AAAAAAAAAAAAAAAAAAAAAAAAAAAA

Programozás alapjai II. 4. ellenőrző dolgozat. 2013.04.18. Kurz/**Terem**: G3/	**20 perc**
Név: **Neptun:**	Összpont:

1. feladat 3.5 pont

Készítsen sablont rendezett párok tárolására (*Pair*)! A pár mindkét eleme legyen sablonparaméterként megadható! Mindkét elem legyen külön-külön lekérdezhető és beállítható, de közvetlenül egyik adat se legyen elérhető! Legyen összehasonlítható két ilyen generikus adat az **==** operatorral! Két rendezett pár akkor azonos, ha adataik sorrendje és értéke azonos. Működjön az elvárásoknak megfelelően az alábbi kódrészlet!

```
Pair<string, long> adat("Homerseklet",12);
std::cout << adat.first();          // kiírja a pár első elemét
std::cout << adat.second();         // kiírja a pár második elemét
adat.second() += 10;                // növeltük a hőmérsékletet
Pair<int, long> tabla[256], kod;    // 2. feladatban ezeket kell használni
if (tabla[0] == kod ) std::cout << "azonos" << std::endl;
```

```
template<class T1, class T2>
class Pair {
    T1 _first;
    T2 _second;
public:
    Pair(const T1& f = T1(), const T2& s = T2()) :_first(f), _second(s) {}
    T1& first() { return _first; }
    T2& second() { return _second; }
    bool operator==(const Pair& rhs) const {
        return _first == rhs._first && _second == rhs._second;
    }
};
```

2. feladat 2.5 pont

Készítsen függvénysablont, ami iterátorokkal megadott adatsorozatban megszámolja, hogy hányszor szerepel egy adott adattól elérő érték! A függvény paraméterként vegye át a két iterátort és a vizsgálandó értéket. A függvény visszatérési értéke egy egész, ami az eltérések számát adja. (1.5p) Kódrészlettel **Mutassa** be, hogy a függvénysablonnal hogyan tudná megállapítani, hogy az előző feladat ***tabla*** tömbjében hány olyan adat volt ami nem egyezik a ***kod*** értékével! (1p)

```
template<class Iter, class T>
int szamol(Iter first, Iter last, const T& ertek) {
    int cnt = 0;
    while (first != last) {
        if(!(*first == ertek)) ++cnt;
        ++first;
    }
    return cnt;
}

cout << szamol(tabla, tabla+256, kod) << endl;
```

BBBBBBBBBBBBBBBBBBBBBBBBBB

Programozás alapjai II. 4. ellenőrző dolgozat. 2013.04.18. Kurz/**Terem**: G3/	**20 perc**
Név: **Neptun:**	Összpont:

1. feladat 3.5 pont

Készítsen sablont rendezett párok tárolására (*Par*)! A pár mindkét eleme legyen sablonparaméterként megadható! Mindkét elem legyen külön-külön lekérdezhető és beállítható, de közvetlenül egyik adat se legyen elérhető! Legyen összehasonlítható két ilyen generikus adat az **==** operatorral! Két rendezett pár akkor azonos, ha adataik sorrendje és értéke azonos. Működjön az elvárásoknak megfelelően az alábbi kódrészlet!

```
Par<string, long> adat("Homerseklet",12);
std::cout << adat.elsoElem();        // kiírja a pár első elemét
std::cout << adat.masodikElem();     // kiírja a pár második elemét
adat.masodikElem() += 10;            // növeltük a hőmérsékletet
Par<int, char> kodtabla[256], kod;   // 2. feladatban ezeket kell használni
if (kodtabla[0] == kod ) std::cout << "azonos" << std::endl;
```

```
template<class T1, class T2>
class Par {
    T1 _elsoElem;
    T2 _masodikElem;
public:
    Par(const T1& f = T1(), const T2& s = T2()) :_elsoElem(f), _masodikElem(s) {}
    T1& elsoElem() { return _elsoElem; }
    T2& masodikElem() { return _masodikElem; }
    bool operator==(const Par& rhs) const {
        return _elsoElem == rhs._elsoElem && _masodikElem == rhs._masodikElem;
    }
};
```

2. feladat 2.5 pont

Készítsen függvénysablont, ami iterátorokkal megadott adatsorozatban megszámolja, hogy hányszor fordul elő egy adott érték! A függvény paraméterként vegye át a két iterátort és a keresett értéket. A függvény visszatérési értéke egy egész, ami az előfordulási számot adja. (1.5p) Kódrészlettel **Mutassa** be, hogy a függvénysablonnal hogyan tudná megállapítani az előző feladat ***kodtabla*** tömbjében a ***kod*** előfordulási számát! (1p)

```
template<class Iter, class T>
int szamol(Iter elso, Iter utolso, const T& ertek) {
    int cnt = 0;
    while (elso != utolso) {
        if(*elso == ertek) ++cnt;
        ++elso;
    }
    return cnt;
}

cout << szamol(kodtabla, kodtabla+256, kod) << endl;
```