

Atvasinātās klases

```
class klases_vārds : { private  
                      protected } bāzes_klase  
                      public  
  
{ locekļu deklarācijas };
```

- n Pēc noklusēšanas mantošanas likums ir `private`.
- n Klase manto visus bāzes klases locekļus, izņemot konstruktorus, destruktoru un metodi `operator=()`.
- n Klases funkcijās var lietot savas bāzes klases `public` un `protected` locekļus.
- n Ja mantošanas likums ir `private`, tad atvasinātajā klasē visi mantotie locekļi kļūst par `private`.
- n Ja mantošanas likums ir `protected`, tad atvasinātajā klasē visi mantotie `public` locekļi kļūst par `protected`.
- n Ja mantošanas likums ir `public`, tad atvasinātajā klasē visi mantotie locekļi saglabā savu pieejamību tādu, kāda tā ir bāzes klasē.

80

Atvasinātās klases (turpinājums)

| | | |
|---|--|---|
| <pre>class B { int a; protected: int b; public: int c; };</pre> | | |
| <pre>class S1 : B { int x; public: int y; };</pre> | <pre>class S2 : protected B { int x; public: int y; };</pre> | <pre>class S3 : public B { int x; public: int y; };</pre> |
| Klasē S1: y – public x, a, b, c – private (a nav pieejams) | Klasē S2: x, a – private (a nav pieejams) y – public b, c – protected | Klasē S3: x, a – private (a nav pieejams) b – protected y, c – public |

81

Atvasinātās klases, to konstruktori un destruktori

- n Kad tiek radīts atvasinātās klases objekts, vispirms tiek izsaukts bāzes klases konstruktors un pēc tam atvasinātās klases konstruktors.
- n Kad tiek likvidēts atvasinātās klases objekts, vispirms tiek izsaukts atvasinātās klases destruktors, un pēc tam bāzes klases destruktors.

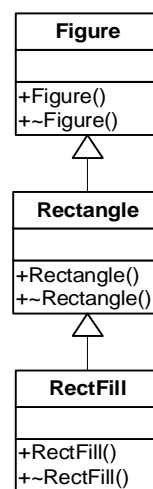
82

Atvasinātās klases, to konstruktori un destruktori

```
class Figure
{
public:
    Figure() { cout<<"Figure created\n"; }
    ~Figure() { cout<<"Figure destroyed\n"; }
};

class Rectangle : public Figure
{
public:
    Rectangle() { cout<<"Rectangle created\n"; }
    ~Rectangle() { cout<<"Rectangle destroyed\n"; }
};

class RectFill : public Rectangle
{
public:
    RectFill() { cout<<"RectFill created\n"; }
    ~RectFill() { cout<<"RectFill destroyed\n"; }
};
```



83

Atvasinātās klases, to konstruktori un destruktori

```
void main()
{   RectFill *prf;           // 1.
    prf = new RectFill;      // 2.
    Rectangle r;             // 3.
    delete prf;              // 4.
}
```

| | | |
|---------------------|---|----|
| Figure created | } | 2. |
| Rectangle created | | |
| RectFill created | } | 3. |
| Figure created | | |
| Rectangle created | } | 4. |
| RectFill destroyed | | |
| Rectangle destroyed | } | 5. |
| Figure destroyed | | |
| Rectangle destroyed | } | |
| Figure destroyed | | |

84

Atvasinātās klases, to konstruktori un destruktori

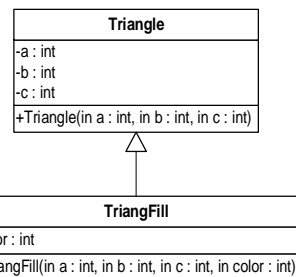
Kā norādīt, kurš no bāzes klases konstruktoriem jālieto, kad rada atvasinātās klases objektu?

```
class Triangle
{   int a, b, c;
    public: Triangle(int a, int b, int c);
    ...
};
```

```
class TriangFill : public Triangle
{   int color;
    public: TriangFill(int a, int b, int c, int color);
    ...
}
```

```
Triangle::Triangle(int a, int b, int c)
{   this->a = a; this->b = b; this->c = c; }
```

```
TriangFill::TriangFill(int aa,int bb,int cc,int color):Triangle(aa,bb,cc)
{this->color = color;}
```



85

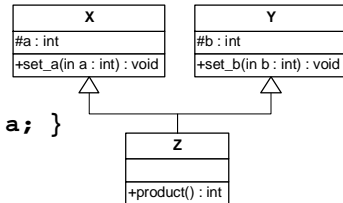
Daudzkārtējā mantošana

- n Klasi var atvasināt no vairākām bāzes klasēm.
- n Klase manto locekļus no visām savām bāzes klasēm.

```
class X
{
    protected: int a;
    public: void set_a(int a) { this->a = a; }
};

class Y
{
    protected: int b;
    public: void set_b(int b) { this->b = b; }
}

class Z : public X, public Y
{
    public: int product() { return a * b }
};
```



86

Daudzkārtējā mantošana (turpinājums)

```
class A
{
    public:
        int a, b;
        A(){a=1; b=2;}
};
```

sizeof(A) == 8

```
class B
{
    public:
        int a, c;
        B(){a=3; c=4;}
};
```

sizeof(B) == 8

```
class C : public A, public B
{
    public: int x, y;
};
```

sizeof(C) == 24

```
void main()
{
    C s;
    //cout << s.a; // KĻŪDA!
    cout << s.A::a << " " << s.B::a << endl;    //izvada: 1, 3
    cout << s.b << " " << s.c << endl;        //izvada: 2, 4
}
```

87

Daudzkārtējā mantošana (turpinājums)

```
class A{  
public:  
    int a;  
    int b;  
    A(){ a = 1; b = 2; }  
};
```

```
class S1 : public A {  
public:  
    int x1;  
    S1(){ x1 = 5;}  
};
```

```
class S2 : public A {  
public:  
    int x2;  
    S2(){ x2 = 6;}  
};
```

```
class S12 : public S1, public S2 {  
public:  
    int x12;  
    S12(){ x12 = 7;}  
};
```

Cik mainīgo būs klases S12 objektam?

88