rendelkezésre álló idő:15 **perc**

Programozás alapja 2. 3. beszámoló dolgozat. 2017.03.17. Kurz/**Terem**: Gy4/	Elért pontszám:
Név: **Neptun:**	

A Pontok osztályt síkbeli pontok (Pont) tárolására készítettük:

```
class Pont {
    int x, y;
public:
    Pont(int x=0, int y=0) :x(x), y(y) { std::cout << x; }
    void setxy(int _x, int _y) { x = _x; y = _y; }
};

class Pontok {
    unsigned siz;
    Pont *p;
public:
    Pontok(unsigned s = 0) :siz(s), p(new Pont[siz]) {}
    unsigned size() const                    { return siz; }
    Pont operator[](unsigned i)const         { return p[i]; }
    ~Pontok() { delete[] p; }
    Pontok(const Pontok& rhs) {
        siz = rhs.siz;
        p = new Pont[siz];
        for (unsigned i = 0; i < siz; i++)
            p[i] = rhs.p[i];
    }




};
```

1. Magyarázza meg, hogy az alábbi kódrészlet futtatásakor miért lép fel memóriakezelési hiba! (1p)

```
{ Pontok pts(3); }
```

Az alapértelmezett destruktor nem jó, mivel az osztály dinamikus memóriaterületet foglal és tart nyilván.

Módosítsa az osztályt, hogy a fenti kódrészlet futtatása ne okozzon hibát! Használja az üres helyeket az osztály deklarációjában! (1p)

Hibajavítás után mit ír a szabványos kimenetre a fenti kódrészlet? 000 (1p)

2. Magyarázza meg, hogy az alábbi függvény futtatásakor miért keletkezik memóriakezelési hiba! (1p)

```
Pontok ketPont(void) { return Pontok(2); }
```

Az objektum értékű függvény a visszatéréskor meghívja az objektum másoló konstruktorát. Az alapértelmezett másoló konstruktor azonban nem jó, mivel az osztály dinamikus memóriaterületet foglal és tart nyilván.

Módosítsa az osztályt, hogy ne lépjen fel a probléma, és az osztály az elvárásoknak megfelelően működjön! Használja az üres helyeket az osztály deklarációjában! (3p)

3. Készítsen olyan extraktor operátort, amivel egy Pont adatai beolvashatók egy std::istream típusú objektumról! Az operátor legyen fűzhető! A koordinátákat vessző választja el egymástól. A vessző előtt és után szóközök is lehetnek. Hibás inputot nem kell kezelnie! (3p)

```
std::istream& operator>>(std::istream& is, Pont& p) {
    int x, y; char ch;
    is >> x >> ch >> y;
    p.setxy(x, y);
    return is;
}
```

BBBBBBBBBBBBBBBBBBBBBBBBBBBBB rendelkezésre álló idő:15 **perc**

Programozás alapja 2. 3. ellenőrző dolgozat. 2017.03.17. Kurz/**Terem**: Gy4/	Elért pontszám:
Név: **Neptun:**	

A Vektorok osztályt helyvektorok (Vec) tárolására készítettük:

```
class Vec {
    double x, y;
public:
    Vec(double x=0, double y=1) :x(x), y(y) { std::cout << y; }
    void set(int _x, int _y) { x = _x; y = _y; }
};

class Vektorok {
    unsigned siz;
    Vec *pData;
public:
    Vektorok(unsigned s = 4) :siz(s), pData(new Vec[siz]) {}
    unsigned size() const { return siz; }
    ~Vektorok() { delete[] pData; }
    Vektorok(const Vektorok& rhs) {
        siz = rhs.siz;
        pData = new Vec[siz];
        for (unsigned i = 0; i < siz; i++)
            pData[i] = rhs.pData[i];
    }









};
```

1. Magyarázza meg, hogy az alábbi kódrészlet futtatásakor miért lép fel memóriakezelési hiba! (1p)

```
{ Vektorok v; }
```

Az alapértelmezett destruktor nem jó, mivel az osztály dinamikus memóriaterületet foglal és tart nyilván.

Módosítsa az osztályt, hogy a fenti kódrészlet futtatása ne okozzon hibát! Használja az üres helyeket az osztály deklarációjában! (1p)

Mit ír a szabványos kimenetre a fenti kódrészlet? 1111 (1p)

2. Magyarázza meg, hogy az alábbi függvény futtatásakor miért keletkezik memóriakezelési hiba! (1p)

```
Vektorok ketVektor() { return Vektorok(2); }
```

Az objektum értékű függvény a visszatéréskor meghívja az objektum másoló konstruktorát. Az alapértelmezett másoló konstruktor azonban nem jó, mivel az osztály dinamikus memóriaterületet foglal és tart nyilván.

Módosítsa az osztályt, hogy ne lépjen fel a probléma, és az osztály az elvárásoknak megfelelően működjön! Használja az üres helyeket az osztály deklarációjában! (3p)

4. Készítsen olyan extraktor operátort, amivel egy vektor (Vec) adatai beolvashatók egy std::istream típusú objektumról! Az operátor legyen fűzhető! A koordinátákat pontosvessző választja el egymástól. A pontosvessző előtt és után szóközök is lehetnek. Hibás inputot nem kell kezelnie! (3p)

```
std::istream& operator>>(std::istream& is, Vec& p) {
    double x, y; char ch;
    is >> x >> ch >> y;
    p.set(x, y);
    return is;
}
```