Kazi Sándor
KAZI(AT)CS.BME.HU

Dinamikus programozás - BFS

1	$(\log n)^k$, ahol $k > 0$	n	$n \cdot \log n$	n^k, ahol $k \geq 2$	c^n, ahol $c > 1$	$n!$	n^n

1. Melyik az a legkisebb egész k, amelyre igaz, hogy az alábbi kifejezés $O(x^k)$?
   (a) $2x^2 + x^3 \log x$
   (b) $(x^4 + x^2 + 1)/(x^4 + 1)$
   (c) $(x^3 + 5 \log x)/(x^4 + 1)$
   (d) $(8x)!$

2. Igaz-e, hogy ha $|a| < |b|$, akkor $2^{|a|} < 2^{|b|}$ $(a, b \in \mathbb{R})$? Igaz-e, hogy ha $f(n) = O(g(n))$, akkor $2^{f(n)} = O(2^{g(n)})$.

3. Legyen $w = w_1 w_2 \cdots w_n$ egy n betűből álló szó. Hívjuk részszónak w egy tetszőleges $w_i w_{i+1} \cdots w_{i+k}$ darabját $(1 \leq i \leq n-1$, $1 \leq k \leq n-i)$. Adjon algoritmust, ami $O(n)$ lépésben meghatározza az összes a-val kezdődő és b-re végződő részszó számát.

4. Egy n és egy m karakterből álló szövegben meg akarjuk találni a legnagyobb azonos darabot, azaz ha az egyik szöveg $a_1 a_2 \cdots a_n$ és a másik $b_1 b_2 \cdots b_m$, akkor olyan $1 \leq i \leq n$ és $1 \leq j \leq m$ indexeket keresünk, hogy $a_{i+1} = b_{j+1}, a_{i+2} = b_{j+2}, \ldots, a_{i+t} = b_{j+t}$ teljesüljön a lehető legnagyobb t számra. Adjon erre a feladatra $O(mn)$ lépést használó algoritmust.

5. Hogyan járja be a BFS (*breadth-first-search*) a K_n és a $K_{n,n}$ gráfokat?

6. Adjunk algoritmust, ami egy n csúcsú fában lineáris időben meghatározza a fában levő leghosszabb út hosszát.

7. Legyen F egy n csúcsú fa, tetszőleges v csúcsának súlya pedig $s(v)$. Adjunk polinom idejű algoritmust, amely meghatározza az F fában található legnagyobb összsúlyú független ponthalmaz súlyát.

8. (*gondolkodtató*) Egy n szóból álló szöveget kell sorokra tördelni. A szöveg i-edik szava ℓ_i karakterből áll, egy sor s karakter hosszú. Ha egy sor a szöveg i-edik szavával kezdődik és a j-edik szóval végződik, akkor az elválasztó szóközöket is figyelembe véve $t = s - (\ell_i + \ell_{i+1} + \cdots + \ell_j + j - i)$ üres hely marad a sor végén. Egy ilyen sor hibája legyen t^2. A tördelés hibája a nem üres sorok hibáinak összege. Adjon $O(n^2)$ lépéses algoritmust egy legkisebb hibájú tördelés meghatározására! (A szavak sorrendje rögzített.)

9. Egy játékban egy $n \times m$ rács bal felső sarkából kell eljutnunk a jobb alsó sarokba. Egy lépés során a rács mentén vizszintesen vagy függőlegesen tudunk a következő rácspontba lépni. Azonban van néhány kereszteződés, ahova nem szabad lépnünk. Ezek helyét az R tömb írja le, $R[i, j] = 1$, ha az (i, j) kereszteződésbe nem léphetünk, egyébként $R[i, j] = 0$. Adjon $O(nm)$ futási idejű algoritmust annak meghatározására, hogy pontosan $n + m - 2$ lépést téve a rácson hányféleképpen tudjuk a célt elérni.

10. (*gondolkodtató*) *Levenshtein-távolság:* Adott két füzér, melyek rendre az $a_1, a_2, \ldots, a_n$ és a $b_1, b_2, \ldots, b_m$ karakterekből állnak. Határozzuk meg annak a legrövidebb műveletsornak a hosszát, amely az egyik füzért a másikba viszi át, ha a megengedett (azonosan egy költségű) műveletek: karakter törlés, karakter átírás, karakter beszúrás. Adjunk $O(n \times m)$ költségű algoritmust. Mire lehet ez a mérőszám jó?

11. Panka és Janka 1000 piros és 1000 zöld koronggal a következő játékot játssza. Kezdetben mindkettőjüknek 1000-1000 korongja van, tetszőleges eloszlásban. Ezután Panka minden lépésben mond egy x számot 1 és 500 között, mire Jankának vagy x darab piros korongját kell zöldre cserélnie Pankával, vagy x darab zöldet pirosra. (Így tehát egyik lánynak x-szel több, a másiknak x-szel kevesebb korongja lesz.) Janka célja, hogy a játék végén éppen 500 piros korongja legyen. Tegyük fel, hogy előre ismerjük a Panka által megadott számok $x_1, x_2, \ldots, xn$ sorozatát, valamint azt, hogy kezdetben hány piros illetve zöld korong van Jankánál. Adjon $O(n)$ lépésszámú algoritmust, amely ez alapján eldönti, hogy Janka elérheti-e a célját! *(Vizsga 2009.01.14.)*

12. Adottak $t_1, t_2, \ldots, t_n$ tárgyak, az i-edik tárgy súlya egy s_i egész szám $(0 < s_i < 100)$. Ezeket a tárgyakat a sorrendet megőrizve (azaz mindig csak a soron következő elemet elhelyezve) ládákba kell pakolnunk, minden ládába összesen legfeljebb 100 súlyú tárgyakat rakhatunk. Minden láda után adót kell fizetnünk: ha egy ládába s összsúlyú tárgyakat rakunk, akkor $(100 - s)^2$ gyémántfélkrajcárt kell fizetnünk. Adjunk algoritmust, amely meghatároz egy legolcsóbb pakolást $O(n^2)$ időben!

13. Adottak $M_1, M_2, \ldots, M_n$ munkák $H_1, H_2, \ldots, H_3$ határidőkkel és $P_1, P_2, \ldots, P_n$ profitokkal. Mindegyik munka elvégzése pontosan 1 napunkba kerül (egy-egy napon csak egy munkát végezhetünk el). Adjunk hatékony algoritmust, amely meghatározza, hogy mely munkákat vállaljuk el a profit maximalizálása érdekében!

14. Éllistával adott a súlyozott élű $G = (V, E)$ gráf. Tegyük fel, hogy az élek súlyai az 1,2,3 számok közül valók. Javasoljunk $O(n + e)$ uniform költségű algoritmust az $s \in V$ pontból az összes további $v \in V$ pontokba vivő legrövidebb utak hosszának a meghatározására. (Itt n a G gráf csúcsainak, e pedig az éleinek a száma.)