AAAAAAAAAAAAAAAAAAA rendelkezésre álló idő:20 **perc**

Programozás alapja 2. 2. ellenőrző dolgozat. 2015.03.23. Kurz/**Terem**: G1/	Elért pontszám:
Név: **Neptun:**	

1. Tervezzen egy olyan osztályt (Ido), ami idő (óra, perc) tárolására alkalmas! Az osztálynak 7p
   - legyen paraméter nélkül hívható konstruktora, ami 0 óra 0 percet állít;
   - legyen olyan beállító függvénye (set), amivel az óra és a perc is beállítható; amennyiben csak az órát adják meg paraméterként, úgy adott óra 0 percet állítson be a függvény;
   - legyenek olyan lekérdező függvényei, amelyekkel az óra és a perc külön-külön lekérdezhető;
   - legyen olyan tagfüggvénye, amivel el lehet dönteni, hogy két időpont közül melyik a korábbi (<);

   Valósítsa meg az osztályt C++ nyelven úgy, hogy az osztálynak legyen legalább 1 nem inline tagfüggvénye és ne lehessen az adattagokhoz kívülről hozzáférni. A beállító függvények dobjanak const char * hibát, ha hibás értékparaméterrel hívják azokat (pl. negatív perc)!
   Működjön helyesen az alábbi kódrészlet:

```
Ido t0, t10, t13;
t10.set(10);                                  // 10 óra 00 percet állít be
t13.set(13,21);                               // 13 óra 21 percet állít be
std::cout << std::boolalpha << (t13 < t10);  // false-t ír ki
```

2. Az előző feladatban megtervezett osztályhoz készítsen olyan fűzhető inserter operatort (<<), ami képes a tárolt időt **o:p** formátumban kiírni egy ostream típusú objektumra! Egy rövid kódrészleten mutassa be az inserter használatát! 3p

megoldás:

```
class Ido {
    int ora;
    int perc;
public:
    Ido() :ora(0), perc(0) {}
    int getOra() const { return ora; }
    int getPerc() const { return perc; }
    void set(int, int = 0);
    bool operator<(const Ido& t) const {
        return ora*60+perc < t.ora*60 + t.perc; }
};

void Ido::set(int o, int p ) {
    if (o < 0 || o >= 24 || p < 0 || p >= 60)
        throw "Hiba";
    ora = o; perc = p;
}

std::ostream& operator<<(std::ostream& os, const Ido& t) {
    os << t.getOra() << ":" << t.getPerc();
    return os;
}

std::cout << t10 << std::endl;
```

BBBBBBBBBBBBBBBBBBBBB rendelkezésre álló idő:20 **perc**

Programozás alapja 2. 1. ellenőrző dolgozat. 2015.03.23. Kurz/**Terem**: G1/	Elért pontszám:
Név: **Neptun:**	

1. Tervezzen egy olyan osztályt (Tort), ami két egész szám hányadosaként tárol valódi törteket! Az osztálynak 7p
   - legyen paraméter nélkül hívható konstruktora, ami 0/1 értéket állít be;
   - legyen olyan beállító függvénye (set), amivel a számláló és a nevező is beállítható;
   - legyenek olyan lekérdező függvényei, amelyekkel a számláló és a nevező külön-külön lekérdezhető;
   - legyen olyan tagfüggvénye, amivel el lehet dönteni, hogy két tört közül melyik a nagyobb (>);

   Valósítsa meg az osztályt C++ nyelven úgy, hogy az osztálynak legyen legalább 1 nem inline tagfüggvénye és ne lehessen az adattagokhoz kívülről hozzáférni. A beállító függvény dobjon const char * hibát, ha hibás értékparaméterrel hívják (pl. számláló nem kisebb a nevezőnél)!
   Működjön helyesen az alábbi kódrészlet:

```
Tort t0, t56, t34;
t56.set(5,6);                                  // 5/6 értéket állít
t34.set(3,4);                                  // 3/4 értéket állít be
std::cout << std::boolalpha << (t56 > t34);  // true-t ír ki
```

2. Az előző feladatban megtervezett osztályhoz készítsen olyan fűzhető inserter operatort (<<), ami a törtet A/B formátumban kiírja egy ostream típusú objektumhoz, ahol „A” számláló, „B” nevező! Egy rövid kódrészleten mutassa be az inserter használatát! 3p

megoldás:

```
class Tort {
    int szamlo;
    int nevezo;
public:
    Tort() :szamlo(0), nevezo(1) {}
    int getSzamlo() const { return szamlo; }
    int getNevezo() const { return nevezo; }
    void set(int, int);
    bool operator>(const Tort& t) const {
        return (double)szamlo/nevezo > (double)t.szamlo/ t.nevezo; }
};

void Tort::set(int sz, int n ) {
    if (sz >= n || n == 0)
        throw "Hiba";
    szamlo = sz; nevezo = n;
}

std::ostream& operator<<(std::ostream& os, const Tort& t) {
    os << t.getSzamlo() << "/" << t.getNevezo();
    return os;
}

std::cout << t10 << std::endl;
```