Összefoglalása az eddigieknek

- Dinamikus adattagokat tartalmazó osztályok

~ futás alatt fogja lefoglalni a helyet
~ilyen classoknak kell:
- destruktor
- sekély másolat nem elég → sekély másolat: a másolat ugyanarra mutat
- másoló konstruktor
- op= …

Az öröklés

- „tehén az egy állat” – tehén → állat

a) tagváltozók és tagfüggvények öröklése
b) behelyettesíthetőség, bárhová behelyettesíthető a „tehén” ahol „állat van

Tagváltozók és tagfüggvények öröklése

```
class Person
{
        string name;
        int birthYear;
public:
        Person(string name, int birthYear):
                name(name), birthYear(birthYear) {} //tagváltozók inicializálása

        void print() { cout << name << ' ' << birthYear << endl;

        void setBirthYear(…) {…}
};

class Employee: public Person //ezzel örökölte az összes tagváltozót és tagfüggvényt
{
        int employmentYear;
public:

        void setEmploymentYear(…) {…} //így megírtuk azokat a függvényeket,
                                        //amikben      különbözik Employee és Person
        //konstruktor

        Employee(string name, int birthYear, int employmentYear): Person(name, birthYear),
        employmentYear(employmentYear) {}
};
```

```
//hogyan kell inicializálni?
…
Employee e(„Ubul”, 1964, 1965);  //első két tag az ősosztály konstruktora által hívódik meg
e.setBirthday(1967);                    //ez az ősosztály függvénye

Person p(„Garfield”, 1965);
p.setBirthYear(1977);
e.setEmploymentYear(1978);
p.setEmploymentYear(1978);       //HIBA, mert Person-nek nincs ilyen tagfüggvénye
```

```
Person *pp = &p;
pp = &e;

pp->setBirthYear(1971);    //e-nek így beállítjuk a birthYearjét OK
pp->setEmploymentYear(1972);

Employee* pe = (Employee*) pp;
```
[*Person* pointer nem tudjuk hogy Personra vagy leszármazottjára mutat* ➔ *polimorfizmus***]**
```
pe->setEmploymentYear(1973);   //OK
```

```
class Employee: public Person
{
        int employmentYear;
public:
        void print()
        {
                Person :: print(); cout<<employmentYear;
        }
}

e.print(); // Employee print() hívódik
p.print(); // Person print() hívódik

pe->print();  // Employee print() hívódik
pp->print();  //nem tudjuk mire mutat → pointer csak az első két változót látja→Person-ként kezeli
```

---

csináljunk inkább új print() függvényt:

```
…

print()
{
        cout<< name << ’ ’ << birthYear << ’ ’ << employmentYear;
                                //ez fordítási hiba mert private tagváltozókra hivatkozunk
}
```

➔ private és public sem jó ~ itt jön be a ***protected***

protected: olyan mint a private de öröklött osztályok látják

Tag/öröklés	***public***	***protected***	***private***
public	public	protected	private
protected	protected	protected	private
private	---	---	---