Hlavná logika je že pôjdem od konca a prepočítam si dynamickým programovanim pre každé vypočitam a zapamätám.

Začneme tým že pôjdem od konca so šlapkami čo sú na "začiatočnom" políčkju a že koľko mi to bude trvať. Pri tom sú 3 možností

1- krokom prejdem do ciela tak si zapisem ze mi to trvalo b[i]=kroky alebo ak som pred tým stupil na lepšie políčko a bolo to lepšie tak to.

2- stúpim na políčko s rovnakymi slapkami tak si zapíŠem b[i]=kroky+b[pozícia] (kedže uz to mám vypočitane s tymito slapakami) alebo ak

som pred tým stupil na lepšie políčko a bolo to lepšie tak to.

3- stupim na políčko a zístim že kolko mi bude trvat keby som sa prezul a zapišem si (ak stúpim na ďalšie zapisem si min()).

takto prejdem cele pole.

Keď sa pozrieme na časovú zložitosť tak si môžeme všimnúť že sa pohybujem rychlosťou šlapiek resp mi to bude trvať n/šlapky.

najhoršie by bolo keby som mal šlapky 1 ale ak by som stretol ďalšie 1-kové tak to už mám predpočítanie takže by som prešiel dopkopy n.

Ale najhoršie by bolo keby neboli rovnake -> napr. 1 2 2 3 3 3 takto nestúpim a so všetkymi musim dojsť do konca ale s 2 prejdem iba n/2,

s 3 prejdem n/3.

Najhoršie možnosť je že by to išlo takto od najmenšich ale môŽeme si všimnuť že toto mi trva iba n*3 resp najväčšie čislo (v tejto najhoršiej postiupnosti

ostatne budú mať lepší čas)(keby tam bolo napr ine číslo namiesto 3-ky ak menšie tak 1 alebo 2 sa "prepojí"s dalšou do iba n resp n/2 ak väšie tak bude lepsie

lebo namiesto n/3 bude napr n/4) takze vyjadrime si najväčšie n=(1+posledne)*posledne/2 čo je všeobecná rovnica pre takýto súčet. takze si vyjadrim

posledne=1+-√1+8n / 2 ked odstranim konštanty posldené=√n Takže zložitost je n*√n

časová zložitosť- o(n*√n).

pamäťová zložitosť- pamätáme si iba vstup a dynaiku o(n)