matfyz

Computer Graphics Charles University

Vyplňování souvislé oblasti

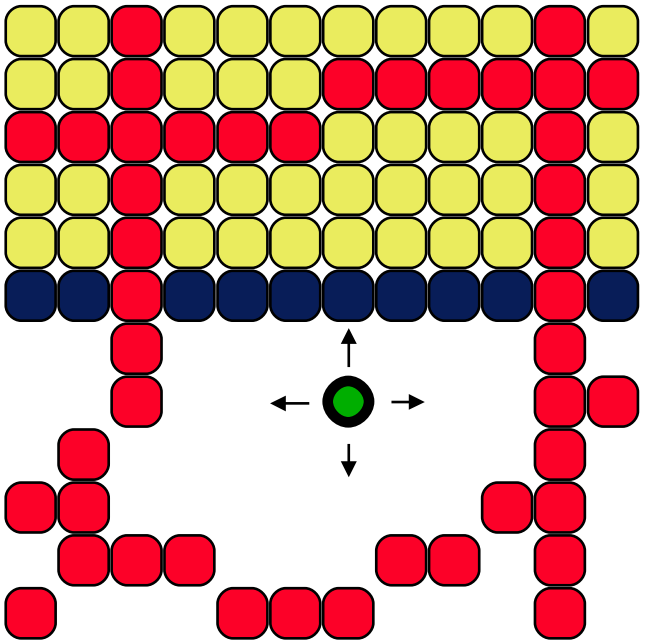
© 1995-2015 Josef Pelikán

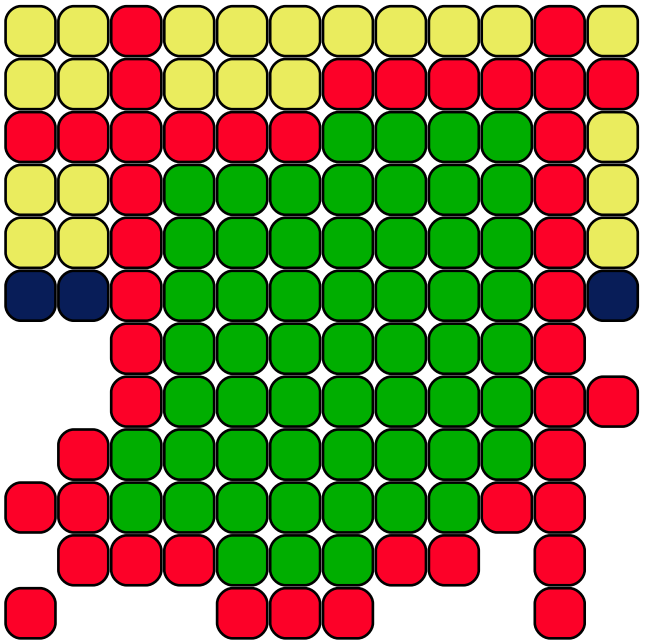
CGG MFF UK Praha

pepca@cgg.mff.cuni.cz

http://cgg.mff.cuni.cz/~pepca/

Hraniční vyplňování

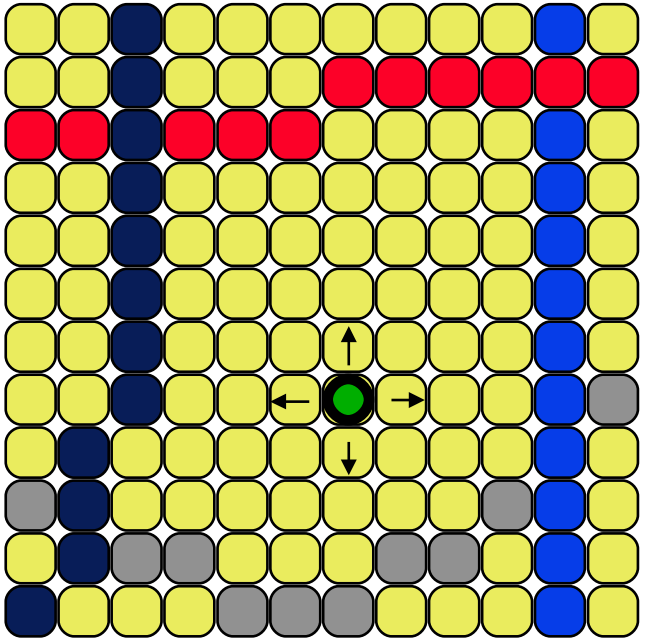


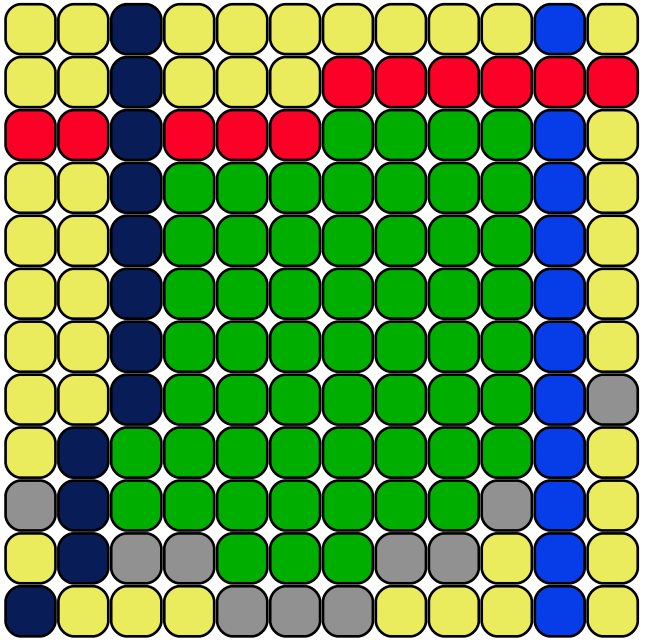


vyplnění až ke hranici dané barvy

GetPixel(x,y) <> barva_hranice

Záplavové vyplňování

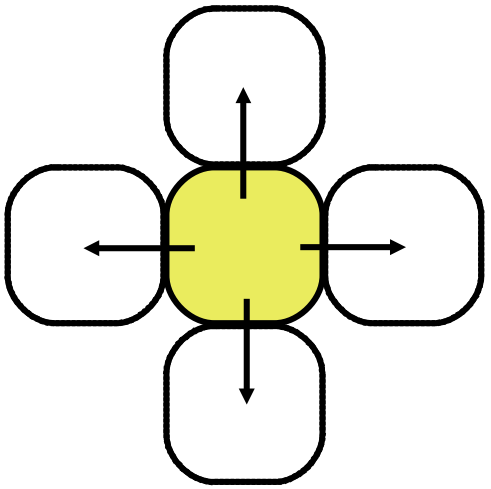


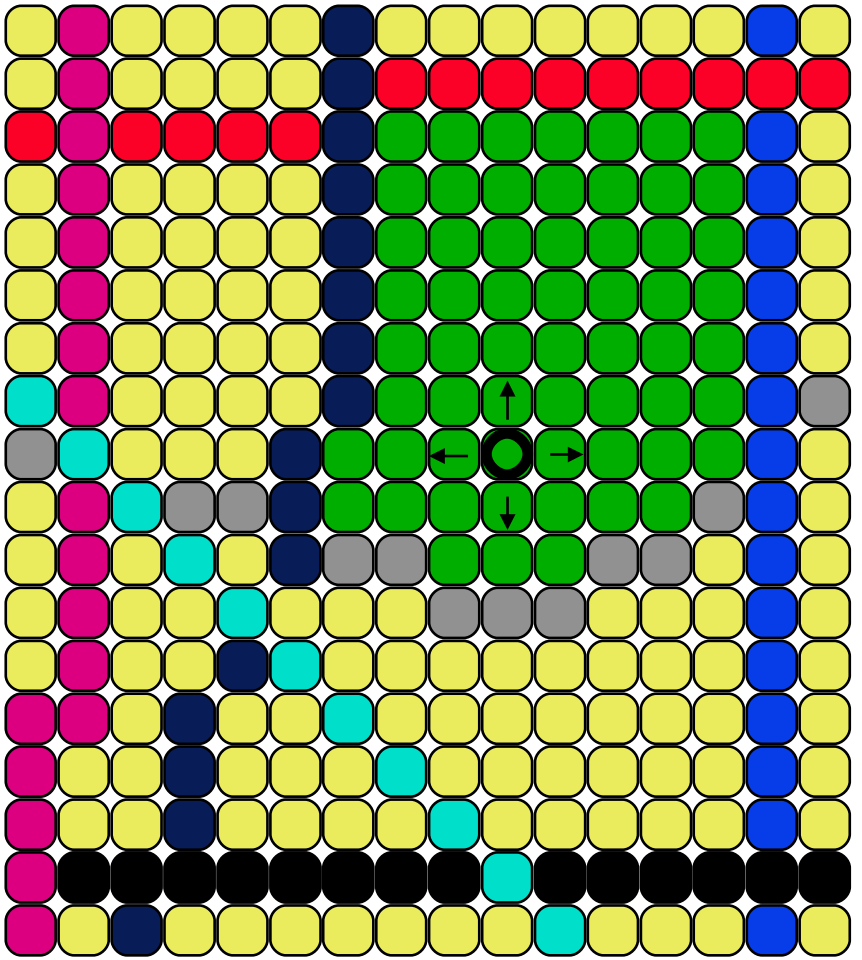


přebarvení pixelů dané barvy

GetPixel(x,y) = původní_barva

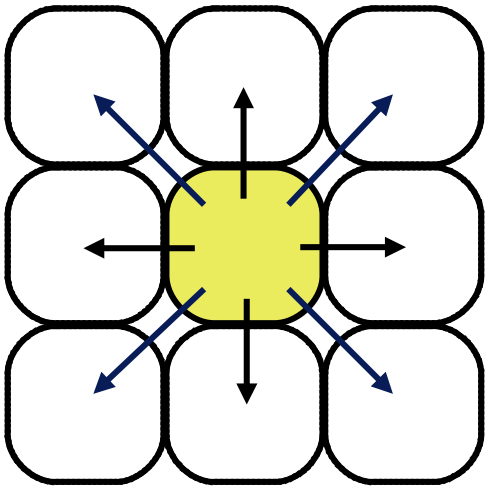
4-souvislá oblast

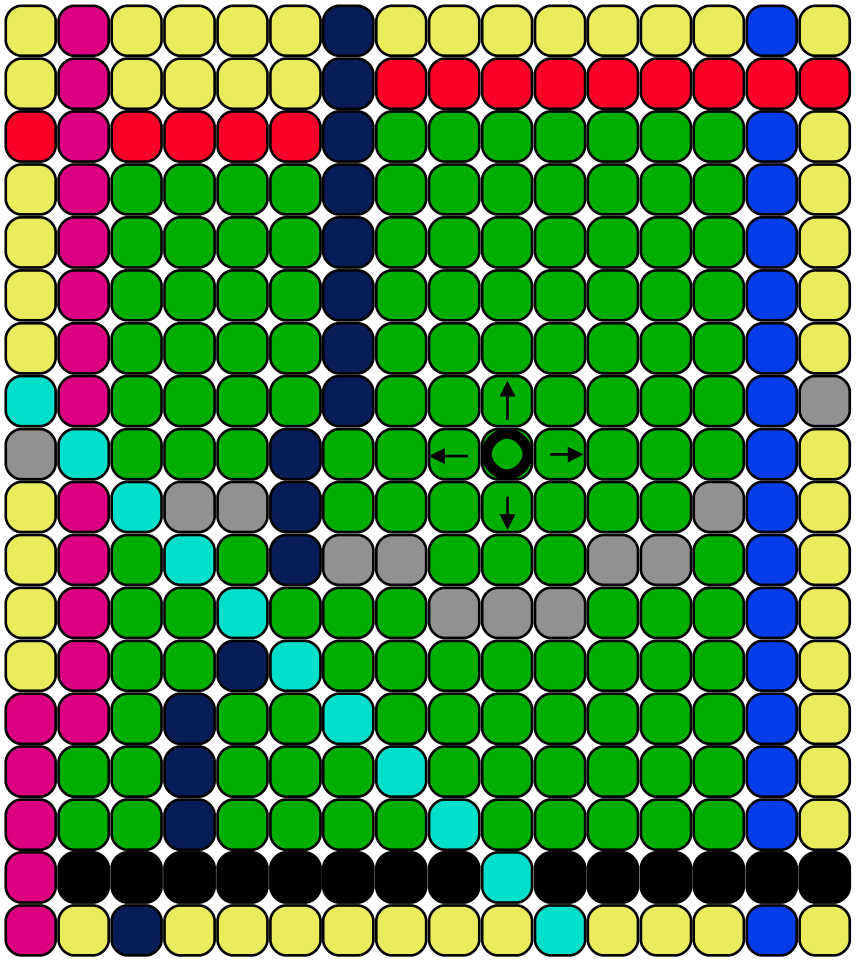




záplavová varianta

8-souvislá oblast





záplavová varianta

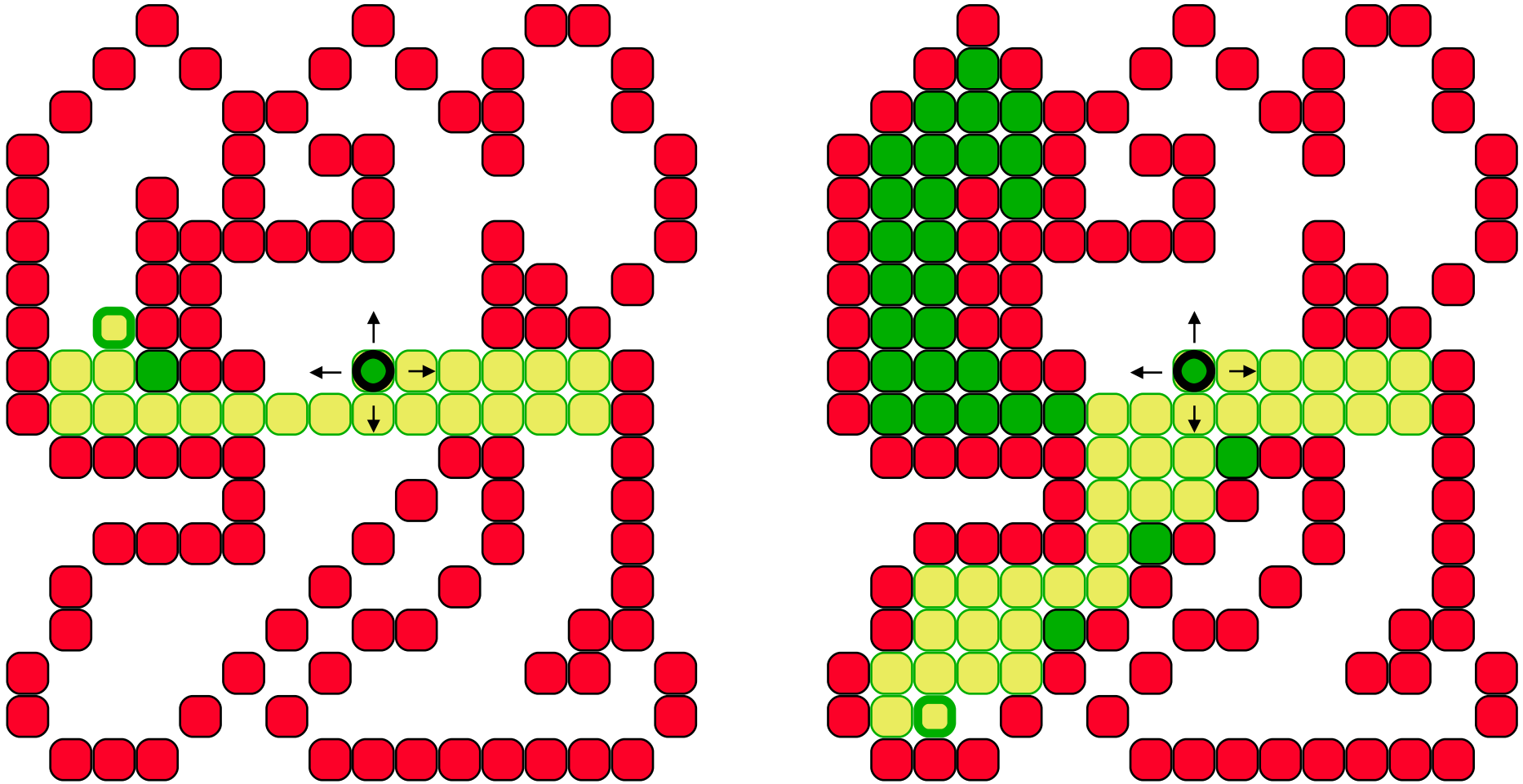
Naivní rekurzivní algoritmus

```
procedure FloodFill4 ( x, y, oldc, newc : integer );
        { záplavová 4-souvislá varianta, oldc <> newc }
begin
  if GetPixel(x,y) = oldc then
    begin             { pixel [x,y] patří do oblasti }
      PutPixel(x,y,newc);
      FloodFill4(x+1,y,oldc,newc);   { čtyři sousedé: }
      FloodFill4(x-1,y,oldc,newc);
      FloodFill4(x,y+1,oldc,newc);
      FloodFill4(x,y-1,oldc,newc);
    end;
end;
```

hraniční varianta:

```
(GetPixel(x,y) <> boundc) and
(GetPixel(x,y) <> newc)
```

Postup vyplňování



hranice vyplněno zásobník

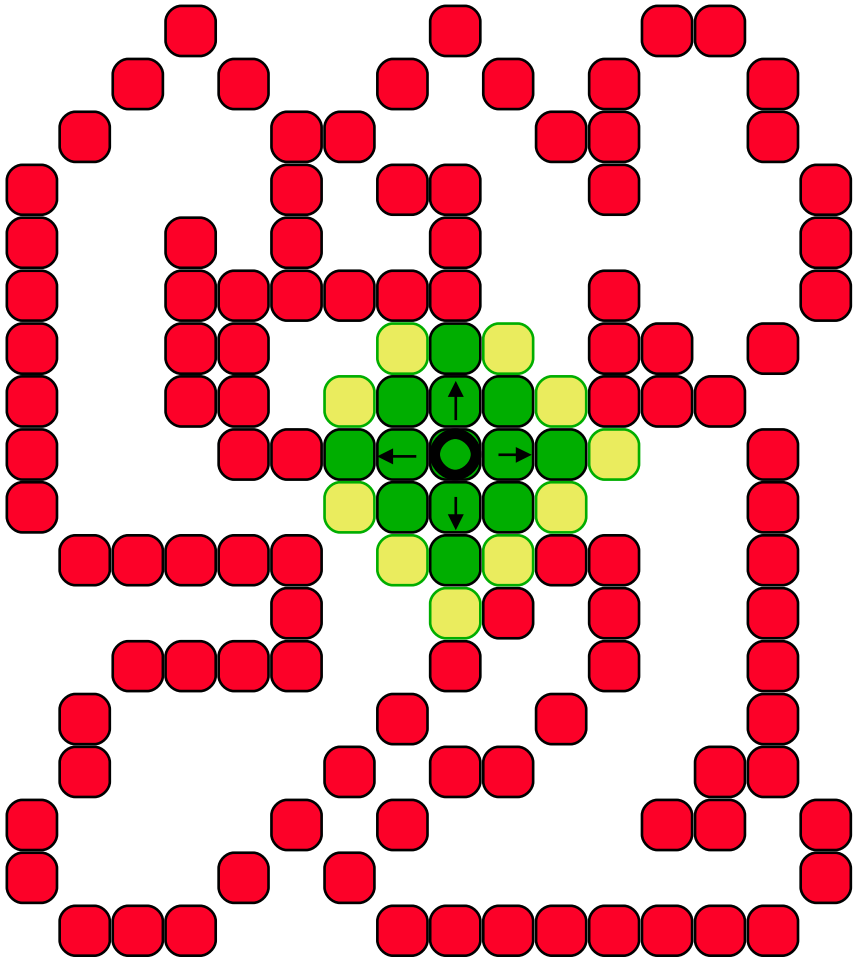
Použití fronty místo zásobníku

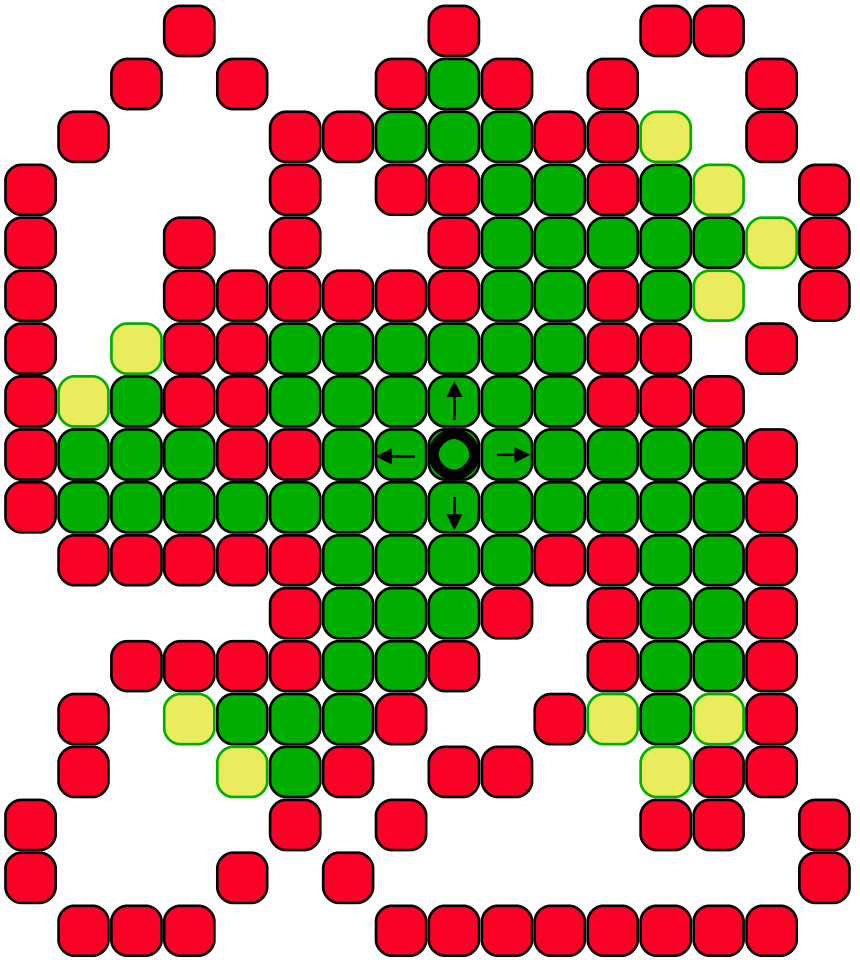
```
procedure FloodFill4 ( x, y, oldc, newc : integer );
        { záplavová 4-souvislá varianta, oldc <> newc }
var Q : Queue;
begin
  Q.Init; Q.Put(x,y);
  repeat
    Q.Get(x,y);
    if GetPixel(x,y) = oldc then
      begin          { pixel [x,y] patří do oblasti }
        PutPixel(x,y,newc);
        Q.Put(x+1,y); Q.Put(x-1,y);
        Q.Put(x,y+1); Q.Put(x,y-1);
      end;
  until Q.Empty;
end;
```

Úspornější varianta

```
procedure FloodFill4 ( x, y, oldc, newc : integer );
      { záplavová 4-souvislá varianta, oldc <> newc }
var Q : Queue;
  procedure NextPixel ( x, y : integer );
  begin      { patří-li pixel do oblasti, uloží ho do fronty }
    if GetPixel(x,y) = oldc then
      begin
        PutPixel(x,y,newc); Q.Put(x,y);
      end;
  end;
begin
  Q.Init; NextPixel(x,y);                  { startovní pixel }
  repeat
    Q.Get(x,y);
    NextPixel(x+1,y); NextPixel(x-1,y);   { čtyři sousedé: }
    NextPixel(x,y+1); NextPixel(x,y-1);
  until Q.Empty;
end;
```

Postup vyplňování



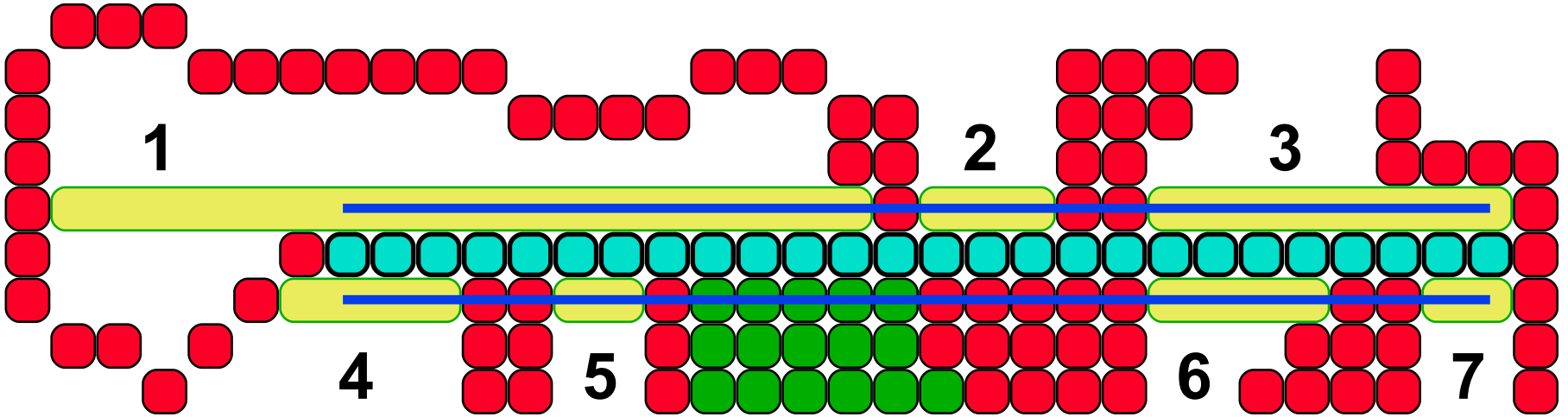


Řádkové vyplňování

```
procedure LineFloodFill4 ( x, y, oldc, newc : integer );
      { záplavová 4-souvislá varianta, oldc <> newc }
var S : Stack;                { položka: [Xmin,Xmax,y] }
    Xmin, Xmax : integer;     { meze na aktuální řádce }
  procedure Search ( Xmin, Xmax, y : integer );
  var Xm : integer;
  begin      { najde všechna pokračování v daném úseku řádky }
    while GetPixel(Xmin-1,y) = oldc do Dec(Xmin);
    repeat             { zkouším [Xmin,y] }
      Xm := Xmin;           { hledám pravý konec úseku: }
      while GetPixel(Xm+1,y) = oldc do Inc(Xm);
      S.Push(Xmin,Xm,y);
      Xmin := Xm+2;         { hledám následující úsek: }
      while (Xmin <= Xmax) and (GetPixel(Xmin,y) <> oldc) do
        Inc(Xmin);
    until Xmin > Xmax;
  end;

...
```

Hledání následníků



hranice

dříve vyplněné pixely

naposledy vyplněné pixely

prohledávané řádky

1-7 nové položky na zásobníku

Řádkové vyplňování

```
...
begin
  S.Init; Search(x,x,y);          { první bod (semínko) }
  repeat
    S.Pop(Xmin,Xmax,y);
    if GetPixel(Xmin,y) = oldc then
      begin                       { úsek ještě nebyl vyplněn }
        Line(Xmin,y,Xmax,y,newc);
        Search(Xmin,Xmax,y-1);
        Search(Xmin,Xmax,y+1);
      end;
  until S.Empty;
end;
```

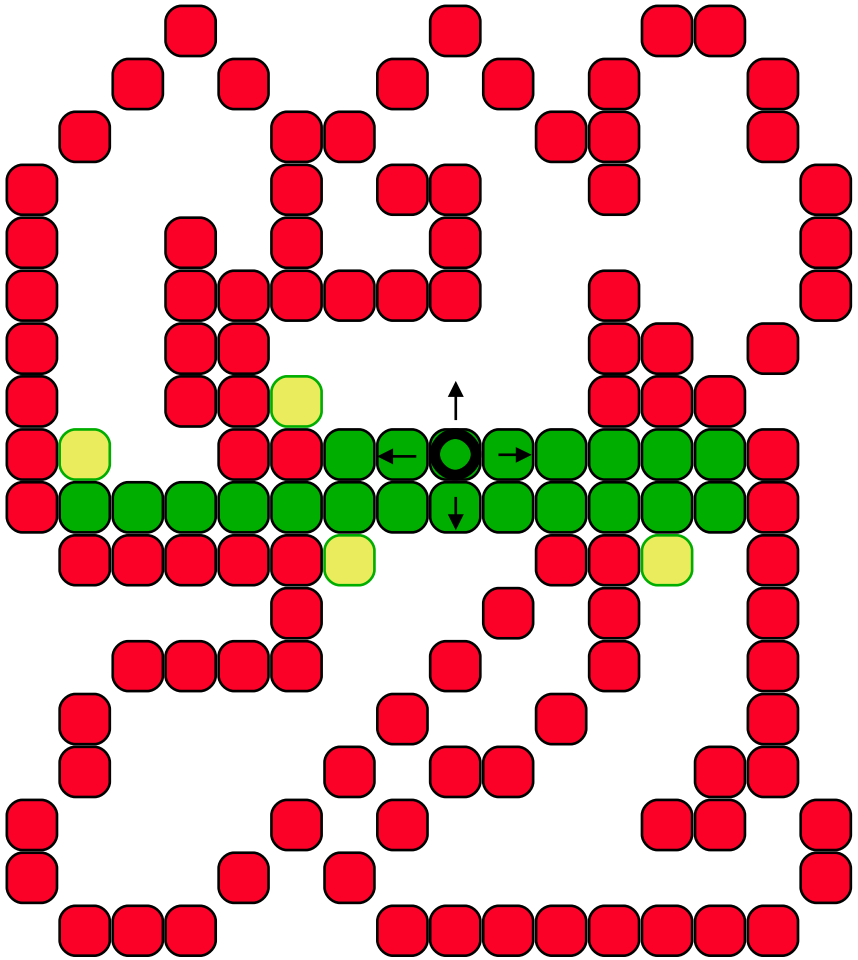
hraniční varianta:

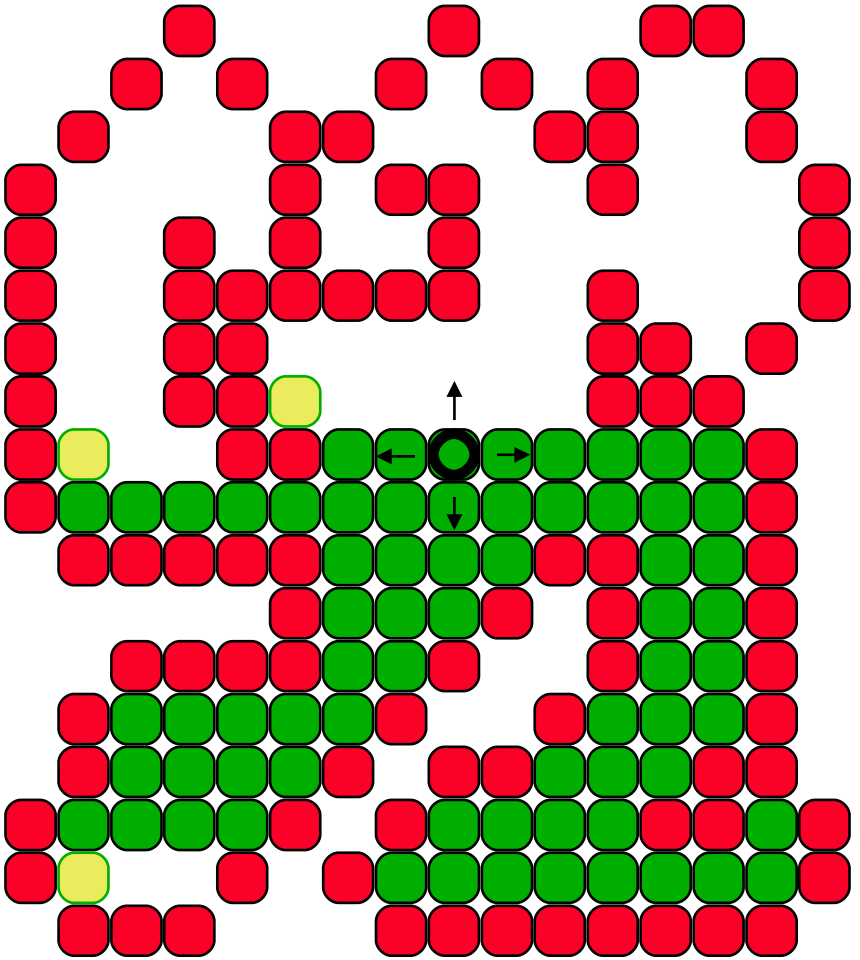
```
(GetPixel(Xmin,y) <> boundc)
and (GetPixel(Xmin,y) <> newc)
```

8-souvislé vyplňování:

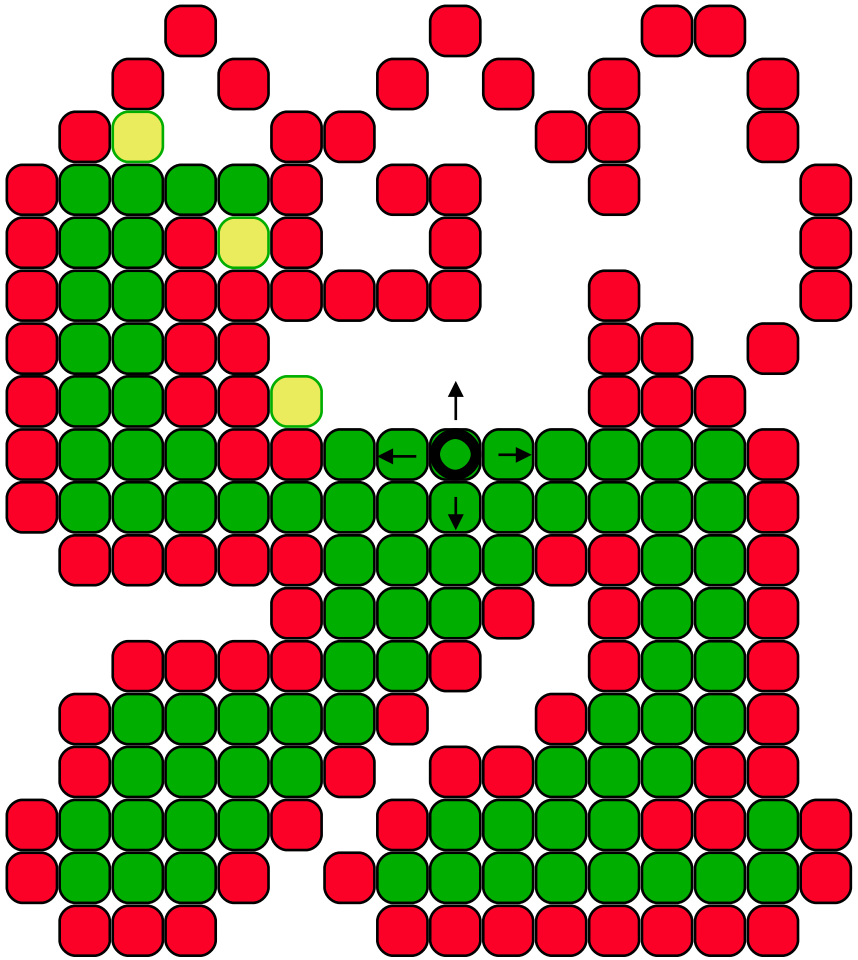
```
Search(Xmin-1,Xmax+1,*)
```

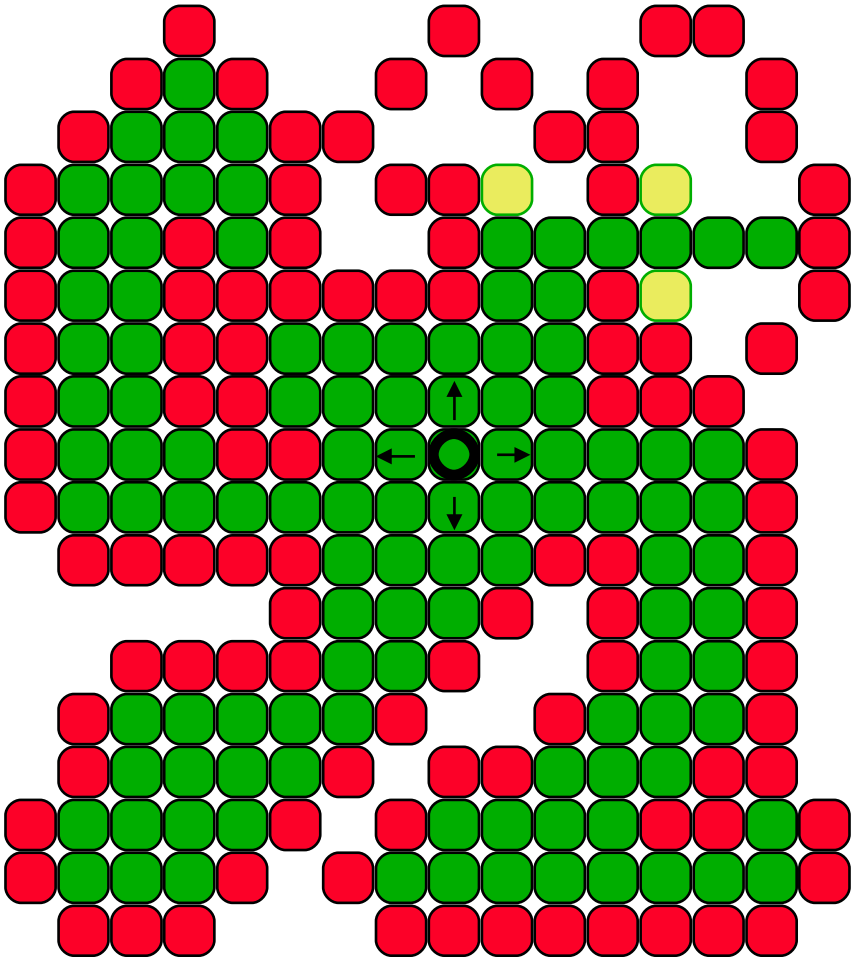
Postup vyplňování





Postup vyplňování





Výhody řádkového algoritmu

- **menší spotřeba paměti**
  - zásobník v běžných případech roste jen pomalu
- **větší rychlost**
  - úspornější přístup do VideoRAM po řádkách
- **zásobník** versus **fronta**:
  - při použití zásobníku je postup vyplňování lokální
  - výhodné při přepínání stránek VideoRAM

Konec

Další informace:

- **J. Foley, A. van Dam, S. Feiner, J. Hughes**: ***Computer Graphics, Principles and Practice***, 979-982
- **Jiří Žára a kol.**: ***Počítačová grafika***, principy a algoritmy, 142-147