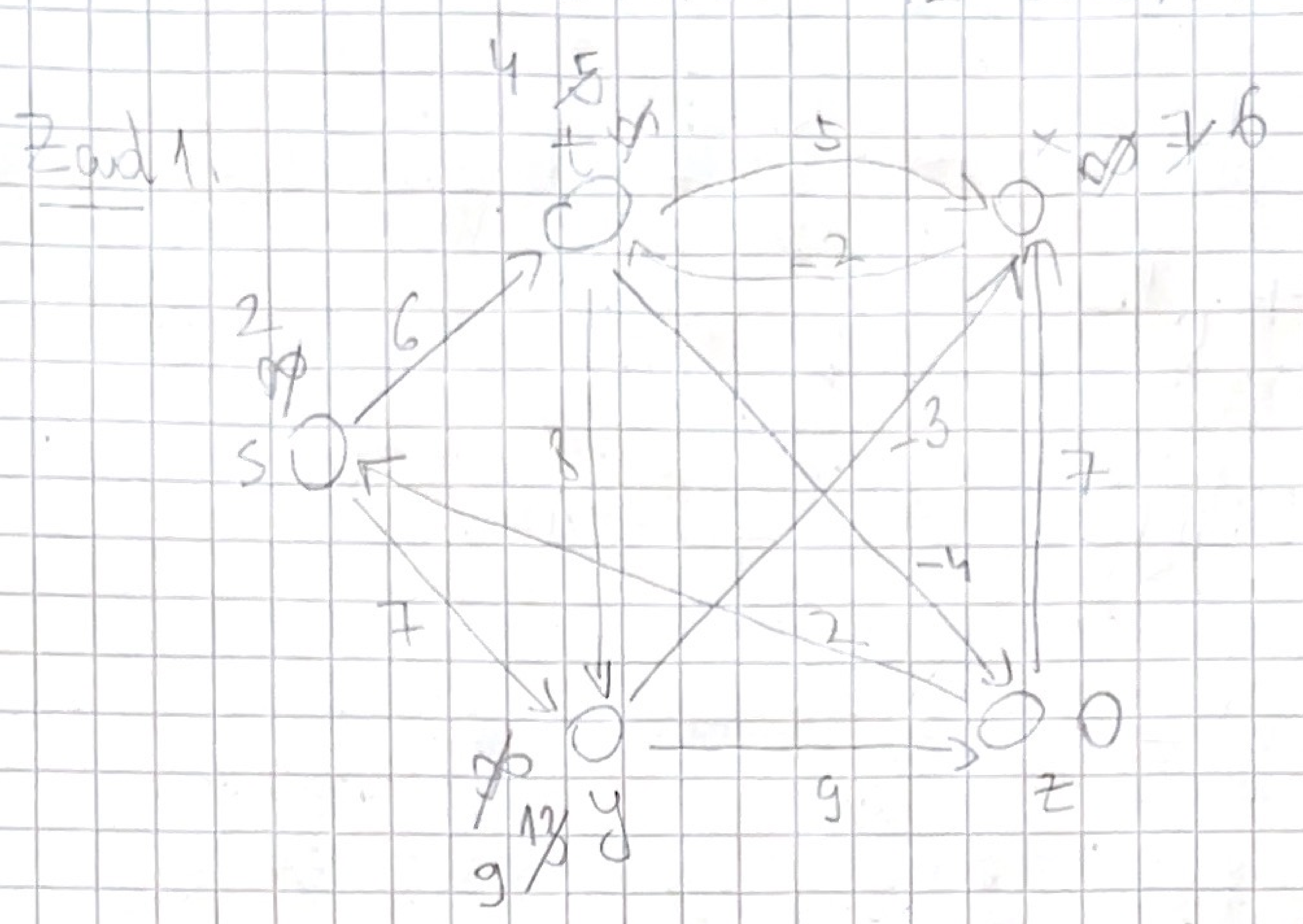
SPA 2 – 1. ZADAĆA

Zad 1.



Imamo 5 vrhova pa ćemo imati 4 obilaska, krećemo od vrha z pa je na njemu u prvom obilasku udaljenost 0, a na ostalim vrhovima je ∞

tj. iteracije

(t, x), (t, y), (t, z), (x, t), (y, x), (y, z), (z, x), (z, s), (s, t), (s, y)

1 – ✓
2 – ✓
3 – ✓
4 – ✓

} iteracije

vrhovi:

z – 0
x – 6
t – 4
s – 2
y – 9

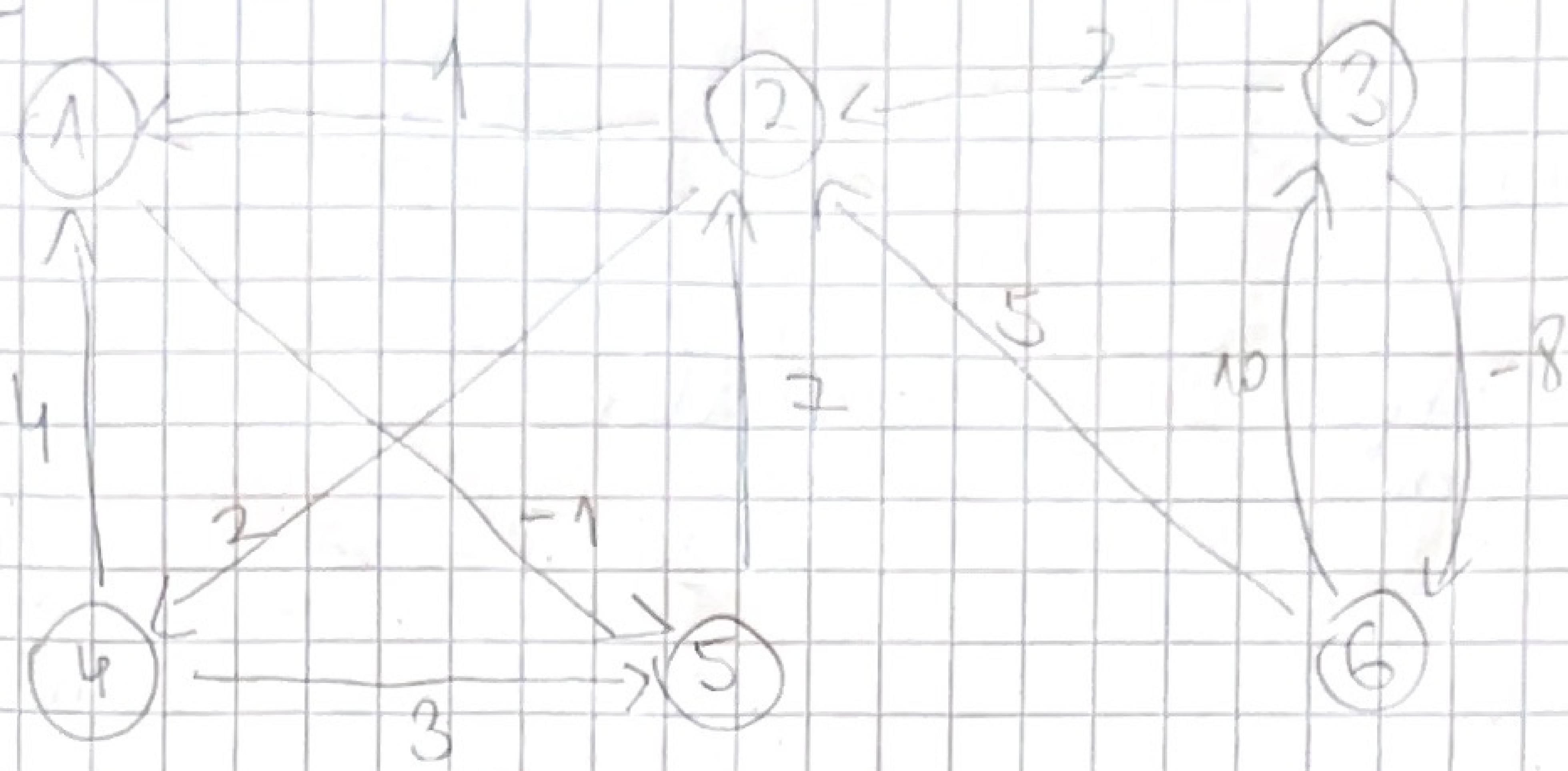
z	x	t	y	s
0, NIL	∞, NIL	∞, NIL	∞, NIL	∞, NIL
0, NIL	7, z	∞, NIL	∞, NIL	∞, NIL
0, NIL	7, z	∞, NIL	∞, NIL	2, z
0, NIL	7, z	8, s	9, s	2, z

Prvi red tablice prikazuje d i π vrijednosti za svaki vrh na početku, drugi red prikazuje isto ali nakon prve iteracije po zadanim bridovima. Treći red prikazuje isto ali nakon izvođenja na bridu (z, s), na drugima se ništa nije promijenilo, nakon toga imamo u 4. redu podatke nakon izvođenja alg. na bridovima (s, t) i (s, y).

Druga iteracija petlje

z	x	t	y	s
0, NIL	7, z	8, s	9, s	2, z
0, NIL	7, z	5, x	9, s	2, z

Zad 2



$$D^0 = \begin{array}{c|cccccc} & 1 & 2 & 3 & 4 & 5 & 6 \\ \hline 1 & 0 & \infty & \infty & \infty & -1 & \infty \\ 2 & 1 & 0 & \infty & 2 & \infty & \infty \\ 3 & \infty & 2 & 0 & \infty & \infty & -8 \\ 4 & -4 & \infty & \infty & 0 & 3 & \infty \\ 5 & \infty & 7 & \infty & \infty & 0 & \infty \\ 6 & \infty & 5 & 10 & \infty & \infty & 0 \end{array}$$

$$D^1 = \begin{array}{c|cccccc} & 1 & 2 & 3 & 4 & 5 & 6 \\ \hline 1 & 0 & \infty & \infty & \infty & -1 & \infty \\ 2 & 1 & 0 & \infty & 2 & 0 & \infty \\ 3 & \infty & \infty & 0 & \infty & \infty & -8 \\ 4 & -4 & \infty & \infty & 0 & -5 & \infty \\ 5 & \infty & 7 & \infty & \infty & 0 & \infty \\ 6 & \infty & 5 & 10 & \infty & \infty & 0 \end{array}$$

$\underset{7}{D^0[5,2]} < \underset{\infty}{D^0[5,1]} + \underset{\infty}{D^0[1,2]}$

$\underset{\infty}{D^0[5,3]} = \underset{\infty}{D^0[5,1]} + \underset{\infty}{D^0[1,3]}$

$\underset{\infty}{D^0[5,4]} = \underset{\infty}{D^0[5,1]} + \underset{\infty}{D^0[1,4]}$

$\underset{\infty}{D^0[5,6]} = \underset{\infty}{D^0[5,1]} + \underset{\infty}{D^0[1,6]}$

$\underset{5}{D^0[6,2]} < \underset{\infty}{D^0[1,2]} + \underset{\infty}{D^0[6,1]}$

$\underset{10}{D^0[6,3]} < \underset{\infty}{D^0[6,1]} + \underset{\infty}{D^0[1,3]}$

$\underset{\infty}{D^0[6,4]} = \underset{\infty}{D^0[6,1]} + \underset{\infty}{D^0[1,4]}$

$\underset{\infty}{D^0[6,5]} = \underset{\infty}{D^0[6,1]} + \underset{-1}{D^0[1,5]}$

$\underset{\infty}{D^0[2,3]} = \underset{1}{D^0[2,1]} + \underset{\infty}{D^0[1,3]} = \infty$

$\underset{2}{D^0[2,4]} < \underset{1}{D^0[2,1]} + \underset{\infty}{D^0[1,4]} = \infty$

$\underset{\infty}{D^0[2,5]} > \underset{1}{D^0[2,1]} + \underset{-1}{D^0[1,5]} = 0$

$\underset{\infty}{D^0[2,6]} = \underset{1}{D^0[2,1]} + \underset{\infty}{D^0[1,6]} = \infty$

$\underset{\infty}{D^0[3,2]} = \underset{\infty}{D^0[1,2]} + \underset{\infty}{D^0[3,1]}$

$\underset{\infty}{D^0[3,4]} = \underset{\infty}{D^0[1,4]} + \underset{\infty}{D^0[3,1]}$

$\underset{\infty}{D^0[3,5]} = \underset{-1}{D^0[1,5]} + \underset{\infty}{D^0[3,1]} = \infty$

$\underset{-8}{D^0[3,6]} < \underset{\infty}{D^0[1,6]} + \underset{\infty}{D^0[3,1]}$

$\underset{\infty}{D^0[4,2]} = \underset{\infty}{D^0[1,2]} + \underset{-4}{D^0[4,1]} = \infty$

$\underset{\infty}{D^0[4,3]} = \underset{-4}{D^0[4,1]} + \underset{\infty}{D^0[1,3]}$

$\underset{3}{D^0[4,5]} \quad \underset{-1}{D^0[1,5]} + \underset{-4}{D^0[4,1]} = -5$

$\underset{\infty}{D^0[4,6]} = \underset{\infty}{D^0[1,6]} + \underset{-4}{D^0[4,1]} = \infty$

$$
D^2 = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{array}
\begin{array}{c}
\begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\
\begin{bmatrix}
0 & \infty & \infty & \infty & -1 & \infty \\
1 & 0 & \infty & 2 & 0 & \infty \\
\infty & \infty & 0 & \infty & \infty & -8 \\
-4 & \infty & \infty & 0 & -5 & \infty \\
7 & 7 & \infty & 9 & 0 & \infty \\
6 & 5 & 10 & 7 & 5 & 0
\end{bmatrix}
\end{array}
$$

$\underset{\infty}{D^1[1,3]} = \underset{\infty}{D^1[2,3]} + \underset{\infty}{D^1[1,2]}$ $\qquad$ $\underset{\infty}{D^1[3,1]} = \underset{1}{D^1[2,1]} + \underset{\infty}{D^1[3,2]}$

$\underset{\infty}{D^1[1,4]} = \underset{2}{D^1[2,4]} + \underset{\infty}{D^1[1,2]}$ $\qquad$ $\underset{\infty}{D^1[3,4]} = \underset{2}{D^1[2,4]} + \underset{\infty}{D^1[3,2]}$

$\underset{-1}{D^1[1,5]} < \underset{0}{D^1[2,5]} + \underset{\infty}{D^1[1,2]}$ $\qquad$ $\underset{\infty}{D^1[3,5]} = \underset{0}{D^1[2,5]} + \underset{\infty}{D^1[3,2]}$

$\underset{\infty}{D^1[1,6]} = \underset{\infty}{D^1[2,6]} + \underset{\infty}{D_1[1,2]}$ $\qquad$ $\underset{-8}{D^1[3,6]} \quad \underset{\infty}{D^1[2,6]} + \underset{\infty}{D^1[3,2]}$

$\underset{-4}{D^1[4,1]} < \underset{1}{D^1[2,1]} + \underset{\infty}{D^1[4,2]}$ $\qquad$ $\underset{\infty}{D^1[5,1]} > \underset{0}{D^1[2,1]} + \underset{7}{D^1[5,2]}$

$\underset{\infty}{D^1[4,3]} = \underset{\infty}{D^1[2,3]} + \underset{\infty}{D^1[4,2]}$ $\qquad$ $\underset{\infty}{D^1[5,3]} = \underset{\infty}{D^1[2,3]} + \underset{7}{D^1[5,2]}$

$\underset{-5}{D^1[4,5]} < \underset{0}{D^1[2,5]} + \underset{\infty}{D^1[4,2]}$ $\qquad$ $\underset{\infty}{D^1[5,4]} > \underset{2}{D^1[2,4]} + \underset{7}{D^1[5,2]}$

$\underset{\infty}{D^1[4,6]} = \underset{\infty}{D^1[2,6]} + \underset{\infty}{D^1[4,2]}$ $\qquad$ $\underset{\infty}{D^1[5,6]} = \underset{\infty}{D^1[2,6]} + \underset{7}{D^1[5,2]}$

$\underset{\infty}{D^1[6,1]} > \underset{1}{D^1[2,1]} + \underset{5}{D^1[6,2]}$ $\qquad$ $\underset{\infty}{D^1[6,5]} > \underset{0}{D^1[2,5]} + \underset{5}{D^1[6,2]}$

$\underset{10}{D^1[6,3]} < \underset{\infty}{D^1[2,3]} + \underset{5}{D^1[6,2]}$

$\underset{\infty}{D^1[6,4]} > \underset{2}{D^1[2,4]} + \underset{5}{D^1[6,2]}$

za treću matricu uzimamo kao bazu 2 red i drugi stupac iz D^2 i ponavljamo postupak kao ovdje i tako sve do 6. matrice (mogao sam isprogramirat pa ti ubacit 6 matrica, al TORZ ...)

ZAD3.

Da bi smo pronašli rutu s najmanjom vjerojatnošću zatvaranja ceste, možemo koristiti Dijkstrin algoritam modificiran za traženje najmanjeg vjerojatnosti umjesto najkraćeg puta.

Prvo inicijaliziramo rastuću vrijednost vjerojatnosti za svaki vrh na beskonačno (osim početnog vrha koji se postavlja na 0). Zatim se krecemo kroz graf koristeći Dijkstrin algoritam, ali umjesto usporedbe na temelju udaljenosti, usporedimo vrhove na temelju njihovih vjerojatnosti. U svakom koraku, ažuriramo vjerojatnosti vrhova koje posjećujemo. Na kraju će se najmanja vjerojatnost zatvaranja nalaziti na ciljnom vrhu.

Evo pseudokoda za modificirani Dijkstrin algoritam:

```
function najmanja_vjerojatnost_zatvaranja(G, početni_vrh, ciljni_vrh):
    inicijaliziraj vjerojatnost[v] za svaki vrh v na beskonačno
    vjerojatnost[početni_vrh] = 0
    prioritetni_red = prazan prioritetni red sortiran prema vjerojatnost[v]
    dodaj početni_vrh u prioritetni_red

    dok prioritetni_red nije prazan:
        trenutni_vrh = ukloni vrh s najmanjom vjerojatnošću iz prioritetnog_reda
        ako trenutni_vrh == ciljni_vrh:
            prekidaj

        za svaki susjed_vrh vrha trenutni_vrh:
            vjerojatnost_prijelaza = vjerojatnost[trenutni_vrh] + vjerojatnost_prijelaza_između(trenutni_vrh,
susjed_vrh)
            ako vjerojatnost_prijelaza < vjerojatnost[susjed_vrh]:
                vjerojatnost[susjed_vrh] = vjerojatnost_prijelaza
                dodaj susjed_vrh u prioritetni_red (ili ažuriraj prioritet)

    vrati vjerojatnost[ciljni_vrh]
```

2. Da bismo pronašli najteži kamion koji može biti poslan od početnog vrha do ciljnog vrha, možemo koristiti modificiranu verziju Dijkstrinog algoritma koja će pratiti najveću dopuštenu težinu na putu do svakog vrha.

Pseudokod:

```
function najteži_kamion(G, početni_vrh, ciljni_vrh):
   inicijaliziraj najveća_težina[v] za svaki vrh v na 0
   najveća_težina[početni_vrh] = beskonačno
   prioritetni_red = prazan prioritetni red sortiran prema najveća_težina[v]
   dodaj početni_vrh u prioritetni_red

   dok prioritetni_red nije prazan:
      trenutni_vrh = ukloni vrh s najvećom najveća_težina iz prioritetnog_reda
      ako trenutni_vrh == ciljni_vrh:
         prekidaj

      za svaki susjed_vrh vrha trenutni_vrh:
         dopuštena_težina = min(najveća_težina[trenutni_vrh],
maksimalna_dopuštena_težina(trenutni_vrh, susjed_vrh))
         ako dopuštena_težina > najveća_težina[susjed_vrh]:
            najveća_težina[susjed_vrh] = dopuštena_težina
            dodaj susjed_vrh u prioritetni_red (ili ažuriraj prioritet)

   vrati najveća_težina[ciljni_vrh]
```

Za pronalaženje najkraćeg puta kojim ovaj kamion može proći, možemo koristiti standardni Dijkstrin algoritam za traženje najkraćeg puta s uvjetom ograničenja težine na svakom koraku.

3. Kada kamion može ići samo istočno od svakog grada koji posjeti, možemo prilagoditi algoritam iz koraka 2 tako da uvjetujemo prijelaz samo na bridove koji vode istočno. To znači da ćemo prilagoditi

uvjet za prijelaz na susjedni vrh samo ako brid vodi istočno. Sve ostale korake možemo zadržati iste kao u koraku 2.