

## ALGORITHMEN ZUR SUCHE IN GRAPHEN (III-B)

Kürzeste Wege:

| <b>ALGORITHMUS</b> <i>Ford</i>                                |              |  |  |
|---|--------------|--|--|
| Input:  | Matrix:      | TAdjazenzmatrix                                  | <i>{ Array[1..n,1..n] of integer }</i>       |
|   | Startknoten: | integer  |  |
| Output:   | Distanz:     | array[1..n] of integer                           |  |
|   | Vorgaenger:  | array[1..n] of integer                           |  |
| Lokal:  | AmEnde:      | boolean  | <i>{ Alle kürzesten Wege sind gefunden }</i> |
|   | von, nach:   | integer  | <i>{ um alle Kanten zu durchlaufen }</i>     |
| <i>{ Initialisierung }</i>                                    |              |  |  |
| • Für Knoten $\leftarrow$ 1 bis n                             |              |  |  |
| tue: • Distanz[Knoten] $\leftarrow$ maxInt                    |              | <i>{ es gibt noch keinen kürzesten Weg }</i>     |  |
| • Vorgaenger[Knoten] $\leftarrow$ Startknoten                 |              | <i>{ der Standard-Vorgänger ist der SK }</i>     |  |
| • Distanz[Startknoten] $\leftarrow$ 0                         |              |  |  |
| • Vorgaenger[Startknoten] $\leftarrow$ -1                     |              |  |  |
| <i>{ Algorithmus }</i>  |              |  |  |
| • AmEnde $\leftarrow$ true                                    |              |  |  |
| • Wiederhole  |              |  |  |
| • Für von $\leftarrow$ 1 bis n                                |              |  |  |
| tue: • Für nach $\leftarrow$ 1 bis n                          |              |  |  |
| tue: • Falls Distanz[von] + Matrix[von, nach] < Distanz[nach] |              | <i>{ d. h. der „Umweg“ über von ist kürzer }</i> |  |
| dann: • Distanz[nach] $\leftarrow$                            |              |  |  |
| Distanz[von] + Matrix[von, nach]                              |              |  |  |
| • Vorgaenger[nach] $\leftarrow$ von                           |              |  |  |
| • AmEnde $\leftarrow$ false                                   |              |  |  |
| bis: AmEnde = true  |              |  |  |

```

procedure TGraph.Ford(Startknoten: integer; pb: TPaintBox; SG: TStringGrid);
type TBesuchteKnotenInfo= record
    distanz: integer;
    vorgaenger: integer;
end;
    TBesuchteKnoten= array[1..maxAnzahl] of TBesuchteKnotenInfo;

var BesuchteKnoten: TBesuchteKnoten;
    a: TAdjazenzmatrix;

procedure Initialisierung;
var i, j: integer;
begin
    a:= Adjazenzmatrix;
    for i:= 1 to maxAnzahl do
        begin
            BesuchteKnoten[i].distanz:= maxEntfernung;
            BesuchteKnoten[i].vorgaenger:= Startknoten;
        end;
        BesuchteKnoten[Startknoten].vorgaenger:= -1;
        BesuchteKnoten[Startknoten].distanz:= 0;

procedure ZeichnePfadZurueck(von: integer; spf, kf: TColor);
var vorgaenger: integer;
begin
    vorgaenger:= BesuchteKnoten[von].vorgaenger;
    if vorgaenger>-1
    then begin
        ZeigeKante(vorgaenger,von,kf,PB,[fett,pfeil]);
        ZeigeEntfernung(vorgaenger,von,BesuchteKnoten[von].Distanz,clBlack,PB);
        ZeigePunkt(von,spf,pb);
        ZeichnePfadZurueck(vorgaenger,spf,kf);
    end;
end;

var von, nach: integer;
    d: integer;
    ende: boolean;
    count: integer;

begin
    ende:= false;
    count:= 0;
    while not ende
    do begin
        ende:= true;
        inc(count);
        for von:= 1 to Anzahl
        do begin
            for nach:= 1 to Anzahl
            do begin
                d:= BesuchteKnoten[von].Distanz+a[von,nach];
                if d < BesuchteKnoten[nach].Distanz
                then begin
                    BesuchteKnoten[nach].Distanz:= d;
                    BesuchteKnoten[nach].Vorgaenger:= von;
                    ende:= false;

                    ZeigeKante(von,nach,clBlue,pb,[fett,pfeil]);
                    ZeigeEntfernung(von,nach,BesuchteKnoten[nach].Distanz,clBlack,PB);
                    ZeigePunkt(von,clBlue,pb);
                    ZeigePunkt(nach,clBlue,pb);
                    if BesuchteKnoten[von].Distanz<maxEntfernung
                    then SG.Cells[nach,count]:= IntToStr(BesuchteKnoten[nach].Distanz)
                    else SG.Cells[nach,count]:= '---';
                end
                else if (a[von,nach]<maxEntfernung) or (a[nach,von]<maxEntfernung)

```

```
        then ZeigeKante(von,nach,clGray,pb,[])
        else ZeigeKante(von,nach,clBtnFace,pb,[])
    end;
end;
warte;
ZeigeDaten(clGray, clGray, clBtnFace, clBtnFace, PB);
ZeigePunktname(Startknoten, 10, clBlack, pb);
ZeigePunkt(Startknoten,clLime,PB);
for von:= 1 to Anzahl
do if BesuchteKnoten[von].Distanz<maxEntfernung
    then ZeichnePfadZurueck(von,clRed,clRed);
end;
end;
```