

Moje řešení tohoto příkladu je inspirováno druhým ukázkovým příkladem zadání, jelikož si jej také můžeme představit za pomoci “vysílání signálů”. Nejdříve vyšle “signál” počáteční vrchol hledané cesty. Každý vrchol, přes který signál projde si poznačí hranu ze které přišel a pošle signál dál. Jakmile signál dorazí k cílovému vrcholu, vyšle po jednom z vrcholů ze kterého signál přišel zpátky druhý signál v podobě výstupního označení. Tento signál už není posílán všem vrcholům, ale pouze tomu, který má daný vrchol “poznačen”.

K samotnému algoritmu: Vrchol nejdříve zjistí, jestli je “cílovým” vrcholem (vstupní označení 2). Pokud ano, nastaví své výstupní označení na 1. Dále vrchol zjistí, zda-li zná cestu – číslo hrany, která vede k “počátečnímu” vrcholu (vstupní označení 1). Pokud ne, a některý ze sousedních vrcholů ji zná, nebo je některý ze sousedů vrcholem počátečním, uloží si místní číslo hrany vedoucí k tomuto sousednímu vrcholu do proměnné. V dalším kroku vrchol zjistí, zda-li je některý z jeho sousedů označen výstupní značkou. Pokud ano a daný vrchol je u tohoto vrcholu uveden jako “další skok” na cestě k cílovému vrcholu, označí touto značkou i sám sebe. Nyní už zbývá jen vyřešit, zda-li může výpočet skončit. Toto je ponecháno na “cílovém” vrcholu – pokud je označen, znamená to že je nalezena cesta a je možno skončit, jinak se pokračuje dále. Všechny ostatní vrcholy na konci každého taktu provedou příkaz stop.

Nalezení cesty bude trvat $2 \cdot (N-1)$, kdy N je počet vrcholů nalezené cesty, včetně počátečního a cílového. Signál musí dojít tam i zpátky, přičemž cestou tam je vynechán počáteční vrchol, cestou zpět koncový.

```

var  x: 0..2;
     y: 0..1 = 0;
     { cislo hrany, po ktorej vede cesta k vrcholu 1 }
     nexthop: 0..3 = 0;
     { pocitadlo pro for }
     i: 1..3 = 0;
begin
  { vrchol, ze ktereho se bude sirit signal zpet }
  if x=2 then y := 1;

  { pokud je to mozne, poznamcime si cestu k vrcholu 1 }
  if nexthop = 0 then begin
    for i := 1 to 3 do begin
      if (S[i].x = 1) or (S[i].nexthop <> 0) then nexthop := i;
    end;
  end;

  { pokud ma nas soused vystupni oznaceni a my jsme jeho dalsi vrchol
    na ceste k 1, oznacime se taky }
  for i:= 1 to 3 do begin
    if (S[i].y = 1) and (S[i].nexthop = P[i]) then y := 1;
  end;

  { pokud jsme vrchol 1 a jsme oznaceni, muzeme skoncit }
  if x=1 then begin
    if y=1 then stop;
  end
  { pokud jsme jiny vrchol, nechame to na vrcholu jedna - povolime ukonceni }
  else stop;
end.

```