Implementáljunk okosan

EGOI felkészítő tábor 2024

Készítette: Éles Júlia

Bevezetés

- **Legtöbb esetben az ötlet a fontos, az implementáció másodlagos**
- **Ettől függetlenül érdemes mintákat, trükköket megjegyezni, mert vannak olyan feladatok, ahol nagyon el lehet csúszni egy rosszul megszervezett implementációval – még ha az ötlet egyszerű is**

Gyakori hibák és elkerülésük

- **Rekurzió:**
  - Ha nem gondoljuk át, hogy mi a megállási feltétel, akkor könnyen végtelen rekurziót eredményez, ami minimum a memóriát nyírja ki
  - Kezdjük a megállási feltétellel!
  - Példa1: gráfban A-ból B-be vezető út megadása rekurzívan
  - Példa2: bináris keresés rekurzívan

Gyakori hibák és elkerülésük

- **While ciklus:**
  - Picit hasonlít a rekurzióhoz: rosszul átgondolt megállási feltétel, általában a memóriával nincsen gond, de időlimit az lesz (vagy épp nem :))
  - Először a változó léptetésekkel kezdjük!
  - Példa1: “ugráltatás”
  - Példa2: 2 pointer módszer

Függvényhasználat – ahogyan én szoktam

- Feladatoknál előfordul, hogy ugyanazt a függvényt különbözőféleképpen lehet megírni
- Ilyenkor hasznos lehet, ha még azelőtt használod, mielőtt megírnád – így képet kapsz arról, hogy mit is csináljon a függvény
- Leginkább az dől el, hogy mit adjon vissza a függvény
- Példa: A-tól legmesszebb lévő csúcs meghatározása

Példa – 1956A(Nene’s Game)

- N ember áll egy sorban, illetve adott egy M elemű A tömb
- Minden körben a sorból kiesnek azok az emberek, akiknek a sorszáma benne van az A tömbben
- Ha senki sem esik ki, véget ér a játék, a bentmaradtak a nyertesek
- Hány nyertesünk van?

Példa – 1956A(Nene’s Game)

- **Az ötlet nem nehéz: egyszerűen le kell szimulálni a játékot**
- **Nehézségek:**
  - Nyilván kell tartani, ki van még bennt az adott körben
  - Ki kell ejteni a megfelelő embereket
  - Ellenőrizni kell, hogy van-e még értelme folytatni
- **Ötletek?**

Példa – 1956A(Nene’s Game)

- **Általános megközelítés: két vektor, amiket váltogatunk a körök közben**
- **Kicsit szebb: egy kételemű tömb, melynek elemei vektorok**
- **Implementáció**
- **Javítási ötlet?**

Példa – 1956A(Nene’s Game)

- Kell az x vektor? Miért?
- Kell az r változó? Miért?
- Jobb változat
- Fontos: általában szükség van az x-re és az r-re

Példa – 1956A(Nene’s Game)

- **Tanulságok:**
  - A legtöbb feladattípusnak vannak “típusimplementációik” – ezeket érdemes gyakorolni
  - Ha valamilyen információra nincs szükségünk, akkor azt ne is tároljuk

Példa – 1976C (Job Interview)

- Egy cég fel akar venni P programozót és T tesztelőt
- Van P + T + 1 jelöltünk, mindenkinek ismerjük a programozó és a tesztelő skilljét (különböznek)
- A felvétel úgy zajlik, hogy végigmegyünk a jelölteken, és ha van még azon a pozíción hely, amiben ő jobb, akkor arra vesszük fel. Ha már nincs, akkor a másikra. A kiválasztás összege a felvettek releváns skilljeinek összege
- Mindenkiről meg kell mondani, hogyha őt kihagyjuk, akkor a kiválasztásnak mennyi lenne az összege

Példa – 1976C (Job Interview)

- **Ötlet:**
  - Ha az utolsó ember marad ki, elég csak szimulálni - trivi
  - Nézzük meg, hogy mennyi lenne az összeg, ha mindenkit arra a pozira vennénk fel, amiben ő jó, és számoljuk, hogy ez mennyivel csökken a megadott feltételek mellett
  - Ha valakit kihagyunk, akkor az ő jobbik skilljével biztosan csökken, emelett tudjuk, hogy hány ember kerülne valamelyik pozícióra
  - Ha hiányzik valamelyikből, mondjuk k db programozó, akkor az utolsó k db tesztelőről tudjuk, hogy ők programozók – tehát annyival csökken az összeg, amennyi a két skill közötti különbség ezeknek az embereknek; a többieket meg felvesszük a jobb skillekre

Példa – 1976C (Job Interview)

- **Implementáció:**
  - Az eredeti összeg számítása triviális, inkább az “utolsó k db tesztelő programozó kell legyen” résszel van a baj
  - Ötlet: számoljunk két vektort: az egyikben legyen az, hogy a programozó→tesztelő átváltásnál mennyit veszítünk, ha az első I programozót váltjuk át, a másikban ugyanez fordítva
  - Figyelni kell arra, hogy ha az utolsó k-ban benne van az épp kihagyott személy, akkor csúsztatni kell

Példa – 1976C (Job Interview)

- **Hogyan ne csináljuk :’(**
- **Mi a baj az implementációval?**
  - ~~gyakorlatilag minden~~
  - Sok az ismétlődés, bele lehet bonyolódni a változókba
  - Versenyen kb 1 órán keresztül implementáltam
- **Min lehet javítani?**
- **Egy szebb változat**

Példa – 1976C (Job Interview)

- **Tanulságok:**
  - Próbáljunk mintákat felfedezni a számolási menetben
  - Általában ha a számolásban csak az változik, hogy melyik adatokkal számolunk, de ugyanazt csináljuk velük, akkor érdemes indexelni őket, és 1 függvényben megírni őket, aminek egy paramétere az, hogy akkor most melyik adattal számolunk

Általános jótanácsok

- **Beszédes változó- és függvénynevek**
- **Függvénybe szervezés**
- **Változók szerepének leírása szóban pl. dp[i][j] – hány pont gyűjthető az (i, j) pozícióig**
- **Algoritmus lépéseinek leírása szóban pl. rendezzük távozási idő szerint, majd…**
- **Maradj konzisztens, ne a versenyen kísérletezz**
- **Jó ha van sablonkódod (nekem is van), de ne legyenek alapalgoritmusok előre implementálva benne**

Módszerek

- **Gyakorlás közben:**
  - Miután megoldottunk egy feladatot, tegyük fel a következő kérdéseket:
    - Össze tudok vonni részleteket egy egyszerűbb függvénybe?
    - Minden adatra, részeredményre szükségem van?
    - Meg lehet egyszerűbben is csinálni? Ha igen, akkor mi vezet ahhoz, hogy egyszerűbb legyen (beépített függvény, adatszerkezet stb)?
    - Mi lenne, ha az X adatszerkezet helyett Y-t használnék?
  - Feladatmegoldás közben először írjuk le papírra, hogy mikkel számolnánk, lehet észreveszünk valamilyen mintát
  - Amennyiben van lehetőségünk, nézzük meg mások implementációját is – akik a listák élén vannak, általában profik ebben. Hasonlítsuk össze a sajátunkkal

Módszerek

- **Verseny közben:**
  - A papírra leírás itt is működik – de nem az utolsó fél órában
  - Ne kapkodjunk, ne kezdjünk el azonnal kódolni
  - Ökölszabály: ha valamit már másodjára írnál le → függvénybe vele
  - Tartsd magad a konvencióidhoz – ha eddig dp-nek nevezted a gráf szomszédsági listáját, akkor ne a versenyen kezdd el a dinamikus tömbre használni

Köszönöm a figyelmet!