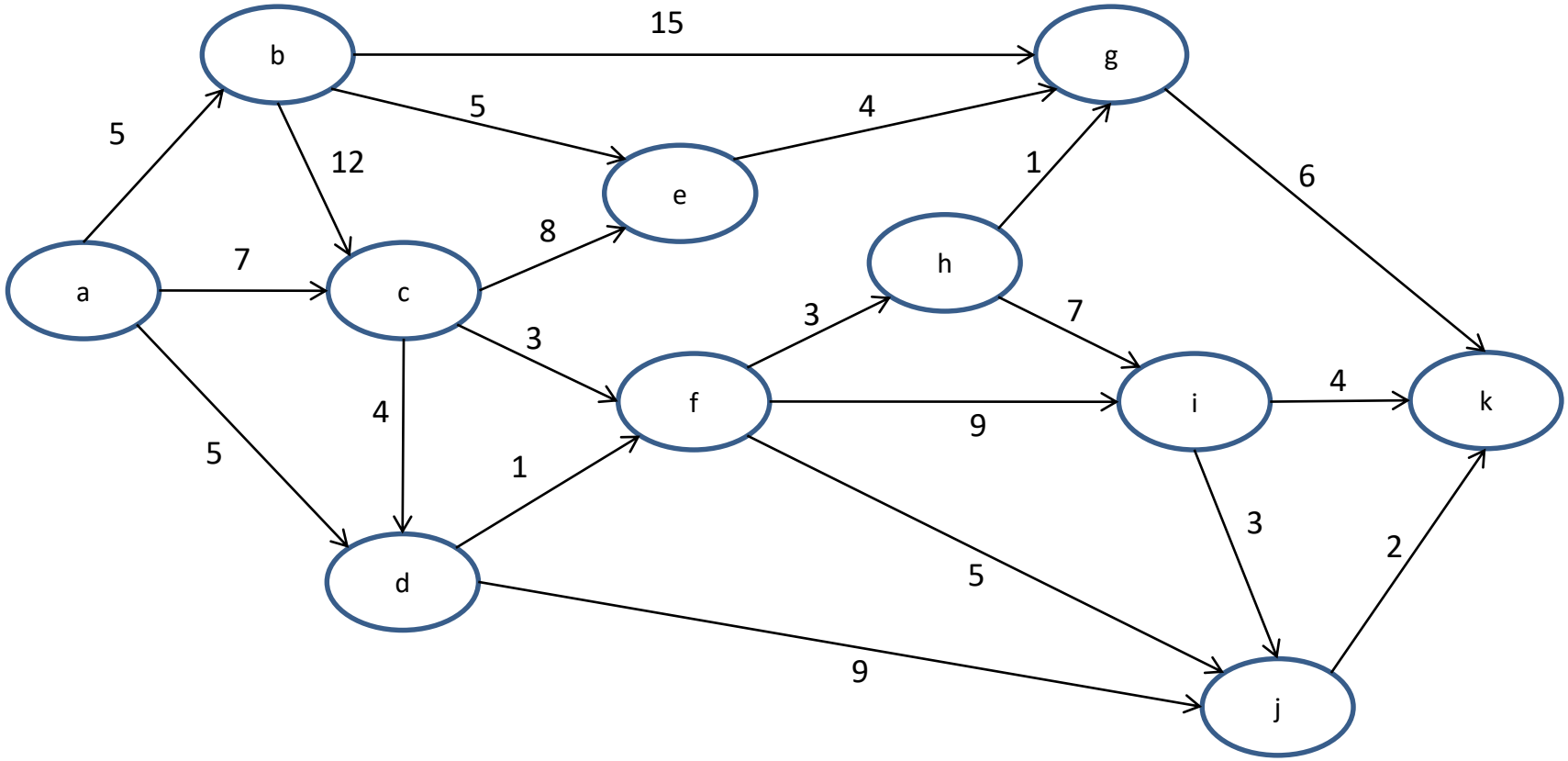
ANALIZA KRITIČNE POTI

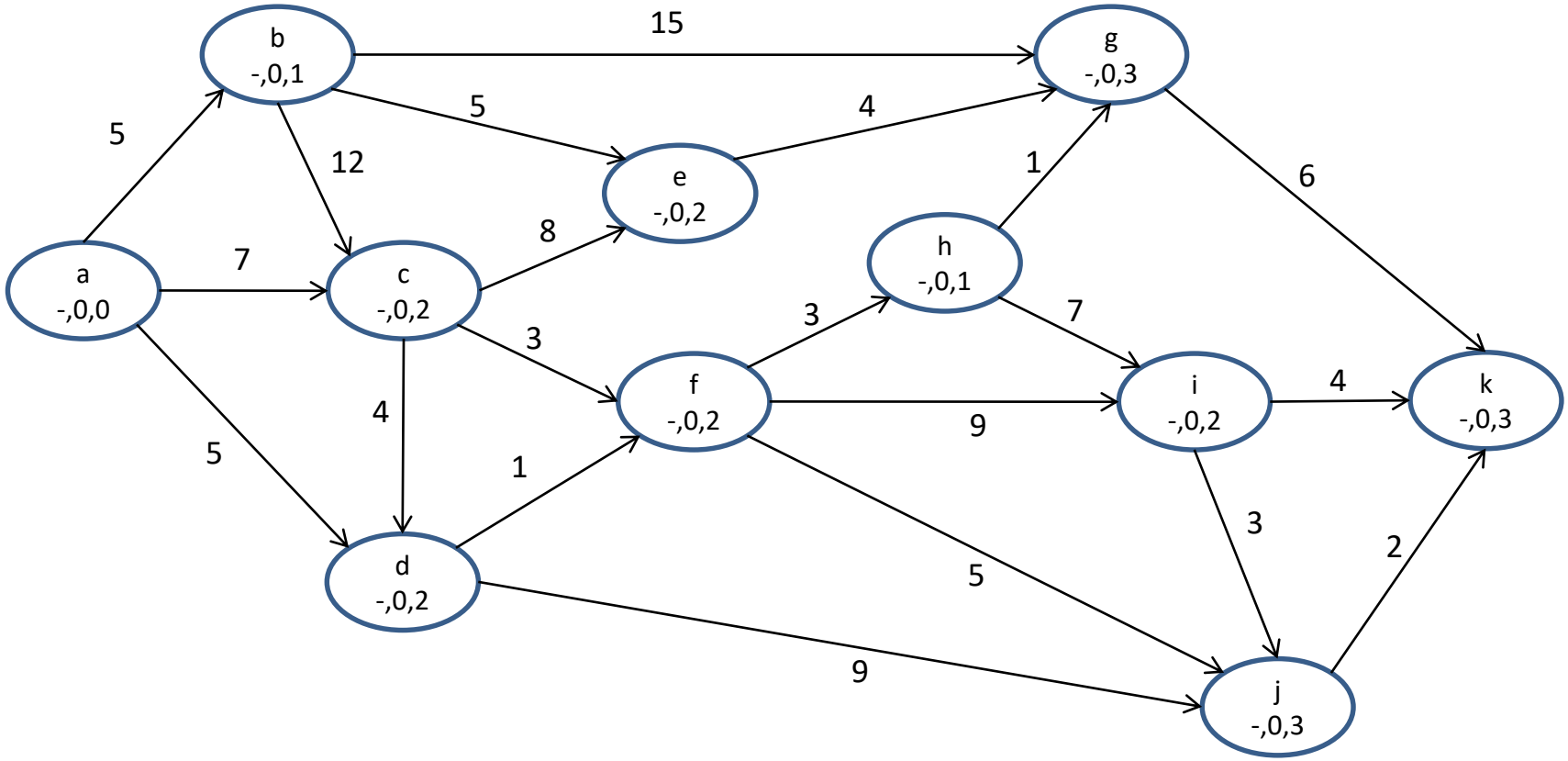
Podan je graf na sliki. Izračunajte kritično pot v grafu.



ANALIZA KRITIČNE POTI

Inicializacija

v vozliščih shranimo predhodno vozlišče, maksimalen čas do vozlišča, število še nepregledanih vhodov

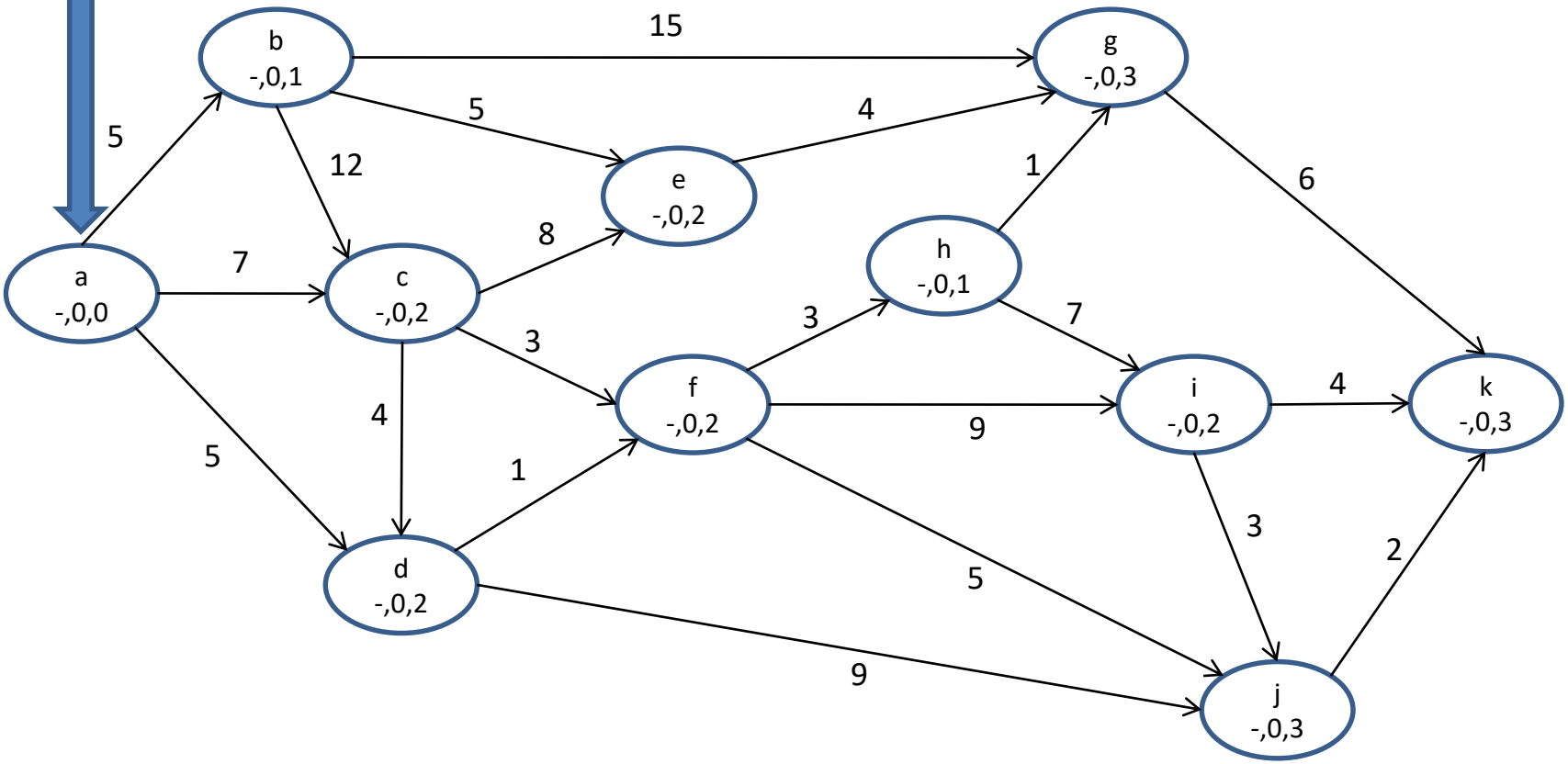


ANALIZA KRITIČNE POTI

Seznam vozlišč, katerih naslednikov še nismo pregledali:

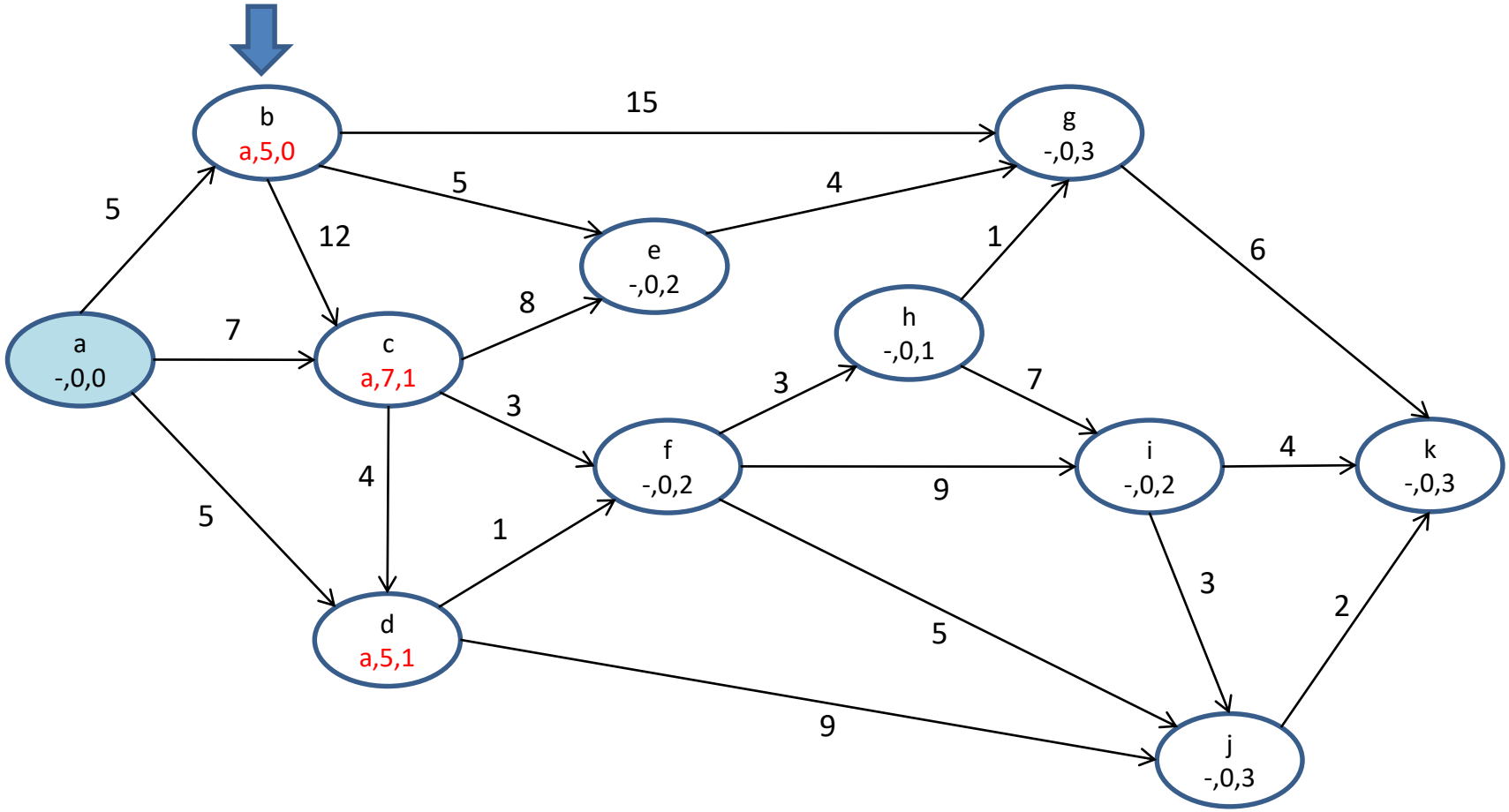
a

v seznam dodajamo samo vozlišča, katerim smo pregledali vse vhode



ANALIZA KRITIČNE POTI

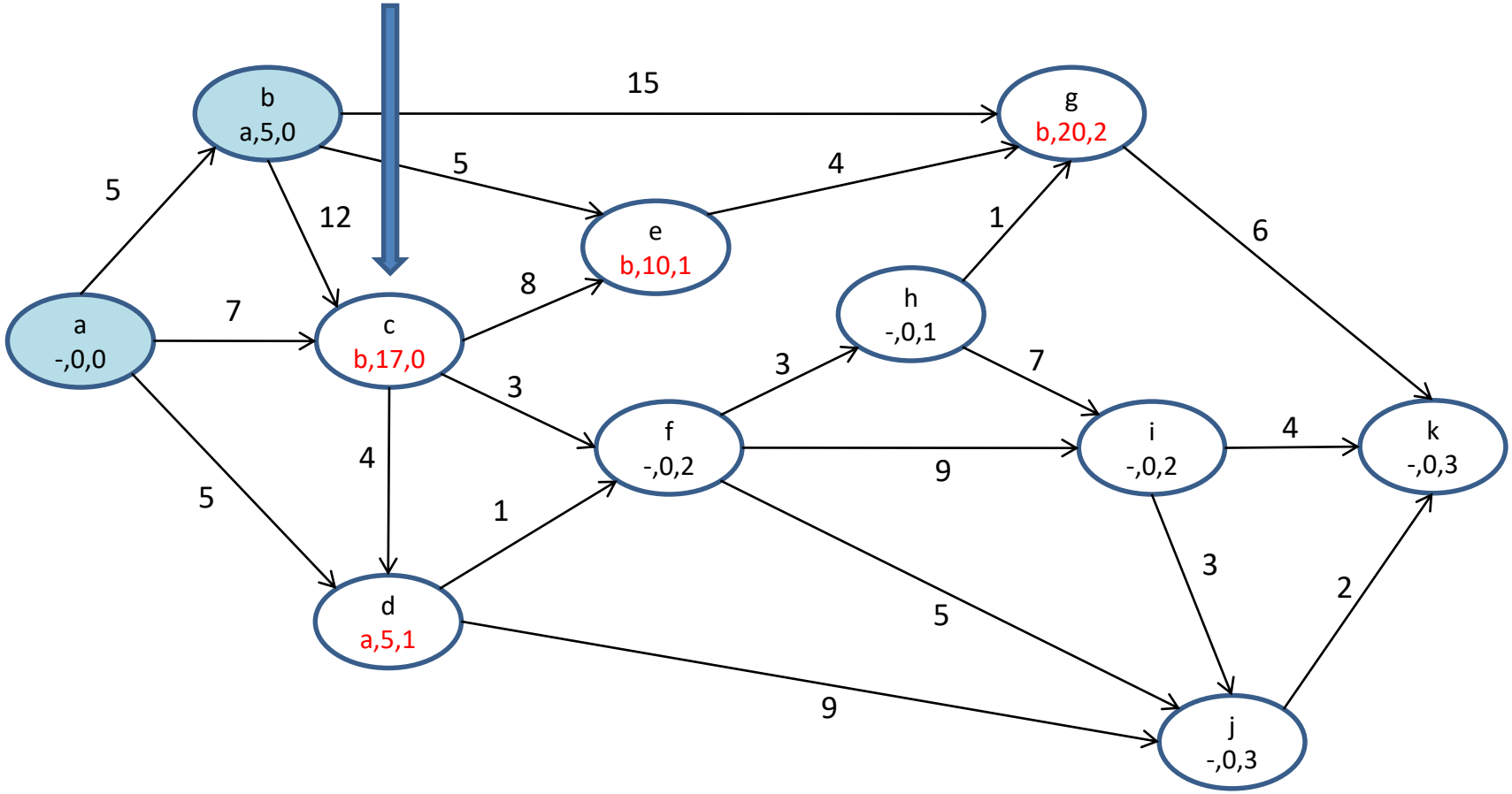
Seznam vozlišč, katerih naslednikov še nismo pregledali:



ANALIZA KRITIČNE POTI

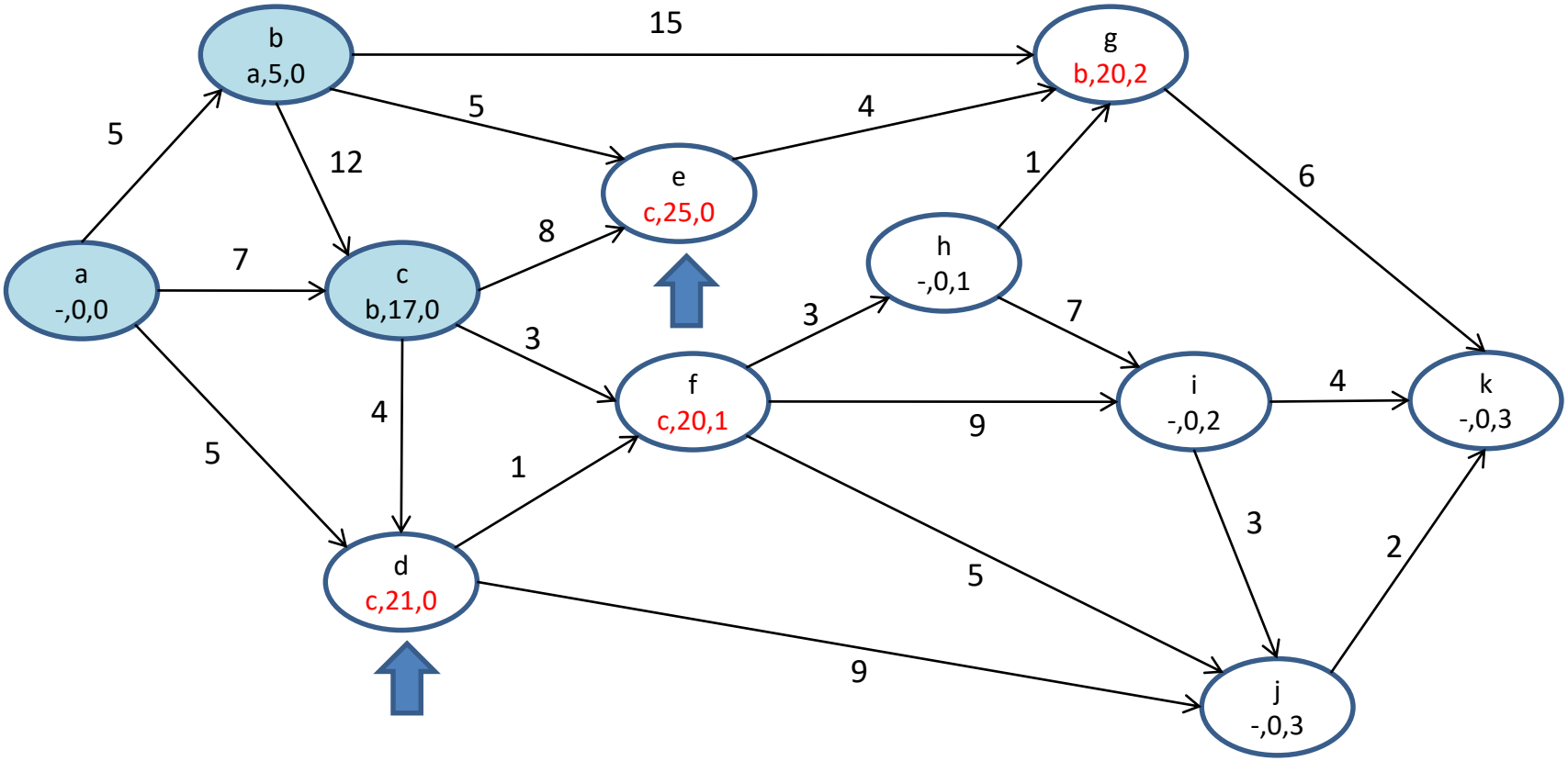
Seznam vozlišč, katerih naslednikov še nismo pregledali:

našli smo daljšo pot: razdalja vozlišča ‘b’ + povezava od ‘b’ do ‘c’



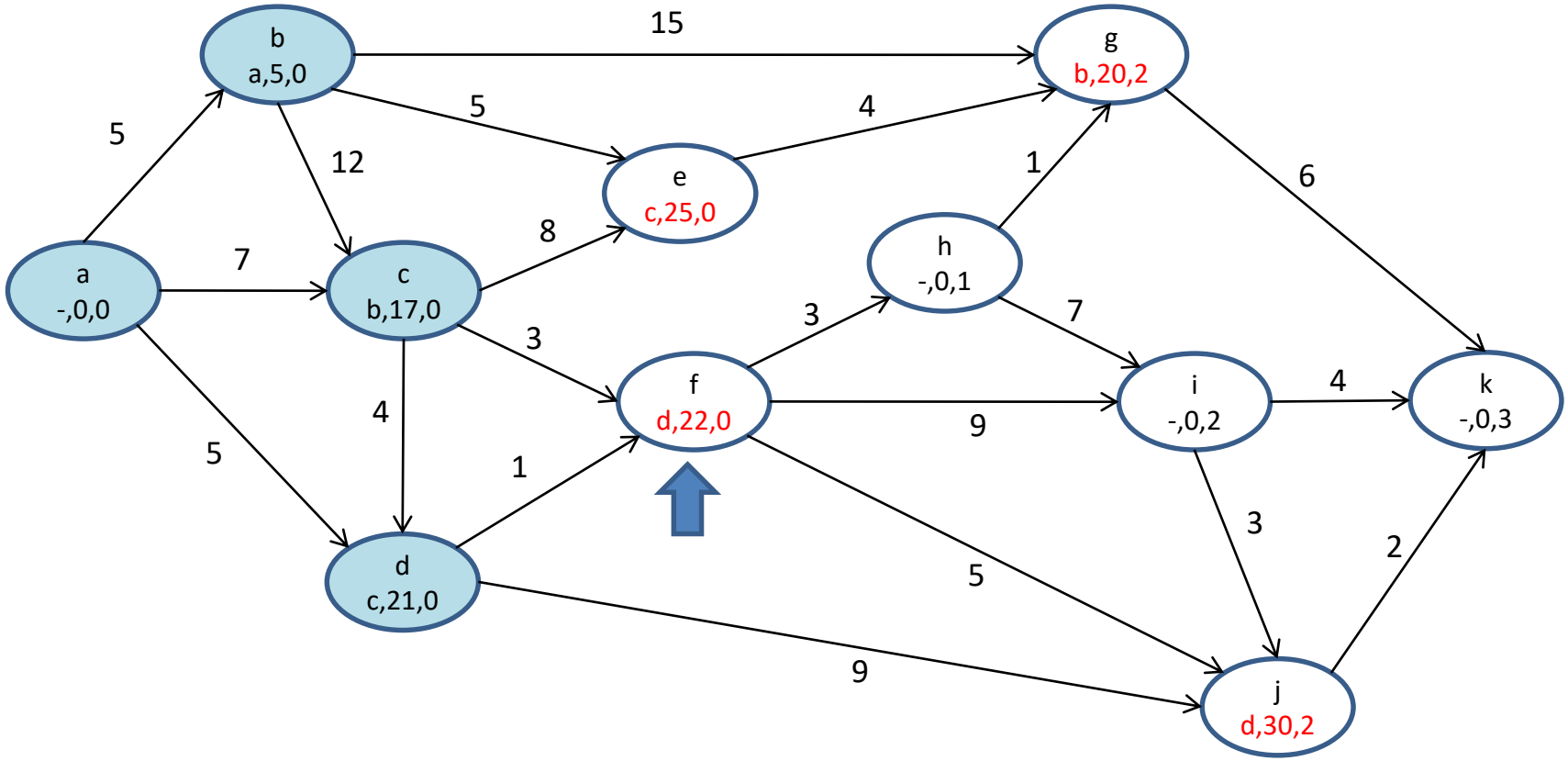
ANALIZA KRITIČNE POTI

Seznam vozlišč, katerih naslednikov še nismo pregledali:



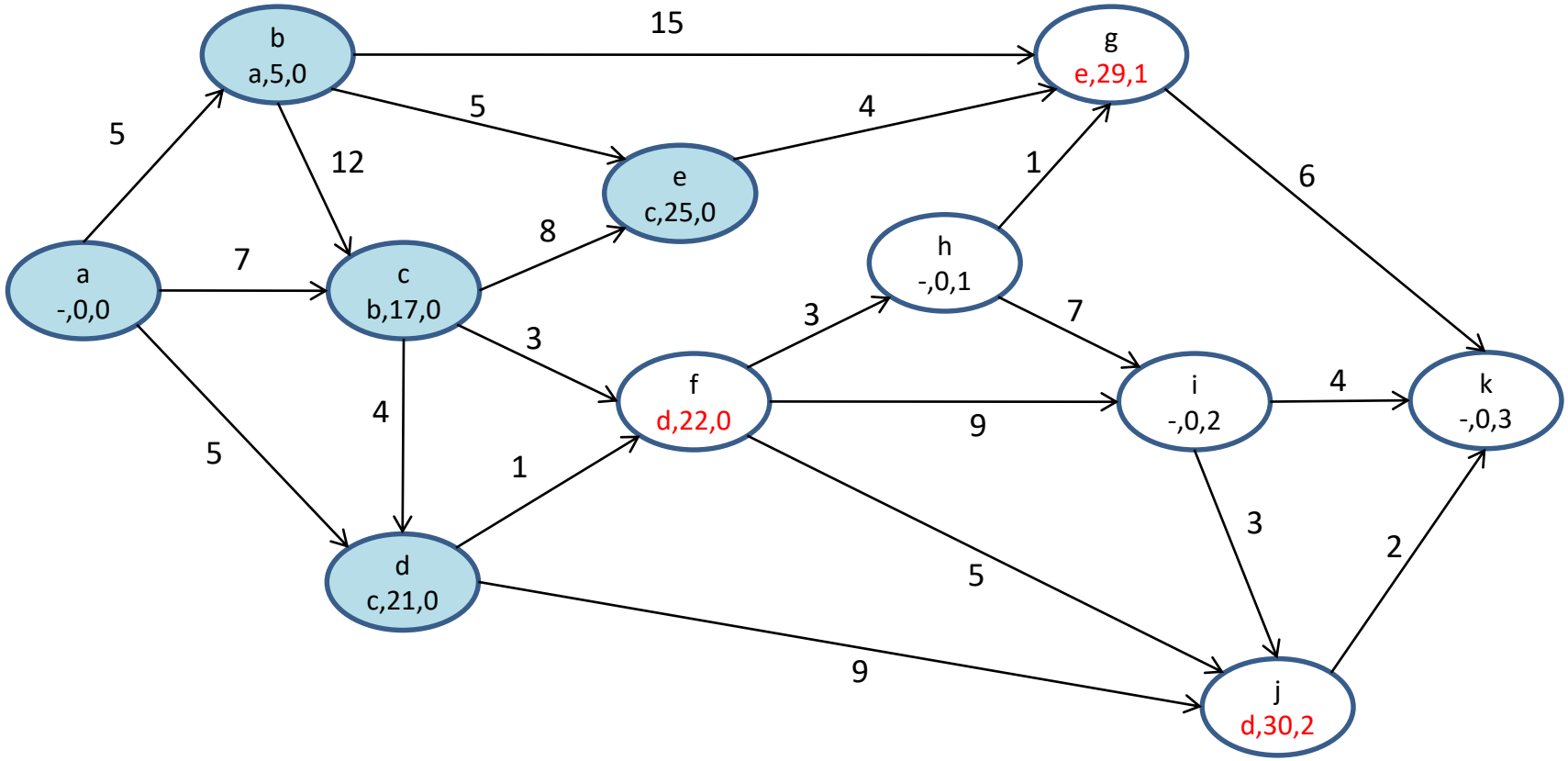
ANALIZA KRITIČNE POTI

Seznam vozlišč, katerih naslednikov še nismo pregledali:



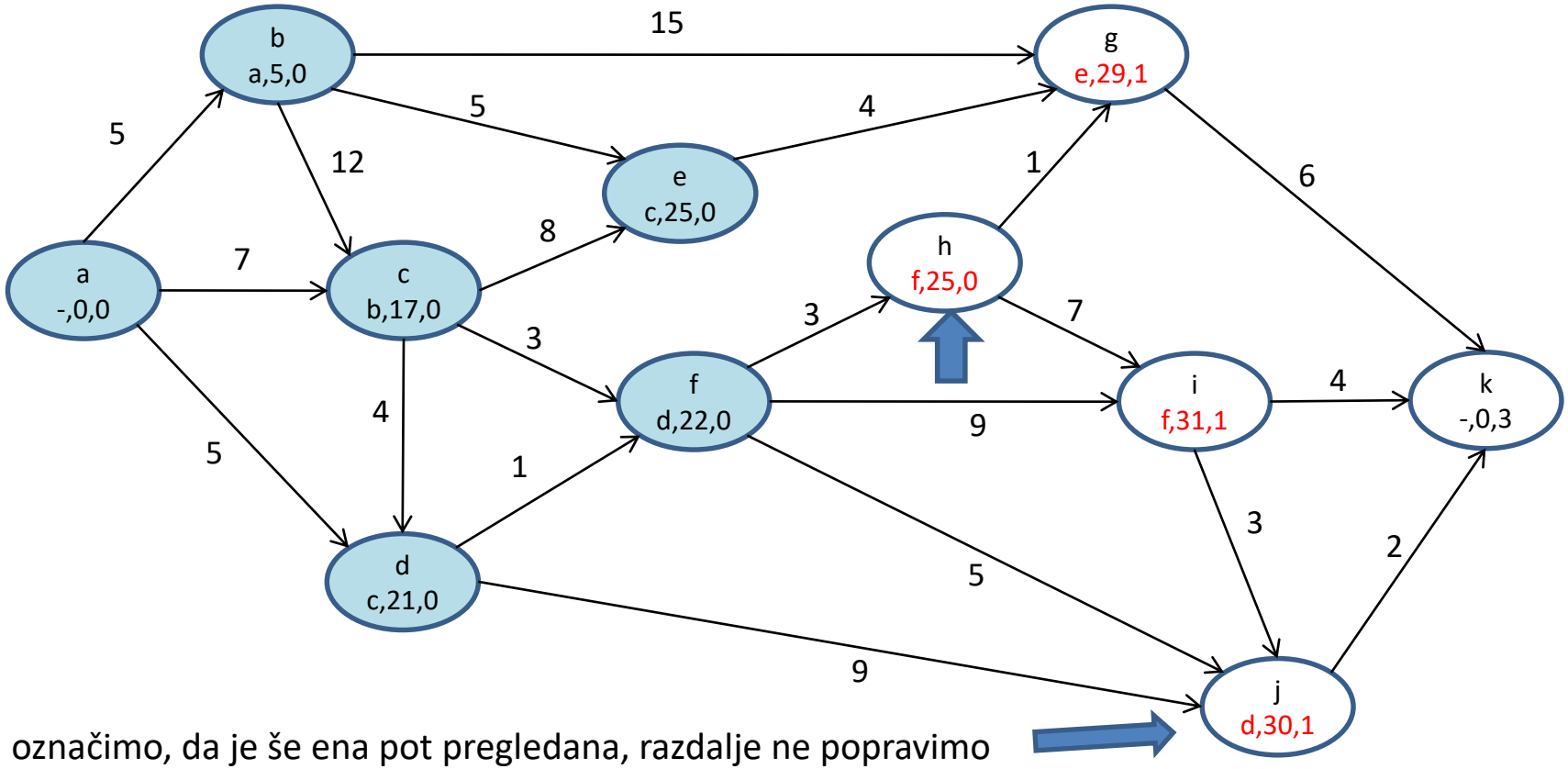
ANALIZA KRITIČNE POTI

Seznam vozlišč, katerih naslednikov še nismo pregledali:



ANALIZA KRITIČNE POTI

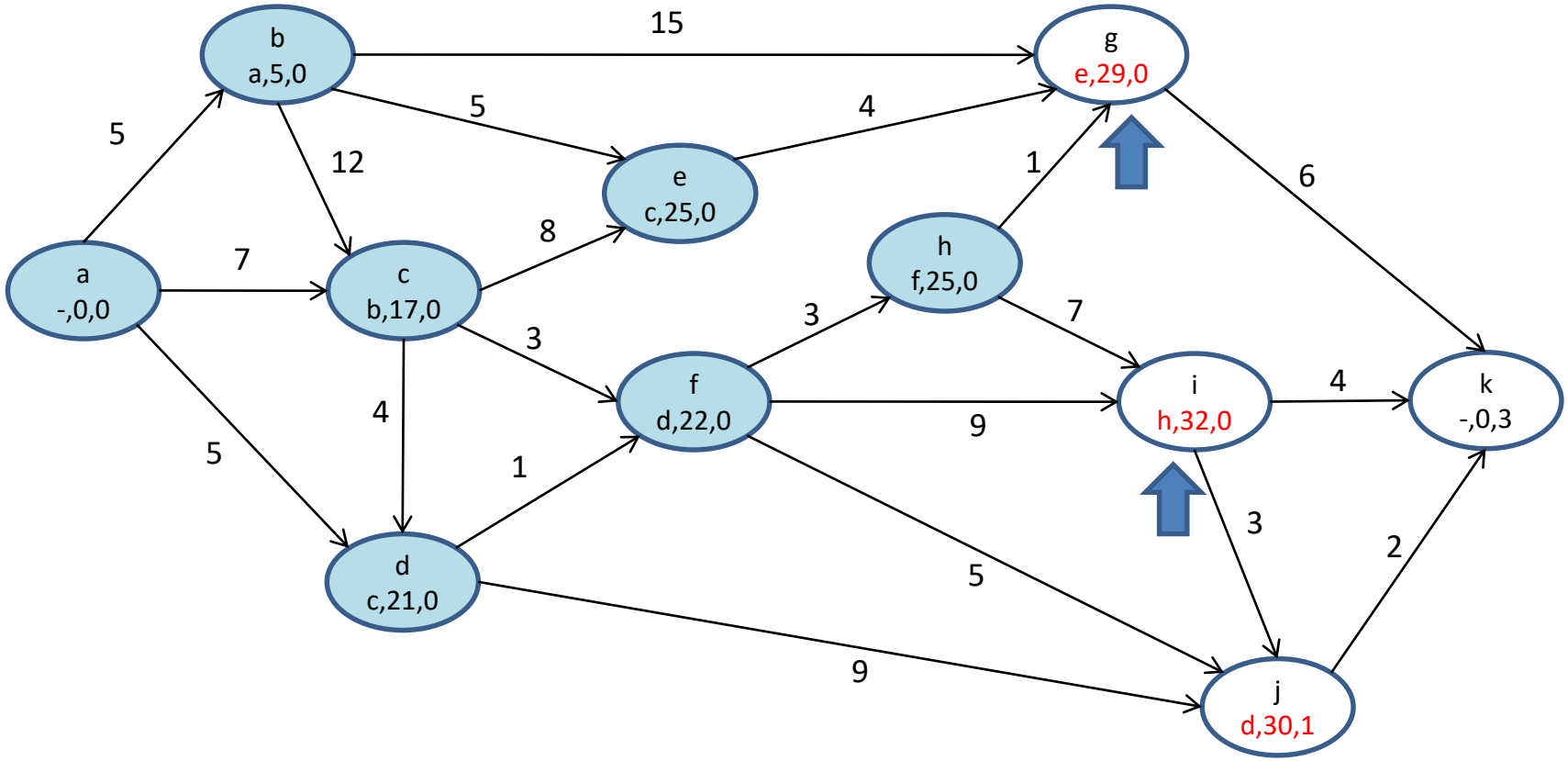
Seznam vozlišč, katerih naslednikov še nismo pregledali:



označimo, da je še ena pot pregledana, razdalje ne popravimo

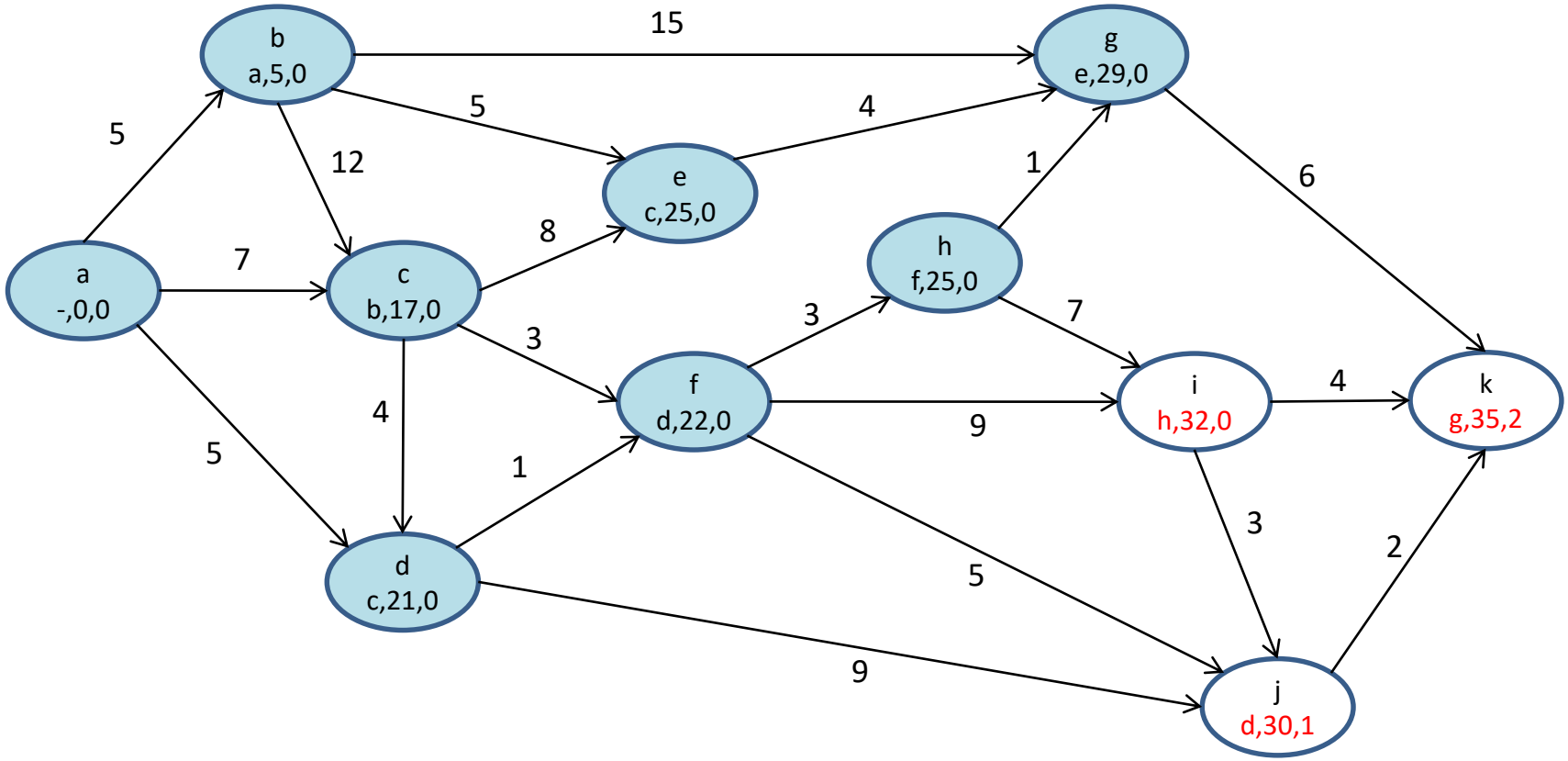
ANALIZA KRITIČNE POTI

Seznam vozlišč, katerih naslednikov še nismo pregledali:



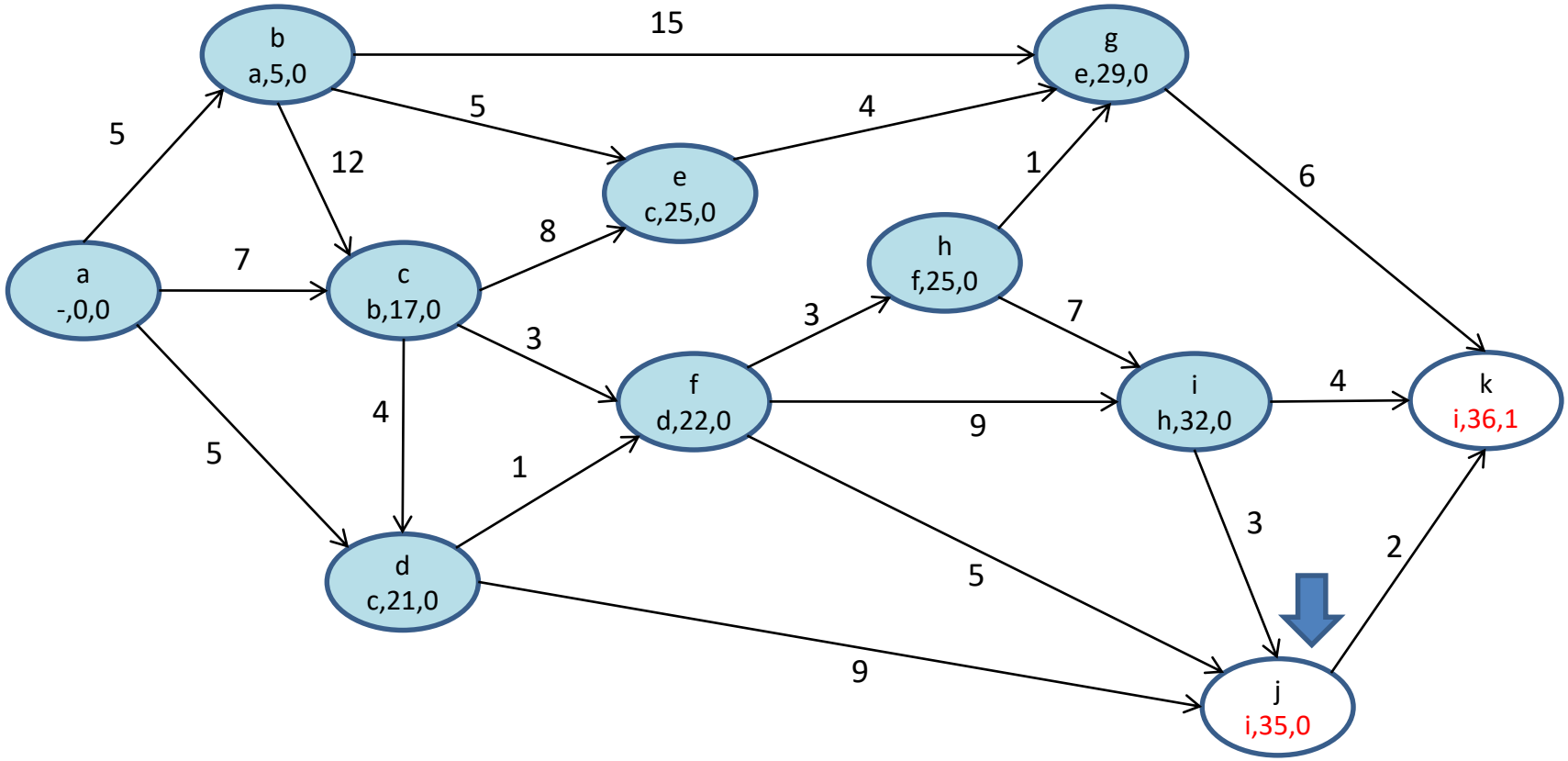
ANALIZA KRITIČNE POTI

Seznam vozlišč, katerih naslednikov še nismo pregledali:



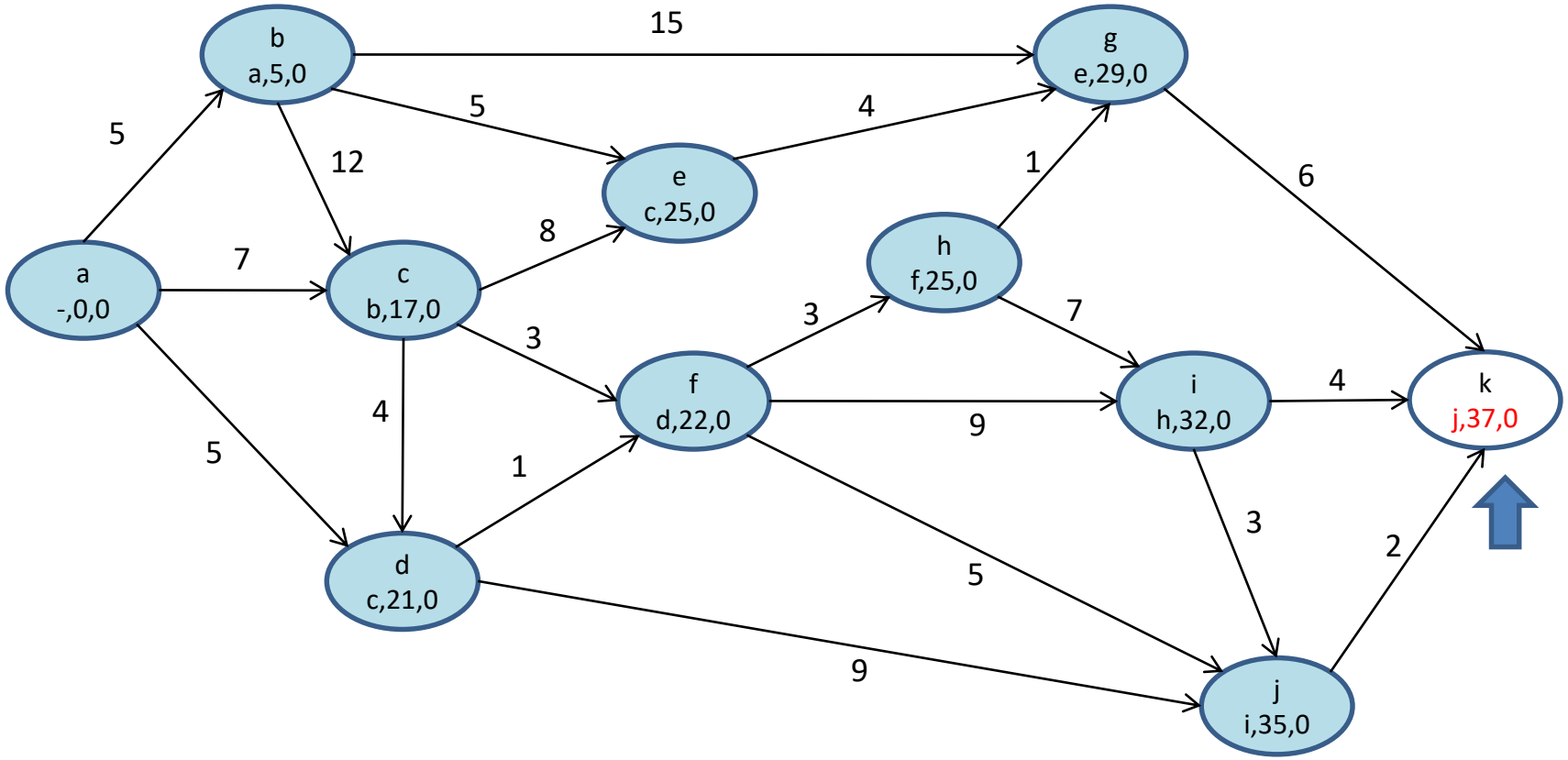
ANALIZA KRITIČNE POTI

Seznam vozlišč, katerih naslednikov še nismo pregledali:



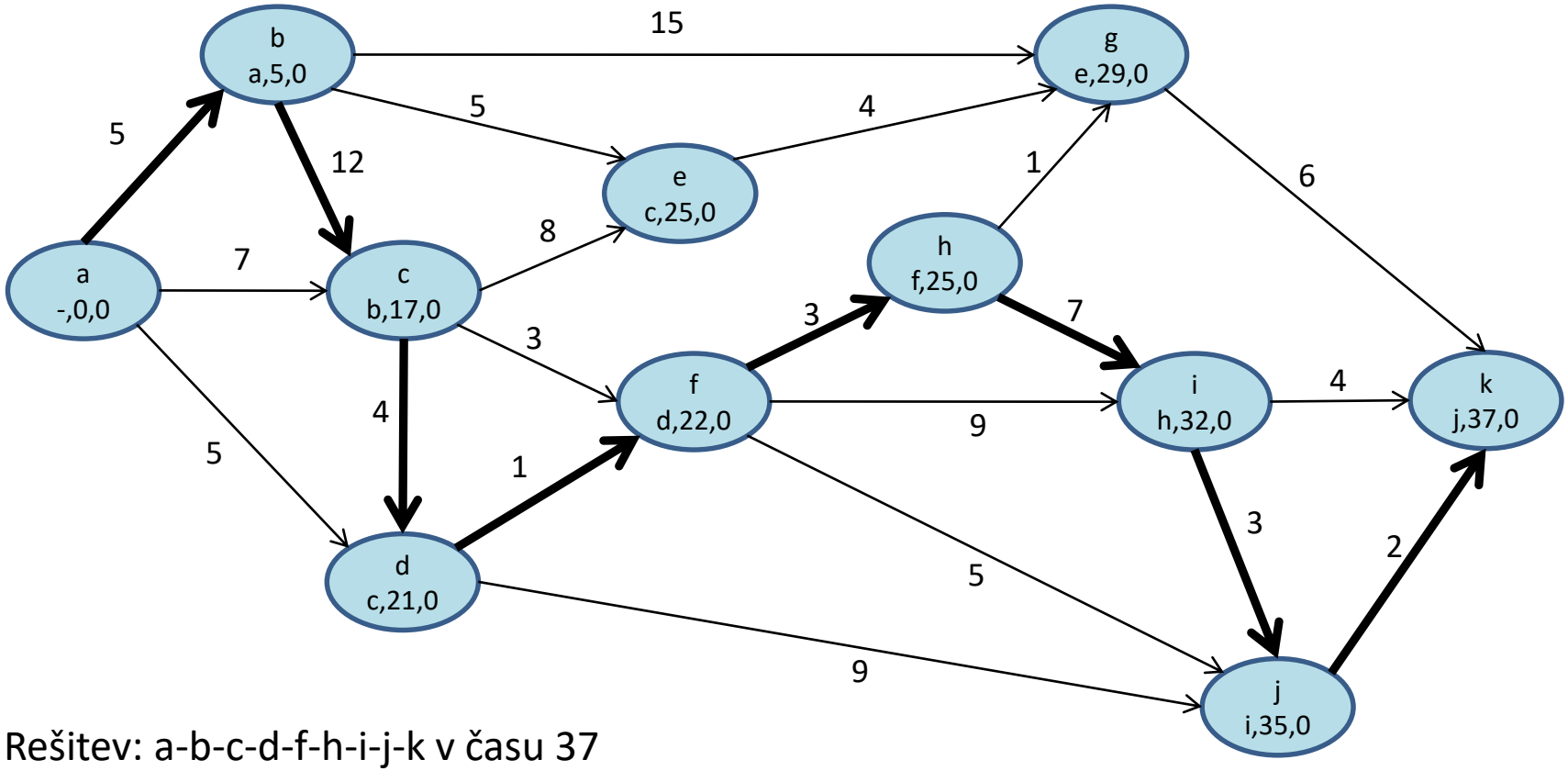
ANALIZA KRITIČNE POTI

Seznam vozlišč, katerih naslednikov še nismo pregledali:



ANALIZA KRITIČNE POTI

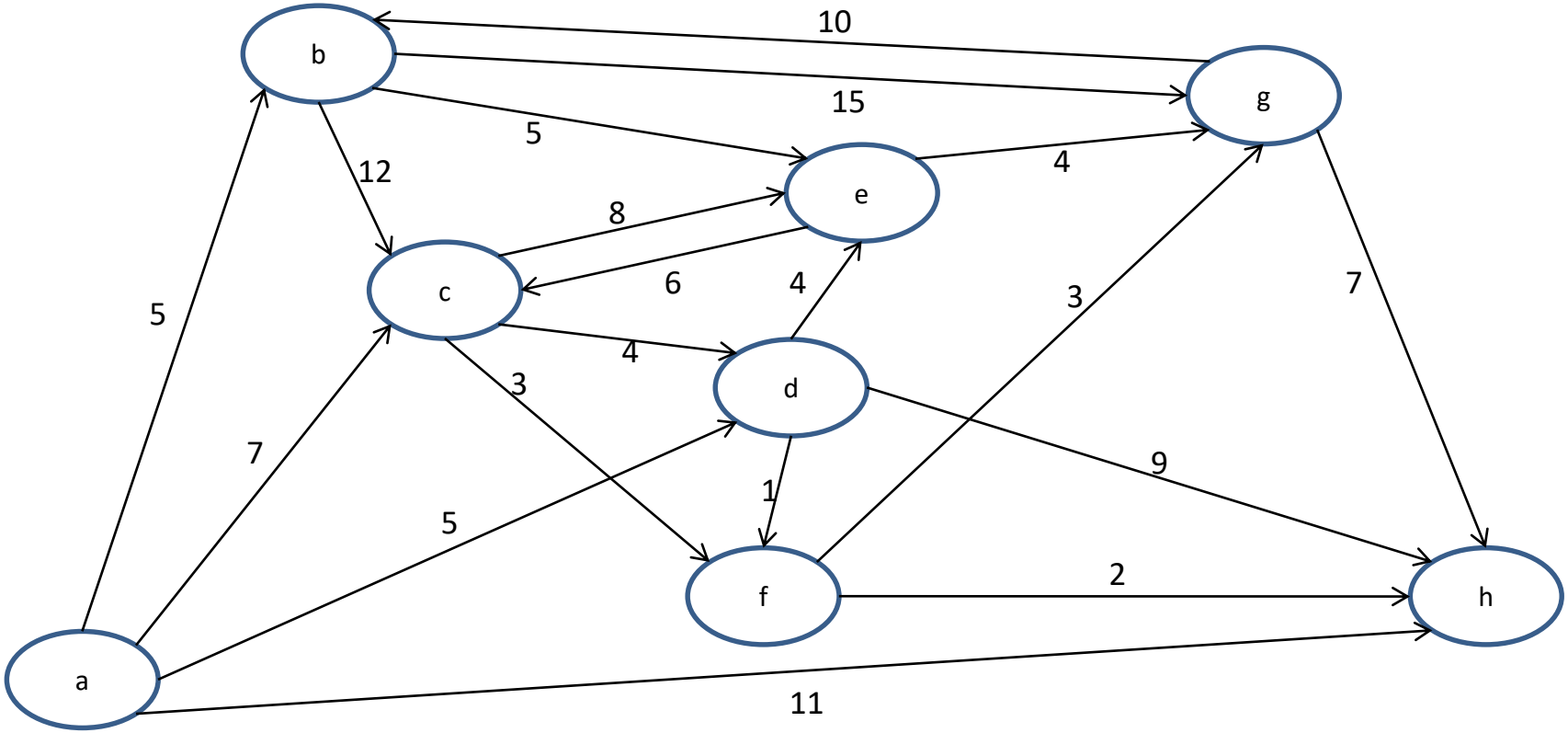
Seznam vozlišč, katerih naslednikov še nismo pregledali:



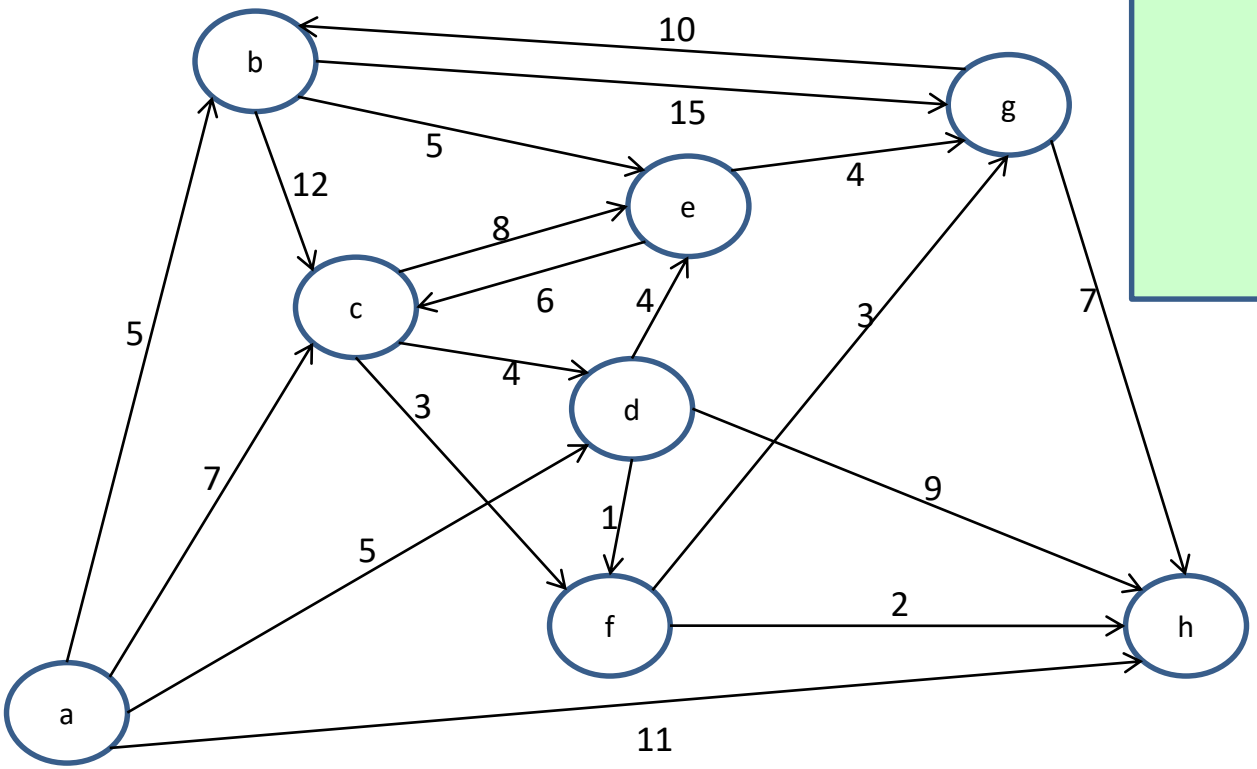
Rešitev: a-b-c-d-f-h-i-j-k v času 37

DIJKSTRA

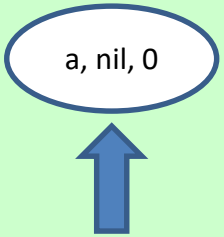
Podan je graf na sliki. Poiščite najkrajše poti od vozlišča ‘a’ do vseh ostalih vozlišč v grafu.



DIJKSTRA



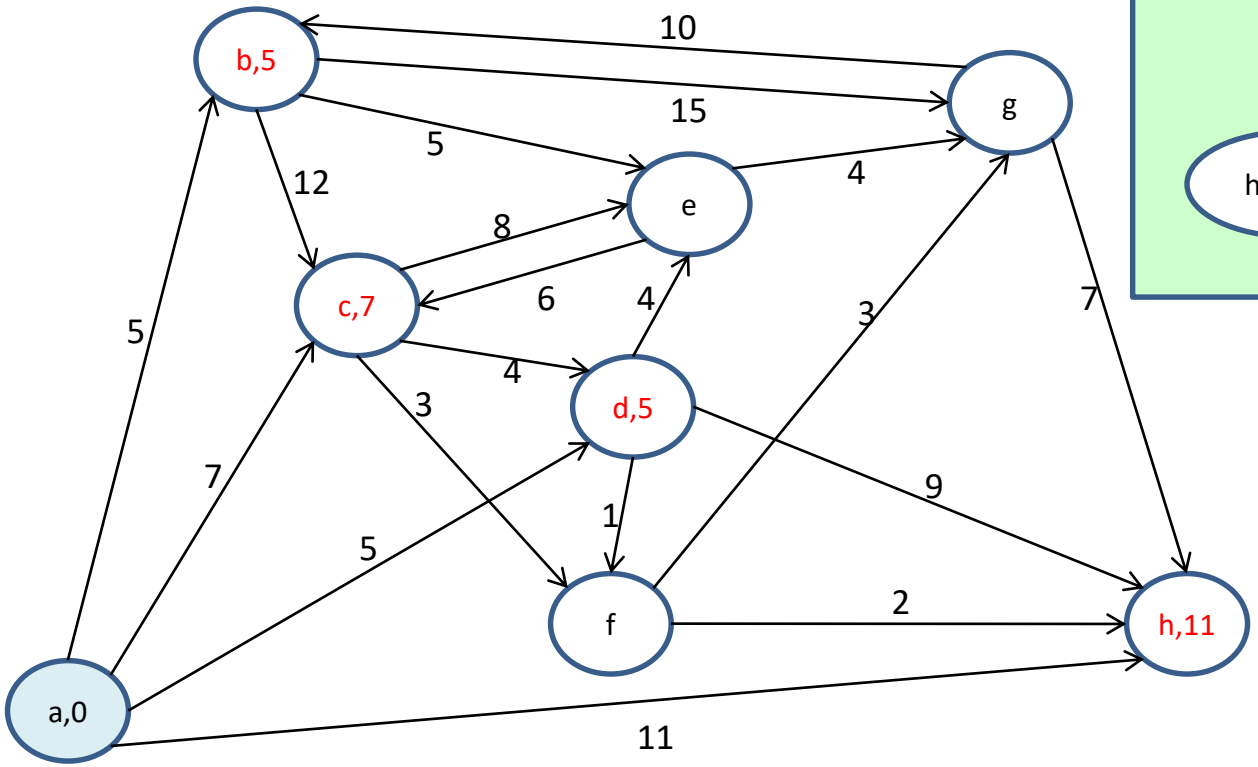
Prioritetna vrsta (kopica):



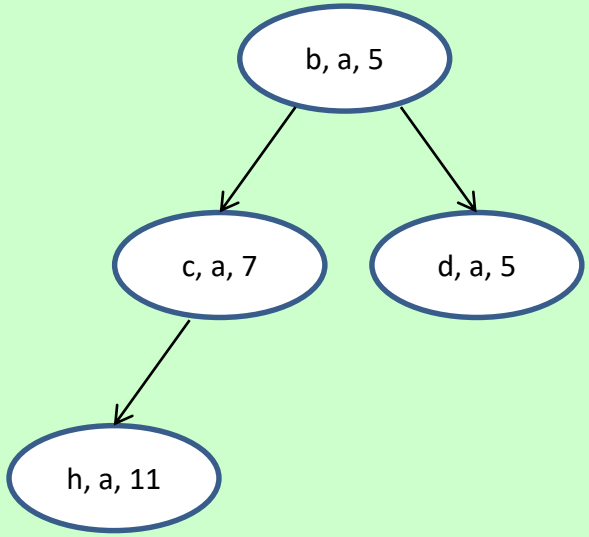
zapis vsebuje:

- ime vozlišča
- ime očeta v vpetem drevesu
- dolžino najkrajše znane poti od začetnega vozlišča

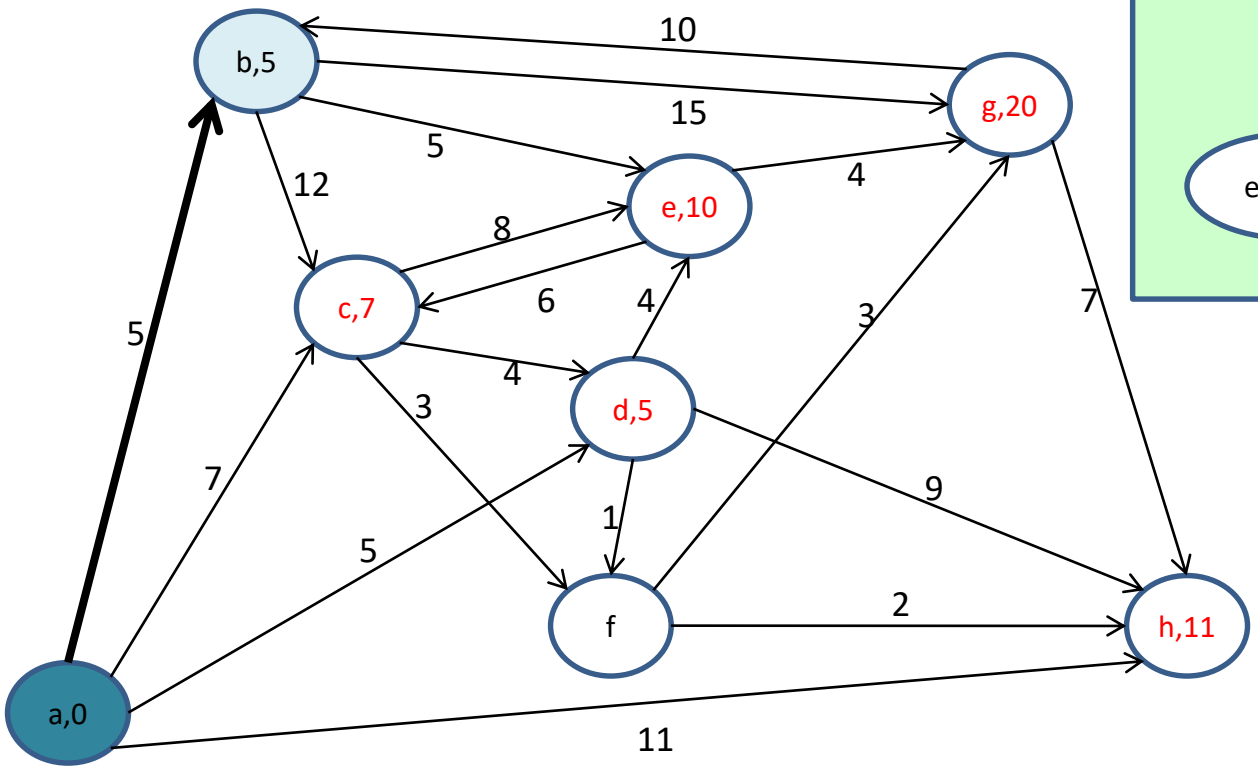
DIJKSTRA



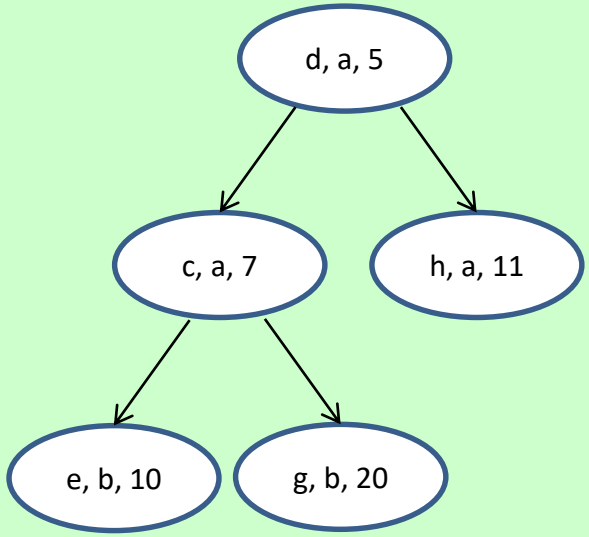
Prioritetna vrsta (kopica):



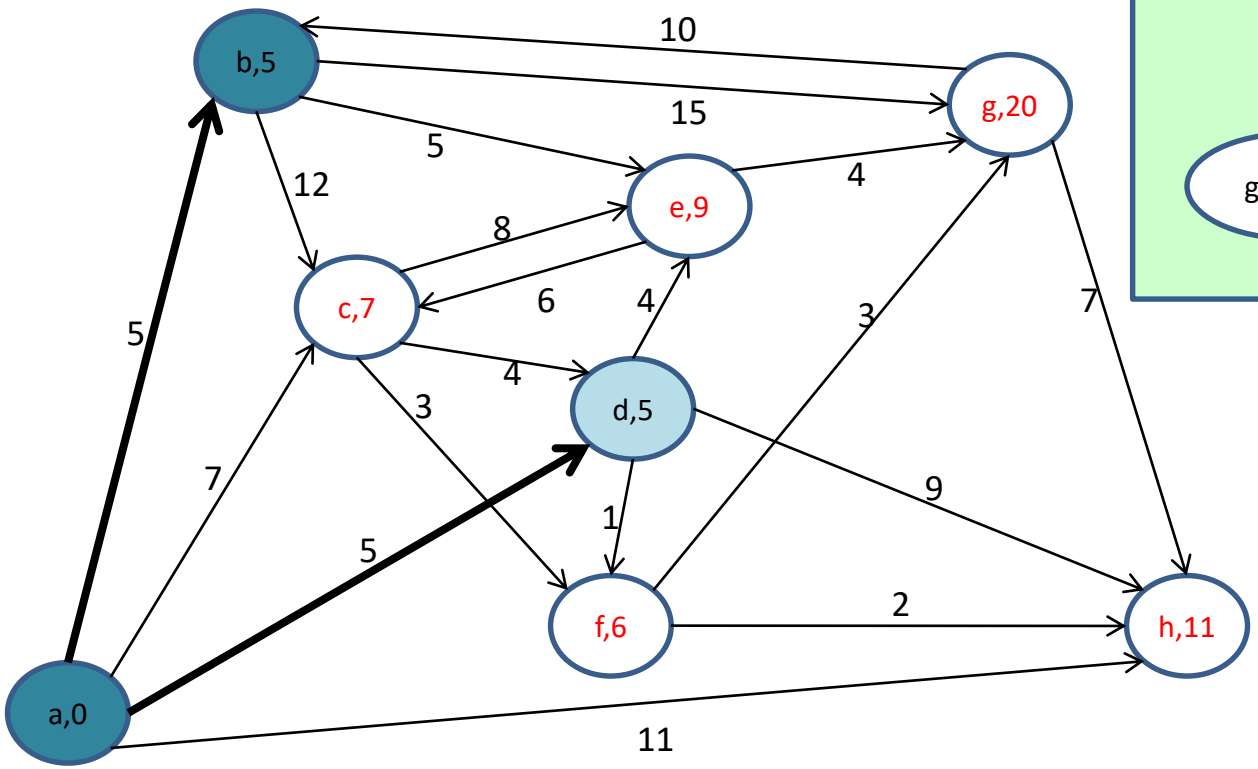
DIJKSTRA



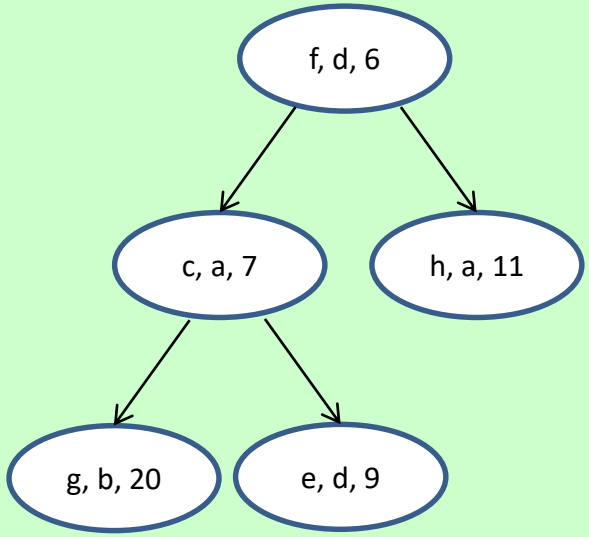
Prioritetna vrsta (kopica):



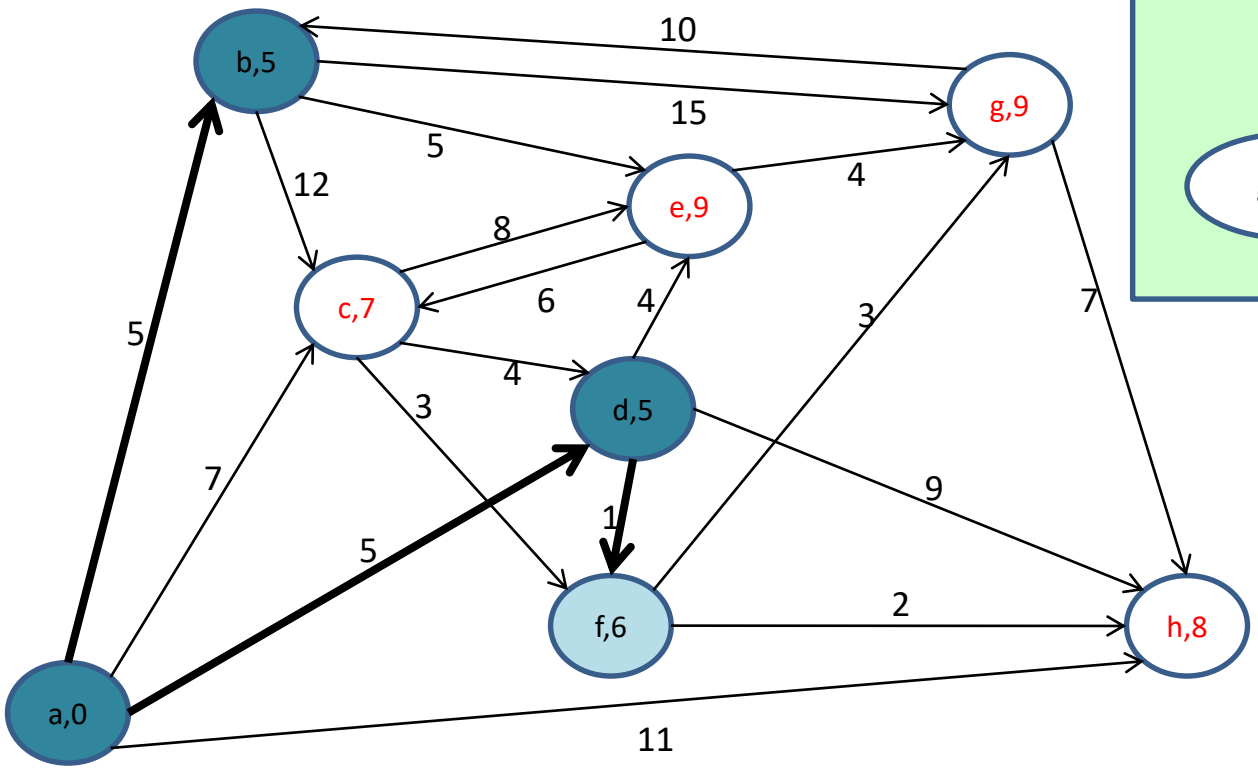
DIJKSTRA



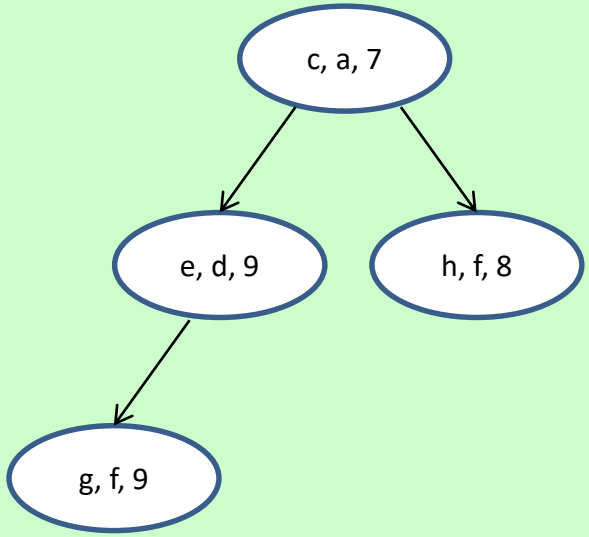
Prioritetna vrsta (kopica):



DIJKSTRA

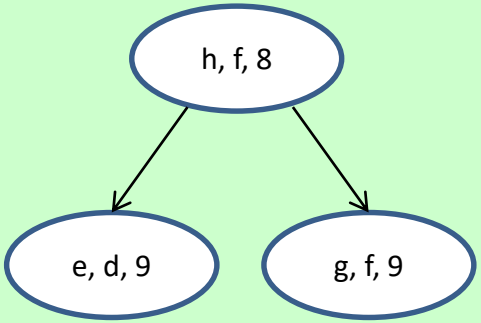


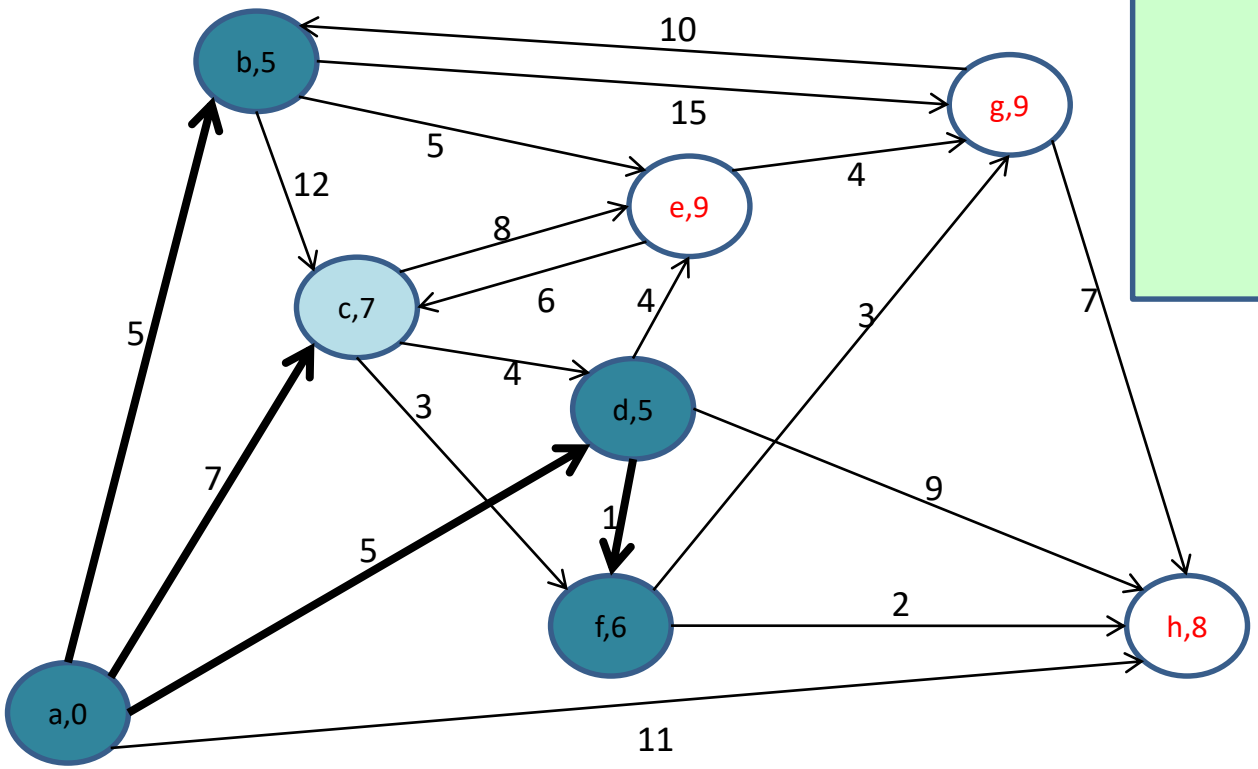
Prioritetna vrsta (kopica):



DIJKSTRA

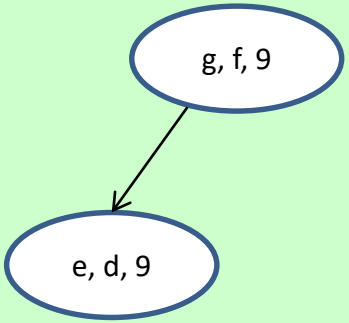
Prioritetna vrsta (kopica):

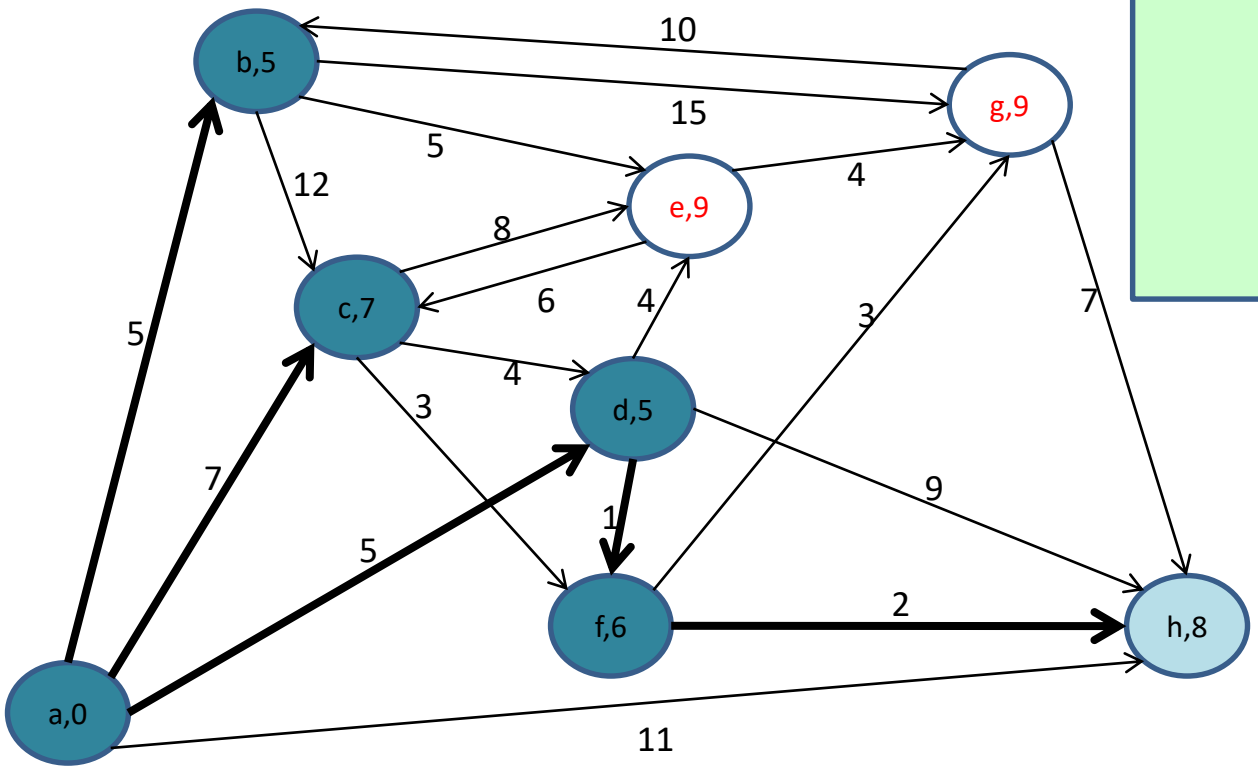




DIJKSTRA

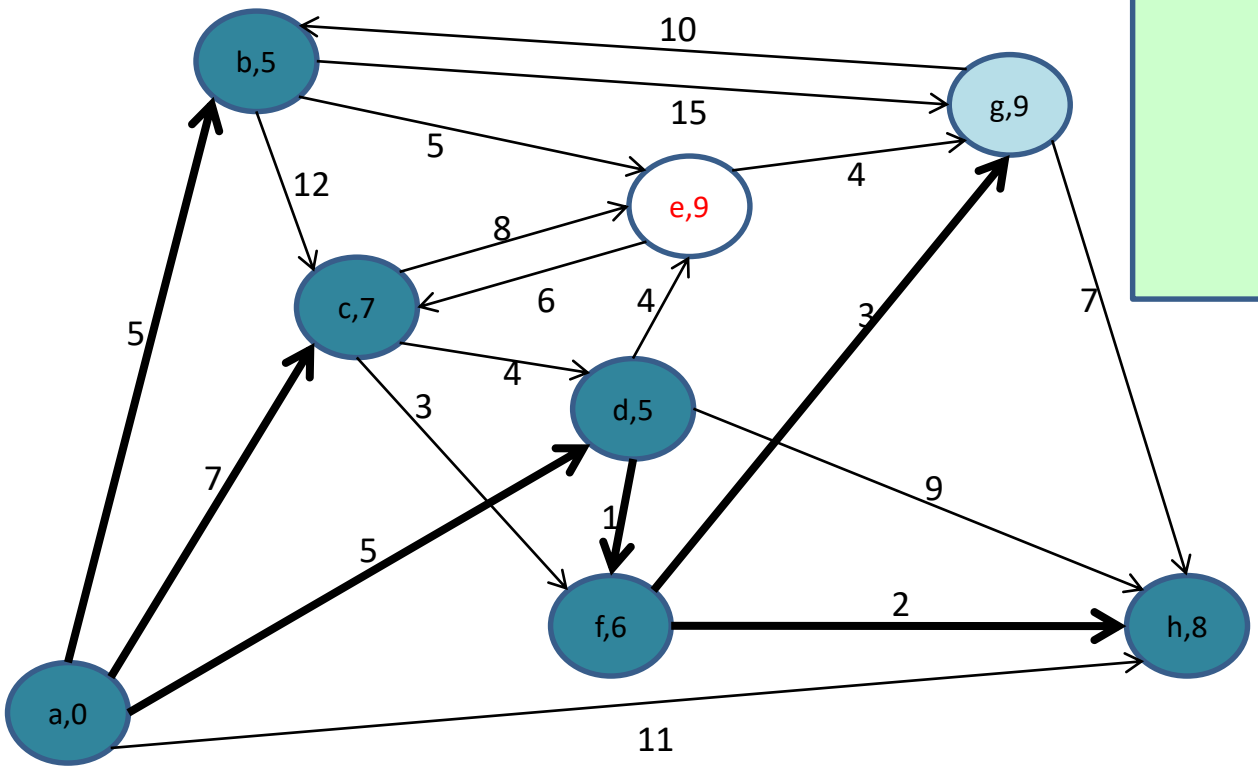
Prioritetna vrsta (kopica):





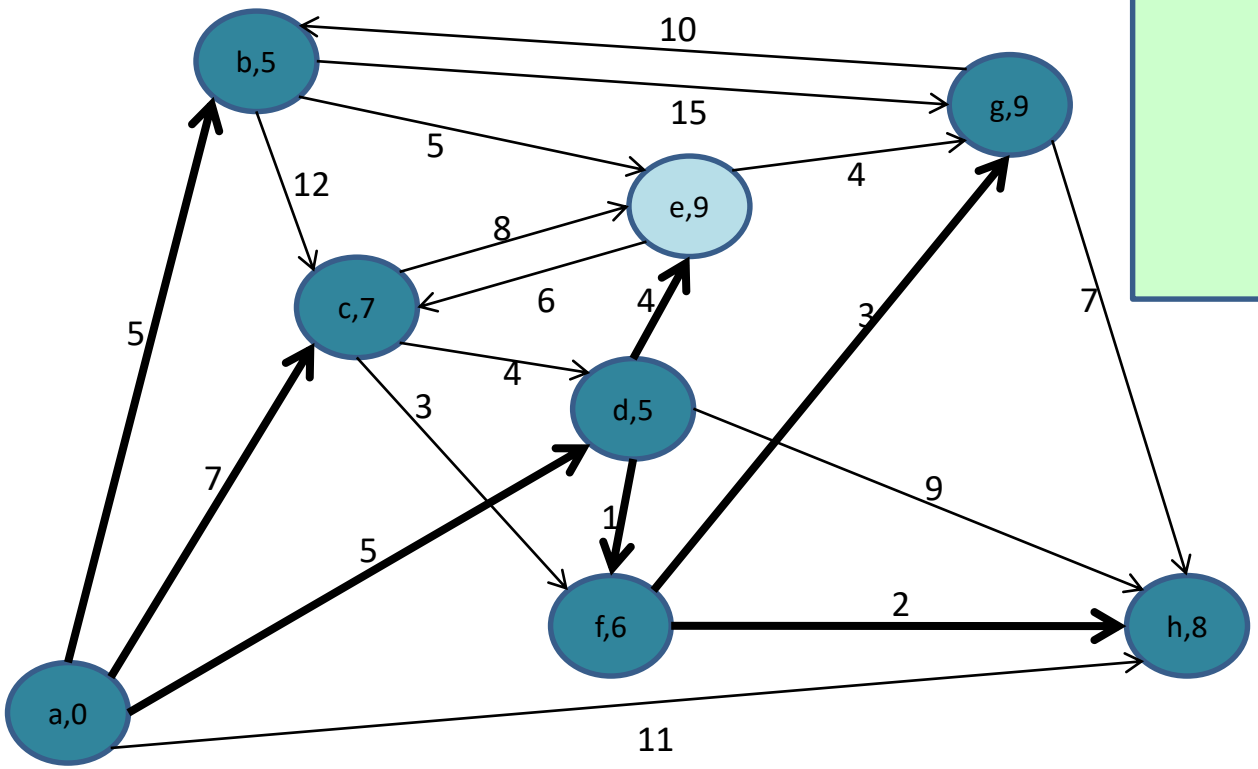
DIJKSTRA

Prioritetna vrsta (kopica):



DIJKSTRA

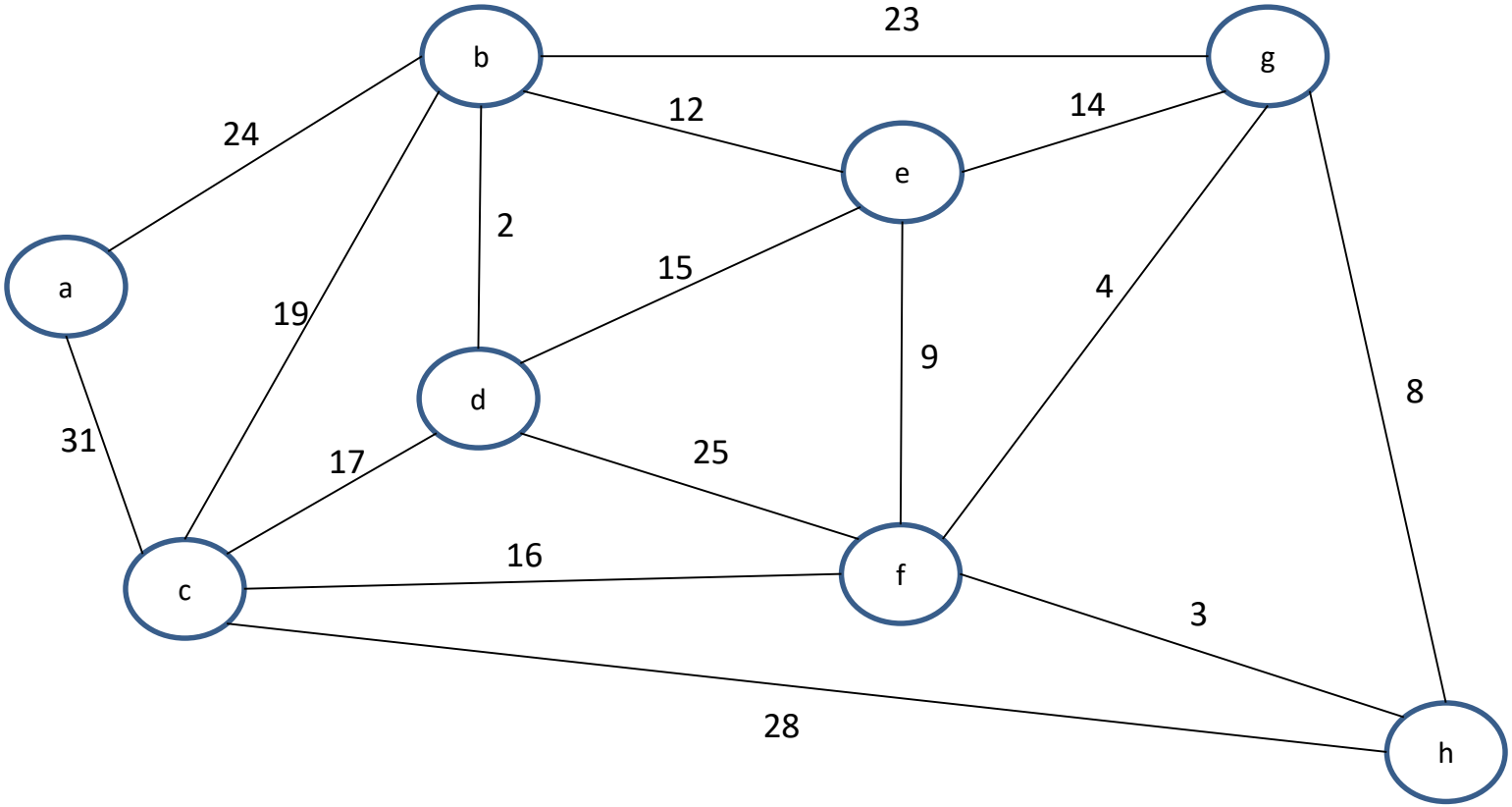
Prioritetna vrsta (kopica):



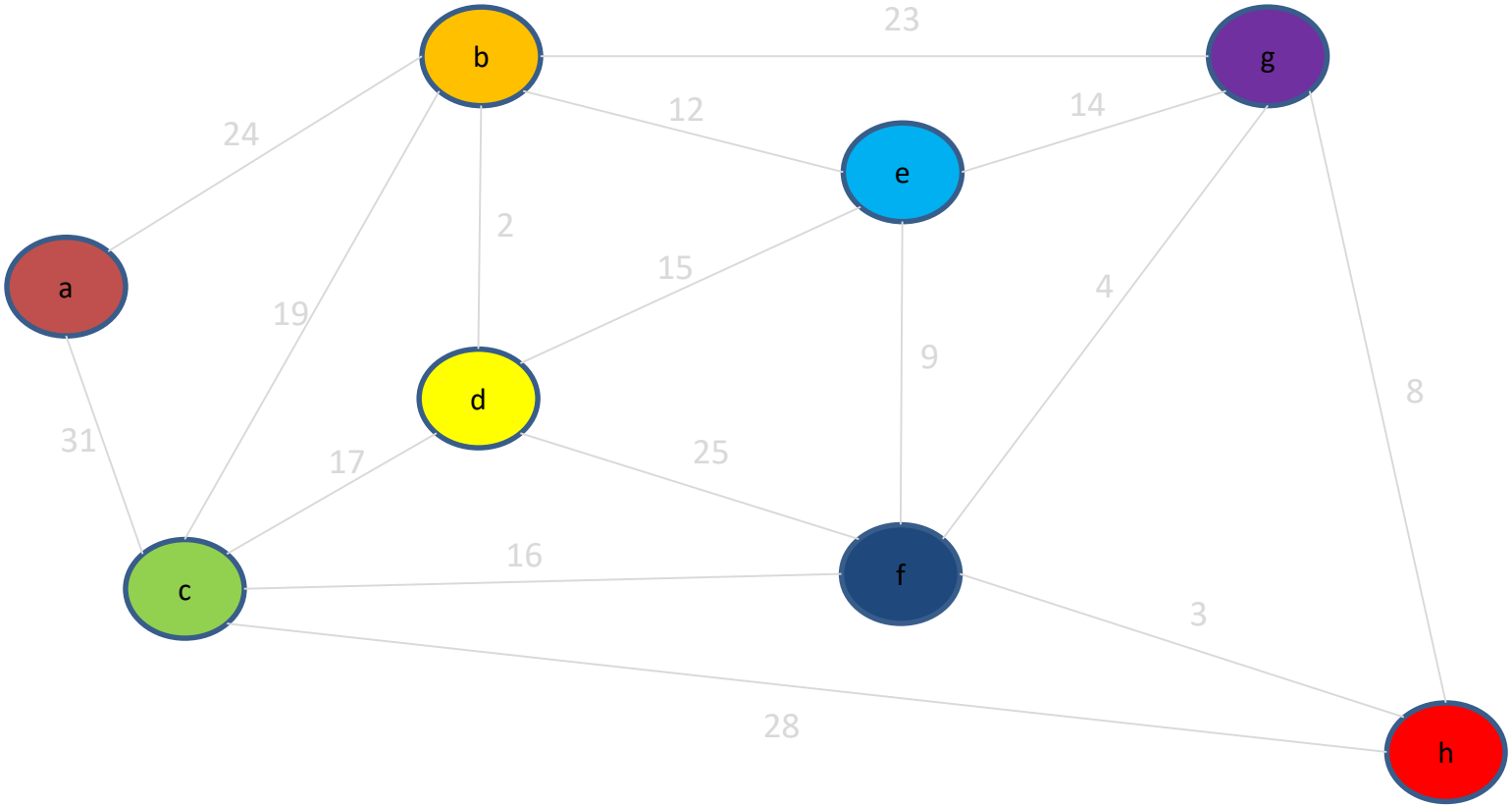
prioritetna vrsta je prazna kar pomeni, da je postopek končan

MINIMALNO VPETO DREVO

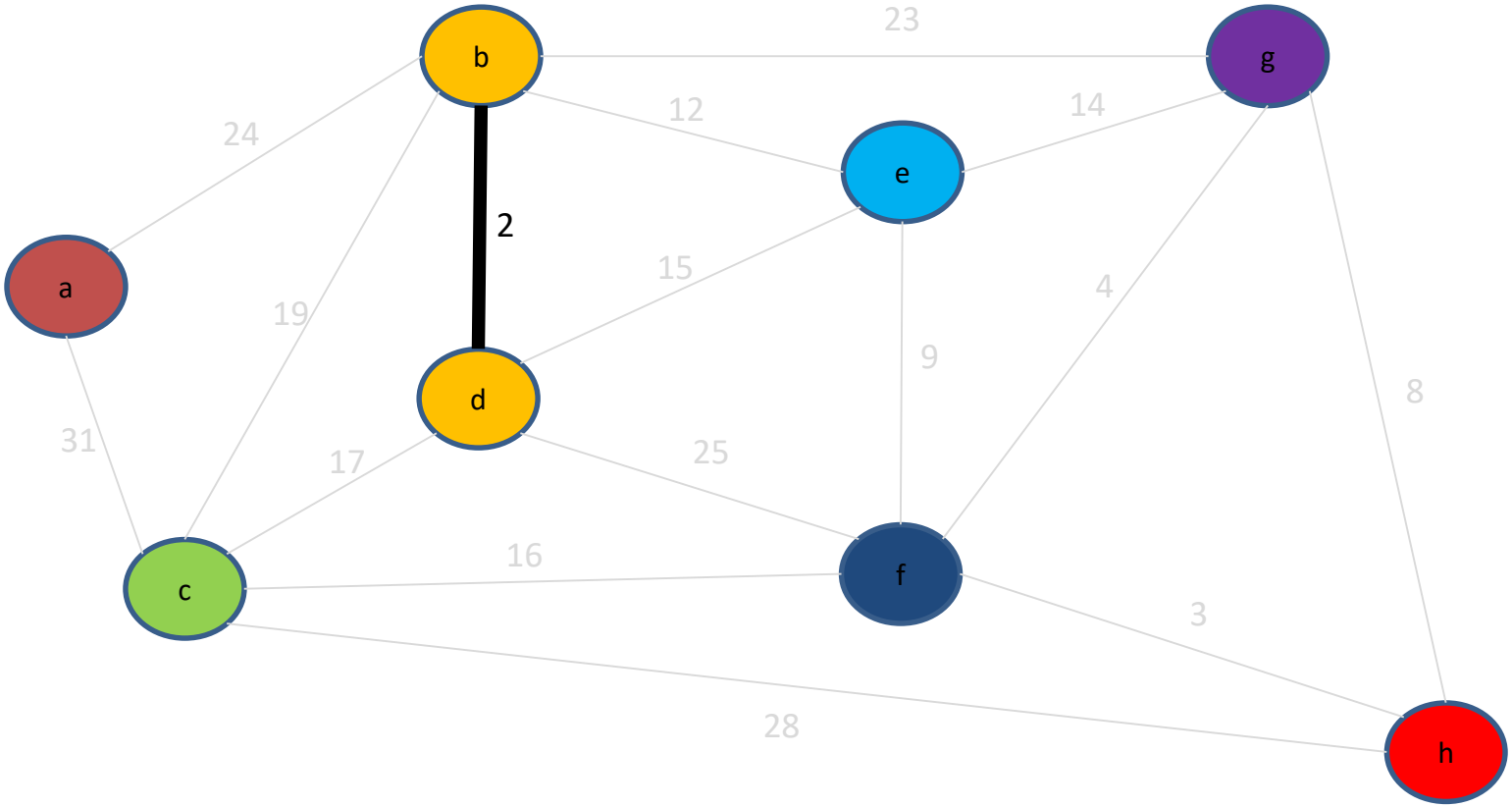
Za graf na sliki poiščite minimalno vpeto drevo z izvajanjem Kruskalovega algoritma.



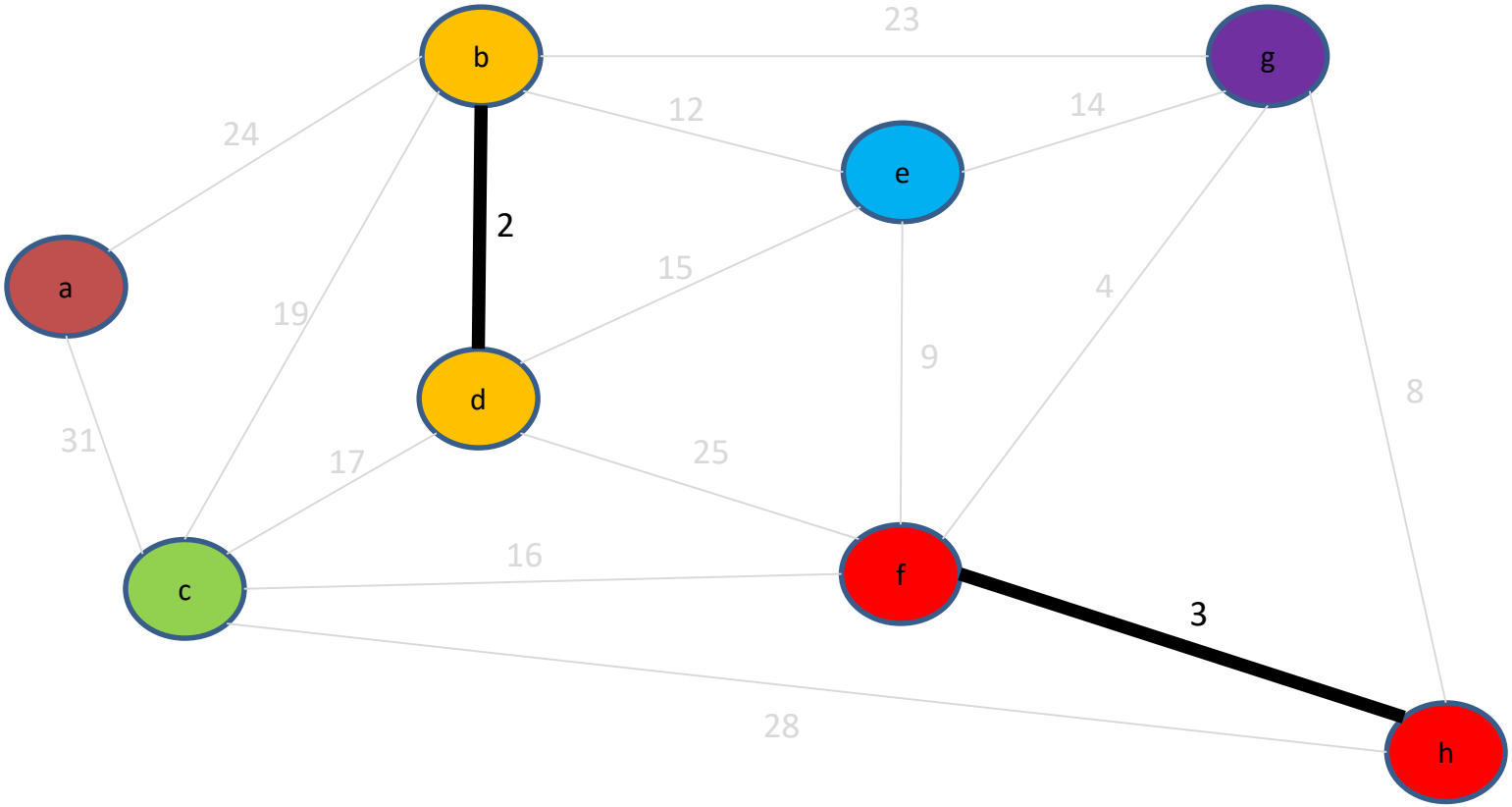
MINIMALNO VPETO DREVO



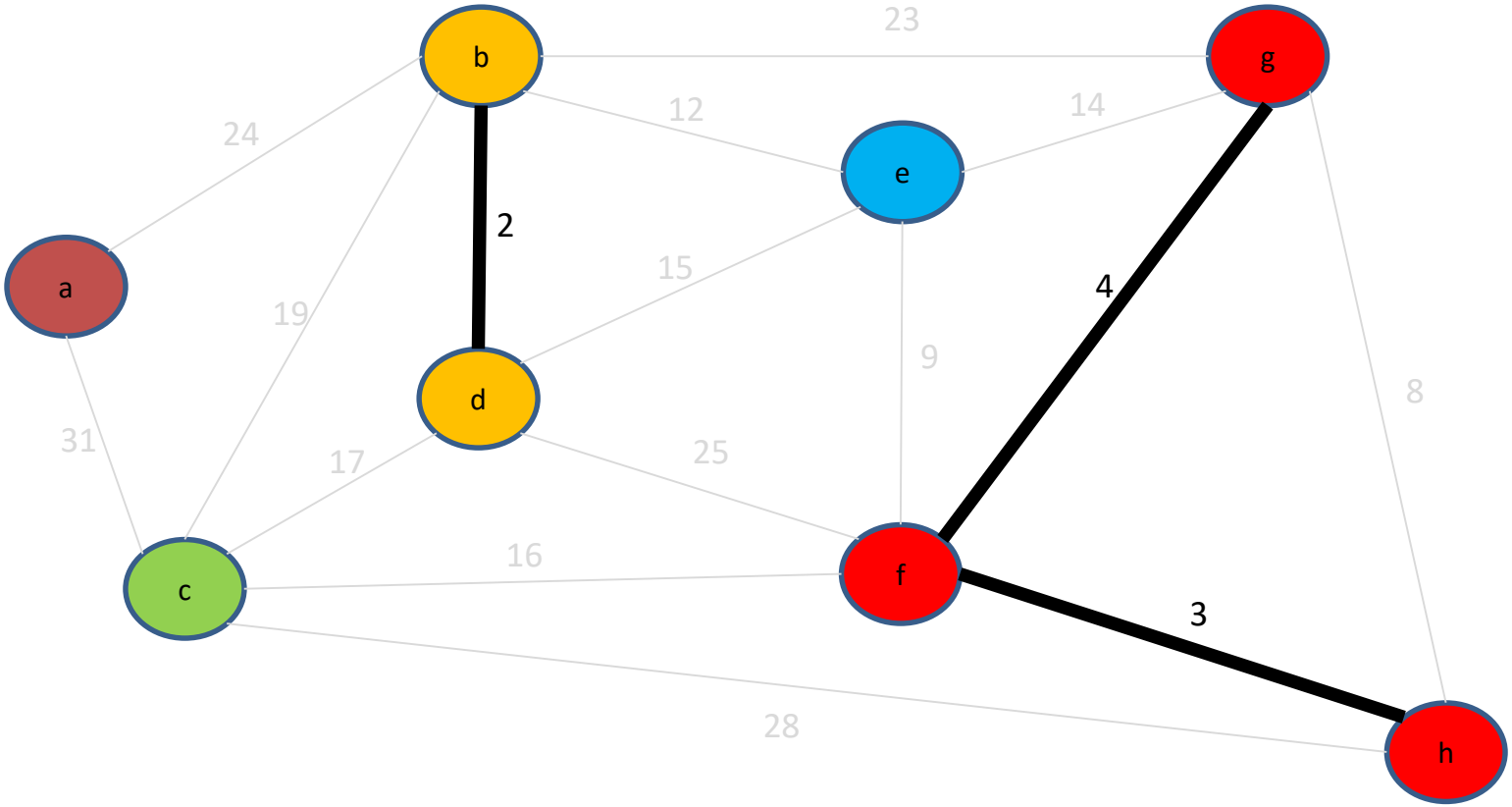
MINIMALNO VPETO DREVO



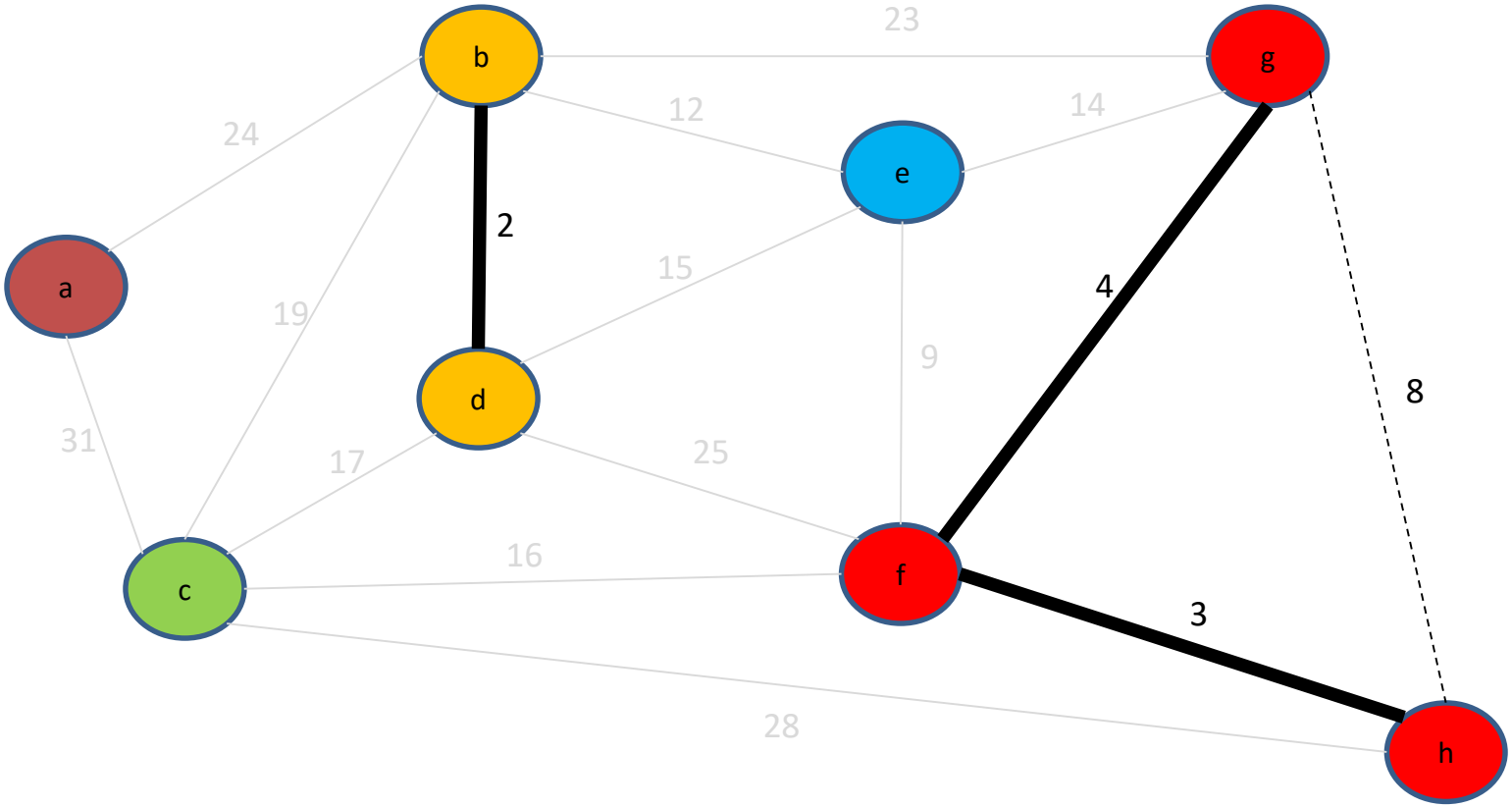
MINIMALNO VPETO DREVO



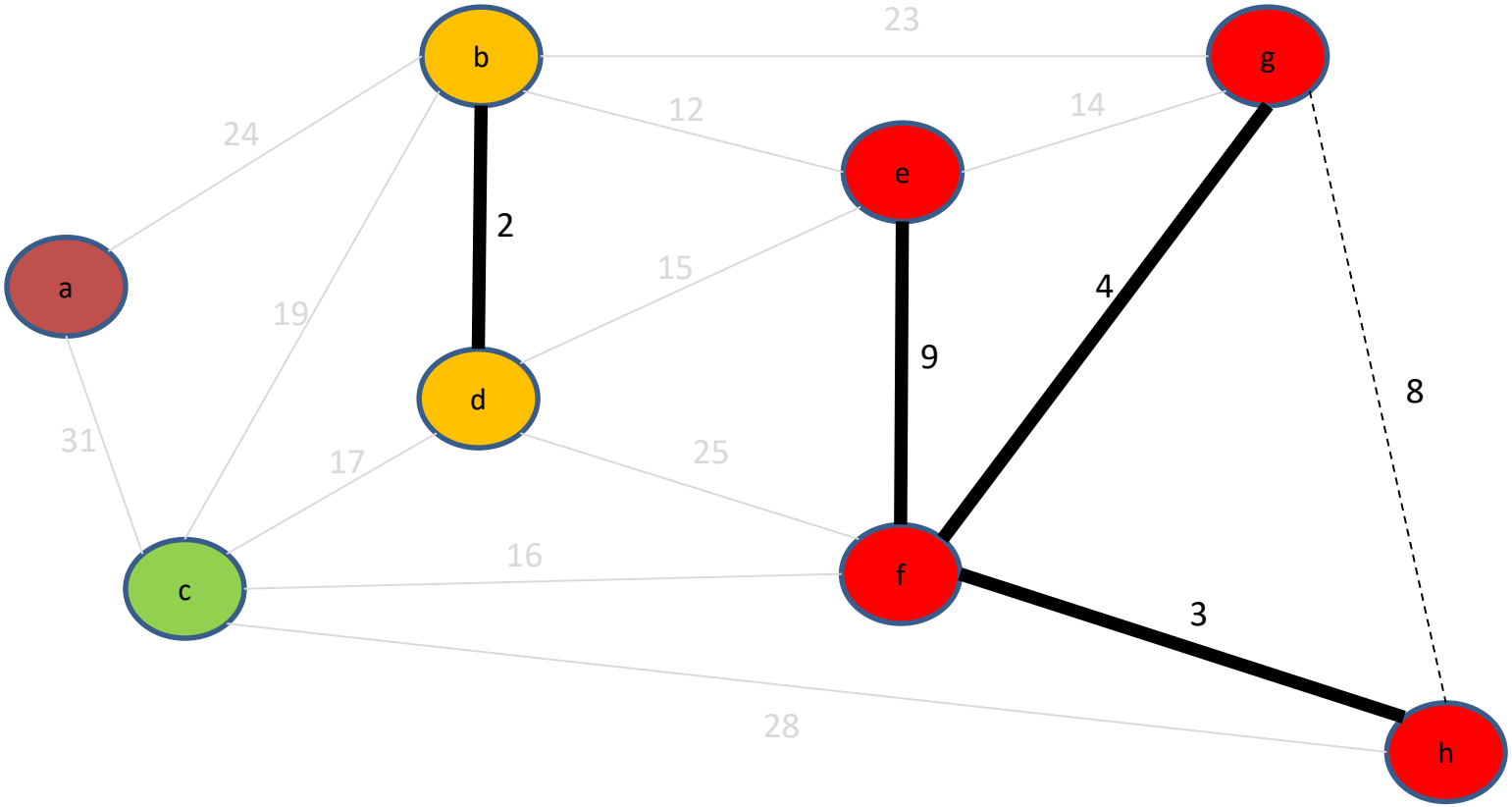
MINIMALNO VPETO DREVO



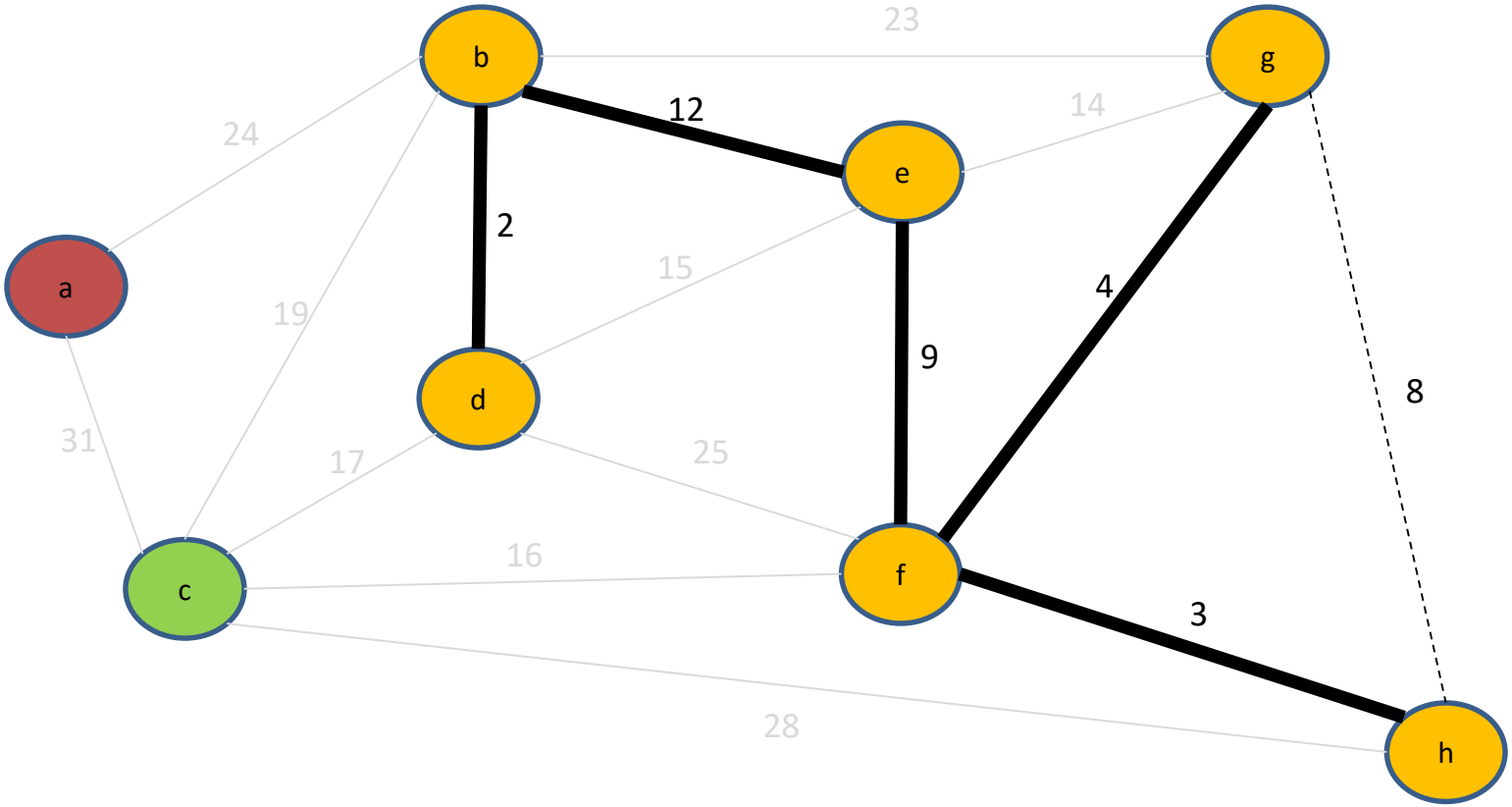
MINIMALNO VPETO DREVO



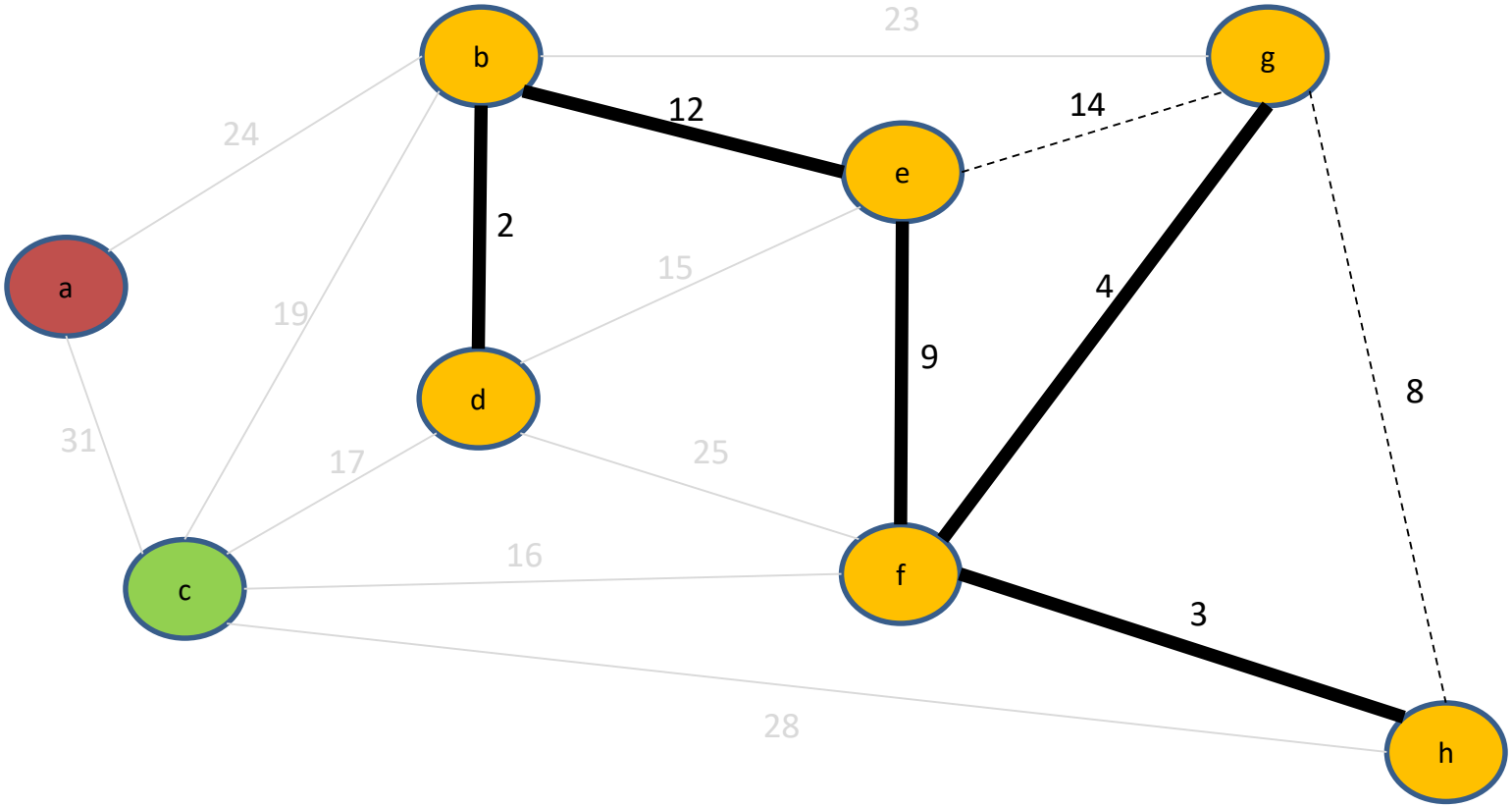
MINIMALNO VPETO DREVO



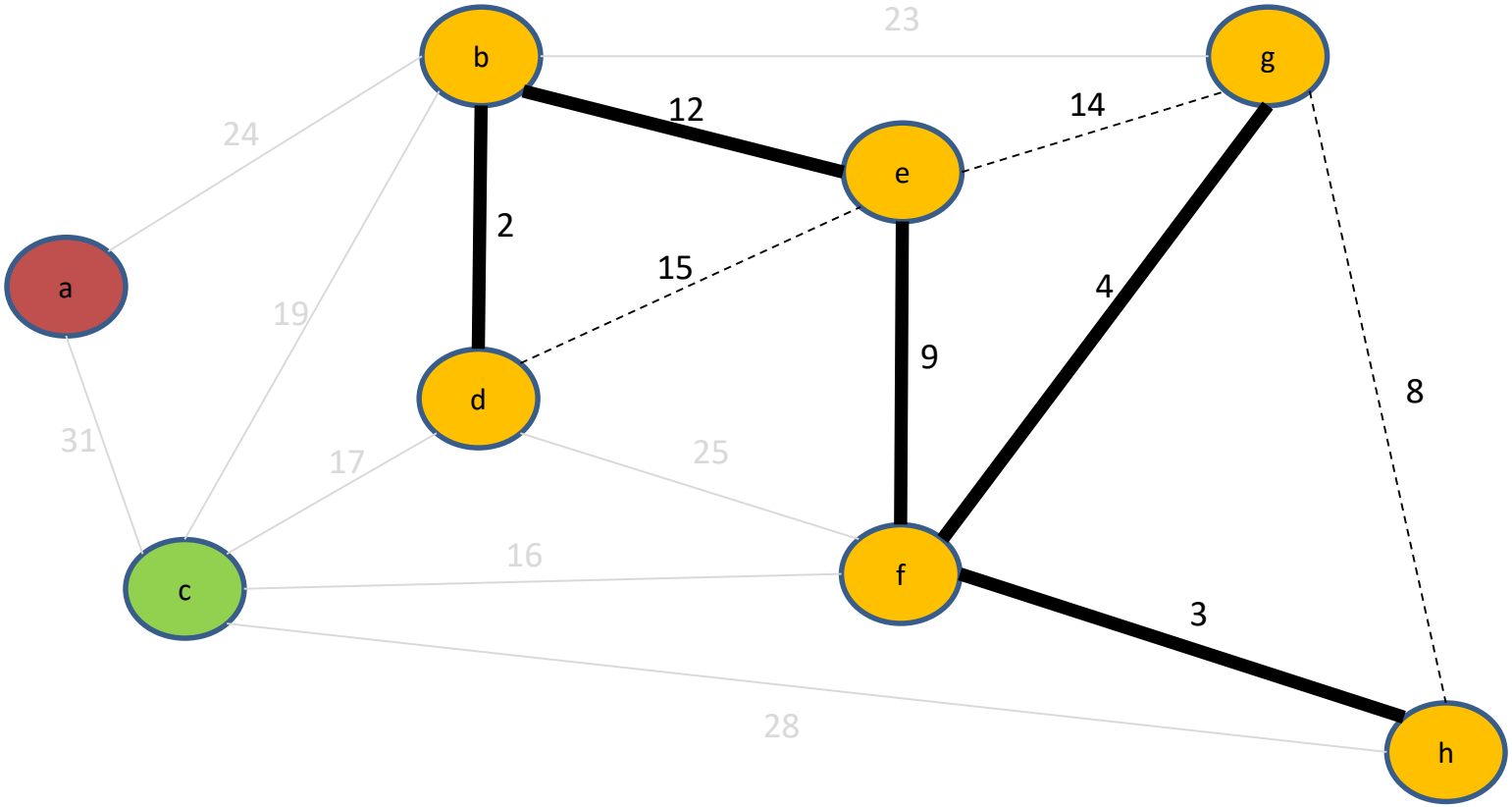
MINIMALNO VPETO DREVO



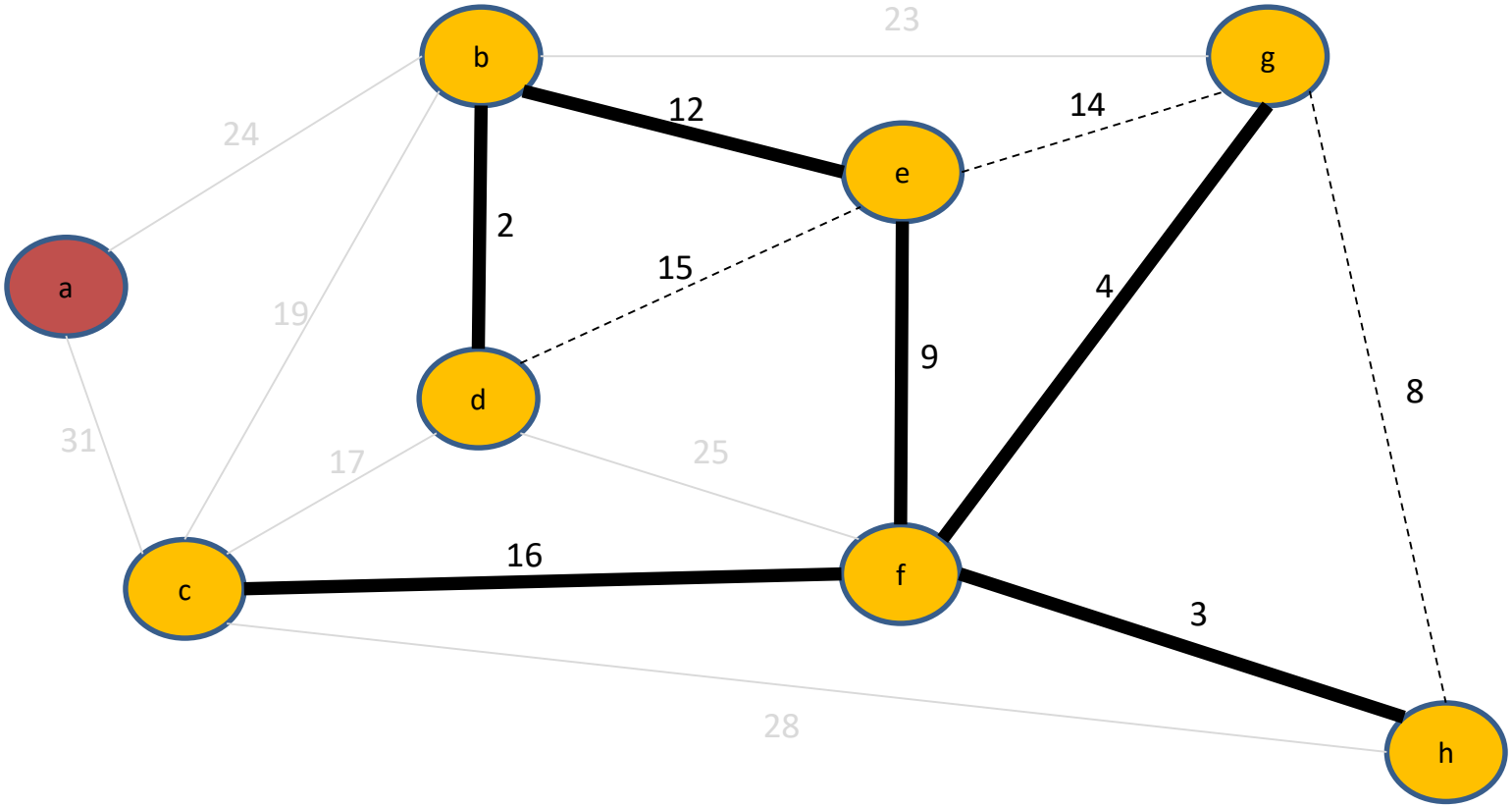
MINIMALNO VPETO DREVO



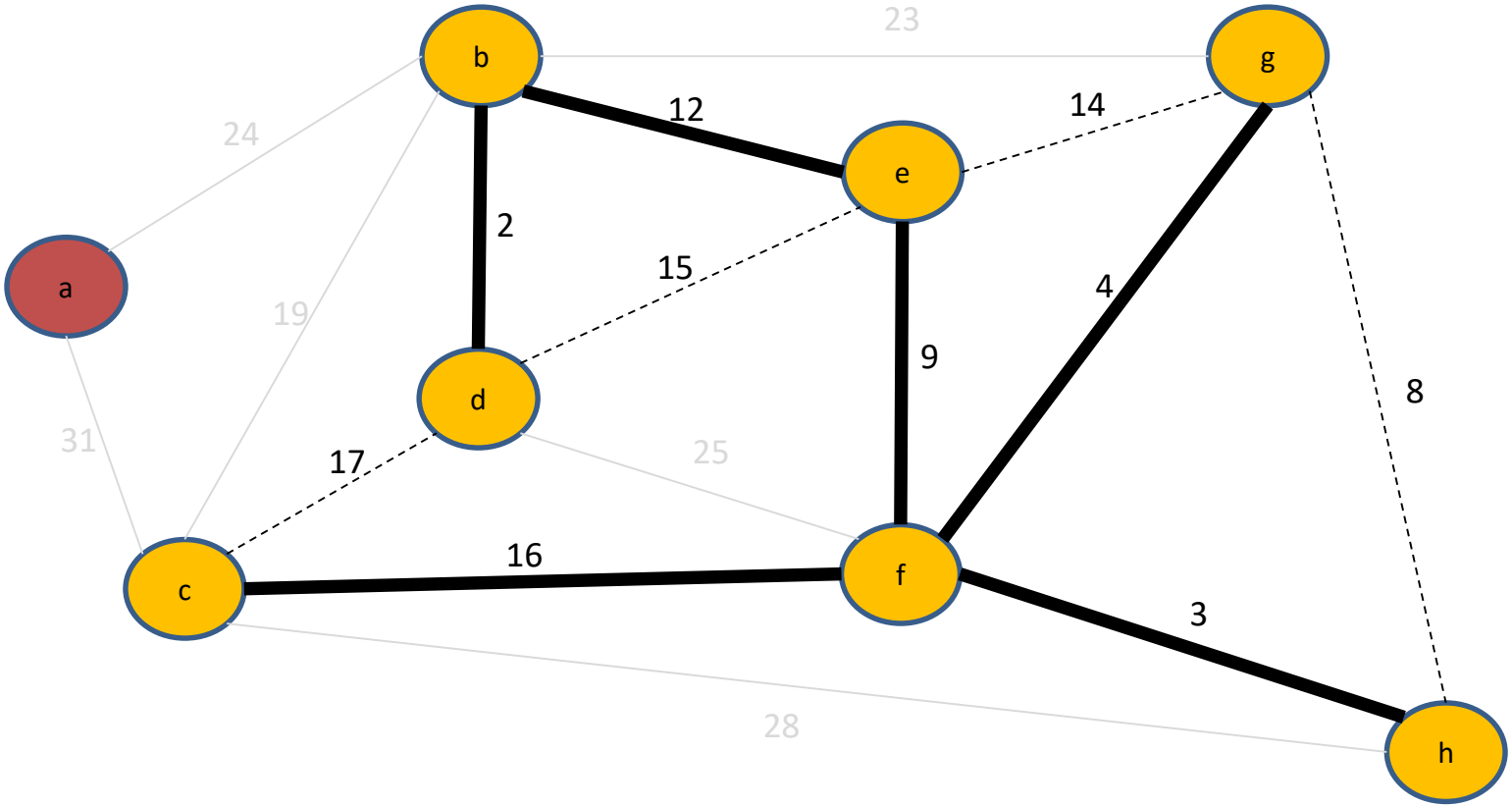
MINIMALNO VPETO DREVO



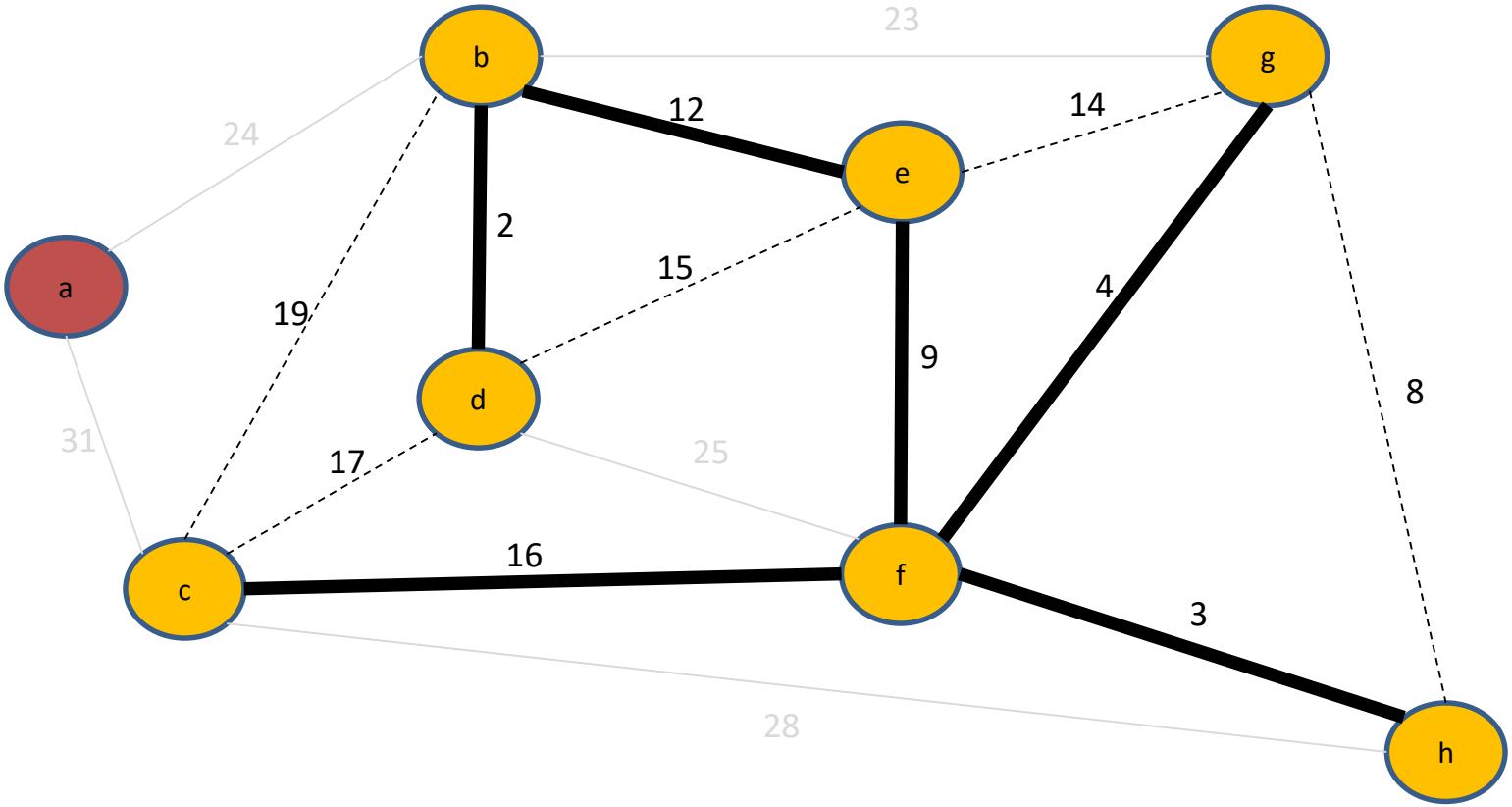
MINIMALNO VPETO DREVO



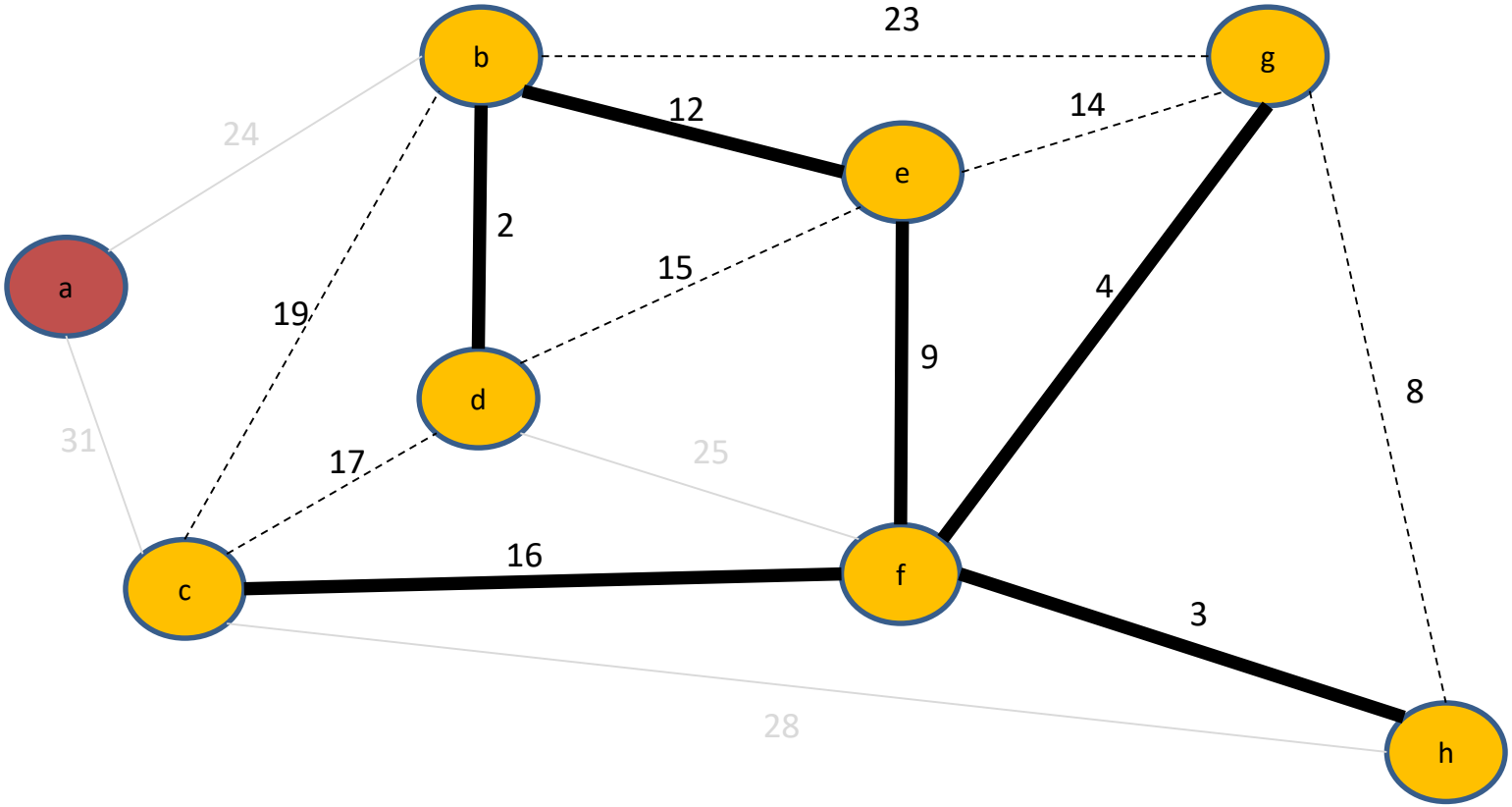
MINIMALNO VPETO DREVO



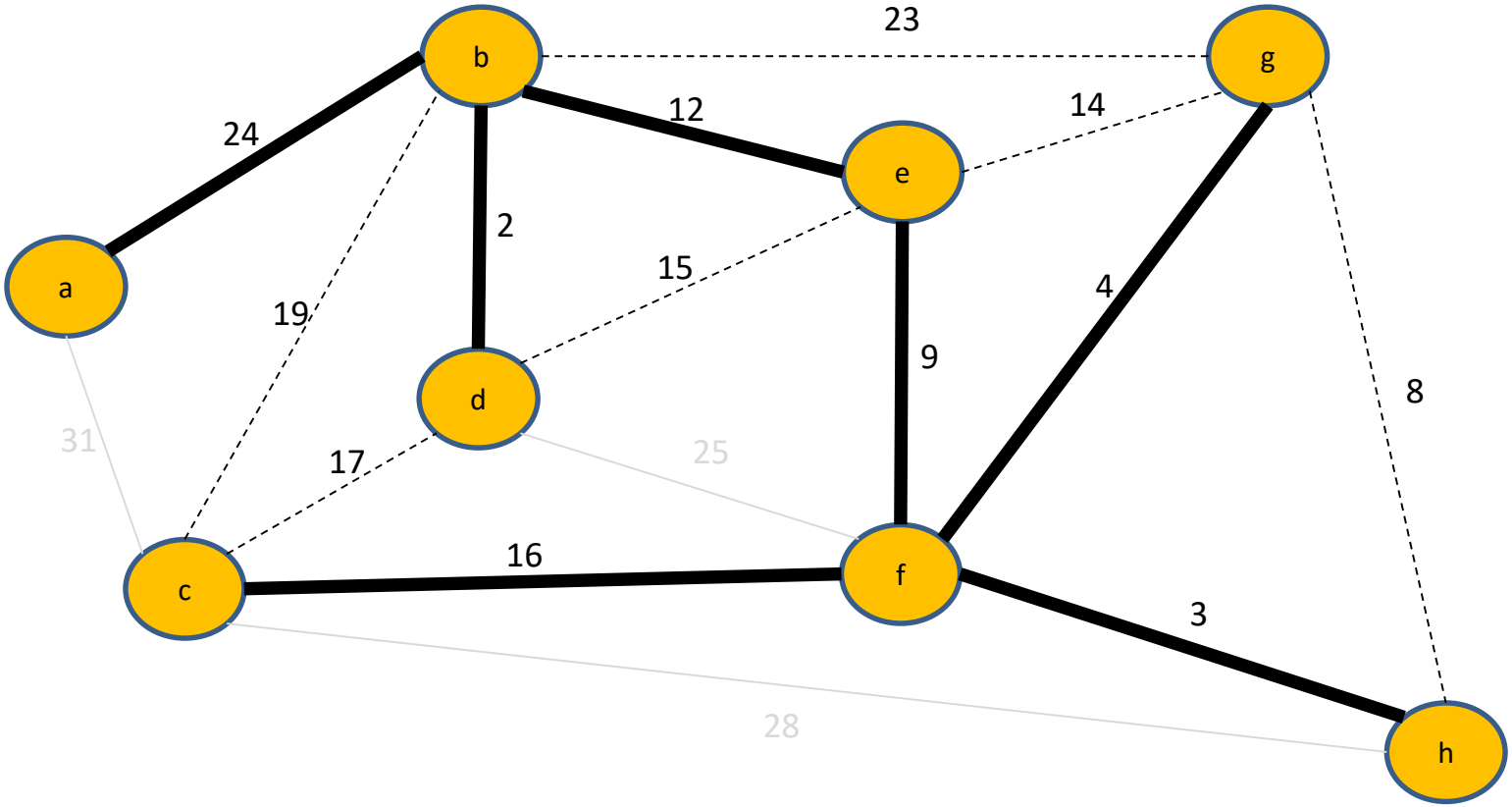
MINIMALNO VPETO DREVO



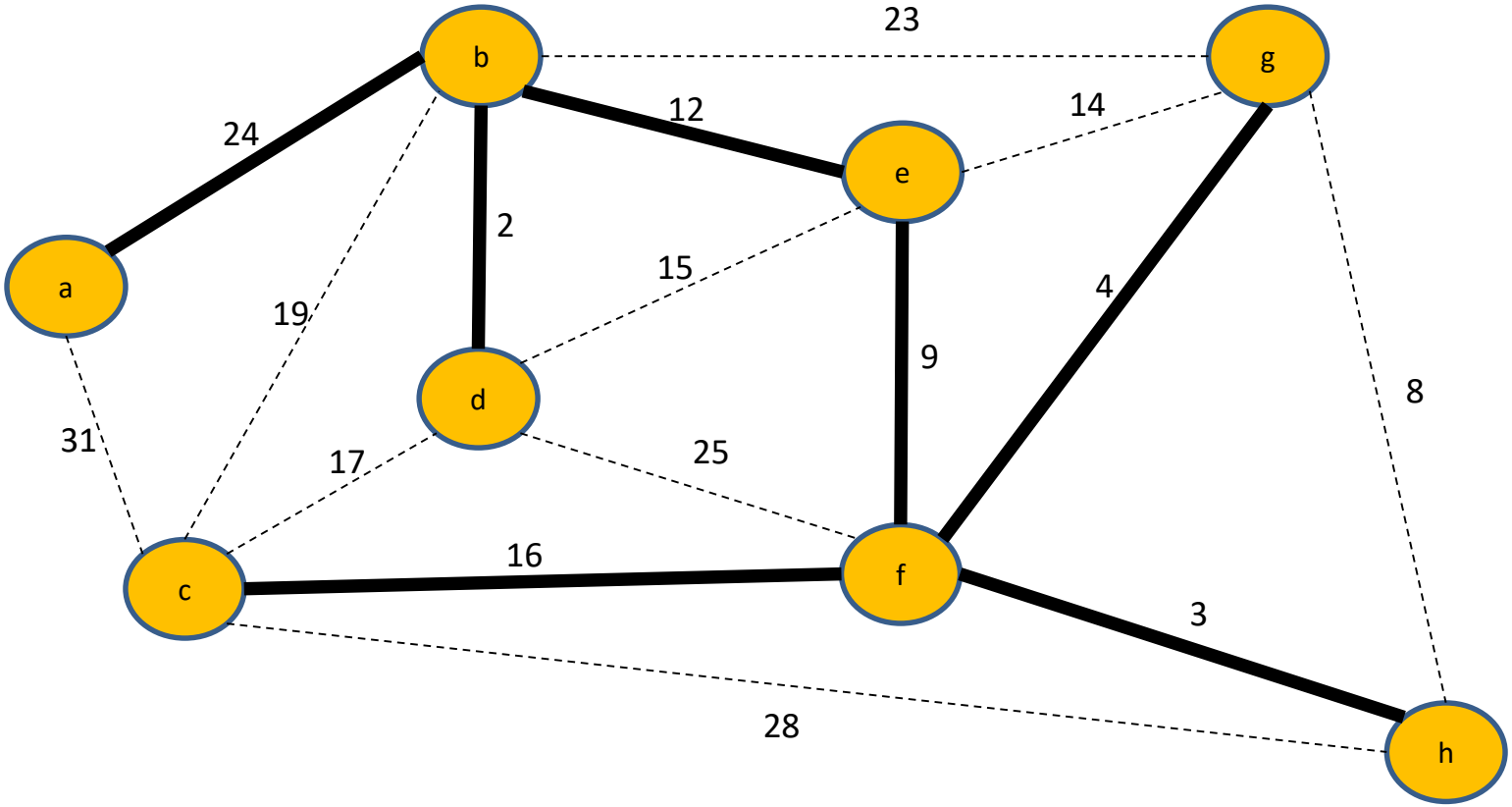
MINIMALNO VPETO DREVO



MINIMALNO VPETO DREVO



MINIMALNO VPETO DREVO



NALOGA

V Primovem algoritmu za gradnjo minimalnega vpetega drevesa prioritetno vrsto realiziramo z

a) urejenim seznamom
b) neurejenim seznamom
c) zgoščeno tabelo
d) AVL drevesom

Ocenite časovno zahtevnost vseh variant algoritma za gradnjo minimalnega vpetega drevesa v odvisnosti od števila vozlišč n in števila povezav m.

Primov algoritem: n operacij INSERT, n operacij DELETEMIN, m-(n-1) krat DECREASE_KEY (torej O(m) krat)

Prior. vrsta	INSERT	DELETEMIN	DECREASE_KEY	SKUPAJ
Ur. seznam	O(n)	O(1)	O(n)	O(n*O(n) + n*O(1) + m*O(n)) = O(n^2 + m*n)
Neur. seznam	O(1)	O(n)	O(1)	O(n*O(1) + n*O(n) + m*O(1)) = O(n^2 + m)
Zg. tabela	O(1)	O(n)	O(1)	O(n*O(1)+n*O(n) + m*O(1)) = O(n^2 + m)
AVL	O(log n)	O(log n)	O(log n)	O(n*O(log n) + n*O(log n) + m*O(log n)) = O(m*log n), m >= n-1