1. Az alábbi algoritmusban lépésnek az értékadásokat, az összeadásokat, a szorzásokat és az eredmény visszaadását tekintjük.

Input: egy $n \geq 2$ darab számot tartalmazó T tömb

ciklus: i **fut** 1**-től** 2025**-ig**
  ciklus: j **fut** i**-től** $(n-1)$**-ig**
    $T[j] \coloneqq i * T[j]$
  ciklus vége
ciklus vége
`osszeg` $\coloneqq 0$
ciklus: k **fut** 0**-tól** $(n-1)$**-ig**
  `osszeg` $\coloneqq$ `osszeg` $+ T[k]$
ciklus vége
return `osszeg`

(a) Mutassuk meg, hogy az algoritmus lépésszáma $O(n)$.

Az első, külső ciklus magja 2025-ször, azaz $O(1)$-szer fut le.
A belső ciklus magja legfeljebb n-szer (avagy $O(n)$-szer) fut le.
A belső ciklus magján belül 2, azaz $O(1)$ műveletet végzünk: egy szorzást és egy értékadást.
Ezután van egy értékadás, ami $O(1)$ lépés.
A harmadik ciklus magja is n-szer (avagy $O(n)$-szer) fut le.
Ennek a ciklusnak a magján belül 2, azaz $O(1)$ műveletet végzünk: egy összeadást és egy értékadást.
Végül visszaadjuk az eredményt, ami $O(1)$ lépés. *(5 pont)*
Ez összesen

$$O(1) \cdot O(n) \cdot O(1) + O(1) + O(n) \cdot O(1) + O(1)$$

lépés, azaz az előadáson tanultak szerint $O(n)+O(1)+O(n)+O(1) = O(n)$ lépés. *(3 pont)*

(b) Mutassuk meg, hogy az algoritmus lépésszáma $O(n^3)$.

Az előadáson tanultak szerint minden $O(n)$-es lépésszámú algoritmus egyben $O(n^3)$-ös is. *(2 pont)*

2. Adott egy n darab számot tartalmazó T tömb, és egy m darab számot tartalmazó S tömb. Adjunk $O(n+m)$ lépésszámú algoritmust annak eldöntésére, hogy igaz-e, hogy a T tömb minden eleme nagyobb-e, mint az S tömb minden eleme, vagy a T tömb minden eleme kisebb-e, mint az S tömb minden eleme.

   (a) Algoritmus leírása (pszeudokóddal vagy szöveggel):

   Az előadáson tanult algoritmussal (és annak értelemszerű, a gyakorlaton látott módosításával) keressük meg a T, illetve az S tömbben is a legkisebb és a legnagyobb elemet: legyenek ezek $t_{\min}$, $t_{\max}$, $s_{\min}$, $s_{\max}$.
   Ha $t_{\min} > s_{\max}$ vagy $t_{\max} < s_{\min}$, akkor a válaszunk `igaz`, különben `hamis`.
   (Természetesen más helyesen leírt algoritmus is elfogadott.) *(5 pont)*

   (b) Algoritmus helyességének bizonyítása:

   Ha $t_{\min} > s_{\max}$, akkor a T tömb minden eleme nagyobb lesz az S minden eleménél, (hiszen $T[i] \geq t_{\min} > s_{\max} \geq S[j]$ minden szóba jövő i és j indexre); ha viszont $t_{\min} \leq s_{\max}$, akkor ez nem lesz igaz.
   Hasonlóan, ha $t_{\max} < s_{\min}$, akkor a T tömb minden eleme kisebb lesz az S minden eleménél, ellenkező esetben viszont ez nem teljesül.
   Vagyis minden esetben jó választ adtunk. *(2 pont)*

   (c) Lépésszám meghatározása (indoklással):

   A T, illetve az S legkisebb és legnagyobb elemének megkeresése $2 \cdot O(n) + 2 \cdot O(m)$ lépés (hiszen egy n-elemű tömb legkisebb elemét $O(n)$ lépésben meg tudjuk találni, és hasonlóan a legnagyobb elemet is).
   Ezután elvégzünk legfeljebb két összehasonlítást, és visszaadjuk a végeredményt, ami $O(1)$ lépés.
   Ez összesen $2 \cdot O(n) + 2 \cdot O(m) + O(1) = O(n+m)$ lépés. *(3 pont)*

3. Adott egy n darab különböző (nem feltétlenül egész) számot tartalmazó **rendezett** tömb. Adjunk $O(\log_2 n)$ lépésszámú algoritmust annak eldöntésére, hogy a 2, 37, 101, számok mindegyike szerepel-e a tömbben.

   (a) Melyik tanult algoritmus segítségével (és azt milyen bemeneteken futtatva) lehet megoldani ezt a feladatot?

   Három darab bináris keresést futtatunk, melyekkel az adott tömbben keressük a 2, a 37, illetve a 101 számokat. *(4 pont)*
   Ha mindhárom elemet megtaláljuk, akkor a válaszunk `igen`, különben pedig `nem`. *(2 pont)*

   (b) Lépésszám meghatározása (indoklással):

   Az előadáson tanultak szerint a bináris keresés lépésszáma $O(\log_2 n)$. *(3 pont)*
   Vagyis három darab bináris keresés futtatása és azok eredménye alapján a feladat kérdésére a válasz meghatározása összesen

   $$3 \cdot O(\log_2 n) + O(1) = O(\log_2 n)$$

   lépés. *(1 pont)*

4. Az alábbi tömbön, ami csupa különböző egész számot tartalmaz, futtattuk a kiválasztásos, illetve beszúrásos rendezést is.

5	x	9	3	7	1

A kiválasztásos rendezést futtatása közben valamikor az

1	3	9	x	7	5

közbülső állapotot láttuk, míg a beszúrásos rendezés során valamikor a

x	5	9	3	7	1

állapotot láttuk. Határozzuk meg az x egész szám összes lehetséges értékét (és indokoljuk meg, hogy x értéke nem lehet más).

A kiválasztásos rendezés először megkeresi a legkisebb elemet a tömbben, ami az 1 vagy az x, és ezt kicseréljük a tömb 0-s indexű cellájában lévő elemmel. Mivel ezt az elemet többet nem mozdítjuk el innen, ezért $1 < x$.

1	x	9	3	7	5

Ezután az így kapott T tömb $T[1 : n-1]$ résztömbjében megkeressük a legkisebb elemet, ami a 3 vagy az x, és ezt kicseréljük a tömb 1-es indexű cellájában lévő elemmel. Mivel ezt az elemet többet nem mozdítjuk el innen, ezért $3 < x$.

1	3	9	x	7	5

Ezzel meg is kaptuk a feladatbeli közbülső állapotot. *(5 pont)*

A beszúrásos rendezés először a bemenetként kapott tömb az 1-es indexű cellájában lévő elemet, azaz az x-et mozgatja egyesével balra addig, amíg vagy az előtte lévő elem kisebb nem lesz nála vagy el nem érjük a tömb elejét. Mivel az x és az 5 elemek felcserélődtek, ezért $x < 5$.

x	5	9	3	7	1

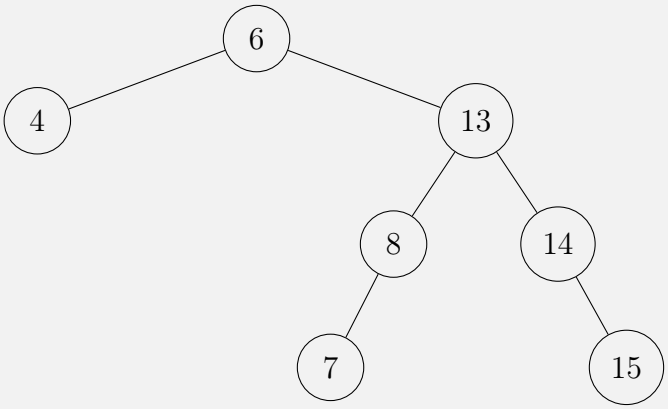
Ezzel meg is kaptuk a feladatbeli közbülső állapotot. *(4 pont)*

Vagyis $3 < x < 5$, így mivel a feladat szövege szerint x egész szám, ezért $x = 4$. *(1 pont)*

5. (a) Építsünk beszúrásokkal bináris keresőfát az alábbi sorrendben érkező számokból:

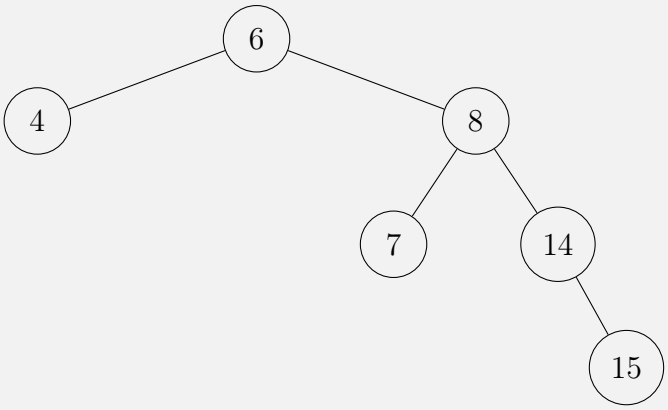
$$6, \quad 13, \quad 8, \quad 4, \quad 7, \quad 14, \quad 15.$$

(Kivételesen nem kell indokolni, hanem elég a végső fát felrajzolni.)



(5 pont)

(b) Töröljük a kapott bináris keresőfából a 13-at.
(Elég a törlés után kapott bináris keresőfát felrajzolni, nem kell indokolni.)



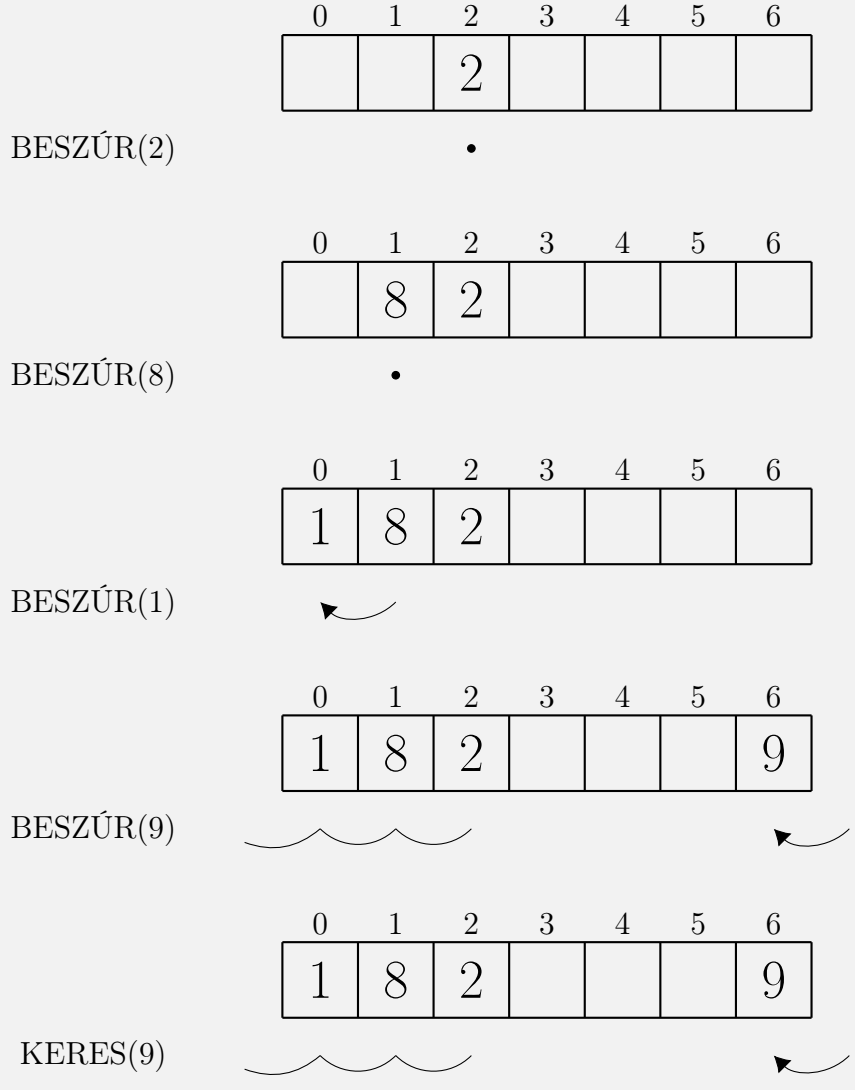
(5 pont)

6. Egy kezdetben üres, 7-méretű hash-táblába nyílt címzéssel, a $h(K) = K \pmod 7$ hash-függvényt és lineáris próbát használva szúrtunk be néhány egész számot. Lehetséges-e, hogy a KERES(9) művelet során a 2, 8, 1, 9 elemeket láttuk ebben a sorrendben?

   (Ha igen, akkor rajzoljunk fel egy ilyen hash-táblát és mutassuk meg, hogy milyen sorrendben szúrhattuk bele a benne tárolt elemeket. Ha nem, akkor indokoljuk meg, hogy miért nem.)

Például az alábbi módon lehetséges. *(0 pont)*

A kezdetben üres hash-táblába a 2, 8, 1, 9 elemeket szúrjuk be ebben a sorrendben (a megadott módon).



(10 pont)