**2.**

Program využíva vlastnosť, že ak majú byť v poli veľkosti všetky čísla od do , tak hodnoty v poli môžeme použiť aj ako indexy do toho istého poľa. Takéto pole sa dá potom chápať ako orientovaný graf, kde z každého bodu vedie hrana do bodu . Takýto graf môže mať niekoľko podôb:

1. Obsahuje niekoľko komponentov, pričom každý z nich je tvorený iba jedným kompletným orientovaným cyklom. To zaručí, že do každého vrchola vchádza práve jedna hrana, čo je ekvivalentné s tvrdením, že každé číslo sa v poli nachádza práve raz. Toto je teda príklad správnych hodnôt v poli.
2. Existuje vrchol, do ktorého vchádzajú dve alebo viac hrán. To znamená, že aspoň v dvoch políčkach poľa je rovnaká hodnota, čiže pole nespĺňa požiadavky.
3. Medzi vrcholmi neexistuje taký, do ktorého vchádzajú dve alebo viac hrán, avšak je medzi nimi taký, do ktorého nevchádza žiadna hrana. Nakoľko z každého vrchola nejaká hrana vychádza (žiadne políčko poľa nie je prázdne), tak hrán je , a keďže do niektorého z  vrcholov nevchádzala hrana, tak musela táto hrana ísť do vrchola mimo tohto intervalu. To znamená, že v poli sa nachádza číslo , a teda sa v ňom podľa Dirichletovho princípu už nemôžu nachádzať všetky čísla od do .

Daný algoritmus nasleduje cesty v takomto grafe a sleduje či nenastal jeden z prípadov 2 alebo 3. Hodnotou si označuje, že z daného vrchola už vychádzala hrana, a nemá touto hranou pokračovať aby sa nezacyklil. Premenná i označuje v ktorom vrchole začal prehľadávať komponent grafu, a teda aj vrchol v ktorom musí skončiť, ak sú čísla v poli korektné (každé sa vyskytuje práve raz, prípad 1). Podmienka j != i testuje či som naozaj prehľadávanie komponentu skončil v tom istom vrchole, ako som začal – ak je nesplnená tak som „prešiel“ do už existujúceho cyklu, čím nastal prípad 2. Podmienka a[j] >= n zas testuje prípad 3.

Algoritmus teda sleduje trasu v komponente grafu, a keď už v rámci komponentu uzavrie cyklus, lineárnym prehľadávaním nájde prvý vrchol, ktorý ešte nepatrí žiadnemu komponentu (posledný while cyklus), a opäť v ňom sleduje trasu. Prejdenie trás vo všetkých komponentoch zaberie presne prechodov (pretože graf má hrán), čiže je . Hľadanie ďalšieho vrchola, ktorý nepatrí komponentu prebieha lineárne od začiatku po koniec poľa, pričom nikdy „necúva“ (i sa vždy iba zväčšuje), teda tiež . Celková zložitosť je teda .

Aby ľubovoľný algoritmus detekoval napr. prípad č. 3 (v poli je číslo ), musí nutne aspoň raz prejsť všetky prvky poľa. To znamená, že nemôže existovať algoritmus, ktorý by úlohu riešil rýchlejšie ako . Nakoľko uvedený algoritmus sme odhadli na , a optimálny algoritmus je taktiež aspoň , nemôže existovať nižší odhad uvedeného algoritmu, a teda odhad je asymptoticky tesný .