# 08. Rekurze (princip, příklady)

Metoda, která uvnitř svého těla volá sama sebe. Obvykle na jednodušší zadání.

## Obecná definice

* Sebeopakování, které je realizováno bez použití cyklu.
* Rekurzivní algoritmus musí obsahovat podmínku, při které dojde k jejímu ukončení, aby nepokračovala do nekonečna.
* Po každém kroku musí dojít ke zjednodušení problému, jinak je postup neefektivní.
* Důležité je nalezení vztahu mezi řešenými problémy.
* Každý rekurzivní algoritmus lze přepsat na iterační (pomoci cyklů, podmínek, zásobníku).
* Lze ji použít, pokud počet rekurzivních volání roste lineárně.

## Nevýhody

* Rekurzivní algoritmy jsou neefektivní pro velké programy, při každém novém volání funkce dochází k vytvoření nové sady lokálních proměnných (HW náročnost).
* Pro rozsáhlé datové soubory může dojít paměť dříve, než je nalezeno řešení.
* Stručný, ale nepříliš přehledný kód.
* Nedoporučuje se pro Fibonacciho posloupnost (roste exponenciálně)

## Použití

* Faktoriál
* Hanojské věže
* Binární vyhledávání
* Převod z desítkové do dvojkové soustavy
* Fibonacciho posloupnost
* Quick sort

#### public class test {

#### /\*\*

#### \* Vypočítá rekurzivně faktoriál zadaného čísla

#### \* @param number cislo >=0

#### \* @return faktoriál čísla, -1 v případě neplatného vstupu

#### \*/

#### public static int faktorial(int number){

#### if(number < 0) return - 1;

#### if(number == 0 || number == 1) return 1;

#### return number\*faktorial(number - 1);

#### }

#### }