Računarska grafika - vežbe

10 – Popunjavanje oblasti

Zadatak 1

U memoriji rasterskog prikazivača se nalazi na crvenoj pozadini "šahovska tabla" sa plavim i žutim poljima.

a) Napisati proceduru koja koristi jednostavni rekurzivni algoritam za izmenu boje svih polja jedne (plave ili žute) boje, za zadate koordinate piksela koji pripada proizvoljnom polju date početne boje

b) napisati sekvencu poziva gornje procedure koja će zameniti boje polja (plava da postanu žuta i žuta da postanu plava). Pretpostaviti da je jedno izvršenje procedure mnogo kraće od 1/24 s.

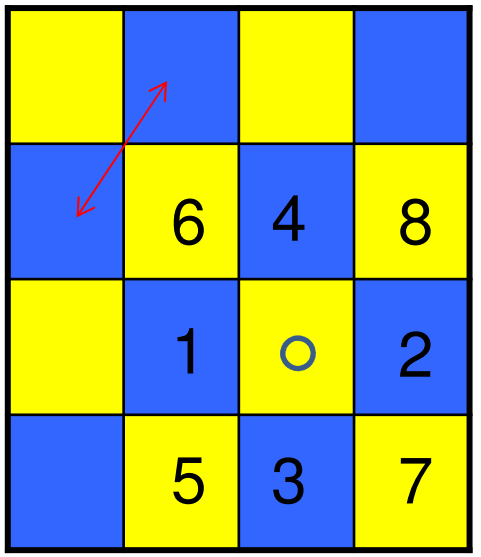
c) Koji su nedostaci navedenog algoritma pod (a)?

d) Kako se nedostaci navedeni pod (c) mogu otkloniti? Ukratko obrazložiti.

Zadatak 1 – rešenje (a)

- Pošto se radi o šahovskoj tabli
  - treba popunjavati osmosusednu oblast
- Pošto tekuću boju jednih polja treba zameniti nekom bojom
  - oblast je definisana unutrašnjošću

```
void FloodFill8(int x, int y, Color new_val, Color old_val){
    if(GetPixel(x,y)==old_val)    {
      SetPixel(x,y,new_val);
      FloodFill8(x-1,y, new_val, old_val);
      FloodFill8(x+1,y, new_val, old_val);
      FloodFill8(x,y-1, new_val, old_val);
      FloodFill8(x,y+1, new_val, old_val);
      FloodFill8(x-1,y-1, new_val, old_val);
      FloodFill8(x-1,y+1, new_val, old_val);
      FloodFill8(x+1,y-1, new_val, old_val);
      FloodFill8(x+1,y+1, new_val, old_val);
    }
}
```



Zadatak 1 – rešenje (b)

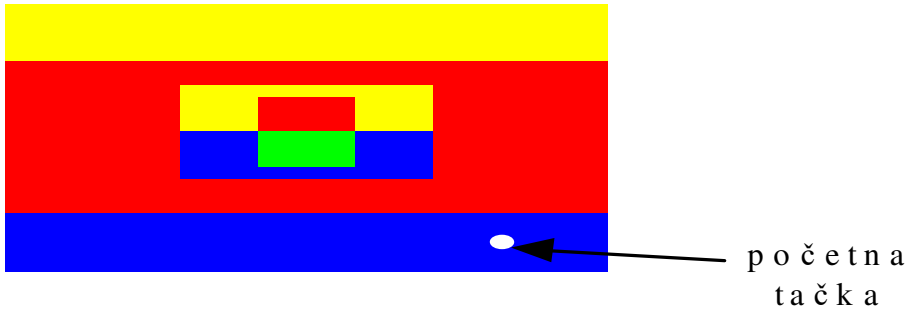
```
FloodFill8(x1,y1,zelena,plava);
   // (x1,y1) u plavom polju
FloodFill8(x2,y2,plava,zuta);
   // (x2,y2) u zutom polju
FloodFill8(x1,y1,zuta,zelena);
```

Zadatak 1 – rešenje (c,d)

- Glavni nedostatak algoritma je rekurzija
  - sporo se izvršava
  - primenljivo je samo za relativno male oblasti (ograničena dubina programskog steka)
- Poboljšanja
  - provera da li pikselu treba promeniti boju pre rekurzivnog poziva
  - napraviti kuržni bafer i iterativno rešenje

Zadatak 2

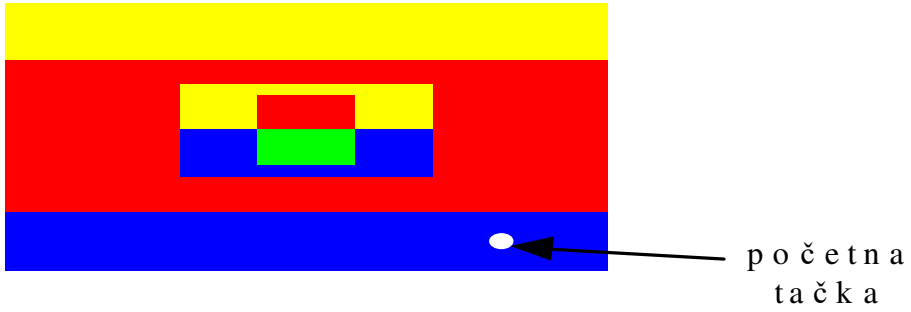
- Data je sledeća slika na jednom izlaznom uređaju



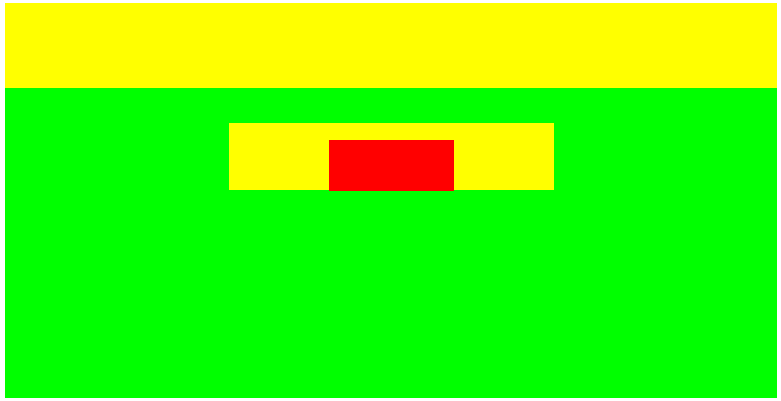
- Ako se na prikazani slučaj primeni jednostavan rekurzivni algoritam popunjavanja:
  - oivičene četvorosusedne oblasti (Boundary_fill4) sa vrednostima:
    - boundary_value = žuta i new_value=zelena
  - oivičene osmosusedne oblasti (Boundary_fill8) sa vrednostima:
    - boundary_value = žuta i new_value=zelena

  skicirati efekat nakon izvršenja odgovarajuće procedure, precizno navodeći boje oblasti.

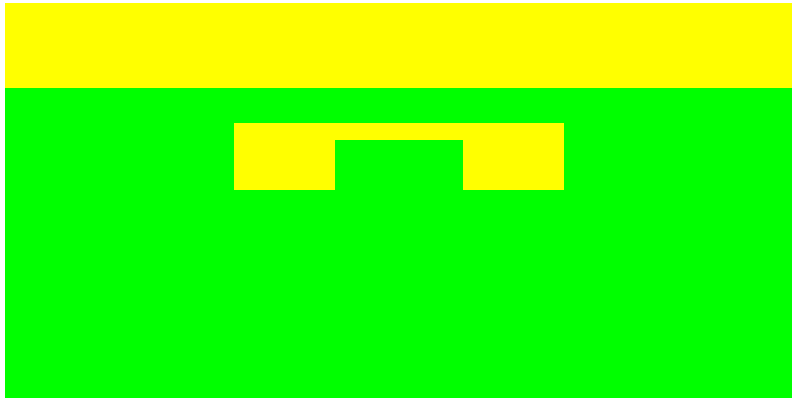
Zadatak 2 – rešenje



boundary=žuta, new=zelena



a) 4-susedna



b) 8-susedna

Linijski rekurzivan algoritam za popunjavanje

```
#include "fill.h"
int HLine(int x, int y)  {
   int from=x, to=x+1;

   if(getPixel(x,y)) return to;
   while(from>0 && !getPixel(from,y))
       setPixel(from--,y,1);
   while(to < limitx && !getPixel(to,y))
       setPixel(to++,y,1);

   if(y<limity) for (int f=from+1;f<to;f=HLine(f,y+1));
   if(y>0) for (int f=from+1;f<to;f=HLine(f,y-1));
   return to;
}

void Fill(int x, int y){
   HLine(x,y);
}
```