```
/* függvény pointer: <return-
type> (*functionName)(<parameter - tipusoklista>);
... fv(..., int(*amitmeghiv)(tipus, tipus)) { ... }

C++:
::hatókör operátor
alapértelmezett paraméterek
fvnév(tipus változó = érték) híváskor elhagyható, csak a záró paramétere
k.
overload : azonos nevü, más paraméterezésü.a fordító választja ki az oda
illöt c++
                egyetlen eltérés a visszatérö érték nem lehet.
template - k : általános osztályok, amiből

          template <typename identifier> function_declaration; typename é
s class ugyanazt jelenti.
hivásnál fvnév<tipus>(paraméterlista)
*/
#include "stdafx.h"
// ma nem lesz szinusz.
#include <iostream> // cout nincs .h
#include <iomanip>  // i/o manipulativ: pl setw
#include <cmath> // első utasítása: math.h
#include <ctime> // time.h

using namespace std;

// függvények fejlécei
double tombatlag(double * t, int meret);
double tombatlag(int * t, int meret);// overload
double tombatlag(double mennyilegyen = 0); // értelmetlen, de ezt is meg
irjuk
template <typename Tipus> // vagy template <class Tipus>
void csere(Tipus &a, Tipus &b);

int _tmain(int argc, _TCHAR* argv[])
{
      int *ti,n,i;
      double *td,atlag;
      time_t ido;
      time(&ido); srand((unsigned int)ido);
      setlocale(LC_ALL, "hun");
      cout << "hány tömbelemet átlagoljak "; cin >> n;
      ti = new int[n];
      td = new double[n]; // ettől még pointerek
      for (i = 0; i < n; i++)
      {
            ti[i] = rand() - RAND_MAX / 2;
            td[i] = ((double)rand() / RAND_MAX)*200-100; // -
100 és 100 közti valós
      }
      if (n < 20)
      {
            cout << "int :";

      for (i = 0; i < n; i++) cout << setw(7) << ti[i]; // előjelesen cs
ak kifér
            cout << endl;
            cout << "double :";

      for (i = 0; i < n; i++) cout << setw(8) <<setprecision(5)<< td[i];
            cout << endl;
      }
      atlag = tombatlag();  // paraméter nélkül, 0
```

```
	cout << "Ha nincs paraméter, az átlag: " << atlag << endl;
	cout << "Ha megadjuk a paramétert : " << tombatlag(4.5) << endl;
	atlag = tombatlag(ti, n); // ti-
ből tudja, hogy az int típusút kell meghívni
	cout << "Az int-ek átlaga: " << atlag << endl;
	atlag = tombatlag(td, n);
	cout << "A double-k átlaga: " << atlag << endl;
	if (n>1) // két elem már megcserélhető
	{
		csere<int>(ti[0], ti[1]);

	cout << "csere<int> után ti[0]=" << ti[0] << ", ti[1]=" << ti[1] <
< endl;
		csere<double>(td[0], td[1]);

	cout << "csere<double> után td[0]=" << td[0] << ", td[1]=" << td[1
] << endl;
	}
	delete td;
	delete ti;
	cin.get(); cin.get();
	return 0;
}

// átlagolom a tömb elemeit. a legnagyobbat és a legkisebbet elhagyom.
// ha az elemszám háromnál kisebb, 0-t adok
double tombatlag(double * t, int meret)
{
	int i, nagy_index = 0, kis_index = 0;// egyből értékadás
	double szumma;
	if (meret < 3) return 0;
	// megkeresem a legnagyobbat és a legkisebbet
	for (i = 1; i<meret; i++) // 1-től elég menni
	{
		if (t[i]>t[nagy_index]) nagy_index = i;
		if (t[i]<t[kis_index])  kis_index = i;
	}
#ifdef _DEBUG // csak ha debugolok. a fg ne irjon a konzolra.
	cout<<"double függvény fut: legkisebb a " << kis_index<<" legnagyo
bb a " <<nagy_index<<
		" indexü elem"<<endl;
#endif
	// átlagolok. a legnagyobb és a legkisebb indexűt nem rakom bele
	// az összegbe. ki is vonhatnám őket, de ne engedjük a boszorkányt
 a házba
	szumma = 0; // nem felejtettem el kinullázni.
	for (i = 0; i<meret; i++)
	{
		if (i != nagy_index && i != kis_index) szumma += t[i];
	}
	return szumma / (meret - 2);
}

// blokk copy után overload. C++
double tombatlag(int * t, int meret)
{
	int i, nagy_index = 0, kis_index = 0;
	int szumma;// int !!!
	if (meret < 3) return 0;
	for (i = 1; i<meret; i++)
	{
		if (t[i]>t[nagy_index]) nagy_index = i;
		if (t[i]<t[kis_index]) kis_index = i;
	}
```

```
#ifdef _DEBUG
	cout << "int függvény fut: legkisebb a " << kis_index << " legnagy
obb a " << nagy_index <<
		" indexü elem" << endl;
#endif
	szumma = 0;
	for (i = 0; i<meret; i++)
	{
		if (i != nagy_index && i != kis_index) szumma += t[i];
	}
	return (double)szumma / (meret - 2); // különben egész osztás
}

double tombatlag(double mennyilegyen) // ide már nem adhatom meg az =0-
t, ha fent szerepelt
{
#ifdef _DEBUG
	cout << "az alapértelmezett paraméteres függvény fut" << endl;
#endif
	return mennyilegyen;
}

// ezzel a függvénnyel mindent meg tudok cserélni, amire értelmezve van
az értékadó operátor.
template <typename Tipus>
void csere(Tipus &a, Tipus &b)
{
	Tipus tmp = a; // ha xor-
ral vagy műveletekkel végezzük, lassabb lesz
	a = b; // itt viszont + memóriára van szükség tmp-nek
	b = tmp;  // memória ma van, idő nincs!
}
```