Rekurzió

Konzultáció

3. alkalom

Rekurzió

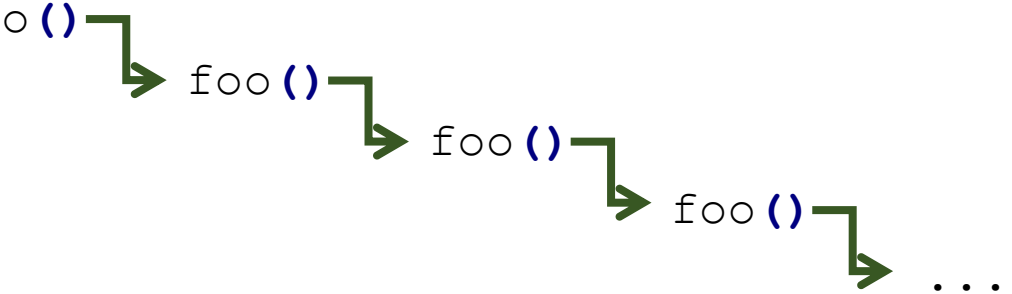
- Rekurzió: egy eljárást rekurzívnak nevezünk, ha előfordul saját magának a hívása
  - Hogy elkerüljük a végtelen ciklust, definiálni kell egy **bázis esetet**

```
void foo()
{
   foo();
}
```

Rekurzió – futás közben

```
void foo()
{
        foo();
}

int main()
{
        foo();
        return 0;
}
```

foo() -> foo() -> foo() -> ...



Bázis eset

- Bázis eset: biztosítja hogy a rekurzió terminálјon
  - A bázis eset mindig input specifikus (feladat függő)

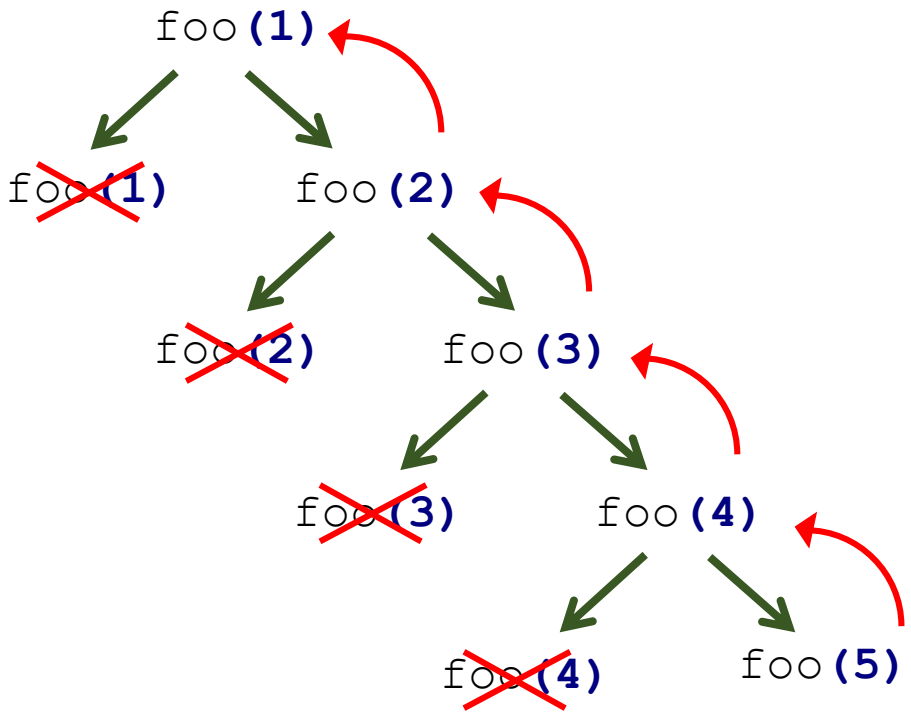
```
int foo(int n)
{
        if(n < 5)
        {
                return foo(n+1);
        }
        else
        {
                return n;
        }
}

int main()
{
        printf("%d\n", foo(1));
}
```

Bázis eset, folytatás

```
foo(1) → if(n < 5) ➡ true → foo(1 + 1) ➡ foo(2)
foo(2) → if(n < 5) ➡ true → foo(2 + 1) ➡ foo(3)
...
foo(5) → if(n < 5) ➡ false → 5
```

Rekurziós fa

- Rekurziós fa: a rekurzió megjelenítésére szolgál, segítségével könnyebb megérteni mi történik egy-egy rekurzió közben



Mi is történt?

```
int foo(int n) // n = 1
{
        if(n < 5) return foo(n+1);
        else return n;
}

int foo(int n) // n = 2
{
        if(n < 5) return foo(n+1);
        else return n;
}

int foo(int n) // n = 3
{
        if(n < 5) return foo(n+1);
        else return n;
}

int foo(int n) // n = 4
{
        if(n < 5) return foo(n+1);
        else return n;
}
```

1 + 1

2 + 1

3 + 1

4 + 1

```
int foo(int n) // n = 5
{
        if(n < 5) return foo(n+1);
        else return n;
}
```

Feladat – I

- Írj (rekurzív) programot amely kiszámítja a megadott szám faktoriálisát!

A faktoriális függvény formális definíciója:

$$n! = \prod_{k=1}^{n} k \quad \text{minden } n \geq 0 \text{ egész számra.}$$

Például:

$$5! = 1 \cdot 2 \cdot 3 \cdot 4 \cdot 5 = 120$$

Megoldás – I

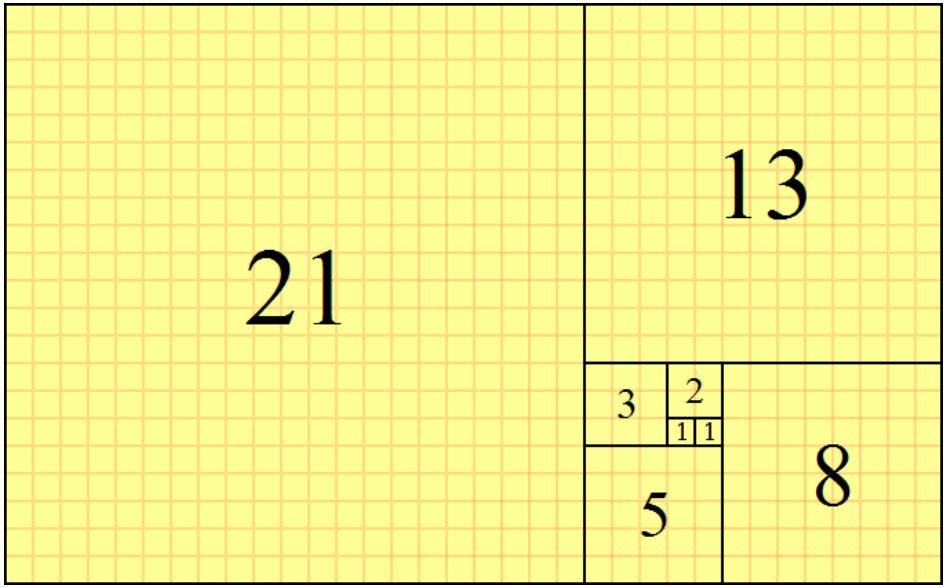
- Pszeudokód: nem fontos a szintaxis, csak a működést írjuk le, a részleteket majd később kitaláljuk (https://hu.wikipedia.org/wiki/Pszeudokód)

```
factorial(n) {
   if n == 1 {
      return n
   }
   else {
      return n * factorial(n - 1)
   }
}
```

Feladat – II

- Írj (rekurzív) programot amely kiszámítja az n-edik fibonacci számot!

$$F_n = \begin{cases} 0, & \text{ha } n = 0; \\ 1, & \text{ha } n = 1; \\ F_{n-1} + F_{n-2}, & \text{ha } n \geq 2. \end{cases}$$



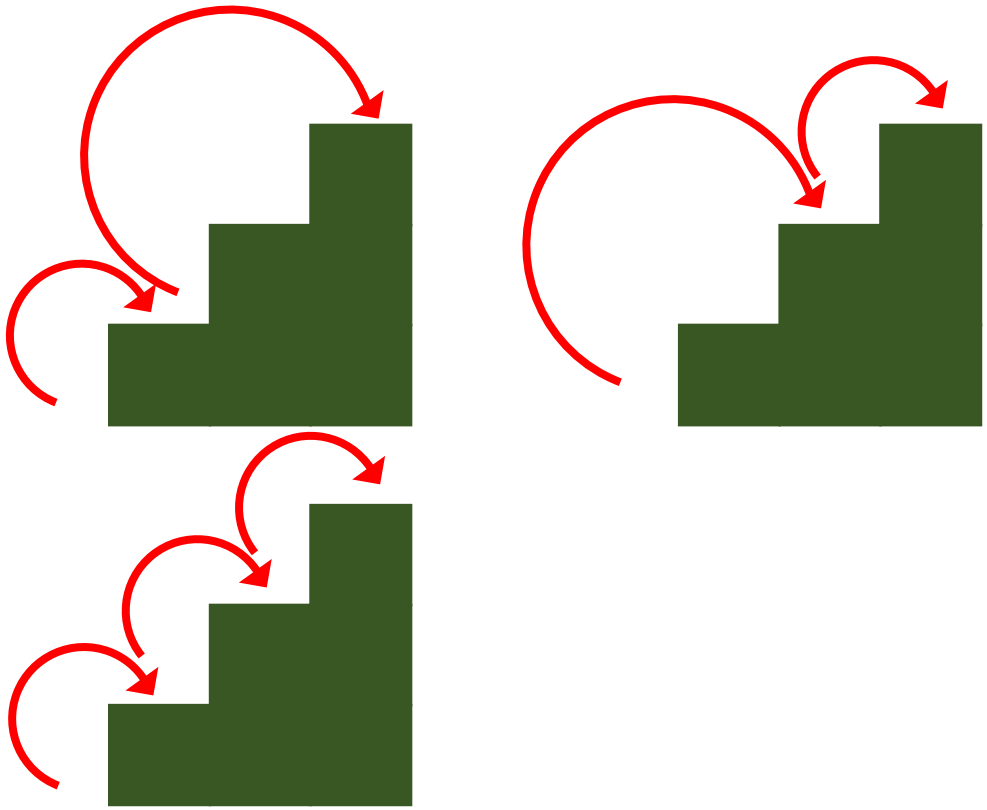
Megoldás – II

```
fibo(n) {
    if n == 0 {
        return 0
    }
    if n == 1 {
        return 1
    }
    else {
        return  fibo(n-1) +
                fibo(n-2)
    }
}
```

Feladat – III

- Írj (rekurzív) programot amely kiszámítja hányféleképpen mehetünk fel egy n lépcsőfokból álló lépcsőn, ha egyszerre csak 1 vagy 2 lépcsőfokot léphetünk!

n = 3-ra:



Megoldás – III

```
lepcso(n) {
   if n == 1 {
      return 1
   }
   if n == 2 {
      return 2
   }
   else {
      return  lepcso(n-1) +
              lepcso(n-2)
   }
}
```