1. Mit csinál a következő program?

Input: n (pozitív egész)
if $n \geq 2$ **then**
  $i := 2$
  while $(i < \sqrt{n})$ **and** $(n$ **MOD** $i \neq 0)$
    i++
  if $(i < \sqrt{n})$ **then**
    print „NEM”
  else
    print „IGEN”
else
  print „1”

2. Van n súlyunk és egy kétkarú mérlegünk, amivel egyszerre egy-egy súlyt tudunk összehasonlítani.

(a) Hány mérésre van szükségünk, hogy biztosan meg tudjuk találni a legnehezebb súlyt?

(b) Hány mérésre van szükségünk, hogy biztosan meg tudjunk találni egyet (akármelyiket) a 10 legnehezebb súly közül?

(c) Hány mérésre van szükségünk, hogy biztosan meg tudjuk találni a legnehezebb és a legkönnyebb súlyt?

(d) Hány mérésre van szükségünk, hogy biztosan meg tudjuk találni a két legnehezebb súlyt?

3. A kincstárban 10 zsákban tárolják a pénzérméket, minden zsákban pontosan 100 érme van. Egy kivételével az összes zsákban szabályos, 1 gramm súlyú érmék vannak, de az egyik zsákban hamis, félgrammos érmék vannak.

(a) Hány méréssel tudjuk megtalálni a hamis zsákot egy egykarú mérleggel, ha nem bonthatjuk ki a zsákokat?

(b) Hány méréssel tudjuk megtalálni a hamis zsákot egy egykarú mérleggel, ha a zsákokból tetszőleges számú érmét kivehetünk?

Definíció: Ha $f(n)$ és $g(n)$ természetes számokon értelmezett, pozitív értékű függvények, akkor $f = O(g)$ azt jelenti, hogy $\exists c, n_0$ számok, amikre $\forall n \geq n_0$ esetén $f(n) \leq c \cdot g(n)$.

4. Igaz-e hogy

(a) $2^{n+1} = O(2^n)$?

(b) $2^{2n} = O(2^n)$?

5. Becsüljük meg az alábbi függvényeket az $O()$ jelöléssel!

(a) $T(n) = 4n^2 - 12n + 450$ (b) $T(n) = n^{10} + 2^n$ (c) $T(n) = n \log n + \log^2 n$

Ha egy modern számítógép közelítőleg 10^9 lépést tud elvégezni másodpercenként, és az alábbi függvények egy-egy algoritmus lépésszáma, akkor nagyjából mekkora n bemenetre tudjuk megoldani a feladatokat 10 másodperc alatt? És ha egy 1000-szer gyorsabb szuperszámítógépen futtatjuk? (A log kettes alapú logaritmust jelöl.)

6. A következő pszeudokódokkal megadott algoritmusok futási idejére írj fel rekurziós képletet, majd oldd meg őket! Add meg $O()$ jelölést használva a futási időt!

Bubble Sort:

```
BubbleSort(arr, n)
    if n == 1
        return
    for i from 0 to n-2
        if arr[i] > arr[i+1]
            swap(arr[i], arr[i+1])
    BubbleSort(arr, n-1)
```

TowerOfHanoi:

```
TowerOfHanoi(n, from_rod, to_rod, aux_rod):
    if n == 0:
        return
    TowerOfHanoi(n-1, from_rod, aux_rod, to_rod)
    Movedisk(n, from_rod,to_rod)
    TowerOfHanoi(n-1, aux_rod, to_rod, from_rod)
```

7. Oldjuk meg a következő rekurziókat! (Feltehetjük, hogy $T(1) = 1$.) Adjunk minél jobb becslést a T függvényre az $O()$ jelölést használva!

(a) $T(n) \leq n + T(n-1)$

(b) $T(n) \leq 2T(n-1) + 2$

(c) $T(n) \leq 5 + T(n-2)$

(d) $T(n) \leq n + 2T(\lceil n/2 \rceil)$

(e) $T(n) \leq \log n + T(n-1)$

(f) $T(n) \leq 3T(\lceil n/2 \rceil)$

Házi feladatok:

8. (1 pont) Írj $O(a+b)$-nél gyorsabb pszeudokódot, ami megadja az a és b számok legnagyobb közös osztóját!

9. (2 pont) Pistike gondolt egy 1 és 16 közötti számra. Egy kérdésben azt kérdezhetjük meg tőle, hogy a gondolt szám benne van-e egy adott számhalmazban (például ,,Nagyobb a gondolt szám 7-nél?" vagy ,,A gondolt szám az 1,2,5 számok valamelyike?"). Mindig elég a válasz után eldöntenünk, hogy mi legyen a következő kérdés. Pistike néha hazudik, de minden barkochba játékban legfeljebb egyszer. Hány kérdésből tudjuk biztosan kitalálni, hogy melyik számra gondolt? *Nem kell bizonyítani hogy a megoldásod optimális, de a teljes pontszám az optimális megoldásra jár, nem optimális megoldással részpontszám szerezhető.*