3. házi feladat: Polinomok tömbök segítségével

Írjunk programot, mely kiszámolja egy legföljebb 10-edfokú polinom fokát és helyettesítési értékét! A polinomokat egyszer?en együtthatóinak tömbjeként kezeljük, például a $p_1(x) = x^3 - 5x + 12$ polinomot egy olyan p1 nev? tömbbel adhatjuk meg, melynek elemei: `p1[0]=12`, `p1[1]=-5`, `p1[2]=0`, `p1[3]=1`, `p1[4]=0`,... `p1[10]=0`.

Írjunk két függvényt a main-en kívül:

1. Az egyik függvény legyen az, amely megállapítja, és értékként visszaadja egy polinom fokát! Ennek deklarációja lehet például a következ?:

```
int fok( float polinom[] );
```

hívása például `fok( p1 )`, ahol `p1` egy már definiált polinom. Az algoritmus egyszer?en keresse meg a legnagyobb index? nemnulla értéket a polinomot megadó tömbben. Az azonosan nulla polinom fokát mínusz végtelennek szokás definiálni, ekkor e függvény adja vissza az ábrázolható legkisebb egészt! E függvényhez el kell tudnunk dönteni, hogy egy lebeg?pontosan ábrázolt szám egyenl?-e nullával. A számábrázolás pontatlansága miatt az `(a == 0.0)` feltétel nem lesz jó, ezért egy számról akkor fogjuk azt mondani, hogy 0, ha valamely el?re megadott `EPSILON` értéknél abszolút értékben kisebb. Lehet például `EPSILON = 1e-10`, vagyis 10^{-10}. Használjuk az alábbi kódot (vagy helyettesítsük egy jobbal):

```
int kbNulla( float a ){
    float EPSILON=1e-10;
    if ( a < EPSILON && a > -EPSILON )
        return 1;
    return 0;
}
```

2. Írjunk egy függvényt, mely kiszámolja egy polinom értékét egy megadott helyen a Horner-módszert használva. A Horner-módszer az

$$a_n x^n + a_{n-1} x^{n-1} + \ldots + a_1 x + a_0 = (\ldots((a_n x + a_{n-1})x + a_{n-2})x + \ldots + a_1)x + a_0$$

összefüggésre épül, mellyel hatványozás nélkül, és gyorsan értékelhet? ki egy polinom. Ha e függvény deklarációja

```
float Horner( float polinom[], float x );
```

akkor például a fent megadott p1 polinom és `x=1.0` esetén a `Horner( p1, x )` függvényhívás `8.0`-át ad vissza. E függvényben használjuk a fokszámot megadó függvényt.

A programunkat kezdjük az alábbi sorokkal:

```
#include <stdio.h>
#include <limits.h>
#define MAX_POL 11  // minden polinomon legföljebb 10-edfokú
```

Az utolsó sorban egy konstanst definiáltunk. Ez a makróparancs a fordítás els? fázisában a kód a MAX_POL változó helyébe 11-et helyettesít.

A main függvényben az alábbi polinomokat definiáljuk, majd mindegyiknek írjuk ki a fokát és a helyettesítési értékét az `x = 1.2` helyen!

```
int main(void){
```

```
float p1[MAX_POL]={12,-5,0,1,0,0,0,0,0,0,0};
float p2[MAX_POL]={1,0,0,0,0,0,0,0,0,0,1};
float p3[MAX_POL]={5,0,0,0,0,0,0,0,0,0,0};
float p4[MAX_POL]={0,0,0,0,0,0,0,0,0,0,0};
```

A két függvény és a `kbNulla` függvény mindegyikét a main el?tt deklaráljuk, és a main után definiáljuk!