

## ZV-05



Java nyelven!

Írjon függvényt vagy metódust, amely egy valós szám köbgyökét adja vissza! Használja az  $x_{k+1} = 1/3 \cdot (2 \cdot x_k + a/x_k^2)$  sorozatot, amely a köbgyökéhez konvergál, ha  $x_1 = 1$ .

---

```
public static double cubeRoot(double a) {  
    double x = 1;  
    for (int i = 0; i < 100; i++) {  
        x = (2 * x + a / (x * x)) / 3;  
    }  
    return x;  
}
```

Ez a függvény egy valós szám köbgyökét adja vissza az  $x_{k+1} = 1/3 \cdot (2 \cdot x_k + a/x_k^2)$  sorozatot használva. A ciklus 100 iterációt végez, de a számítások számát a szükséges pontosságtól függően lehet módosítani.

---



Magyarázd el!

---

A fenti kód egy Java függvényt definiál, amelynek neve "cubeRoot" és egy valós számot vesz fel paraméterként. A függvény visszatérési értéke egy valós szám, ami a paraméterként kapott szám köbgyöke.

A függvény belsejében egy "for" ciklust használunk, amely 100 iterációt végez. Az iterációk során a ciklusváltozó (i) értékét nem használjuk, csupán a ciklus számát szabályozza. A ciklus belsejében a köbgyök számítását végző módszert használjuk, az  $x_{k+1} = 1/3 \cdot (2 \cdot x_k + a/x_k^2)$  sorozatot. Az  $x_{k+1}$  következő x értékét jelenti,  $x_k$  pedig az aktuális x értékét jelenti. Az a pedig a paraméterként kapott számot jelenti. A ciklus végén az x változót adja vissza a függvény, ami a köbgyök szám.



Fontos megemlíteni, hogy a sorozat konvergálni fog a köbgyökhöz, ha  $x_1 = 1$ .

---

## ZV-06



Java nyelven!

Írjon függvényt vagy metódust, amely kiszámolja az n-edik Fibonacci számot!

A Fibonacci

sorozatot az  $a_n = a_{n-2} + a_{n-1}$  rekurzióval definiálja ( $n > 2$ ), ahol  $a_1 = a_2 = 1$ .

---

```
public static int fibonacci(int n) {  
    if (n <= 2) {  
        return 1;  
    }  
    return fibonacci(n-1) + fibonacci(n-2);  
}
```

1	1	2	3	5	8	13	21
---	---	---	---	---	---	----	----

Ez a függvény rekurzívan meghívja magát, hogy kiszámolja az n-edik Fibonacci számot. Az első két eleme mindig 1, így ha  $n \leq 2$ , akkor 1-et ad vissza. Ha  $n > 2$ , akkor a függvény a (n-1) és (n-2) Fibonacci számokat kérdezi le, és összeadja őket.

Egy másik megoldás, ami hatékonyabb, ha nagyszámú Fibonacci számokat számolunk, az iteratív megoldás:

```
public static int fibonacci(int n) {  
    int a = 0, b = 1, c;  
    if (n == 0) return a;  
    for (int i = 2; i <= n; i++) {  
        c = a + b;  
        a = b;  
        b = c;  
    }  
    return b;  
}
```

---