Euler–Fermat-tétel, a polinomiális algoritmus fogalma

Bevezetés a számításelméletbe 1

3. gyakorlat

2025.

Az Euler-féle φ-függvény.

Ha $n \geq 2$ egész szám, akkor az 1 és n közé eső, n-hez relatív prímek számát $\varphi(n)$ jelöli.

Tétel.

Ha az $n > 1$ egész szám kanonikus alakja $n = p_1^{\alpha_1} \cdot \ldots \cdot p_k^{\alpha_k}$, akkor

$$\varphi(n) = \left(p_1^{\alpha_1} - p_1^{\alpha_1 - 1}\right) \cdot \ldots \cdot \left(p_k^{\alpha_k} - p_k^{\alpha_k - 1}\right).$$

Euler–Fermat-tétel.

Legyenek $m \geq 2$ és a egész számok. Ha $(a, m) = 1$, akkor $a^{\varphi(m)} \equiv 1 \pmod{m}$.

1. Tekintsük azt a számtani sorozatot, amelynek első tagja 32, differenciája 51. Milyen maradékot ad a sorozat első 32 tagjának szorzata 51-gyel osztva?

2. Tekintsük azt a mértani sorozatot, amelynek első tagja 41, kvóciense 7. Mi az utolsó három számjegye a sorozat első 800 tagjának a szorzatának?

3. Legyen $n = 200704261601$. Határozzuk meg n^n utolsó három számjegyét.

4. Bizonyítsuk be, hogy tetszőleges p prímre $(a + b)^p \equiv a^p + b^p \pmod{p}$.

5. Legyen a egy 2001-hez relatív prím egész szám. Bizonyítsuk be, hogy ekkor

   $$(a^{28} - 1)(a^{24} - a^{22} - a^2 + 1)$$

   osztható 2001-gyel. (Segítségül: $2001 = 3 \cdot 23 \cdot 29$.)

6. Legyenek a és b egymáshoz relatív prím, pozitív egész számok. Mennyi maradékot ad az $a^{\varphi(b)} + b^{\varphi(a)}$ szám ab-vel osztva?

7. Egy algoritmus bemenete egy tízes számrendszerben megadott m pozitív egész szám. Az algoritmus egy ciklusból áll, aminek a magja $2m$-szer fut le, és a ciklusmag minden végrehajtásakor kiírunk egy 1-est a képernyőre.

   (a) Határozzuk meg a bemenet méretét.

   (b) Hányszor fut le a ciklusmag?

   (c) Döntsük el, hogy a ciklusmag lefutásainak száma a bemenet méretében polinomiális-e.

   (d) Határozzuk meg a ciklusmagon belül végrehajtott műveletek lépésszámát.

   (e) Határozzuk meg a teljes algoritmus lépésszámát és döntsük el, hogy ez a bemenet méretében polinomiális-e.

8. Az alábbi két C kód közül az első a bemenetként (10-es számrendszerben) kapott a pozitív egész szám négyzetét, a második pedig az a számjegyeinek az összegét számítja ki. Tegyük fel, hogy a kódok végrehajtásakor a gép az alapműveleteket az (alsó tagozatban tanult) „írásbeli” összeadás, szorzás, stb. segítségével végzi el. Döntsük el, hogy az eljárások polinomiálisak-e. (A `floor(a/10.0)` az $a/10$ alsó egészrészét adja vissza.)

(a)
```
x = a; y = 0;
while (x > 0) {
    x = x-1;
    y = y+a;
}
printf("Eredmény: %d", y);
```

(b)
```
x = 0; y = 0;
while (a > 0) {
    x = floor(a/10.0);
    y = y+a-10*x;
    a = x;
}
printf("Összeg: %d", y);
```

9. Az alábbi C kódok közül az első $\lfloor\sqrt{a}\rfloor$-t, a második $\lfloor\log_2 a\rfloor$-t számítja ki bármely bemenetként (10-es számrendszerben) kapott $a > 0$ egész esetén. Tegyük fel, hogy a kódok végrehajtásakor a gép az (alsó tagozatban tanult) „írásbeli" összeadás, szorzás, stb. segítségével végzi el. Döntsük el, hogy az eljárások polinomiálisak-e.

(a)
```
x = 0; y = 0;
while (y <= a) {
    x = x+1;
    y = x*x;
}
printf("Eredmény: %d", x-1);
```

(b)
```
x = 0; y = 1;
while (y <= a) {
    x = x+1;
    y = 2*y;
}
printf("Eredmény: %d", x-1);
```

10. Az alábbi C kód a bemenetként (10-es számrendszerben) kapott $0 < a < m$ egészek esetén az m-nek az a-nál nemnagyobb osztói közül a legnagyobbat számítja ki. Tegyük fel, hogy a kód végrehajtásakor a gép az (alsó tagozatban tanult) „írásbeli" összeadás, szorzás, stb. segítségével végzi el. Döntsük el, hogy az eljárás polinomiális-e.

```
while (m%a != 0) {
    a = a-1;
}
printf("Eredmény: %d", a);
```

11. Egy képzeletbeli A algoritmus bemenete egy tízes számrendszerben megadott m pozitív egész szám. Döntsük el az alábbi állításokról, hogy igazak-e.

    (a) Ha minden m bemenet esetén A legfeljebb $5m^2$ lépés után megáll, akkor A polinomiális algoritmus.

    (b) Ha minden m bemenet esetén A legfeljebb $100 \cdot (\log_3 m)^5$ lépés után megáll, akkor A polinomiális algoritmus.

    (c) Ha minden m bemenet esetén A legalább m lépést tesz, akkor A biztosan nem polinomiális algoritmus.

    (d) Ha van olyan m, amelyre A legalább m lépést tesz, akkor A biztosan nem polinomiális algoritmus.

    (e) Ha minden páros m bemenet esetén A legalább m lépést tesz, akkor A biztosan nem polinomiális algoritmus.