Floyd-Warshallov algoritmus

```
// initialization
for i:=1 to n do
  for j:=1 to n do
    dist[i,j,0]:=w[i,j];

// main computation
for k:=1 to n do
  for i:=1 to n do
    for j:=1 to n do
      dist[i,j,k]:=min{dist[i,j,k-1],
                       dist[i,k,k-1]+dist[k,j,k-1]}
```

Floyd-Warshallov algoritmus

```
// initialization
for i:=1 to n do
  for j:=1 to n do
**  dist[i,j]:=w[i,j];

// main computation
for k:=1 to n do
  for i:=1 to n do
    for j:=1 to n do
**    if dist[i,k]+dist[k,j]<dist[i,j] then
**       dist[i,j]:=dist[i,k]+dist[k,j];
```

Dijkstrov algoritmus

```
// initialize dist, S, T
S:=0; T:=V;
for all w in V do dist[w]:=infinity;
dist[u]:=0;

// add one vertex at a time to T
while T is non-empty do
**s:=vertex for which dist[s] represent the length
**   of the shortest path from u to s;
  add s to S; remove s from T;
  // update dist to account for enlarged set S
  for all t in out(s) do
    // try to shorten current path to t through s
    if (dist[s]+w[s,t]<dist[t]) then
      dist[t]:=dist[s]+w[s,t];
```

Dijkstrov algoritmus

```
// initialize dist, S, T
S:=0; T:=V;
for all w in V do dist[w]:=infinity;
dist[u]:=0;

// add one vertex at a time to T
while T is non-empty do
  s:=vertex with the smallest dist[s];
  add s to S; remove s from T;
  // update dist to account for enlarged set S
  for all t in out(s) do
    // try to shorten current path to t through s
    if (dist[s]+w[s,t]<dist[t]) then
      dist[t]:=dist[s]+w[s,t];
```

Dijkstrov algoritmus (s rekonštrukciou ciest)

```
// initialize dist, S, T
S:=0; T:=V;
for all w in V do
* dist[w]:=infinity; last[w]:=undefined;
dist[u]:=0;

// add one vertex at a time to T
while T is non-empty do
  s:=vertex with the smallest dist[s];
  add s to S; remove s from T;
  // update dist to account for enlarged set S
  for all t in out(s) do
    // try to shorten current path to t through s
    if (dist[s]+w[s,t]<dist[t]) then
*     dist[t]:=dist[s]+w[s,t]; last[t]:=s;
```

```
// path reconstruction from u to v
w:=v; create an empty path;
while last[w]<>undefined do
  add w to the beginning of the path;
  w:=last[w];
```