Domácí úkol 3:

Mějme mapu malebného městečka se spoustou úzkých uliček a kaváren s předzahrádkami. Chceme z našeho distrubučního skladu rozvážet do kaváren suroviny. Některé ulice jsou ale příliš úzké pro náklaďáky, dodávky, mnohdy i rozumně velké automobily. Chceme pro každou kavárnu najít cestu po které se k ní dostane co nejširší dopravní prostředek.

Na vstupu máme (orientovaný) graf s hranami ohodnocenými šířkou $1, \ldots, \infty$. Je dán výchozí vrchol v. Pro všechny ostatní vrcholy chceme spočítat šířku největšího dopravního prostředku, který by se do daného vrcholu z v mohl dostat. Budeme také chtít spočítat informace, ze kterých můžeme v případě potřeby pro libovolný vrchol w zrekonstruovat celou cestu mezi v a w v rozumném čase (úměrný délce cesty).

Předpokládáme, že vstup je korektně zadaný graf s vrcholem a ohodnocením. Můžeme předpokládat, že všechny hrany, které v grafu jsou mají nenulovou šířku. Pro každý vrchol chceme cestu maximální šířky, neuvažujeme žádnou množinu dostupných dopravních prostředků nebo další omezení.

(+) [tuto část nemusíte řešit ani odevzdávat, nebude hodnocena] ... ale můžete se zkusit samostatně zamyslet co by se na úloze změnilo pokud bychom měli danou množinu aut s šířkami, a chtěli najít cestu pro nejšírší z dostupných aut. Úloha bude téměř totožná, pouze s jedním obskurujícím krokem navíc. Přístupů je ale hned několik.

Řešení by tradičně mělo obsahovat stručný popis metody, pseudokód, komentář ke speciálním případům, pořádný důkaz korektnosti (můžete odkazovat na vlastnosti algoritmů z přednášek, cvičení a průvodce), analýzu časové složitosti, a stručný komentář jak by bylo možné rekonstruovat konkrétní cesty.

Popis: Upravíme Dijkstrův algoritmus aby místo nejkratších cest hledal nejširší (použijeme maximovou haldu a upravíme výpočty hodnot). Pro každý vrchol si ukládáme předchůdce (P) na nejšírší cestě pro rekonstrukci cest.

pseudokód (plný pseudokód, stačilo ukázat pseudokód vnitřního cyklu)

```
Function Dijkstra(G, zdroj v)
  stav[*] ← N; width[*] ← 0; P[*] ← ∅
  stav[v] ← O; width[v] ← ∞
  while ∃ otevřené vrcholy do
    u ← otevřený vrchol s největší šířkou
    for w soused u do
      if width[w] < min(width[u] ; width[uw]) then
        stav[w] ← O
        zlepši odhad šířky w na min(width[u] ; width[uw]), a
          propaguj změnu do haldy
      end
    end
    stav[u] ← Z
  end
  return pole width, P
end
```

Korektnost:

Stejně jako u Dijkstry platí, že prodloužením sledu se jeho hodnota chová monotónně, může pouze klesnout (šířka sledu je minimum z šířek hran). To je ekvivalentní neexistenci záporných hran v klasickém Dijkstrovi.

Odhady šířek vrcholů začínají na 0, a mohou pouze růst. Obdobně jako v Dijkstrovi platí, že při zavírání vrcholu u se nemůže šířka jiného vrcholu zvýšit nad šířku u. To plyne z předchozího, protože všechny nově uvažované hodnoty jsou prodloužením aktuálně nejširšího sledu do u. Stejně jako u Dijkstry odvodíme vlastnost analogickou monotonii, tedy že uzavřené vrcholy mají vždy vyšší šířky než otevřené. Důsledkem je, že každý vrchol je uzavřen právě jednou, a po uzavření se jeho šířka už nezmění.

Nyní dokážeme, že hodnota šířky je při uzavření vrcholu korektní, indukcí dle délky nejširší cesty. Výchozí vrchol má korektní šířku. Nechť u je vrchol s nejširší cestou délky alespoň 2, a v jeho předchůdce na nejširší cestě. Z indukčního předpokladu u má při uzavírání korektní šířku, je tedy relaxován s touto šířkou a v dostane korektní šířku pokud neměl již vyšší. Vyšší šířku ale mít němůže, protože každá šířka odpovídá nějakému sledu. Dostáváme spor.

Složitost:

Stejně jako u Dijkstrova alg. platí, že každý vrchol je uzavřen nejvýše jednou. Složitost je tedy $\mathcal{O}((m+n)\log n)$.