INFORMATIKA LABORATÓRIUM I.

Pointerek

Rekurzió

- A matematikában lehetőség van bizonyos adatok és műveletek rekurzív definiálására.
- A rekurzív algoritmusokban a soron következő lépéshez az előző lépések elvégzésekor kapott eredményeket használjuk fel.
- A programozási nyelvekben a **rekurzióról akkor beszélünk, amikor egy alprogram saját magát hívja meg (*önrekurzió vagy közvetlen rekurzió*), vagy ha két (vagy több) alprogram felváltva hívja egymást (*kölcsönös* vagy *közvetett rekurzió*).**
- Rekurzió használata esetén minden esetben kell lennie egy, a rekurziót leállító feltételnek, különben a programunk “kiakad”, futása megáll.

Rekurzió

- Minden rekurzió triviálisan ciklussá alakítható (egy külön vezetett verem segítségével) és minden ciklus rekurzióvá alakítható.
- Egy algoritmus implementálásának a stílusát érthetőségi és hatékonysági szempontok alapján szokták megválasztani.
- Egy fa-szerű adatszerkezet mélységi bejárása megoldható iteratívan is, de rekurzióval leírva könnyebben követhető és még hatékony is.
- Ahogyan ciklusok esetében előfordulhat véletlenül végtelen ciklus, úgy rekurziók esetében is lehetséges a végtelen rekurzió. Az előbbinek a tünete lehet a „befagyott program” az utóbbinak pedig az „elszállás”. Ilyenkor általában elfogy a hívási verem (call stack) számára fenntartott memória és veremtúlcsordulási hibával (*stack overflow error*) megszakad a program.

Fibonacci ciklussal

```c
void fibonacci(int n) {
  int i, egyik = 0, masik = 1;
  printf("%d ", masik);
  for (i = 1; i <= n; i++) {
    int harmadik = egyik + masik;
    egyik = masik;
    masik = harmadik;
    printf("%d ", masik);
  }
  printf("\n");
}
```

Fibonacci rekurzióval

```
//-------------------------------------------------------------

#include <stdio.h>
#include <conio.h>

int fib(int n);
void main()
{
    int szam;
    printf("Hanyadik elemet szeretned? Max 46! ");
    scanf("%d", &szam);
    printf("Az eredmeny: %d", fib(szam));
    getch();
}
int fib(int n)
{
     int v;
     if (n == 1 || n == 2) return 1;
     else return fib(n-1) + fib(n-2);
    }
//-------------------------------------------------------------
```

Hanoi-tornyai

```
//---------------------------------------------------------------------------

#include <stdio.h>

int lepesszam=0;
void hanoi(int mirol, int mire, int manko, int n);
void main()
{
hanoi(1,2,3,4);
getch();
}
void hanoi(int mirol, int mire, int manko, int n)
{
    if (n>0)
        {
        hanoi(mirol,manko,mire,n-1);
        lepesszam++;
        printf("%2.0d. lepes: %d -> %d\n",lepesszam,mirol,mire);
        hanoi(manko,mire,mirol,n-1);
        }
}
//---------------------------------------------------------------------------
```

Faktoriális

A lenti kód a faktoriálist számolja ki ciklussal, valósítsuk meg rekurzió segítségével!

```
#include <stdio.h>

void main(void)
{
 int i, n, fakt=1;
 printf("Kerem a szamot: ");
 scanf("%d",&n);
 for (i = 1; i <= n; i++) {
  fakt=fakt*i;
 }
 printf("Eredmeny: %d", fakt);
 getch();
 }

//-------------------------------------------------
```