5. labor kis ZH, 2016. május 5., csütörtök 10-12

NÉV (olvasható): NEPTUN kód:

max. 10p, megoldási idő: 20 perc

A feladatok megoldásához közvetlen emberi segítségen kívül bármilyen segítséget igénybe vehetsz (pl. jegyzet, könyv, internet (de chat/facebook/stb. nem)). Javasolt a megoldást erre a lapra írni, de a megoldást vagy annak egy részét elküldheted a pohl@eet.bme.hu-ra is, legkésőbb a ZH beadásának pillanatáig.

1. Adott a `struct Adat{ double jegy, kredit; };` struktúra. Ilyen struktúrákat tárolunk egy std::vectorban. Írj programot, amely std::inner_product algoritmus segítségével kiszámítja majd kiírja a jegyek súlyozott átlagát! (A $\text{súlyozott átlag} = \sum_i (jegy_i \cdot kredit_i) / \sum_i kredit_i$) A számlálót is és a nevezőt is std::inner_producttal számold! A programban ciklus nem lehet. Teljes programot írj, a vector feltöltését …-tal helyettesítsd! Tipp: a jegyeket tároló vektort saját magával skalárszorozzuk; az összegző függvény mindkét esetben lehet az `std::plus<double>()`, a két szorzó függvényt nekünk kell elkészítenünk. (6p)

2. Készíts generikus függvényt, amely egy tömböt kap paraméterként! A tömbben két, egymástól független, azonos méretű adatsor található, méghozzá oly módon, hogy a két adatsor elemei felváltva követik egymást. A függvény térjen vissza igaz értékkel, ha mindkét adatsor szigorúan monoton növekedő! A generikus változó a tárolt elemek típusa. Írj kódrészletet, amelyben meghívod a függvényt (egy tömbbel, a típusát szabadon megválaszthatod). (6p)

*3. Egy téli estén havaseső esett, a http://www.eet.bme.hu/~pohl/kep.jpg kép másnap délelőtt készült. A kövezeten nincs hó, a műanyag borításon van. Mi ennek az oka? (Emberi beavatkozás nem történt.) (2p)

1.

```
#include <iostream>
#include <vector>
#include <numeric>

struct Adat {
    double jegy, kredit;
};

double getszor(const Adat &a, const Adat &b) {
    return a.jegy*a.kredit;
}

double getkr(const Adat &a, const Adat &b) {
    return a.kredit;
}

int main() {
    Adat t[5] = { { 3,5 }, { 4,2 }, { 5,3 }, { 2,10 }, { 4,5 } };
    std::vector<Adat> be(t,t+5); // vector def + atlag + kiírás:

    double sum = std::inner_product(be.begin(), be.end(), be.begin(), 0.0,
                                    std::plus<double>(), getszor);
    double sum_kr = std::inner_product(be.begin(), be.end(), be.begin(), 0.0,
                                       std::plus<double>(), getkr);
    double atlag = sum / sum_kr;

    std::cout << sum << " " << sum_kr << " " << atlag << std::endl;

    return 0;
}
```

2.

```
template<typename T>
bool szigmon(T *t, int n) {
    bool egy = true, ketto = true;
```

```
    for (int i = 2; i < n; i++) {
        if (t[i] <= t[i - 2])
            if (i % 2 == 0)
                egy = false;
            else ketto = false;
    }
    return egy && ketto;
}

int main() {
    double t[10] = { 1, -5, 2, -4, 3, -3, 4, -2, 5, -1 };
    std::cout << szigmon(t, 10) << std::endl;
    return 0;
}
```

3. Hővezetés különbség.