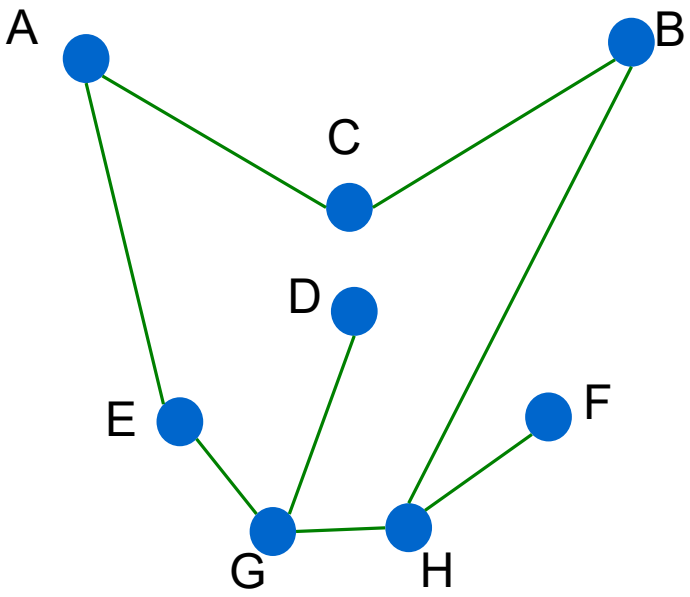
Algoritmi i strukture podataka

- predavanja -

11. Grafovi

Osnovni pojmovi I

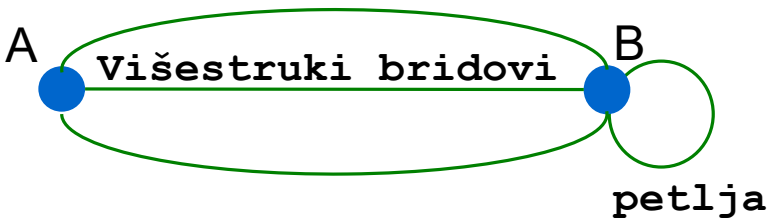
- Graf G = (V, E)
  - objekt (struktura podataka) definiran na skupu točaka **V** (tzv. vrhovi; engl. *vertex*, pl. *vertices*) te spojnica **E** (tzv. bridovi; *edges*) među vrhovima



- skup vrhova V = { A, B, C, D, E, F, G, H }
- skup bridova E = { (A, E), (A, C), (C, B), (B, H), (D, G), (E, G), (G, H), (H, F) }
- konačan graf: skupovi V i E su konačni
- moguće E = Ø, tj. graf ne mora sadržavati bridove (prazan graf)
- graf mora sadržavati najmanje jedan vrh
  - trivijalan graf: graf s jednim vrhom

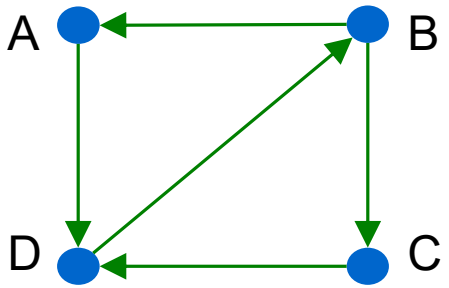
Osnovni pojmovi II

- |V| red grafa: broj vrhova grafa
- |E| veličina grafa: broj bridova grafa
- $d_G(v)$ stupanj vrha grafa (valencija grafa)
  - broj bridova povezanih s vrhom, tj. broj sjecišta male kružnice oko vrha s bridovima dotičnog vrha ➔ petlja se broji dva puta
- vrh v je *izoliran* ako je $d_G(v) = 0$, a *list* ako je $d_G(v) = 1$
- pravi brid (karika)
  - brid koji spaja različite vrhove
- petlja
  - brid spaja vrh s samim sobom
- višestruki bridovi
  - dva ili više bridova spojena s istim parom vrhova



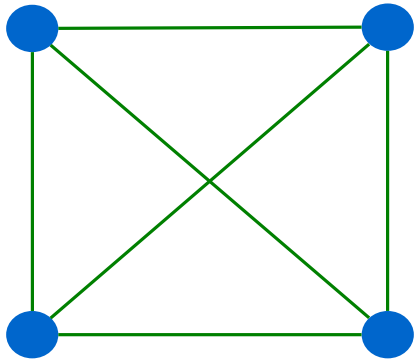
Vrste grafova I

- neusmjereni graf
  - bridovi nemaju orijentaciju: brid (A, B) je identičan bridu (B, A)
- usmjereni graf (*directed graph*, DIGRAF)
  - bridovima se pridružuje smjer (lukovi)
  - (A, B)≠(B, A)
- orijentirani graf (*oriented graph*)
  - usmjereni graf gdje između dva vrha postoji samo jedan usmjereni brid
  - oprez: orijentirani i usmjereni graf se vrlo često koriste kao sinonimi
- miješani graf (*mixed graph*)
  - graf koji se sastoji od usmjerenih i neusmjerenih bridova



Vrste grafova II

- jednostavan graf
  - neusmjeren graf, u literaturi najčešće podrazumijevan graf
  - ne sadrži niti petlje niti višestruke bridove
- potpun graf
  - jednostavan graf gdje je svaki par vrhova spojen bridom

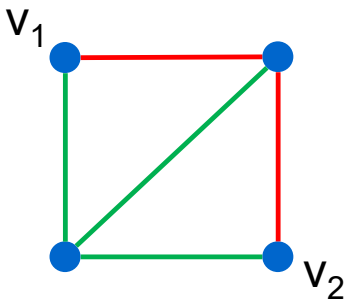


Vrste grafova III

- multigraf
  - neusmjeren graf, dozvoljava višestruke bridove, ali ne i petlje
- pseudograf (multigraf sinonim?)
  - multigraf koji dozvoljava i petlje
  - oprez: vrlo česti sinonim za multigraf
- težinski graf
  - svakom bridu je pridijeljen je broj/težina koja predstavlja cijenu, udaljenost, kapacitet

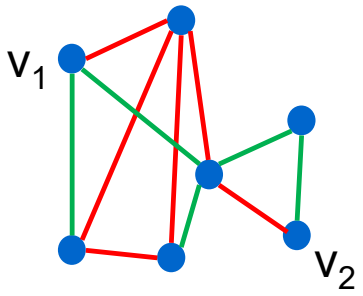
Šetnje, putovi, povezanost I

- šetnja (*walk*) je niz $W=v_0, e_1, v_1, e_2 \ldots, v_k$ gdje su v_i vrhovi, a e_i bridovi grafa takvi da za $1 \leq i \leq k$, brid e_i povezuje vrhove v_{i-1} i v_i ; k – dužina šetnje
  - šetnja može proizvoljni broj puta proći nekim vrhom, tj. bridom
- put (*path*) - šetnja koja ne uključuje niti jedan vrh dva puta
  - eventualno prvi, tj. zadnji vrh mogu biti isti
  - elementarni (jednostavan) put vs. put (sinonimi)

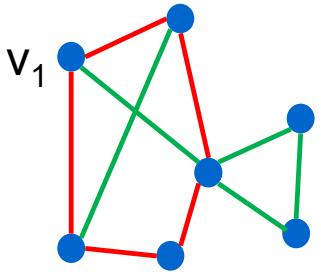


Šetnje, putovi, povezanost II

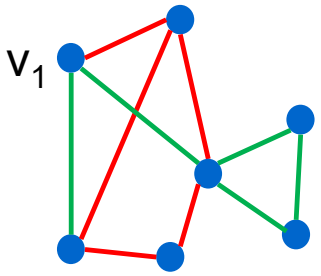
- staza (*trail*) - šetnja koja ne prolazi dva puta istim bridom
  - može uključivati isti vrh dva puta ukoliko se dolazni i odlazni brid razlikuju



- ciklus (*cycle*) - put koji počinje i završava u istom vrhu



- krug (*circuit*) – staza koja počinje i završava u istom vrhu



Šetnje, putovi, povezanost III

- Hamiltonov put - put kroz sve vrhove grafa, svaki vrh obilazi se točno jednom
  - nije nužno proći kroz sve bridove
- Hamiltonov krug/ciklus - poseban slučaj H. puta gdje su prvi i zadnji vrh jednaki
- Eulerova staza (put) – staza (put) kroz sve bridove grafa
  - može proći isti vrh više puta
- Eulerov krug/ciklus - poseban slučaj E. staze gdje su prvi i zadnji vrh jednaki

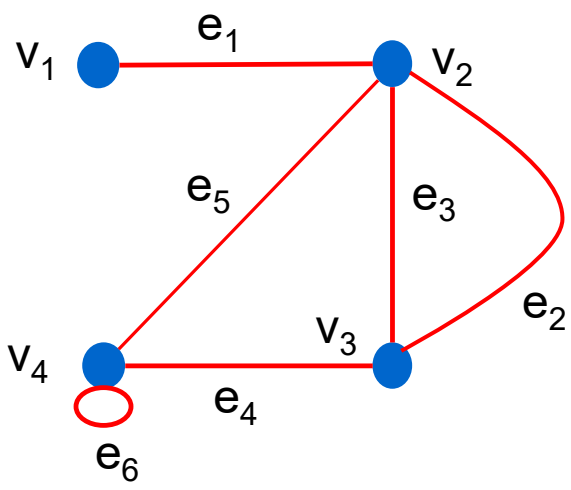
Šetnje, putovi, povezanost IV

- neusmjereni graf je **povezani (*connected*) graf** ako za svaki par vrhova postoji put između vrhova
- usmjereni graf je **strogo povezani (*strongly connected*) graf** ako za svaki par vrhova postoji (usmjereni) put između vrhova
  - ako uz zamjenu usmjerenih bridova s neusmjerenima se dobije (neusmjereni) povezani graf tada ga nazivamo **slabo povezanim (*weakly connected*) grafom**

Reprezentacija grafova I: Lista susjedstva (*adjacency list*)

- za neki G = (V, E) lista susjedstva je skup nesortiranih lista gdje svaka pojedina lista u skupu opisuje za neki vrh s kojim je sve vrhovima i/ili bridovima grafa povezan dotičan vrh
- različite implementacije strukture podataka za zapis liste susjedstva
- polje pokazivača A[|V|] gdje svaki element polja predstavlja jedan vrh i pokazuje na listu susjednih vrhova
- memorijski efikasan način zapisa o grafu ☺
- dohvat nekog elementa – slijedni prolaz kroz listu ☹
- moguće zapisati i informacije o bridovima

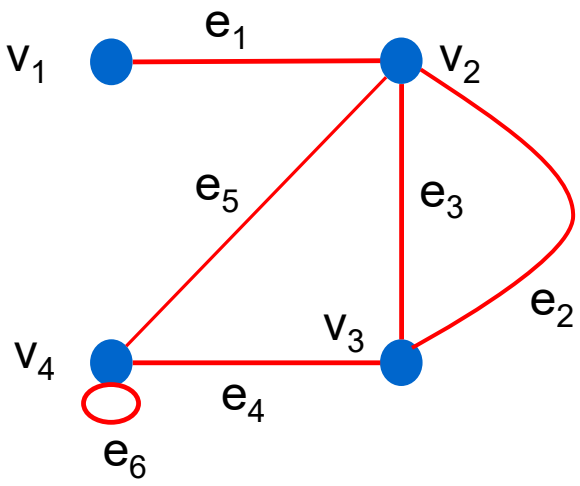
Reprezentacija grafova I: Lista susjedstva (*adjacency list*)



Lista susjedstva	
A[|V|]	Pojedinačne liste
v_1	→ v_2
v_2	→ v_1 → v_3 →v_4
v_3	→ v_2 → v_4
v_4	→ v_2 → v_3 → v_4

Reprezentacija grafova II: Matrica susjedstva (Adjacency matrix)

- matrica susjedstva vrhova grafa
- za neki G = (V, E) te n = |V| matrica susjedstva je kvadratna matrica M_{nxn}
- svakom se vrhu pridružuje red *i* i stupac *j* u matrici;
- m_{ij} broj bridova koji spajaju vrhove *i* i *j*



M	v_1	v_2	v_3	v_4
v_1	0	1	0	0
v_2	1	0	2	1
v_3	0	2	0	1
v_4	0	1	1	1

Reprezentacija grafova II: Matrica susjedstva (Adjacency matrix)

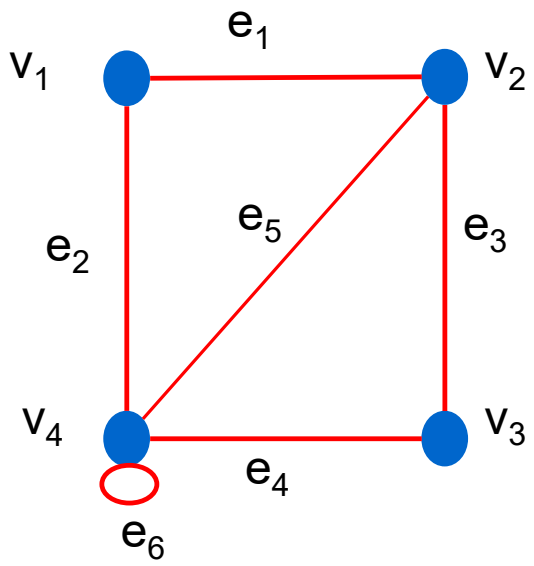
- bez petlji → dijagonalni elementi iznose nula
- neusmjereni graf – simetrična matrica M_{nxn}
  - dovoljno je zapisati samo jednu polovicu matrice
- kompaktan memorijski zapis
- daje odmah odgovor postoji li brid između bilo koja dva vrha
- suma retka (stupca) iznosi broj bridova incidentnih s dotičnim vrhom
  - matrica susjedstva samo broji bridove, ali ih ne razlikuje

Reprezentacija grafova III: Matrica incidencije (Incidence matrix)

- vrh je incidentan nekom bridu ako je povezan s dotičnim bridom
- svakom vrhu pridružuje red *i*, svakom bridu pridružuje se stupac *j* u matrici
- za G = (V, E) te n = |V| i m = |E|, matrica incidencije grafa je dimenzije M_{nxm}

Reprezentacija grafova III: Matrica incidencije (Incidence matrix)

- za neusmjereni graf
  - ako je vrh v_i incidentan rubu e_j, $M_{ij} = 1$
  - ako je vrh v_i nije incidentan rubu e_j, $M_{ij} = 0$



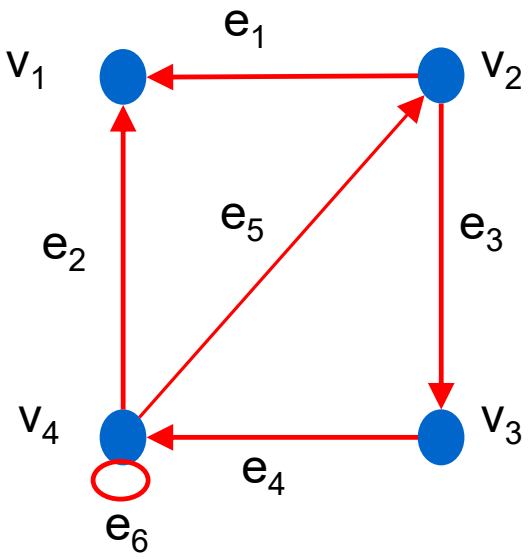
M	e_1	e_2	e_3	e_4	e_5	e_6
v_1	1	1	0	0	0	0
v_2	1	0	1	0	1	0
v_3	0	0	1	1	0	0
v_4	0	1	0	1	1	2

Petlja

- suma po retcima odgovora stupnju (valenciji) vrha
- suma po stupcima iznosi 2 ← svaki brid ulazi/izlazi u/iz jednog vrha

Reprezentacija grafova III: Matrica incidencije (Incidence matrix)

- za usmjereni graf
  - ako brid e_j izlazi iz vrha v_i, $M_{ij} = 1$
  - ako vrh v_i i rub e_j nisu incidentni, $M_{ij} = 0$
  - ako brid e_j ulazi u vrh v_i, $M_{ij} = -1$



M	e_1	e_2	e_3	e_4	e_5	e_6
v_1	-1	-1	0	0	0	0
v_2	1	0	1	0	-1	0
v_3	0	0	-1	1	0	0
v_4	0	1	0	-1	1	0

Petlja

- suma po stupcima = 0 (brid izlazi iz jednog vrha i ulazi u jedan vrh)

Obilazak grafa u dubinu (*depth first search, DFS*)

- Rekurzivni algoritam za DFS grafa *G* krenuvši od vrha *v:*

```
procedura DFSR (G, v)
    označi v kao obiđen
    ispiši v
    za sve neobiđene vrhove w susjedne vrhu v
        DFSR (G, w)
```

- Složenost Θ(|E|+|V|)

DFSR.cpp

Obilazak grafa u dubinu (*depth first search, DFS*)

- Nerekurzivni algoritam zahtijeva vlastitu implementaciju stoga (koji je rekurzijom implicitno ostvaren)

```
procedura DFS (G, v)
    stavi v na stog S
    dok ima elemenata na stogu S
        skini v sa stoga
        ako v nije obiđen
            označi v kao obiđen
            ispiši v
            za sve neobiđene vrhove w susjedne vrhu v
                stavi w na stog S
```

- Složenost Θ(|E|+|V|)

DFS.cpp

Obilazak grafa u širinu (*breadth first search, BFS*)

- Nerekurzivni algoritam za BFS grafa *G* krenuvši od vrha *v:*

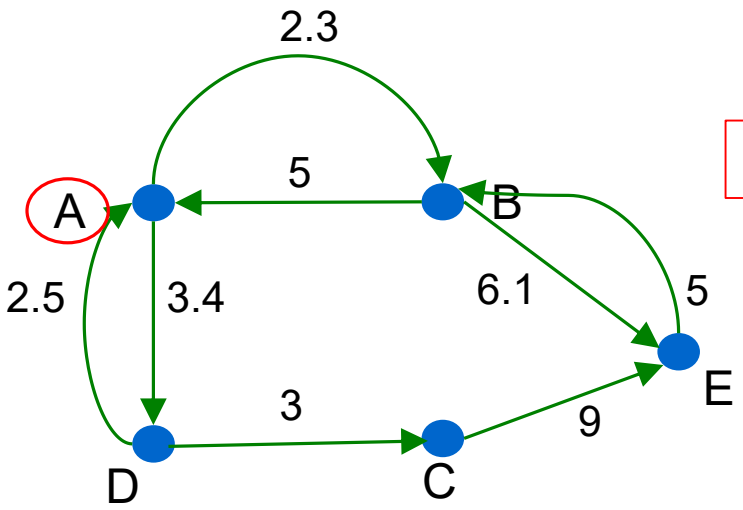
```
procedura BFS (G, v)
    označi v kao obiđen
    ispiši v
    stavi v u red Q
    dok ima elemenata u redu Q
          uzmi v iz reda Q
          za sve neobiđene vrhove w susjedne vrhu v
                ako w nije obiđen
                      označi w kao obiđen
                      ispiši w
                      stavi w u red Q
```

- Složenost Θ(|E|+|V|)

BFS.cpp

Dijkstrin algoritam

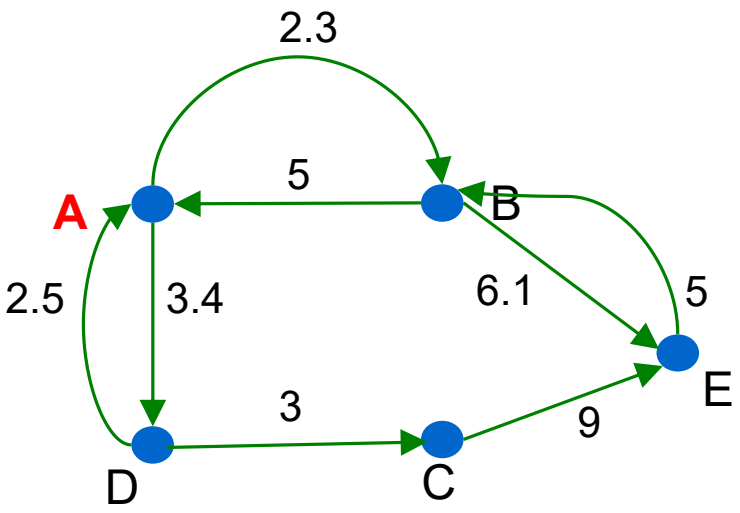
- Dijkstrin algoritam ili Dijkstrin algoritam najkraćeg puta (engl. *Dijkstra's Shortest Path First algorithm, SPF algorithm*)
  (E. W. Dijkstra, 1956.)
- zadan je težinski usmjeren graf G = (V, E) i početni vrh v_0
- Cilj:
  - iz početnog vrha v_0 pronaći najkraći put do svih ostalih čvorova iz V, ili
  - iz početnog vrha v_0 pronaći najkraći put do odredišnog vrha $v_{odredište}$



Ovdje je A je početni vrh.

GraphDijkstra

Dijkstrin algoritam – primjer (1)



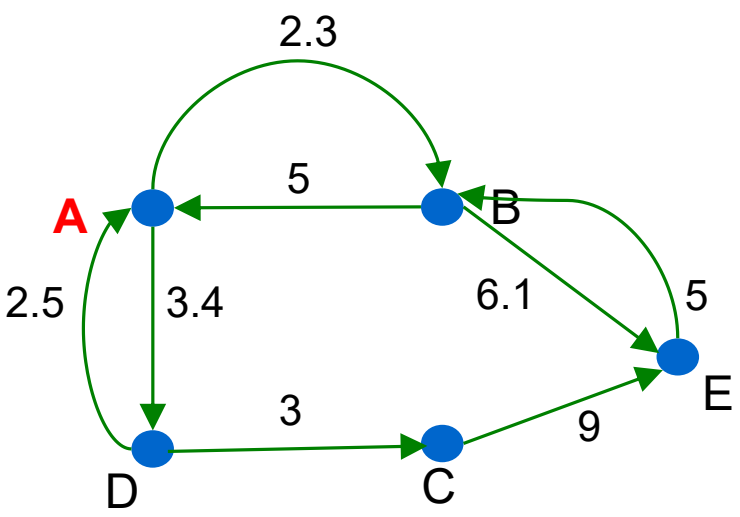

```
// Q: minheap;
// na početku prazan

za svaki v iz V
   udaljenost[v] = ∞
   preth[v] = nedef. // -

stvori minheap Q s udaljen. za
sve v

// A je početni vrh
udaljenost[A] = 0
```

	A	B	C	D	E
udaljenost	0	∞	∞	∞	∞
preth	-	-	-	-	-

$Q = \{0, \infty, \infty, \infty, \infty\}$

Zapravo se pamti (A, 0), a ne 0 (isto i za ostale)!

Dijkstrin algoritam – primjer (2)

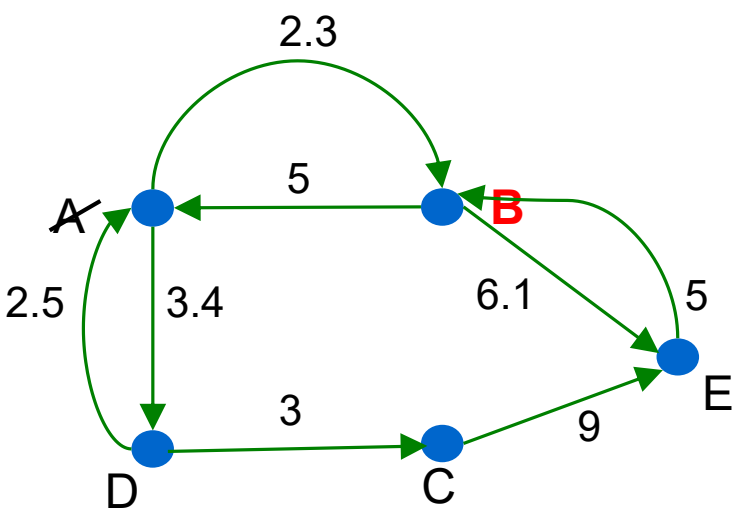



```
dok Q n bude prazan
    u = skini(Q)
    označi da je u obiđen
    za svakog susjeda v od u
        nova = udaljenost[u] +
                    težina(u, v)
        ako je nova < udaljenost[v]
            udaljenost[v] = nova
            preth[v] = u

    stvori minheap Q s novim udaljen.
        za sve neobiđene vrhove
```

	A	B	C	D	E
udaljenost	0	2.3	∞	3.4	∞
preth	-	A	-	A	-

$Q = \{2.3, \infty, 3.4, \infty\}$

Dijkstrin algoritam – primjer (3)

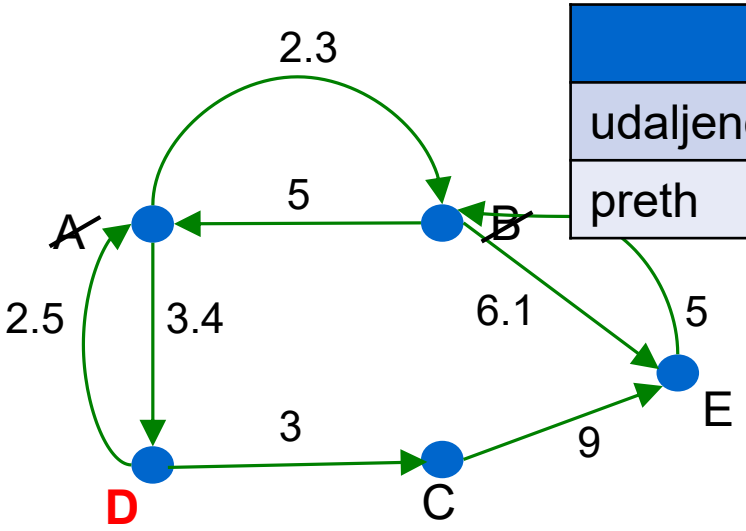



```
dok Q n bude prazan
    u = skini(Q)
    označi da je u obiđen
    za svakog susjeda v od u
        nova = udaljenost[u] +
                    težina(u, v)
        ako je nova < udaljenost[v]
            udaljenost[v] = nova
            preth[v] = u

    stvori minheap Q s novim udaljen.
        za sve neobiđene vrhove
```

	A	B	C	D	E
udaljenost	0	2.3	∞	3.4	2.3 + 6.1
preth	-	A	-	A	B

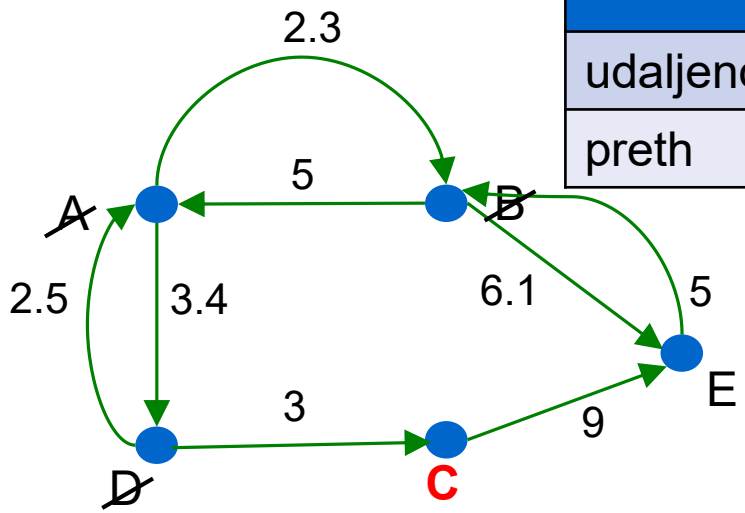
$Q = \{3.4, \infty, 8.4\}$

Dijkstrin algoritam – primjer (4)



	A	B	C	D	E
udaljenost	0	2.3	6.4	3.4	8.4
preth	-	A	D	A	B

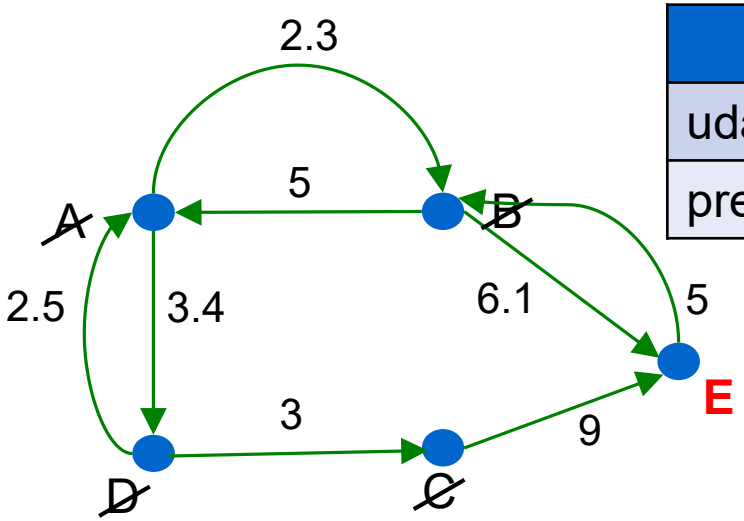
Q = {6.4, 8.4}



	A	B	C	D	E
udaljenost	0	2.3	6.4	3.4	8.4
preth	-	A	D	A	B

Q = {8.4}

Dijkstrin algoritam – primjer (5)



	A	B	C	D	E
udaljenost	0	2.3	6.4	3.4	8.3
preth	-	A	D	A	B

Q = { }