AAAAAAAAAAAAAAAAAAAAAA

Szoftver laboratórium II. 5. ellenőrző dolgozat. 2013.05.06. Kurz/**Terem**: L3/	**15 perc**
Név: **Neptun:**	Összpont:

1. feladat **4.5 pont**

Az *Adat* osztály egy részecskefizikai kísérlet adatait tárolja. Az osztály belső szerkezetét nem ismerjük, de minden tárolt adata lekérdezhető és beállítható *getter/setter* függvényekkel, melyek interfészét pontosan ismerjük:

Függvény	**Funkció**
`void Adat::setA(long int);`	azonosító beállítása
`long int Adat::getA();`	azonosító lekérdezése
`void Adat::setE(double);`	energia beállítása
`double Adat::getE();`	energia lekérdezése

Az osztálynak van default konstruktora és számos egyéb tagfüggvénye is, melyek pontos funkcióját/leírását a fizikus kollegák ismerik, mi nem. Tudjuk, hogy az osztály nem perzisztens. **Hozzon** létre az *Adat* osztályból példányosított objektumok tárolására egy *AdatTar* osztályt, ami perzisztens és úgy viselkedik, mint egy *std::vector*! Legyenek az *AdatTar* osztálynak az *std::vector* tárolónál megismert funkciójú konstruktorai (ld. megjegyzés)! **Deklarálja** és **valósítsa** meg az *AdatTar* osztályt úgy, hogy az alábbi programrészlet az elvárásoknak megfelelően működjön. Tételezze fel, hogy a fájlok írása/olvasása közben nem lép fel hiba!

```
Adat a1;
a1.setA(1234); a1.setE(12.3);
AdatTar t1, t2(5), t3(10, a1), t4(&a1, &a1+1);   // KONSTRUKTOROK
...
std::ofstream ofs("Adat.dat"); t3.write(ofs);    // adat kiírása fájlba
...
std::ifstream ifs("Adat.dat"); t2.read(ifs);     // adat beolvasása fájlból
```

```
class AdatTar : public std::vector<Adat> {
public:
    AdatTar(size_t s = 0, const Adat& v = Adat()) : std::vector<Adat>(s, v) {}
    template <class Iter>
    AdatTar(Iter first, Iter last) : std::vector<Adat>(first, last) {}
    void read(std::istream& is) {
        size_t siz;
        is >> siz;
        resize(siz);
        for (size_t i = 0; i < size(); i++) {
            long l; double d;
             is >> l >> d;
            this->operator[](i).setA(l);
            (*this)[i].setE(d);
        }
    }
    void write(std::ostream& os) {
        os << size() << std::endl;
        for (size_t i = 0; i < size(); i++)
            os << (*this)[i].getA() << ' ' << (*this)[i].getE() << std::endl;
    }
};
```

2. feladat **1.5 pont**

Tételezze fel, hogy az *AdatTar* osztály elkészült! **Írjon programrészletet**, amelyben létrehoz egy *AdatTar* objektumot 100 adat tárolására, majd az *sdl::count_if* algoritmussal megszámolja, hogy hányszor fordult elő benne az előző feladat *a1* adatával egyező *adat!

```
using namespace std;
AdatTar tar(100);
cout << count_if(tar.begin(), tar.end(),
                 bind2nd<equal_to<Adat> >(equal_to<Adat>(), a1)) << endl;
```