Tak prvým riešením ktoré každého napadne je vyskúšať všetky možnosti.
Tomu sa však ani neodvážim odhadnúť akú by to malo časovú zložitosť.
Preto sa poďme pozrieť, čo sa nám tu všetko opakuje.

Aby sme tu zbytočne neplietli zložité zadanie, tak počty znakov v jednotlivých stĺpcoch si načítajme do jednorozmerného poľa – koľko čiernych a bielych políčok obsahuje daný stĺpec.

Poďme sa pozrieť na nasledovné:

Ak sa vieme dostať jedným spôsobom na pozíciu p s BUNV čiernym stĺpcom(tu končí čierny stĺpec a začína biely na stĺpci vpravo, alebo už žiadny) a iným spôsobom tiež s čiernym stĺpcom ale iným počtom zmenených políčok, tak prečo by sme brali ten spôsob s tým vyšším počtom zmenených políčok??
Mohlo by nám to nejako pomôcť?? keby sme zobrali ten s vyšším počtom ??
Určite nie, pretože ak končíme rovnakou farbou na danom stĺpci, tak ďalšie stĺpce dávame nezávisle od tých čo boli pred nimi, čiže ak by malo byť riešenie s vyšším potom zmenených políčok lepšie, tak potom by sme tam vedeli doplniť také stĺpce, aby sme zmenili menej políčok. Ale takéto isté stĺpce vieme spraviť aj v riešení s menším počtom doplnených políčok po pozíciu p a ak majú mať zvyšok rovnaký, tak potom riešenie s menším počtom doplnených políčok po pozíciu p ,bude mať na konci menej doplnených políčok. Pretože ku každému riešeniu pripočítame rovnaké číslo.
Takže ak sa vieme dostať na nejaké políčko s rovnakou farbou, tak vždy berieme najlepšie riešenie.

Na takéto veci ako nájsť najkratšiu cestu, alebo ako je toto je vhodné BFS.

Keďže na dané políčko sa vieme dostať s 2 rôznymi farbami, tak musíme mať 2 fronty.
Vždy ak sme na políčku v prvej fronte, tak miesta, kde sa sa vieme dostať umiestnime do druhej fronty.

BFS vyzerá nasledovne:

vlož do fronty prvok kde začínaš

pokiaľ nieje fronta prázdna:

prehľadaj kde sa môžeš dostať z daného políčka a vlož tieto políčka do fronty aj sa nane vieme dostať s menšou vzdialenosťou/počtom krokov/časom ako sa naň už vieme dostať
odstráň toto políčko, ktoré si akurát prehľadával z fronty – choď na ďalšie vo fronte u nás to bude znamenať choď na stĺpec vpravo, lebo na tento stĺpec sa už určite nevieme dostať kratšou cestou, pretože sme všetky vyskúšali.

Takže čo my spravíme:

vytvoríme si 2 fronty čiernu a bielu

vložíme do oboch 0, pretože sa nachádzame v 0-tom stĺpci bitmapy.

Spracujeme BUNV čiernu frontu – z čierna fronta znamená, že tu končíme čiernym stĺpcom ideme na všetky možné políčka – stĺpce bitmapy na ktoré možme ísť – aby sme spĺňali rozmery x a y a ak sa vieme dostať na tieto políčka-stĺpce s menším počtom zmenených pixelov, tak na ten stĺpec zapíšem, že tu sa viem dostať s takým a takým počtom zmenených políčok a takou farbou.
Potom idem na ďalší stĺpec- vpravo, lebo na tento sa už neviem dostať kratšou cestou – všetky som

už vyskúšal.

Pred tým ako pôjdem v čiernej fronte na ďalšie políčko spracujem bielu frontu podobne ako čiernu – tam kde sa viem odtiaľto dostať píšem do čiernej fronty – ten druhej.

No a už môžem ísť na ďalšie políčko – vpravo.

Takto sa dostanem možno aj s obidvomi frontami na posledné políčko.

No a teraz už skontrolujem s ktorou frontou sa viem dostať na toto políčko (ak sa vôbec viem dostať) a s čom najmenším počtom zmenených políčok.

Spracovanie bitmapy do jednorozmerného poľa vyzerá asi nasledovne:

for i=1 to r
for j=1 to s
if znak na tomto políčku je čierny stlpec s++

potom nájdenie najkratšej cesty, alebo ako inak sa to dá nazvať robím asi nasledovne:

for i=1 to n begin
//čierna fronta
for j=i+x to i+y
ak sa viem na stĺpec j dostať z políčka i s menším počtom políčok ako som sa tam dovtedy vedel dostať s rovnakou farbou , tak zapíš na toto políčko koľko políčok som potreboval zmeniť aby som sa sem dostal.
//biela fronta
to isté ako v čiernej – určite som v kóde nepoužil ctrl+c
end

ak sa viem na koniec čiernej fronty dostať s menším počtom nahradených pixlov ako v bielej tak vypíšem počet týchto políčok, inak vypíš počet z bielej fronty
jednoducho min(čierna, biela)

ako však zistím koľko políčok potrebujem nahradiť aby som sa dostal jedným pásom z políčka i na políčko j ??

no viem to spočítať počet čiernych (tie mám uložené v poli)v tomto stĺpci ak ich chcem nahradiť bielymi, prípadne počet bielych ktoré vypočítam ako počet všetkých - počet čiernych

keďže z jedného políčka sa viem dostať na y-x rôznych pozícii to aby som spočítaval x+x+1+x+2+x+3..y ?? nie nemusím (preto že ho už vidím :-)) pretože ak si spočítam stĺpce po x, tak jednotlivé pásy so šírkou x a x+1 sa líšia len jedným pásikom – stĺpcom

takže ak si spočítam po x, tak potom mi stačí pripočítať jeden stĺpec a mám x+1 potom znova 1 stĺpec a mám x+2 atd.

Takže čas bude y – pretože každý stĺpec po y pripočítam raz.
Dá sa to rýchlejšie ??
Ano ak by som už mal predpočítané po x

Vieme že súčet stĺpcov od 1 po x a 2 až x+1 sa líšia len 2 stĺpcami – jeden odpočítame a jeden pripočítame, takže pre každý taký súčet (je ich n-x)dĺžky x potrebujeme konštantný počet operácii (2)
takže dokopy čas potrebujeme (s-x)+x=s krokov (krát 2 ??)
takže zložitosť časovú ani pamäťovú nám to nepokazí, tak prečo by sme to nespravili.

Takže vráťme sa k počítaniu počtov nahradených políčok kde sa vieme dostať z daného políčka a tom nám bude trvať y-x času

keďže každé políčko preveríme na kde sa z neho vieme dostať tak zložitosť bude s*(y-x)

načítanie nám bude trvať r*s a výpis konštantu

Takže výsledná časová zložitosť bude O(r*s+s*(y-x))

Čo si musíme pamätať ??
No pre každý stĺpec si pamätáme :

koľko čiernych políčok obsahuje
na koľko nahradených políčok sa vieme sem dostať
súčet x políčok ktorý tu začína

všetko sú to len čísla, takže pamäť bude O(s)

Prečo dá toto riešenie vždy správny výsledok ??
určite vieme, že ak sa vieme dostať na každé políčko najkratšou cestou, tak sa vieme dostať aj na posledné políčko najkratšou cestou, takže vieme nájsť najmenej nájdených políčok.