

# Софтуниада 2023

## Задача 10. Високотехнологично състезаване

Голям фен си на състезанията с коли, от дълго време обмисляш да си напишеш игра за състезаване с коли, но старата формула където всичките коли се състезават на 1 писта е доста изтъркана, затова решаваши да си направиш твоя високотехнологична версия на състезаване между 2 коли, където всяка ще си има неин маршрут.

Дава ни се карта с **N** на брой градове номерирани от **0** до **N-1** и **M** пътища между тях, като всеки път има някаква дължина свързана с него. Задачата ни е да намерим **най-кратките 2 непокриващи се маршрута** между **началният и крайният град**.

- **Началният град** винаги ще е града номериран с **0**, а **крайният град** винаги ще е града номериран с **N - 1**.
- Считаме 2 маршрута за **непокриващи се**, ако **няма път който да е част и от 2та маршрута** (2 непокриващи се маршрута могат обаче да включват един и същи град).
- Дефинираме 2 непокриващи се маршрута за **най-кратки** ако:
  - Покриват горната дефиниция за **непокриващи се**.
  - Сумата от дължините на **маршрутите е най-малката възможна**, за непокриващи се маршрути.
    - Ако съществуват повече от един избори за **най-кратки 2 непокриващи се маршрути**, винаги избираме първо маршрутите които посещават по ниско номерираните градове първо. Примерно маршрут **0 -> 31 -> 76 -> 25** би дошъл преди **0 -> 32 -> 25**, защото **31 < 32**.

### Вход

На първия ред от конзолата ще получим числото **N** – **броят на градовете**, цяло число в диапазона [**3...5000**]

На вторият ред от конзолата ще получим числото **M** – **броят на пътищата**, цяло число в диапазона [**3...100 000**]

- На всеки от следващите **M** реда, ще получим информация за даден път, във формата:
 

**{начален град} {краен град} {разстояние}**

  - Разстоянието ще е цяло число в диапазона [**1...100 000**]
  - Пътищата винаги ще бъдат подадени подредени в нарастващ ред по начален град и след това в нарастващ ред по краен град (виж примерите)

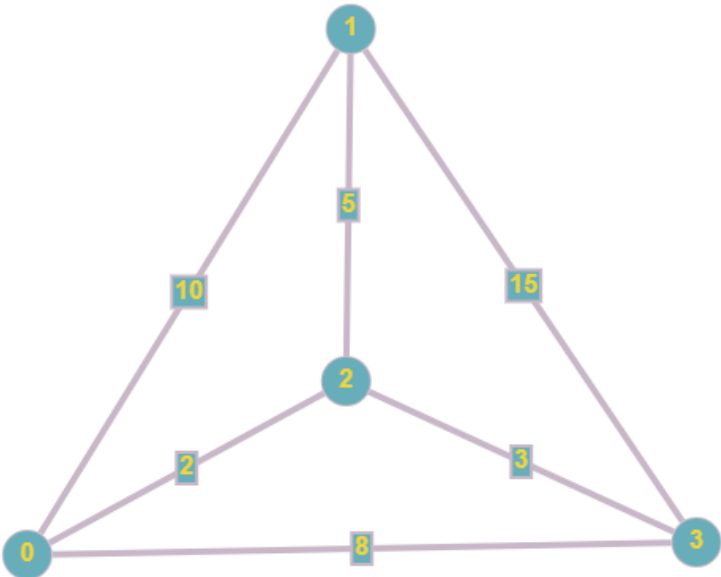
### Изход

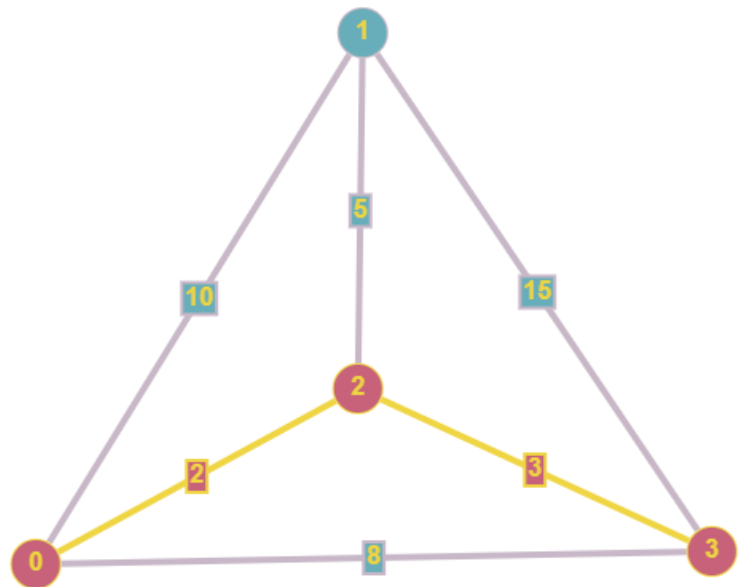
Трябва да изпечатате точно **2 реда** като изход – **2та най-кратки непокриващи се маршрута**, като на **първия ред** трябва да изпечатаме този от 2та който **посещава по-**

ниско номерираните градове първо (виж логиката при повече от един избори за най-кратки 2 непокриващи се маршрута).

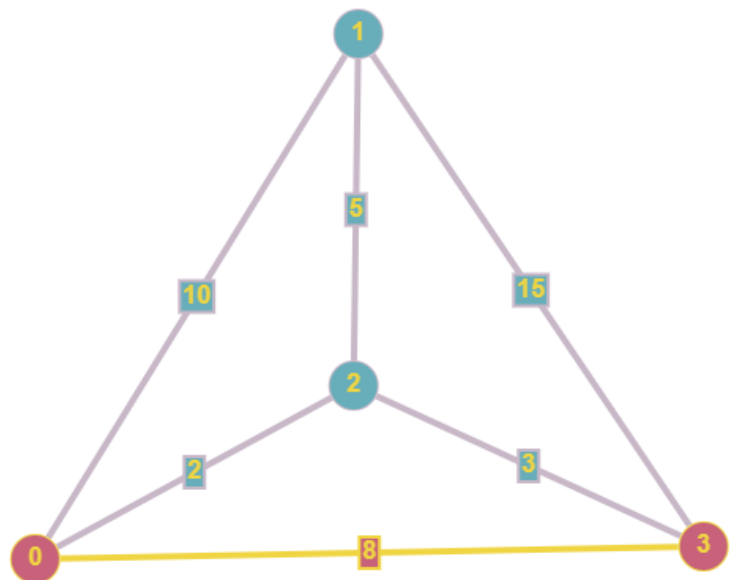
- Всеки маршрут трябва да се изпечата във формата:  
 {начален град} -> {2ри град} -> {3ти град}...{краен град}

## Примерен вход и изход

Вход	Изход	Коментар
4 5 0 1 10 0 2 2 0 3 8 1 2 5 2 3 3	0 -> 2 -> 3 0 -> 3	 <p>От картинката може да видим, че най краткият маршрут е 0 -&gt; 2 -&gt; 3</p>



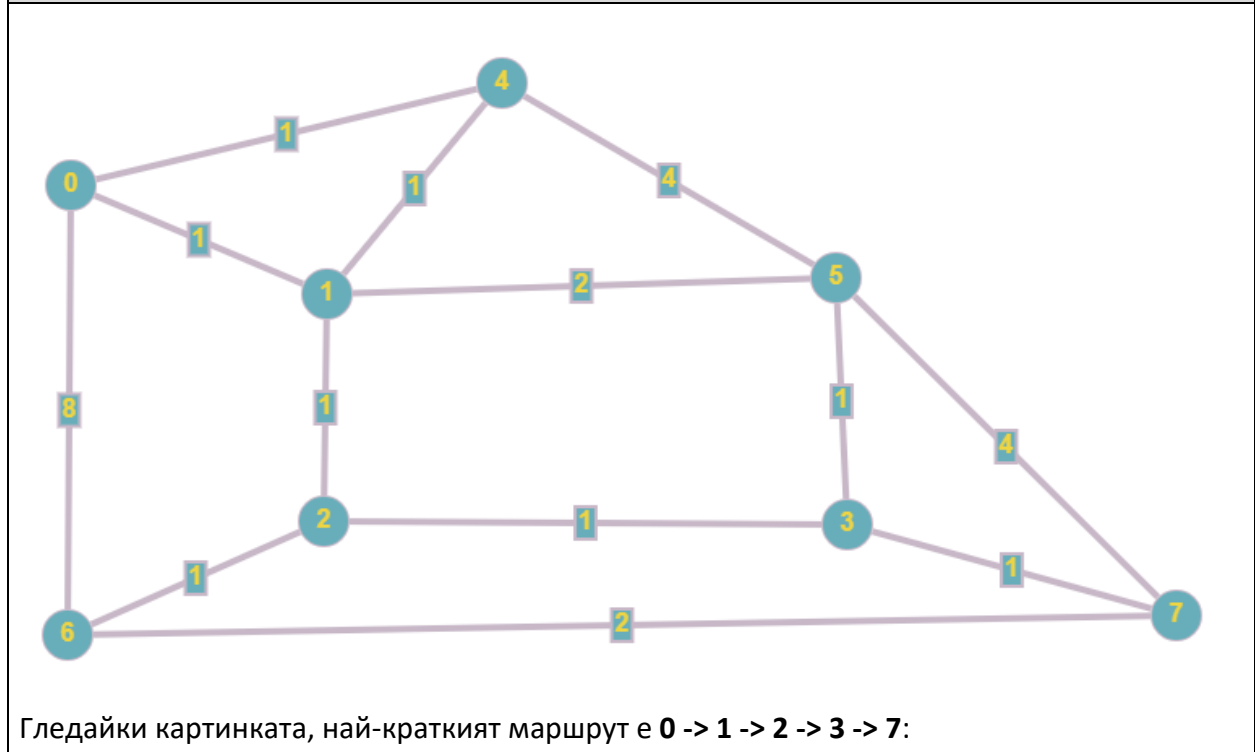
но понеже не можем да използваме същите пътища, 2-рият най-кратък маршрут е **0 -> 3**

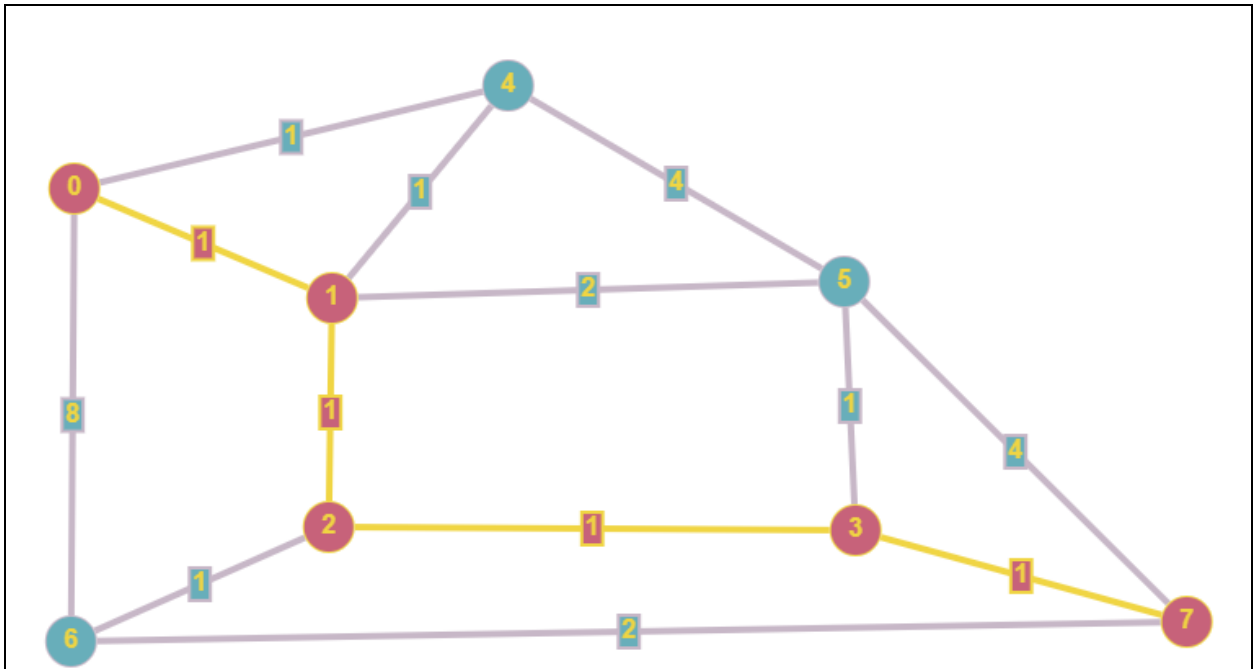


Така че най-кратките маршрути са **0 -> 2 -> 3** и **0 -> 3** и използвайки логиката за подреждане при избор на най-кратки пътища, първо отпечатваме този който посещава по ниско номерирани градове първо и.е. **0 -> 2 -> 3** съответно изхода от програмата е:  
**0 -> 2 -> 3**  
**0 -> 3**

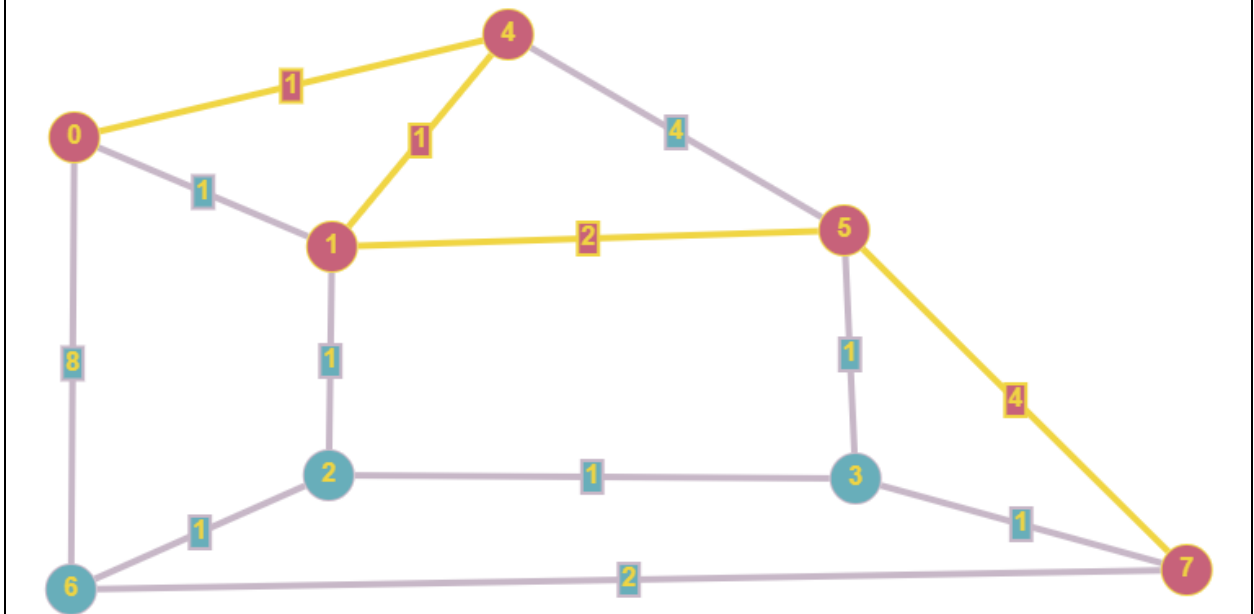
Вход	Исход
8	0 -> 1 -> 2 -> 6 -> 7
13	0 -> 4 -> 1 -> 5 -> 3 -> 7
0 1 1	
0 4 1	
0 6 8	
1 2 1	
1 4 1	
1 5 2	
2 3 1	
2 6 1	
3 5 1	
3 7 1	
4 5 4	
5 7 4	
6 7 2	

**Коментар**

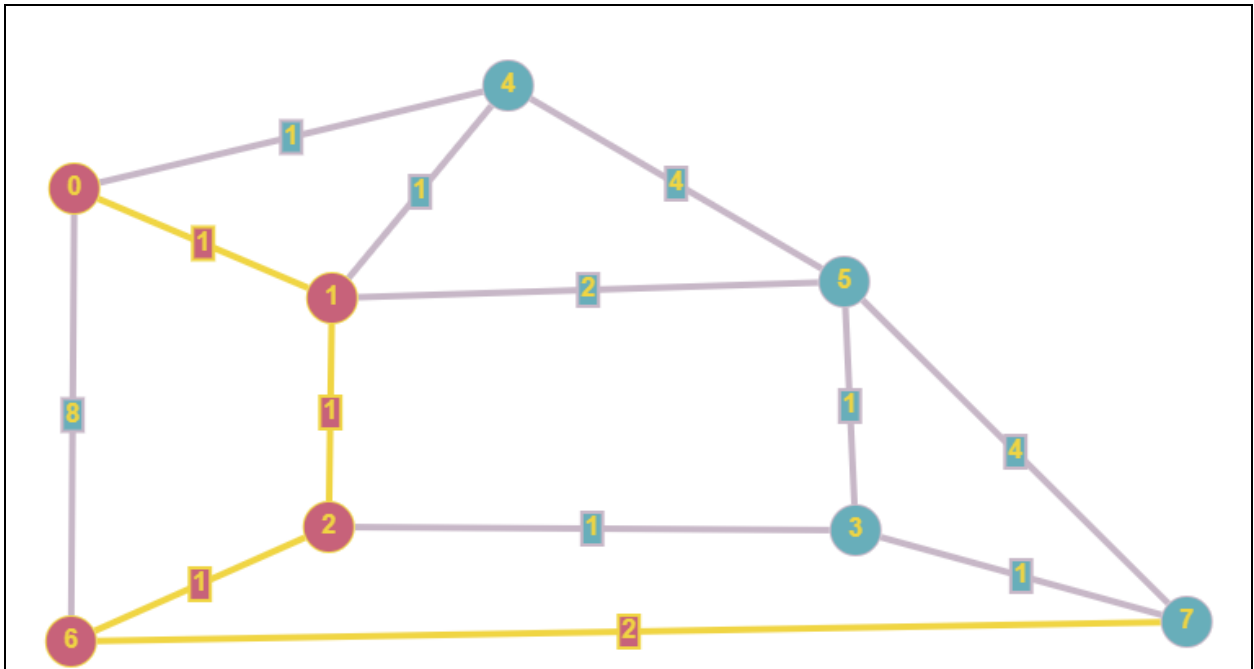




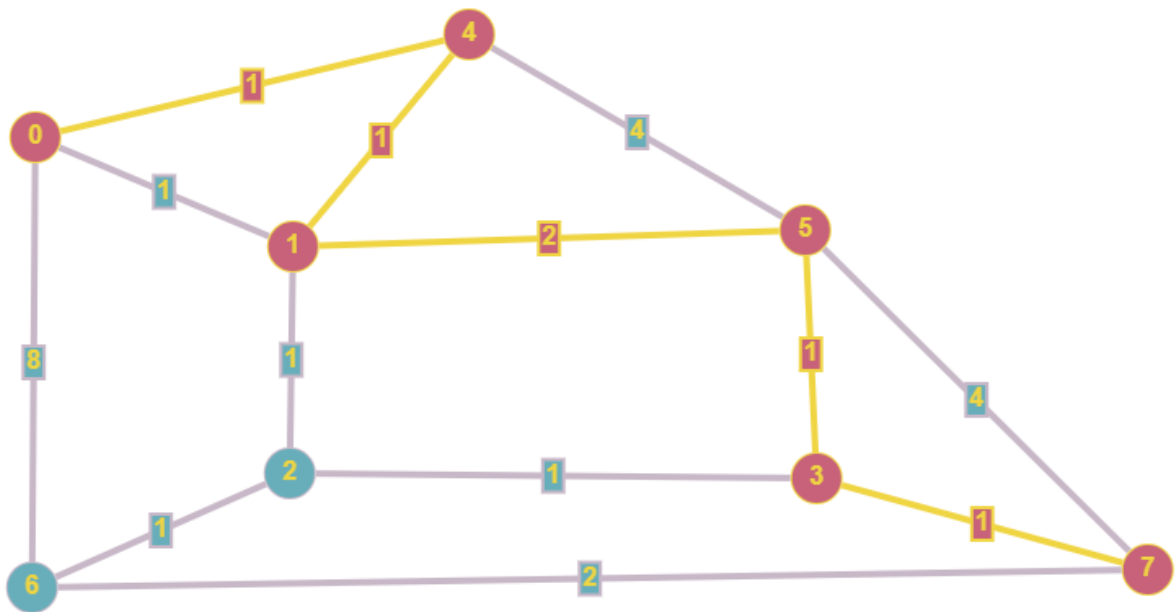
а вторият най-кратък, който не използва еднакъв път е  $0 \rightarrow 4 \rightarrow 1 \rightarrow 5 \rightarrow 7$ :



Ако сметнем сумата на дължините на тези маршрути, обаче получаваме  $0 \rightarrow 1 \rightarrow 2 \rightarrow 3 \rightarrow 7$  има дължина **4**, а  $0 \rightarrow 4 \rightarrow 1 \rightarrow 5 \rightarrow 7$  има дължина **8**, съответно сумата на 2та маршрута е  $4 + 8 = 12$ . Ако погледнем обаче маршрута  $0 \rightarrow 1 \rightarrow 2 \rightarrow 6 \rightarrow 7$ :



И маршрута 0 -> 4 -> 1 -> 5 -> 3 -> 7:



Можем да видим че сумата на маршрут 0 -> 1 -> 2 -> 6 -> 7 е 5, а сумата на маршрут 0 -> 4 -> 1 -> 5 -> 3 -> 7 е 6, което дава сума на 2та маршрута  $5 + 6 = 11$ , която е по малка от сумата на горно намерените 2, така че най-кратките 2 маршрута в този случай са:

**0 -> 1 -> 2 -> 6 -> 7**

**0 -> 4 -> 1 -> 5 -> 3 -> 7**