1-INF-220 Algoritmy a dátové štruktúry, ZS 2026/27
riešenie 1. domácej úlohy

1. príklad

Označme $w_i = p_i - c_i$ a uvažujme prefixové sumy $z_0 = 0$ a $z_i = \sum_{j=0}^{i-1} w_i$ pre $1 \leq i \leq n$. Podmienka zo zadania hovorí, že $z_n \geq 0$. Neformálne povedané, z_i udáva, koľko paliva by bolo v nádrži po príchode do mesta i, ak začíname z 0 a cestou dovolíme aj záporné palivo.

Lema 1. *Nech cesta začína v meste i a dostane sa do mesta j. Ak $i < j < n$, tak stav nádrže po príchode do mesta j je $z_j - z_i$. Ak sa cesta dostane cyklicky do mesta 0 a pokračuje ďalej, tak po príchode do mesta j, $0 \leq j \leq i$, bude stav nádrže $z_n - z_i + z_j$.*

Dôkaz. Pre $i < j < n$ indukciou cez j. Báza pre $j = i+1$: po príchode do mesta $i+1$ je v nádrži $w_i = z_{i+1} - z_i$. Indukčný krok pre $i + 1 < j < n$: ak je po príchode do mesta $j-1$ v nádrži $z_{j-1} - z_i$, tak po príchode do j je $z_{j-1} - z_i + w_{j-1} = z_j - z_i$.

Z predchádzajúceho máme, že po príchode do $n-1$ je v nádrži $z_{n-1} - z_i$. Preto po príchode do 0 je v nádrži $z_{n-1} - z_i + w_{n-1} = z_n - z_i$. Pre $0 < j \leq i$ je indukcia analogická ako pre $i < j < n$. □

Veta 1. *Nech $z_n \geq 0$. Zvoľme k tak, že $z_k = \min\{z_0, \ldots, z_n\}$. Potom sa z mesta k dá obísť celý kruh.*

Dôkaz. Pomocou Lemy 1 ukážeme, že po príchode do každého mesta j je v nádrži nezáporné množstvo paliva. Pre $k < j < n$ je v nádrži $z_j - z_k \geq 0$, lebo z_k je minimum. Pre $0 \leq j \leq k$ je v nádrži $z_n - z_k + z_j = z_n + (z_j - z_k) \geq z_n \geq 0$. □

Program zo zadania vyzerá takto:

```
1  start, nadrz = 0,0
2
3  for i in range(n):
4      nadrz += p[i] - c[i]
5      if nadrz < 0:
6          start = (i + 1) % n
7          nadrz = 0
8
9  print(start)
```

Označme $\mathtt{d_i}$ (resp. $\mathtt{s}_i$) hodnotu premennej `nadrz` (resp. `start`) na začiatku otočenia cyklu s hodnotou i (medzi riadkami 3-4). Hodnoty premenných `nadrz` a `start` po skončení cyklu označme $\mathtt{d_n}$ a $\mathtt{s}_n$.

Invariant 1. *Pre všetky i, $0 \leq i \leq n$ platí*

(1) $\mathtt{s}_i \leq i$
(2) $\mathtt{d_i} = z_i - z_{\mathtt{s}_i}$
(3) $z_{\mathtt{s}_i} = \min\{z_0, \ldots, z_i\}$

Dôkaz. Indukciou cez i. Báza: pre $i = 0$ je $\mathtt{s}_0 = \mathtt{d}_0 = \min\{z_0, \ldots, z_0\} = 0$. Indukčný krok: nech invariant platí pre $i < n$. Po riadku 4 bude podľa (2): $\mathtt{nadrz} = \mathtt{d_i} + w_i = z_i - z_{\mathtt{s}_i} + w_i = z_{i+1} - z_{\mathtt{s}_i}$. Ak $\mathtt{nadrz} < 0$, vykonajú sa riadky 6-7, t.j. $\mathtt{s}_{i+1} = (i+1) \mod n$ a $\mathtt{d_{i+1}} = 0$. Všimnime si, že tento prípad nemohol nastať ak $i = n-1$, lebo v tom prípade by $\mathtt{nadrz} = z_n - z_{\mathtt{s}_i} \geq 0 - z_{\mathtt{s}_i}$. Lenže podľa (3) je $z_{\mathtt{s}_i} \leq z_0 = 0$, takže $\mathtt{nardz} \geq 0$. Preto v prípade $\mathtt{nadrz} < 0$ platí $\mathtt{s}_{i+1} = i + 1$ a $\mathtt{d_{i+1}} = 0 = z_{i+1} - z_{\mathtt{s}_{i+1}}$. Navyše, keďže $\mathtt{nadrz} = z_{i+1} - z_{\mathtt{s}_i} < 0$, je $\mathtt{s}_{i+1} = z_{i+1} < z_{\mathtt{s}_i} = \min\{z_0, \ldots, z_i\}$ podľa (3), a teda $z_{i+1} = \min\{z_0, \ldots, z_{i+1}\}$.

Naopak, ak $\mathtt{nadrz} > 0$, riadky 6-7 sa nevykonajú a ostane $\mathtt{s}_{i+1} = \mathtt{s}_i \leq i$, $\mathtt{d_{i+1}} = \mathtt{nadrz} = z_{i+1} - z_{\mathtt{s}_i}$, a keďže $\mathtt{nadrz} = z_{i+1} - z_{\mathtt{s}_i} > 0$, je $z_{i+1} > z_{\mathtt{s}_i}$ a teda $z_{\mathtt{s}_i} = \min\{z_0, \ldots, z_{i+1}\}$. □

Veta 2. *Uvedený algoritmus je správny.*

Dôkaz. Z Invariantu 1 vyplýva, že $\mathtt{s}_n = \min\{z_0, \ldots, z_n\}$ a z Vety 1 dostávame, že zo `start` sa dá obísť celý kruh. □

2. príklad

Ponecháme si označenie w_i, z_i z prvého príkladu. Z Lemy 1 vidíme, že ak vyrazíme z mesta i a palivo sa nám minie na ceste do mesta $j \leq n-1$, tak j je prvé mesto, pre ktoré $z_j < z_i$. Do mesta j nás dovezie odťahovka a odtiaľ pokračujeme opäť kým sa neminie palivo. Keďže hľadať najbližší menší prvok v poli z budeme potrebovať

veľakrát, môžeme si ich predpočítať do poľa `nxt` algoritmom z prednášky. Takže stačí urobiť niečo ako `nxt[nxt[i]]` a vieme, kde sa nám druhýkrát minie palivo.

Potrebujeme ešte vyriešiť cesty, ktoré prídu na koniec a pokračujú z mesta 0. Na to stačí zobrať dve kópie poľa w. Teraz za každým mestom i, $0 \leq i < n$, máme jeho n nasledovníkov. Ak z mesta i neobehneme celý kruh, palivo sa nám minie skôr ako po n cestách, takže nám stačí rátať prefixové sumy a najbližší menší prvok nad týmto zdvojeným poľom. Zároveň z mesta i vieme obehnúť celý kruh práve vtedy, ak neexistuje najbližší menší prefix. Keď nájdeme $j =$ `nxt[i]`, druhýhrát sa pozrieme na `nxt[j mod n]`

```
from itertools import accumulate
import sys

# precitame vstup
n=int(input())
p = list(map(int,input().split()))
c = list(map(int,input().split()))

w = [x - y for x, y in zip(p, c)]*2  # vyratame w a spravime dve kopie
z = [0] + list(accumulate(w))        # prefixove sumy
nxt = [-1] * (2 * n + 1)             # tu bude najblizsi mensi prefix (alebo -1)

# najblizsi mensi prvok z prednasky
stack = [2 * n]
for i in range(2 * n - 1, -1, -1):
    while stack and z[stack[-1]] >= z[i]:
        stack.pop()
    if stack:
        nxt[i] = stack[-1]
    stack.append(i)

# pre kazde i najdeme nxt(nxt(i))
maxd,start = 0,0
for i in range(n):
    if nxt[i] < 0 or nxt[nxt[i] % n ] < 0: # obehneme cely kruh ?
        maxd, start = n, i
        break
    else:
        j = nxt[i] % n
        d = min(n, nxt[i] - i + nxt[j] - j - 1) # -1, lebo do nxt[j] sme uz nedosli
        if d > maxd:
            maxd, start = d, i

print(f"{maxd} {start}")
```