ПРЕОБРАЗУВАНЕ НА ТИПОВЕ

гл.ас., д-р. Нора Ангелова

Преобразуване на типове

ПРЕОБРАЗУВАНЕ НА ТИПОВЕ

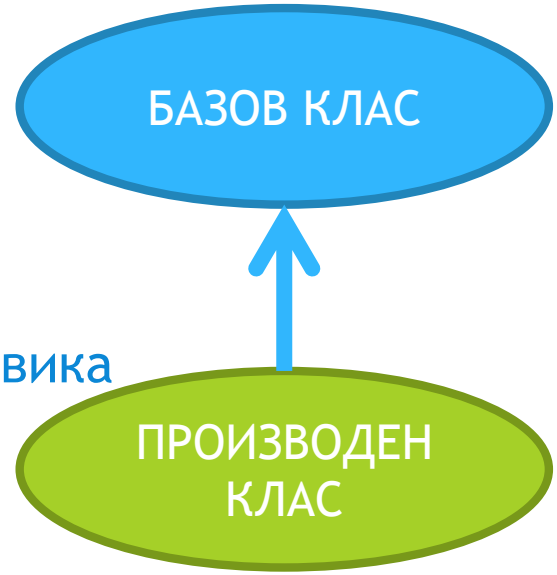
- Ако наследяването на базовия клас е с атрибут public, възможно е заменяне на обекти.
- Могат да се заменят обекти, псевдоними на обекти, указатели към обекти.

Заменянето може да се извършва при:

1. инициализация;
2. присвояване;
3. предаване на параметри на функции;

Две посоки на замяна:

- „производен с основен“ - безопасна;
- „основен с производен“ - може да предизвика проблеми;



„ПРОИЗВОДЕН ОСНОВЕН“

- Обект, псевдоним на обекти,
  указател към обект на производен клас се преобазуват съответно в
  обект, псевдоним, указател към обект на основен клас с неявни стандартни преобразувания.

- Свежда се до използването на наследените компоненти на класа.

Пример:

```
der d;
base b = d;

base& b1 = d;

der * dPtr = &d;
base * bPtr =  dPtr;
```

„ПРОИЗВОДЕН ОСНОВЕН“

Пример:

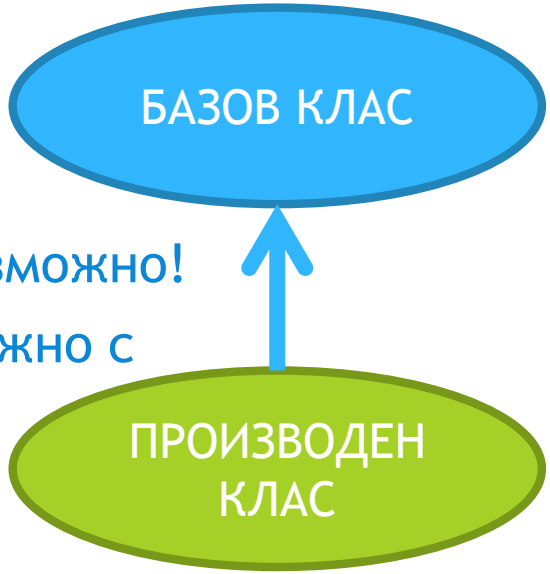
```
der d;
base b = d;


base& b1 = d;


der * dPtr = &d;
base * bPtr =  dPtr;
```

Достъп до собствените компоненти на производен клас:

1. чрез обект на основен клас (b) – не е възможно!
2. указател или псевдоним (b1, b2) – възможно с преобразувания.



„ПРОИЗВОДЕН ОСНОВЕН“

Пример:

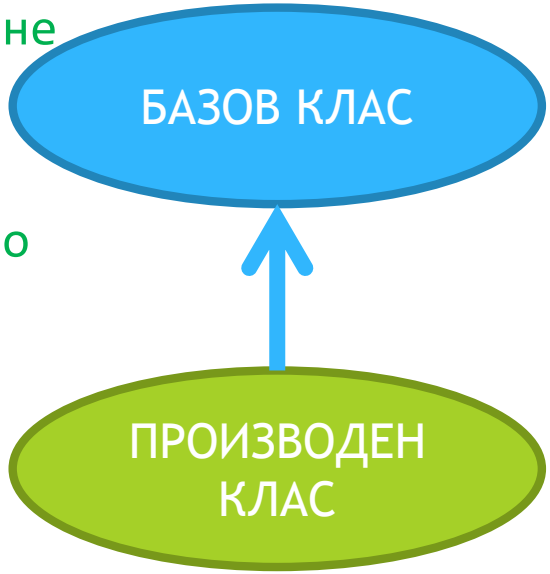
```
der d;
base b = d;

base& b1 = d;

der * dPtr = &d;
base * bPtr =  dPtr;

dPtr->derFunction(); // обичайно извикване
bPtr-> derFunction();// недопустимо

((der*) bPtr)->derFunction(); // възможно
```



* Забежка: ‘->’ и ‘.’ са с по-висок приоритет от преобразуването на типовете.

„ПРОИЗВОДЕН ОСНОВЕН“

Пример:

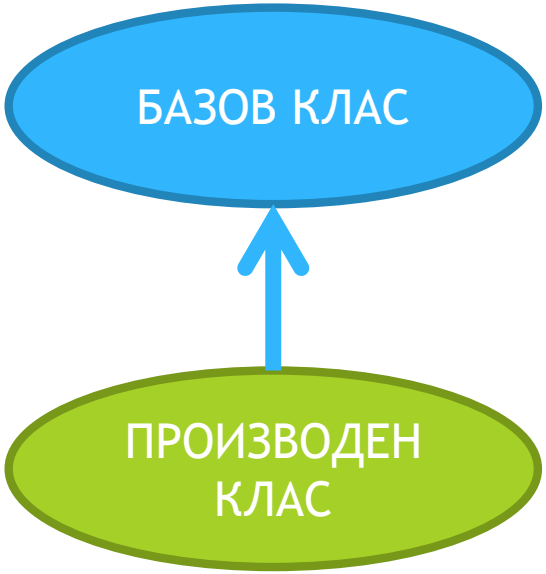
```
der d;
base b = d;

der& d1 = d;
base& b1 = d;

der * dPtr = &d;
base * bPtr =  dPtr;

d1.derFunction(); // обичайно извикване
b1.derFunction(); // недопустимо

((der&) b1).derFunction(); // възможно
```

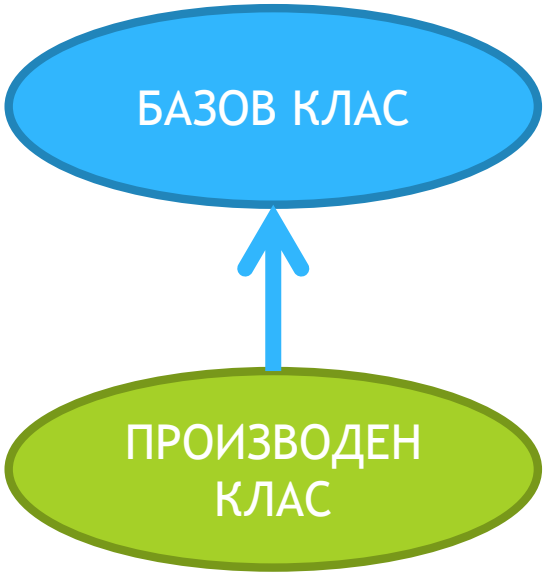


* Забежка: ‘->’ и ‘.’ са с по-висок приоритет от преобразуването на типовете.

„ПРОИЗВОДЕН ОСНОВЕН“

- Основният клас не съдържа собствените компоненти на производния клас.
- Реализира се чрез явно указване.

```
der y;
der * ptr = &y;
base * basePtr = ptr;
```



- Указателите сочат към обекта y.
- Указателите са различни.
- С указателя ptr са допустими извиквания на методи на класа der.
- С указателя basePtr НЕ са допустими извиквания на методи на класа der

Могат да се реализират само с явно преобразуване:

```
((der *) basePtr)->derFunc();
```

„Основен производен“

- Основният клас не съдържа собствените компоненти на производния клас.

- Опасна операция!
- Собствените компоненти остават неинициализирани.
- Опитът за използването на член-данните може да доведе до сериозни последици!
- Съществуват реализации на езика, които няма да извършат това преобразуване.
  В останалите реализицията може да се извърши с явно указване:

```
base x;
der y = (der) x;
```

Указател към метод

- Указател към метод на основният клас

```
void (base::*bPtr)() = base::bFunc;
```

- За да се използва този указател, е необходимо той да се свърже с конкретен обект.

```
base x;
(x.*bPtr)();
```

Възможно е:

```
void (der::*dPtr)() = bPtr;
der y;
(y.*dPtr)();
```

Ще се извърши неявно преобразуване на типовете.
Обектът y съдържа наследената част от основния клас.

Указател към метод

- Обратното присвояване – изисква явно преобразуване и дали ще се използва правилно, зависи единствено от програмиста.

```
void (der::*dPtr)() = der::dFunc;
void (base::*bPtr)() = dPtr;
```

```
void (base::*bPtr)();
bPtr = (void (base::*)()) der::dFunc;
```

Използването може да доведе до грешка или неоднозначност.

КРАЙ