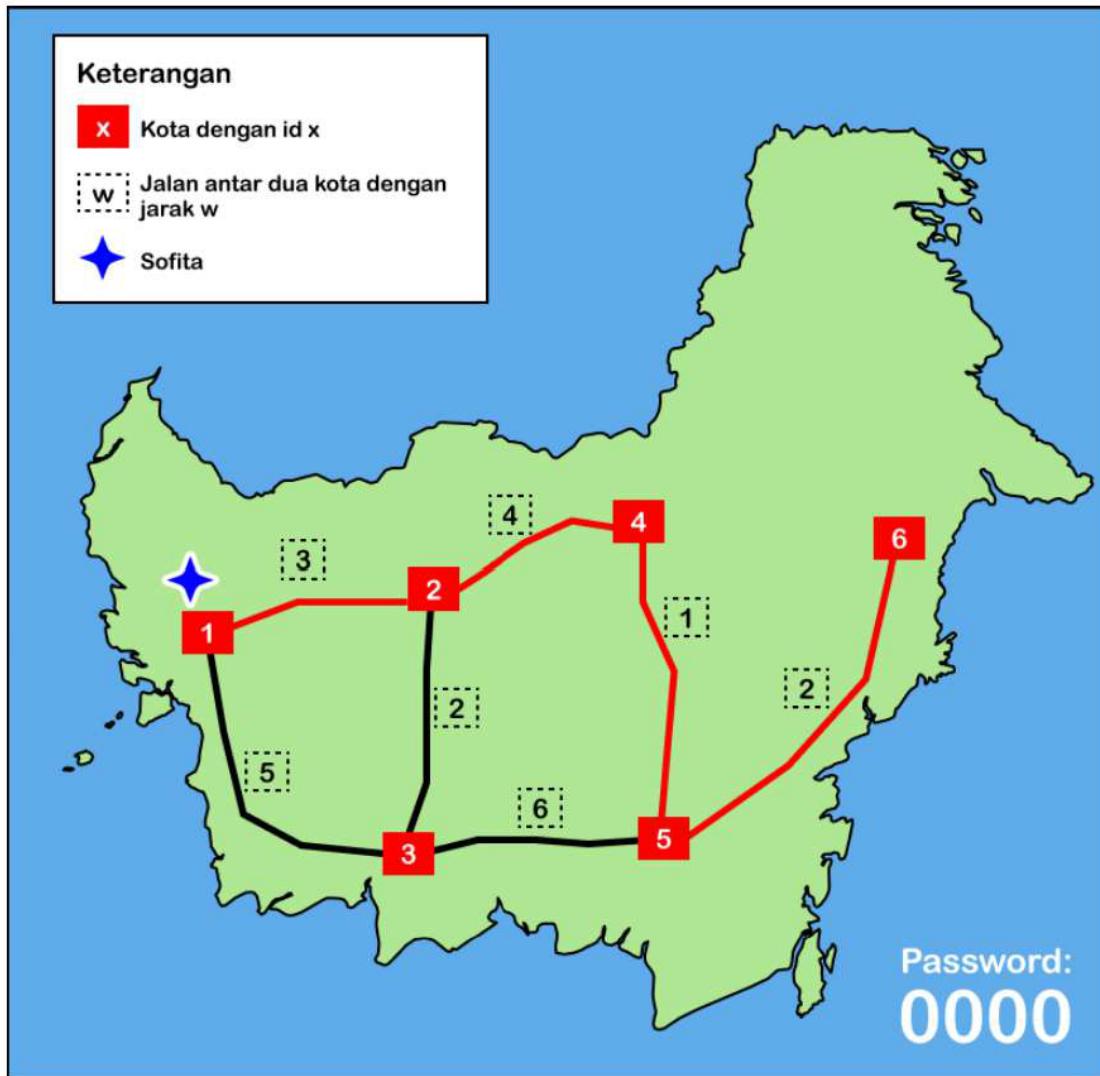
Sofita memiliki energi sebanyak 5. Berikut adalah langkah-langkah kunjungannya ke berbagai kota:

- **Kota ID 2:** Sofita dapat mengunjungi kota ini karena jaraknya dari kota ID 1 kurang dari energi yang dimiliki ( $3 < 5$ ).
- **Kota ID 3:** Sofita dapat mengunjungi kota ini karena jaraknya dari kota ID 1 sama dengan energi yang dimiliki ( $5 == 5$ ).
- **Kota ID 4:** Sofita dapat mengunjungi kota ini dengan langkah berikut:
  - Sofita pergi ke kota ID 2 ( $3 < 5$ ) dan mengisi kembali energi menjadi 5.
  - Dari kota ID 2, Sofita melanjutkan ke kota ID 4 karena jaraknya kurang dari energi yang dimiliki ( $4 < 5$ ).
- **Kota ID 5:** Sofita dapat mengunjungi kota ini dengan langkah berikut:
  - Sofita pergi ke kota ID 2 ( $3 < 5$ ) dan mengisi kembali energi menjadi 5.
  - Dari kota ID 2, Sofita pergi ke kota ID 4 ( $4 < 5$ ) dan mengisi kembali energi menjadi 5.
  - Melanjutkan ke kota ID 5 karena jaraknya kurang dari energi yang dimiliki ( $1 < 5$ ).
- **Kota ID 6:** Sofita dapat mengunjungi kota ini dengan langkah berikut:
  - Sofita pergi ke kota ID 2 ( $3 < 5$ ) dan mengisi kembali energi menjadi 5.

- Dari kota ID 2, Sofita pergi ke kota ID 4 ( $4 < 5$ ) dan mengisi kembali energi menjadi 5.
- Dari kota ID 3, Sofita pergi ke kota ID 5 ( $1 < 5$ ) dan mengisi kembali energi menjadi 5.
- Dari kota ID 5, Melanjutkan ke kota ID 6 karena jaraknya kurang dari energi yang dimiliki ( $2 < 5$ ).

Kesimpulan: Dengan energi sejumlah 5, Sofita dapat mengunjungi total 5 kota.

### 3. F 6

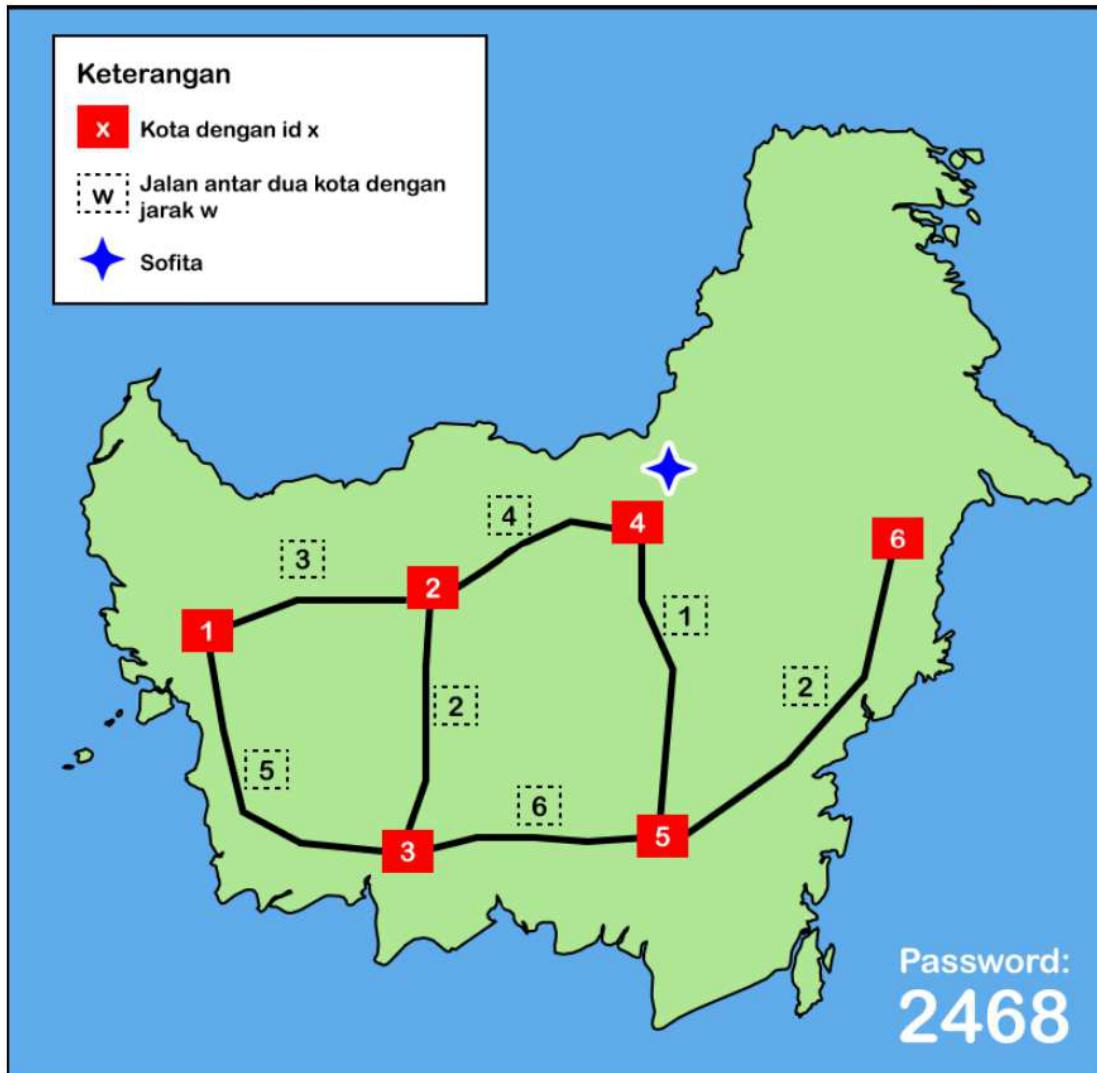


Sofita mulai dari kota ID 1

- ID 1 = 0
- ID 1 - ID 2 = 0 + 3 = 3
- ID 1 - ID 2 - ID 4 = 0 + 3 + 4 = 7
- ID 1 - ID 2 - ID 4 - ID 5 = 0 + 3 + 4 + 1 = 8
- ID 1 - ID 2 - ID 4 - ID 5 - ID 6 = 0 + 3 + 4 + 1 + 2 = 10

Jarak minimum dari kota ID 1 ke kota ID 6 adalah 10

4. M 4 2468



Sofita berpindah ke kota ID 4 dan berusaha memecahkan password rumahnya yang merupakan **2468**. Terdapat 4 kombinasi angka yang tersedia: 1234, 4321, 5678, dan 8765.

Pada langkah pertama, Sofita menggunakan kombinasi angka 1234.

Prosesnya:

- $0 + 1 = 1 \rightarrow$  ambil 1
  - $0 + 2 = 2 \rightarrow$  ambil 2
  - $0 + 3 = 3 \rightarrow$  ambil 3
  - $0 + 4 = 4 \rightarrow$  ambil 4
  - Sehingga  $0000 + 1234 = 1234$

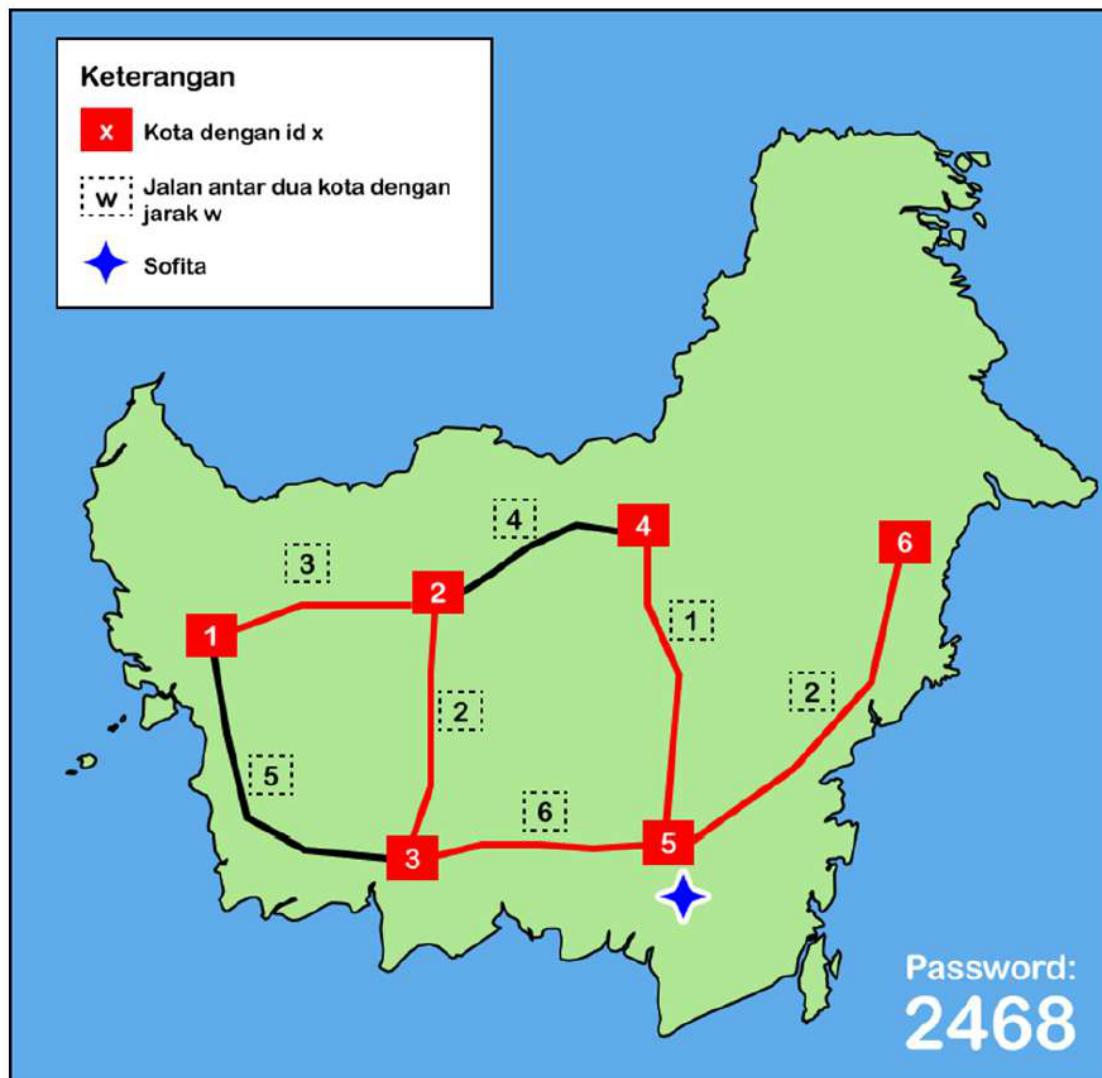
Pada langkah kedua, Sofita kembali menggunakan kombinasi angka yang sama, yaitu 1234.

Prosesnya:

- $1 + 1 = 2 \rightarrow$  ambil 2
  - $2 + 2 = 4 \rightarrow$  ambil 3
  - $3 + 3 = 6 \rightarrow$  ambil 6
  - $4 + 4 = 8 \rightarrow$  ambil 8
  - Sehingga  $1234 + 1234 = 2468$

Dengan demikian, Sofita berhasil memecahkan password rumahnya dalam **2** langkah.

5. J5

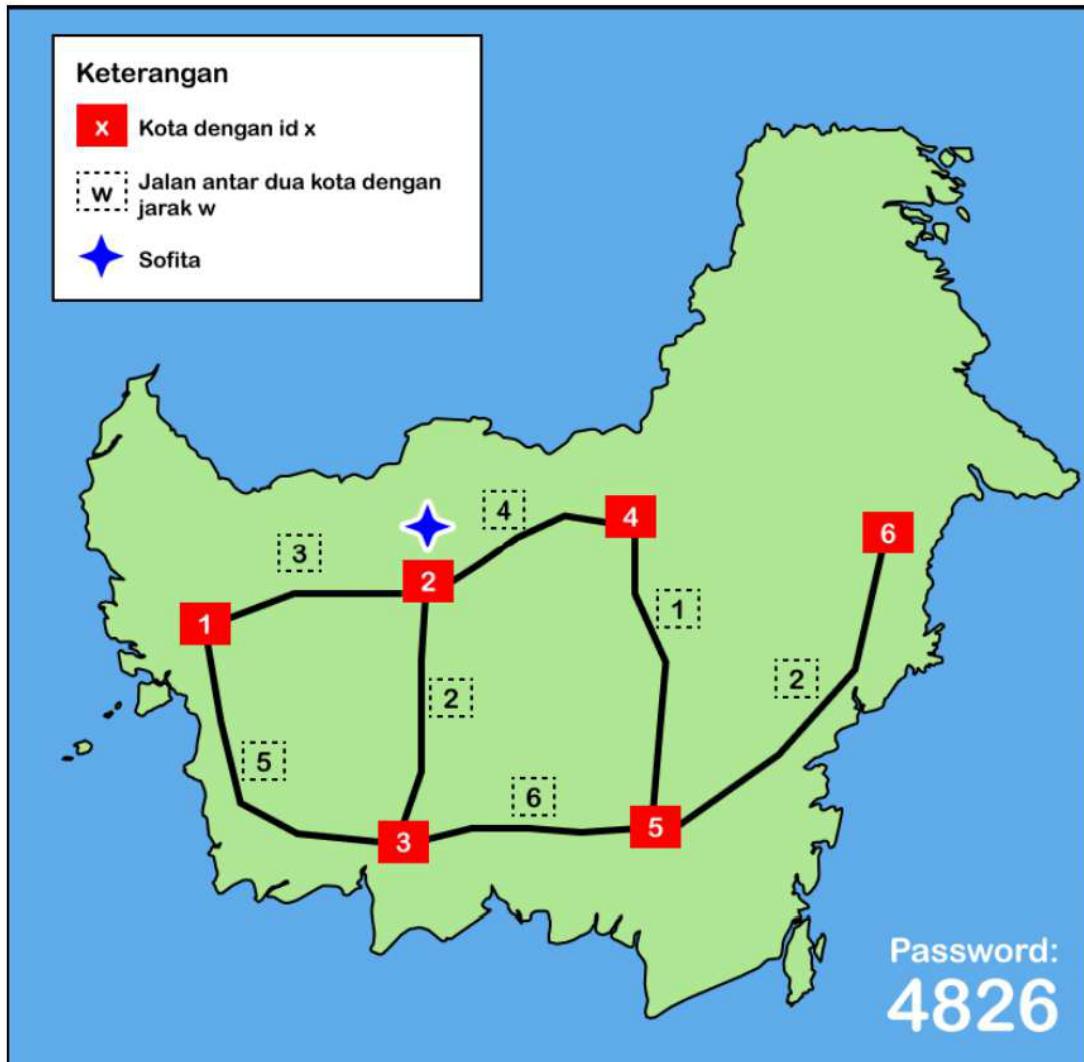


Sofita mulai dari kota ID 5, untuk mengunjungi seluruh kota dengan jarak terendah

- Kota ID 5 = 0
- Kota ID 4 = 1
- Kota ID 6 = 2
- Kota ID 3 = 6
- Kota ID 2 = Melewati kota ID 3 = 2
- Kota ID 1 = Melewati kota ID 3 dan melewati kota ID 2 = 3

Sehingga total jarak terpendek untuk ke semua kota adalah 14

## 6. M 2 4826



Sofita berpindah ke kota ID 2 dan berusaha memecahkan password rumahnya yang merupakan **4826**. Sebelumnya Sofita mengunci dengan *password* **2468**. Terdapat 4 kombinasi angka yang tersedia: 1234, 4321, 5678, dan 8765. Pada langkah pertama, Sofita menggunakan kombinasi angka 1234.

Prosesnya:

- $2 + 1 = 3 \rightarrow$  ambil 3
- $4 + 2 = 6 \rightarrow$  ambil 6
- $6 + 3 = 9 \rightarrow$  ambil 9
- $8 + 4 = 12 \rightarrow$  ambil 2
- Sehingga  $2468 + 1234 = 3692$

Pada langkah kedua, Sofita kembali menggunakan kombinasi angka yang sama, yaitu 1234.

Prosesnya:

- $3 + 1 = 4 \rightarrow$  ambil 4
- $6 + 2 = 8 \rightarrow$  ambil 8
- $9 + 3 = 12 \rightarrow$  ambil 2
- $2 + 4 = 6 \rightarrow$  ambil 6
- Sehingga  $3692 + 1234 = 4826$

Dengan demikian, Sofita berhasil memecahkan password rumahnya dalam **2** langkah.

**Contoh Masukan 2**

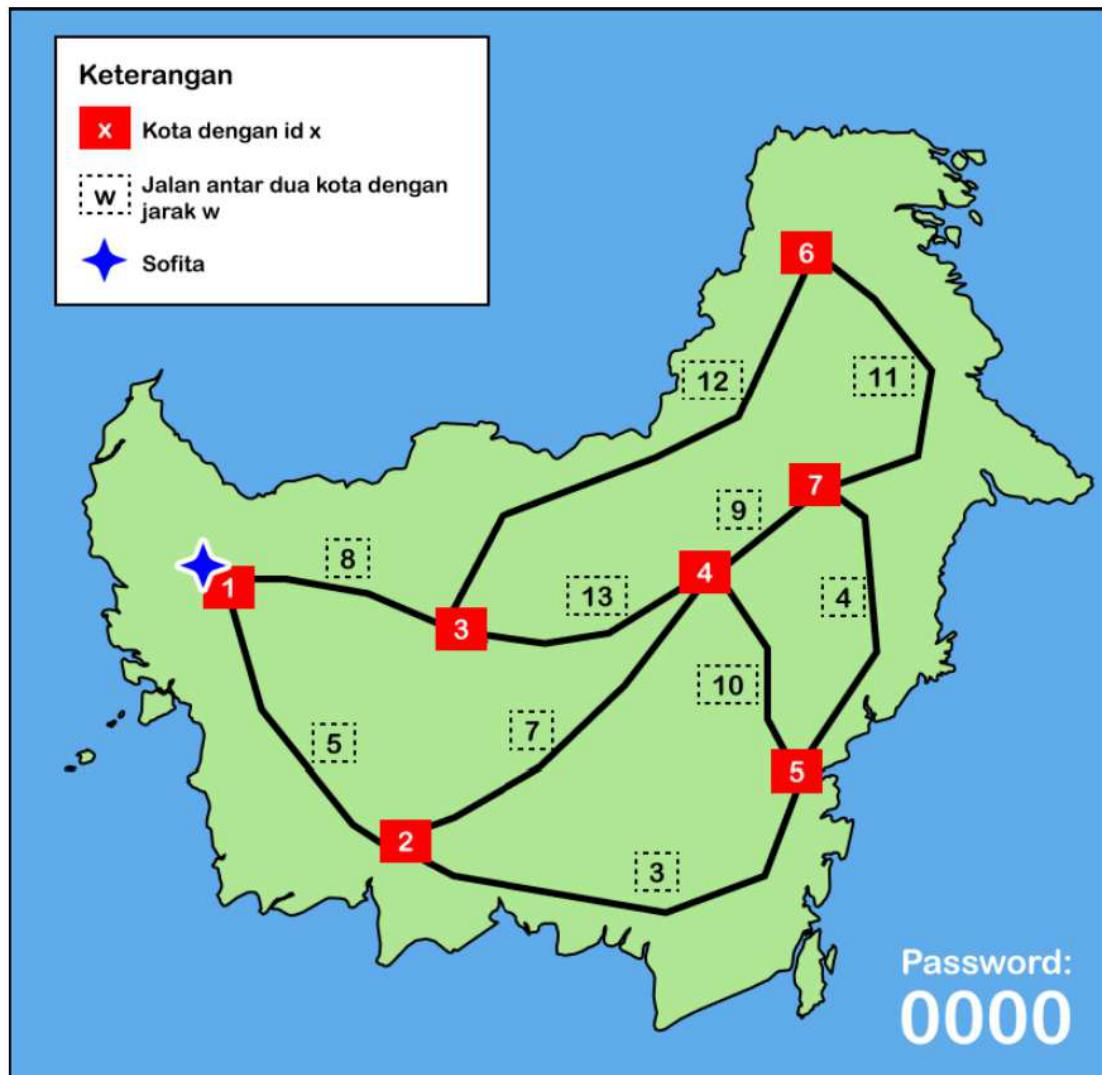
```
7 10
1 2 5
1 3 8
2 4 7
2 5 3
3 4 13
3 6 12
4 5 10
4 7 9
5 7 4
6 7 11
4
1111
8082
4444
1808
8
R 5
M 2 2456
M 3 4444
R 10
M 6 7777
R 10
M 3 6557
J 4
```

**Contoh Keluaran 2**

```
3
-1
1
5
3
-1
2
55
```

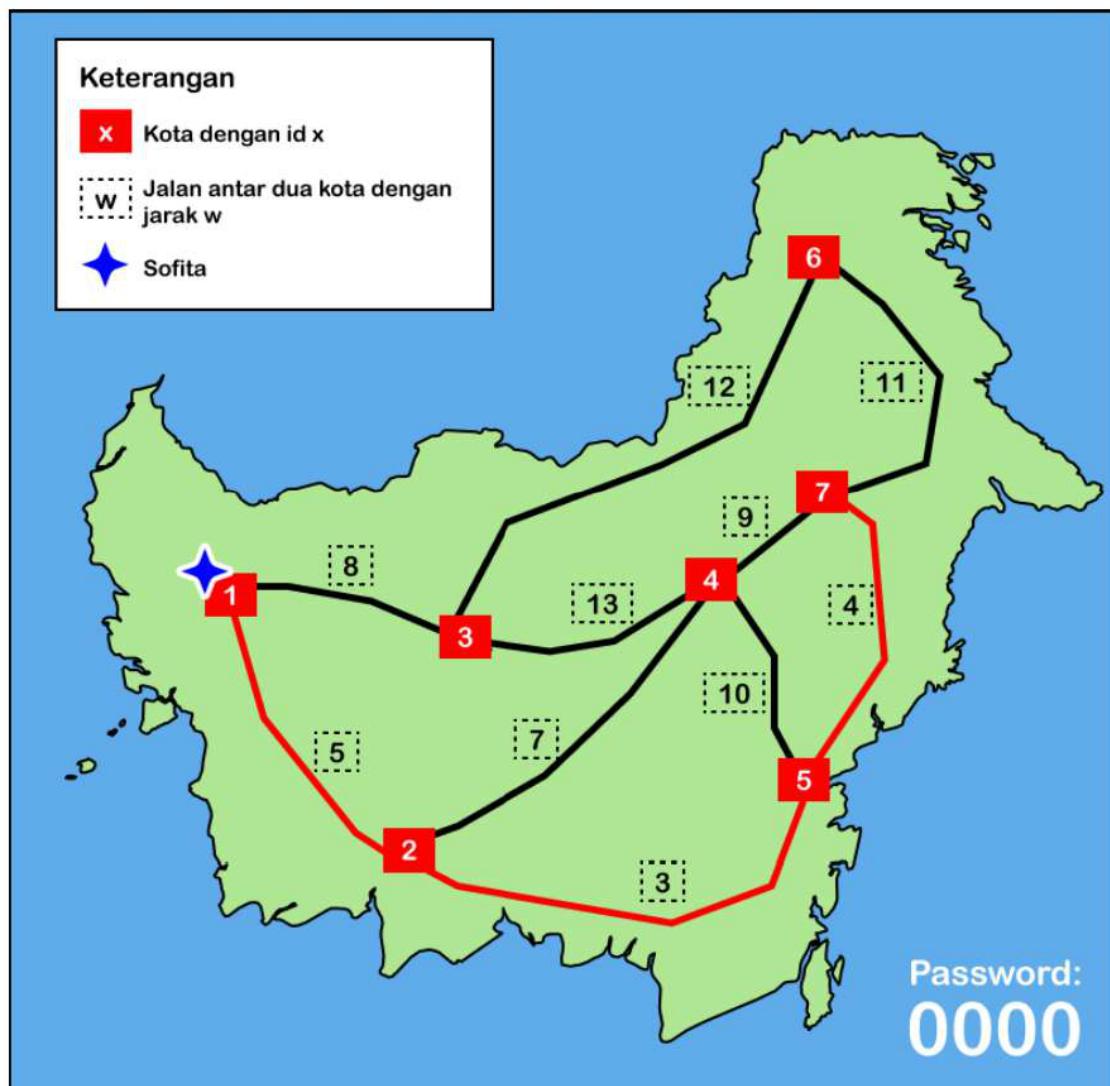
## Penjelasan Contoh 2

### 1. Kondisi awal



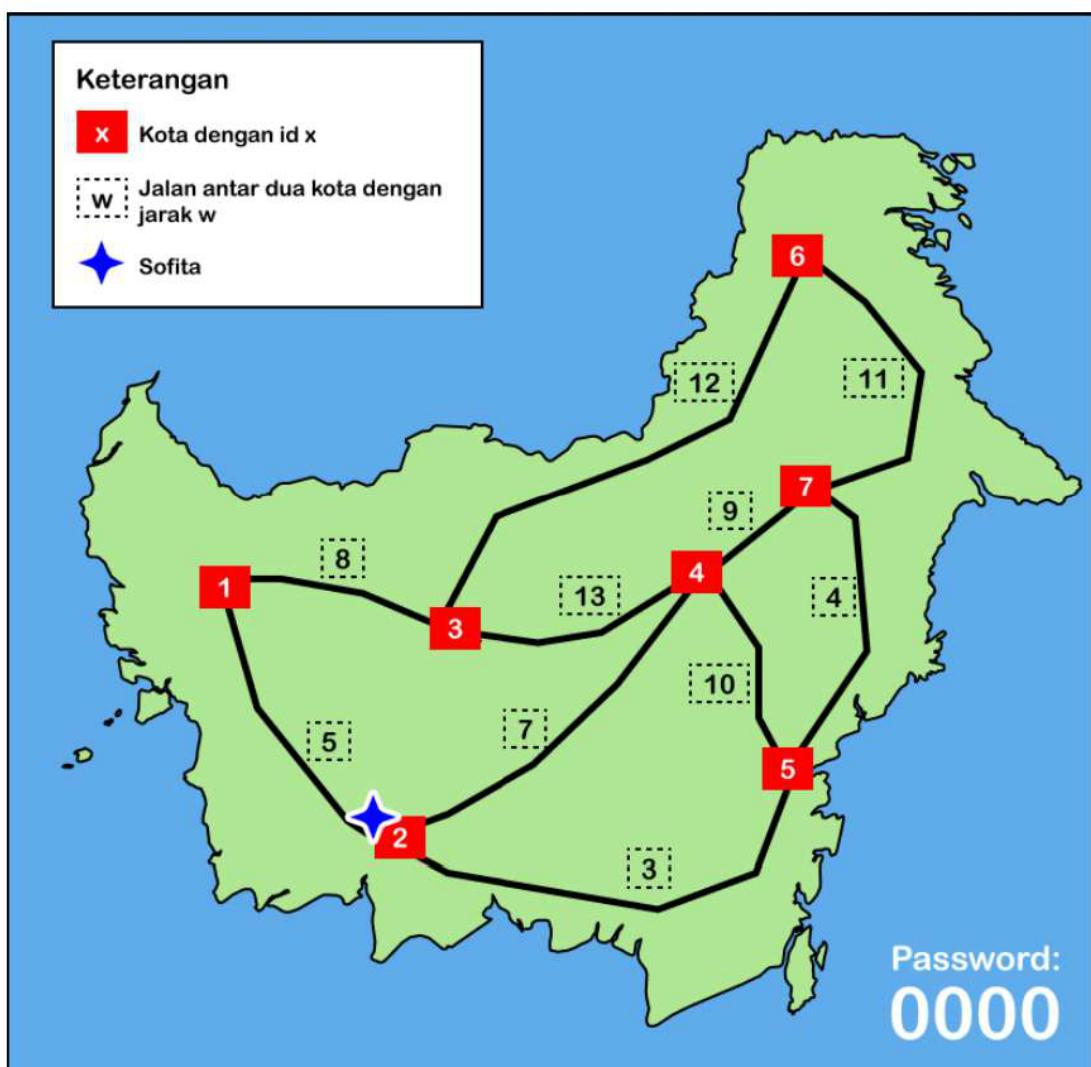
Sofita berada di rumahnya yang terletak di kota dengan id 1 dan dengan *password* inisial 0000.

## 2. R 5



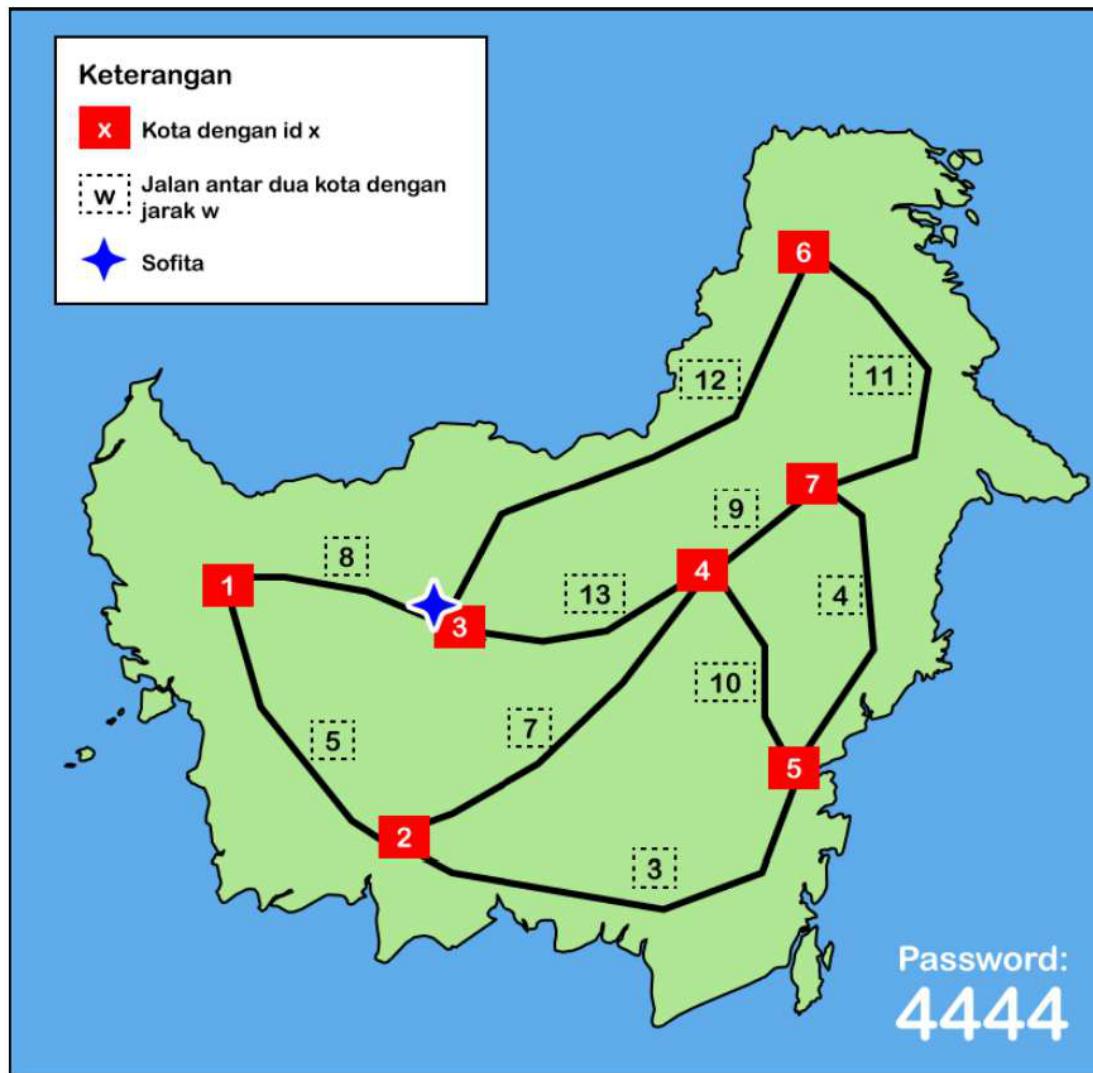
Sofita menempuh perjalanan antar kota dengan maksimal 5 ENERGI dari kota id 1. Setiap Sofita mencapai suatu kota, Sofita mengisi penuh ENERGI menjadi 5 kembali. Dengan ini, ia dapat mengunjungi kota id 2, 5 dan 7, sehingga output kueri ini adalah 3.

### 3. M 2 2456



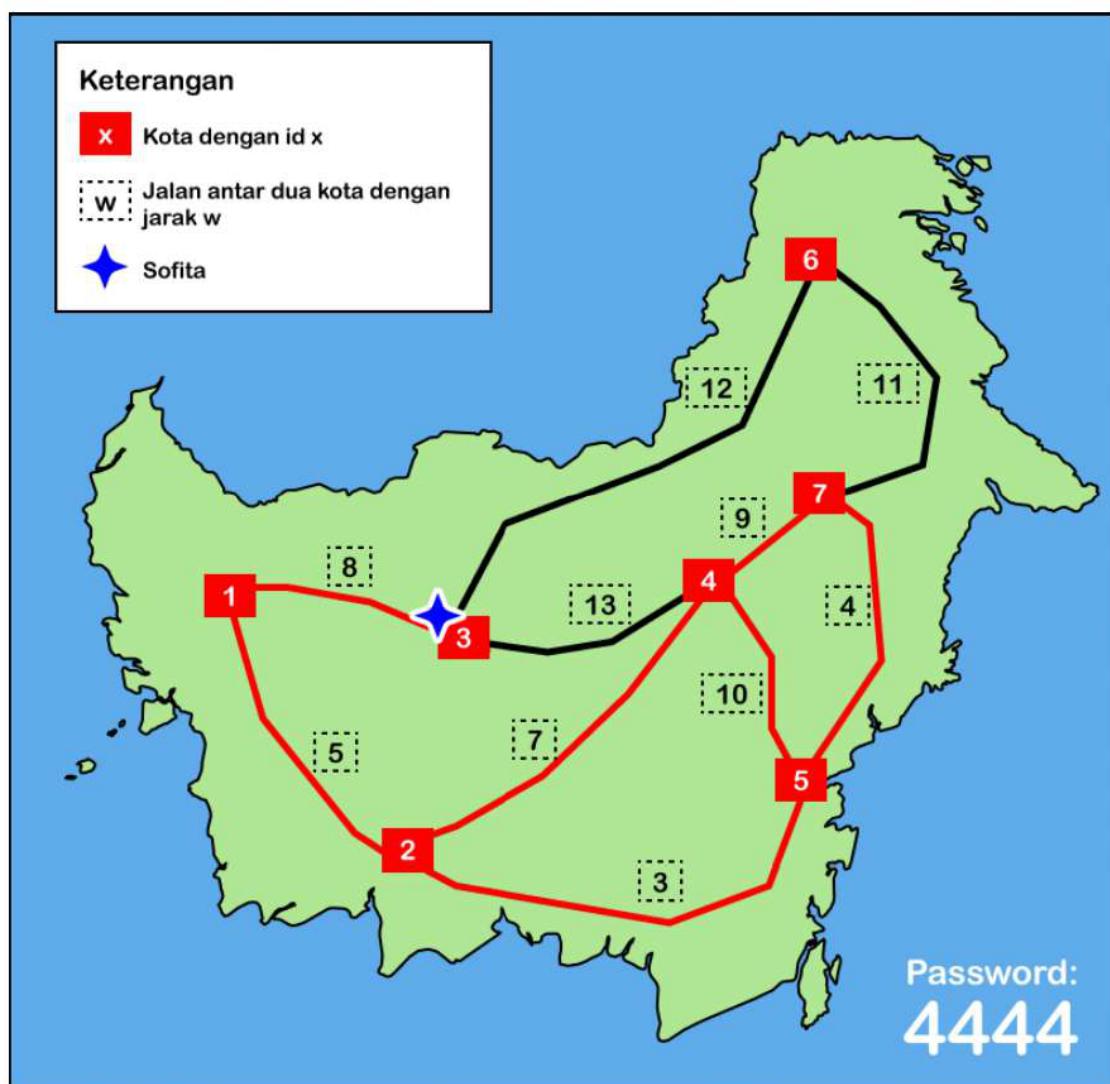
Sofita ingin berpindah ke salah satu rumahnya di kota dengan id 2. Ia membutuhkan kombinasi *password* 2456. Sebelumnya, Sofita mengunci dengan password 0000. Tidak ada kombinasi yang memungkinkan dari 1111, 8082, 4444, dan 1808 yang dapat digunakan Sofita untuk mendapatkan 2456 dari 0000, sehingga output kueri ini adalah -1. Meskipun begitu, **Sofita masih berpindah ke rumahnya di kota dengan id 2.**

#### 4. M 3 4444



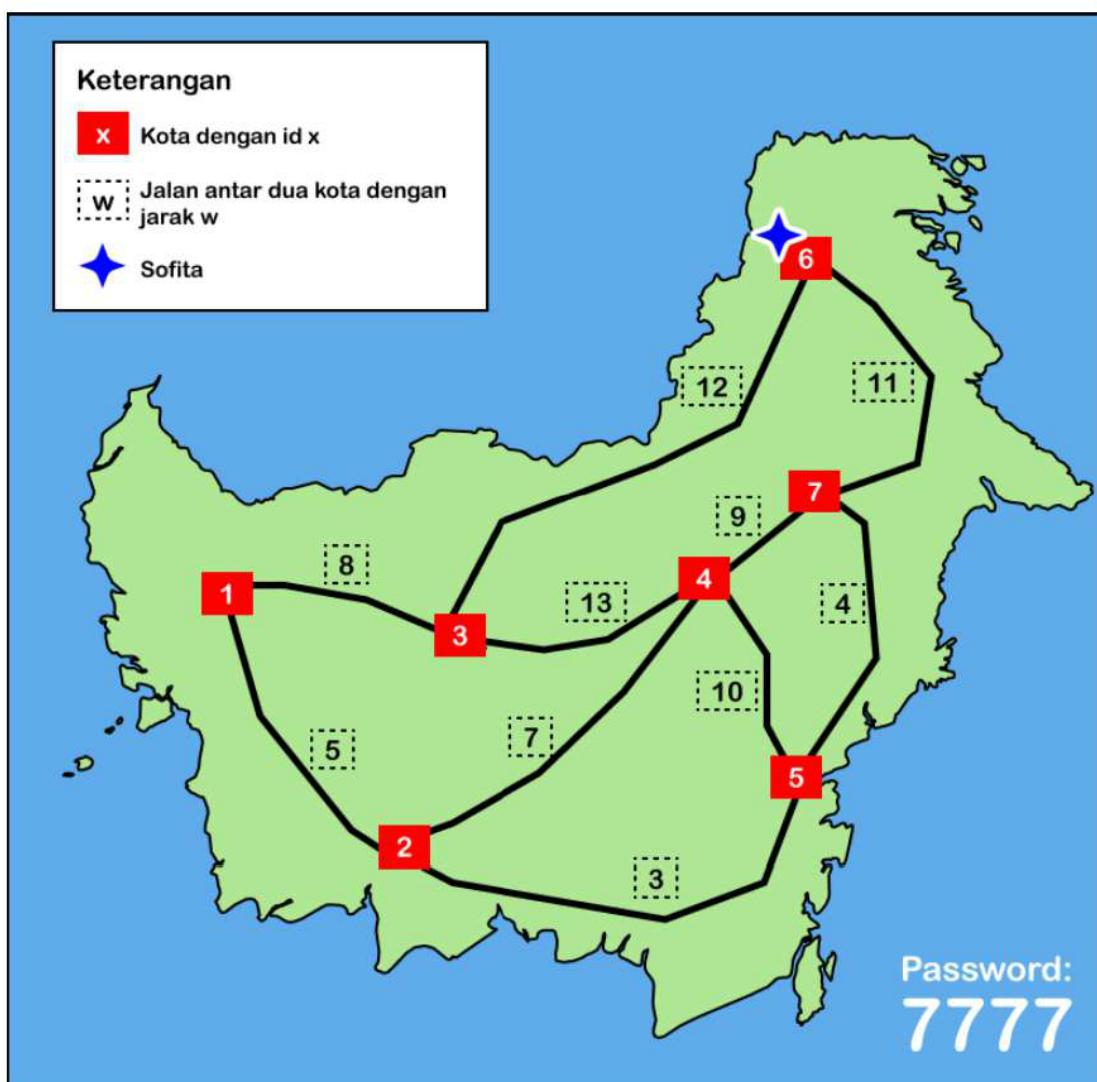
Sofita ingin berpindah ke salah satu rumahnya di kota dengan id 3. Ia membutuhkan kombinasi *password* 4444. Sebelumnya, Sofita mengunci dengan password 0000. Terdapat kombinasi yang memungkinkan, yaitu  $0000 + 4444 = 4444$ , sehingga output kueri ini adalah 1 dan Sofita berpindah ke rumahnya di kota dengan id 3, serta *password* berubah menjadi 4444.

5. R 10



Sofita menempuh perjalanan antar kota dengan maksimal 10 ENERGI dari kota id 3. Setiap Sofita mencapai suatu kota, Sofita mengisi penuh ENERGI menjadi 10 kembali. Dengan ini, ia dapat mengunjungi kota id 1, 2, 4, 5, dan 7, sehingga output kueri ini adalah 5.

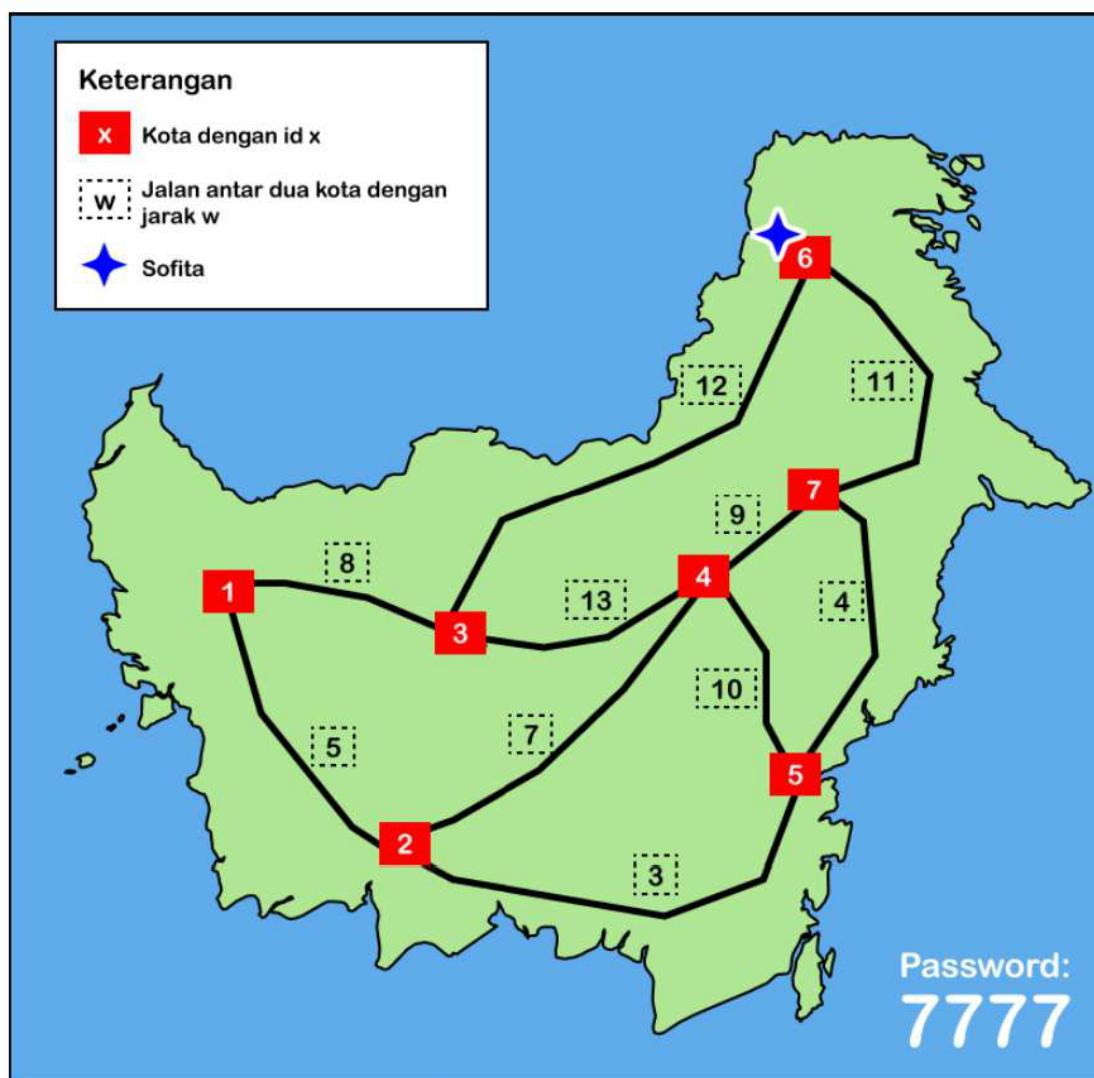
## 6. M 6 7777



Sofita ingin berpindah ke salah satu rumahnya di kota dengan id 6. Ia membutuhkan kombinasi *password* 7777. Sebelumnya, Sofita mengunci dengan password 4444.

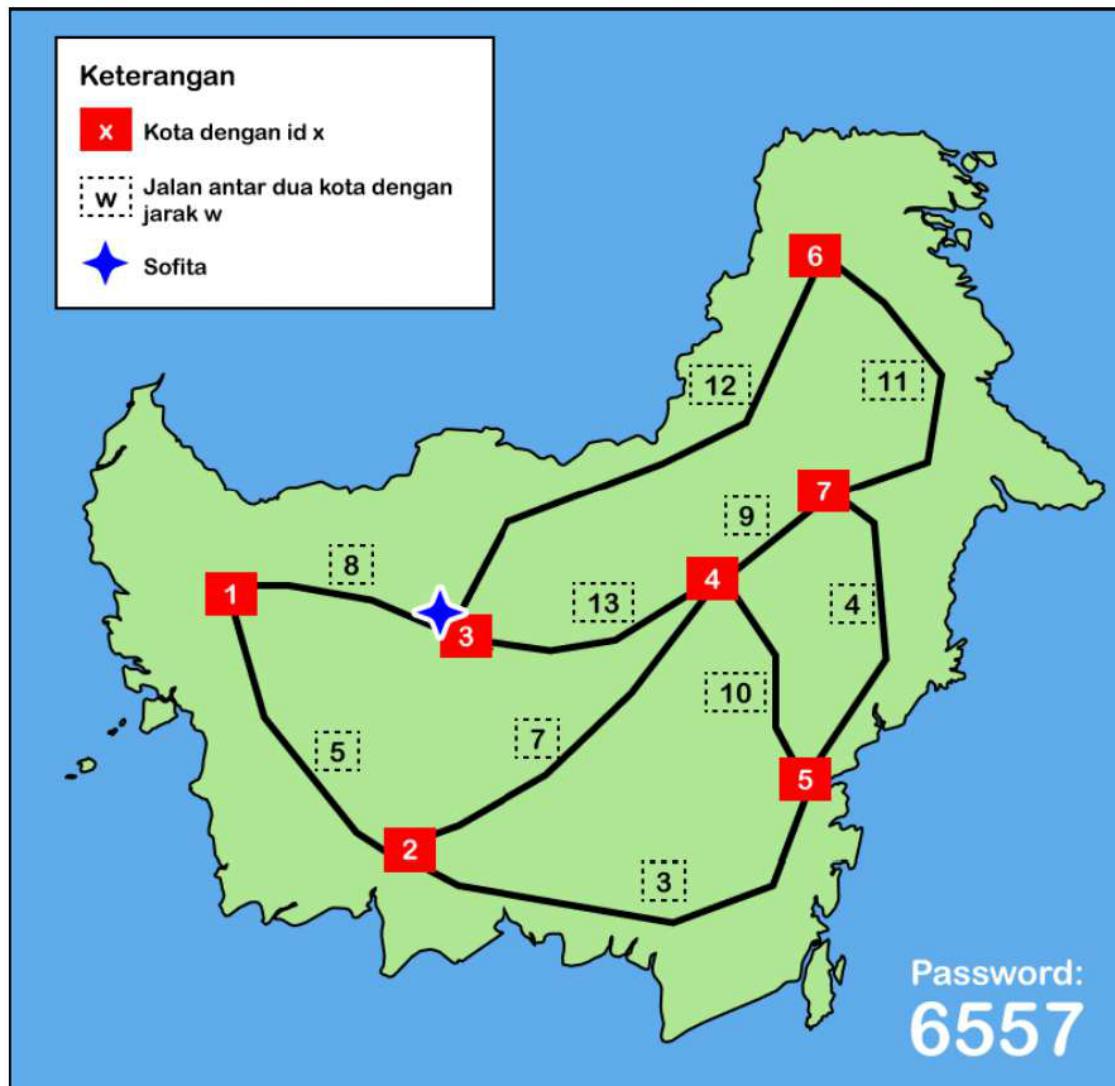
Terdapat kombinasi yang memungkinkan, yaitu  $4444 + 1111 + 1111 + 1111 = 7777$ , sehingga output kueri ini adalah 3 dan Sofita berpindah ke rumahnya di kota dengan id 6, serta *password* berubah menjadi 7777.

7. R 10



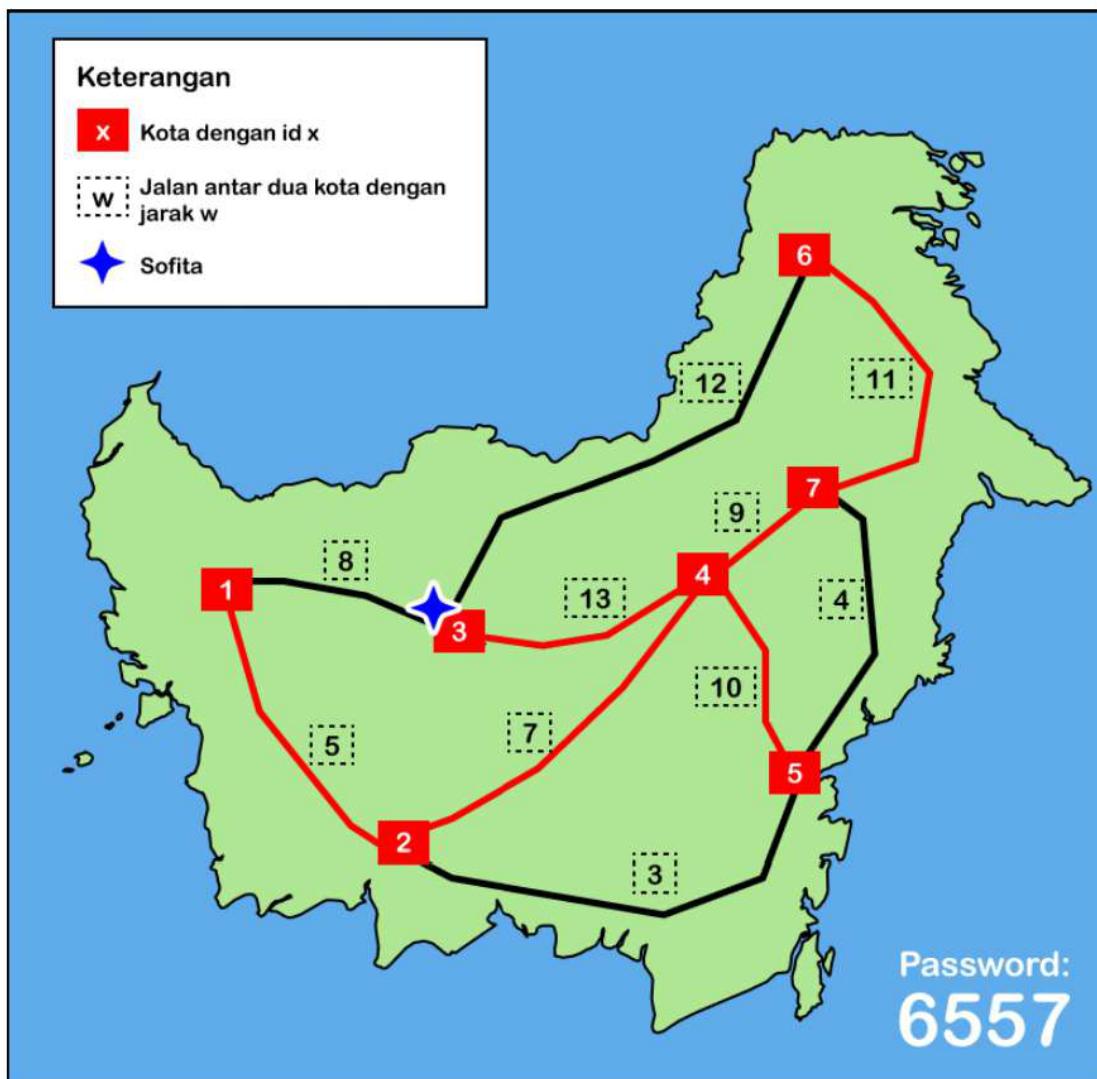
Sofita menempuh perjalanan antar kota dengan maksimal 10 ENERGI dari kota id 6. Tidak ada kota yang dapat dikunjungi dengan ENERGI 10, sehingga output kueri ini adalah -1.

8. M 3 6557



Sofita ingin berpindah ke salah satu rumahnya di kota dengan id 3. Ia membutuhkan kombinasi *password* 8575. Sebelumnya, Sofita mengunci dengan password 7777. Terdapat kombinasi yang memungkinkan, yaitu  $7777 + 8082 + 1808 = 6557$ , sehingga output kueri ini adalah 2 dan Sofita berpindah ke rumahnya di kota dengan id 3, serta *password* berubah menjadi 6557.

9. J 4



Sofita berandai-andai apakah dia bisa menemukan jarak terpendek yang menyambungkan seluruh kota yang ada di Indonesia dimulai dari kota dengan id 4 dengan seluruh jalan yang terhubung langsung dengan kota tersebut juga diperhitungkan dalam perhitungan jarak.

Jalan terpendek yang diandaikan Sofita:

- Kota id 4 ke kota id 2 dengan jarak 7
- Kota id 4 ke kota id 3 dengan jarak 13
- Kota id 4 ke kota id 5 dengan jarak 10
- Kota id 4 ke kota id 7 dengan jarak 9
- Kota id 1 ke kota id 2 dengan jarak 5
- Kota id 6 ke kota id 7 dengan jarak 11

Total jarak tersebut adalah 55. Maka dari itu, output kueri ini adalah 55.

### Informasi Tambahan Test-case

Testcase	Query
R	0 - 10
J	15 - 30
F	30 - 40
R, M	35 - 50
F, M	50 - 65
F, M, J	65 - 90
R, F, M, J	90 - 100

### Peraturan Tambahan

Sesuai dengan scope materi, solusi yang Anda gunakan wajib mengimplementasikan:

- **Heap dan Priority Queue**

Apabila menggunakan heap dan priority queue, mahasiswa wajib mengimplementasikan heap sendiri tanpa *built-in function*.

**Note:** Konsekuensi terhadap pelanggaran peraturan keterangan tambahan akan dikenakan sanksi poin akhir 0