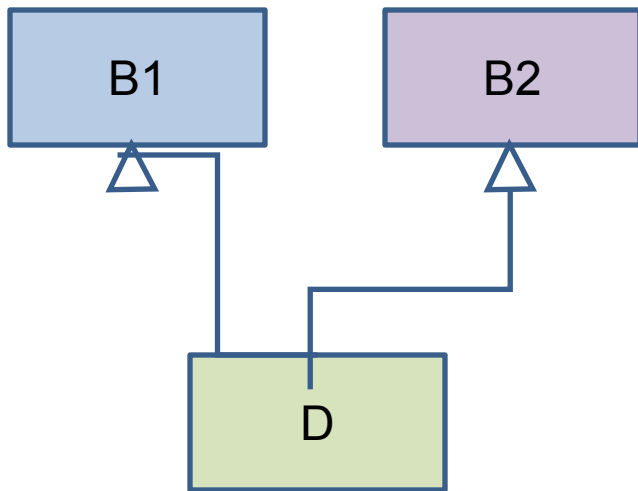


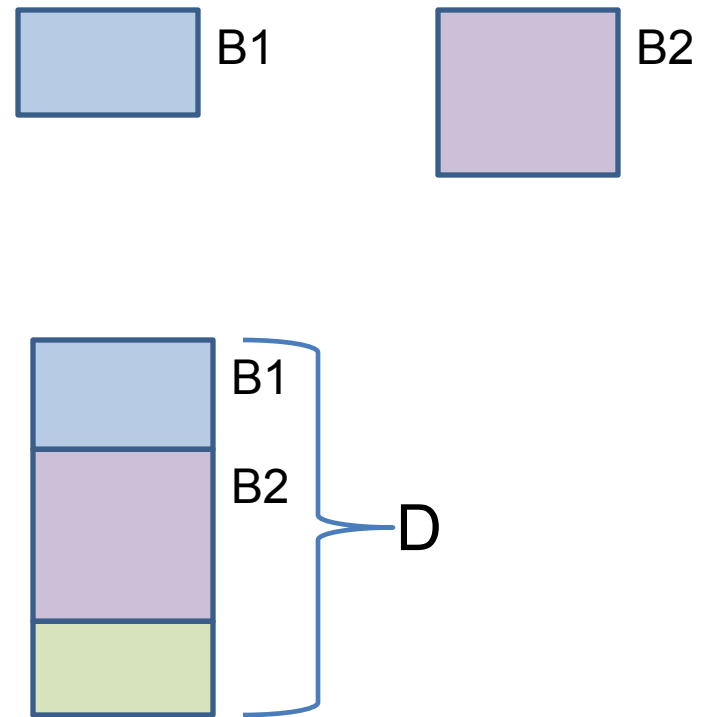
13. Множественное наследование

13.1. Множественное наследование

Иерархия классов



Состояние классов



13.1. Множественное наследование (продолжение)

```
class B1 { . . . };
```

```
class B2 { . . . };
```

```
class D: public B1, public B2 { . . . };
```

Важно:

```
class D: public B1, B2 { . . . };
```

эквивалентно

```
class D: public B1, private B2 { . . . };
```

13.2. Конструкторы и деструктор класса

Конструктор производного класса:

```
D::D( . . . ) : B1( . . . ), B2( . . . ) { . . . }
```

Порядок вызова конструкторов базовых классов

Деструктор производного класса

```
D::~~D() { . . . }
```

Вызов деструкторов базовых классов

13.3. Методы класса

```
class B1 {  
    . . .  
    public:  
        void f1( . . . );  
        void f( . . . );  
  
    . . .  
};
```

```
class B2 {  
    . . .  
    public:  
        void f2( . . . );  
        void f( . . . );  
  
    . . .  
};
```

13.3. Методы класса

(продолжение)

```
class D: public B1, public B2 {  
    . . .  
public:  
    void g( . . . );  
    void f( . . . );  
    . . .  
};
```

13.3. Методы класса

(продолжение)

```
void D::g( . . . )
{
    f1( . . . );
    f2( . . . );
}
void D::f( . . . )
{
    B1::f( . . . );
    B2::f( . . . );
}
```

```
D ob;

ob.g( . . . );
ob.f( . . . );
ob.f1( . . . );
ob.f2( . . . );

ob.B1::f( . . . );
ob.B2::f( . . . );
```

13.4. Указатели на классы

```
B1 *p1;
```

```
B2 *p2;
```

```
...
```

```
p1 = new D( . . . );
```

```
p2 = new D( . . . );
```

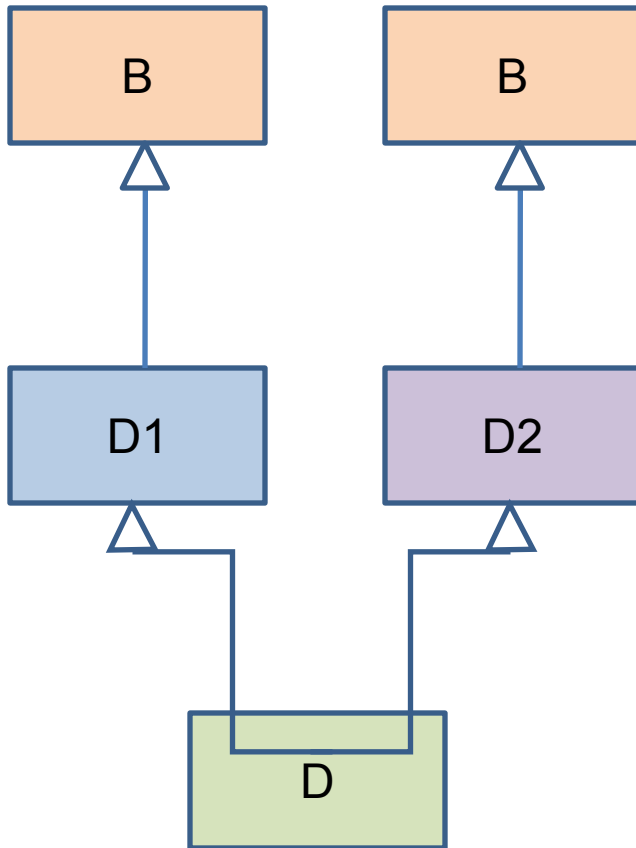
```
p1->f1( . . . );
```

```
p2->f2( . . . );
```

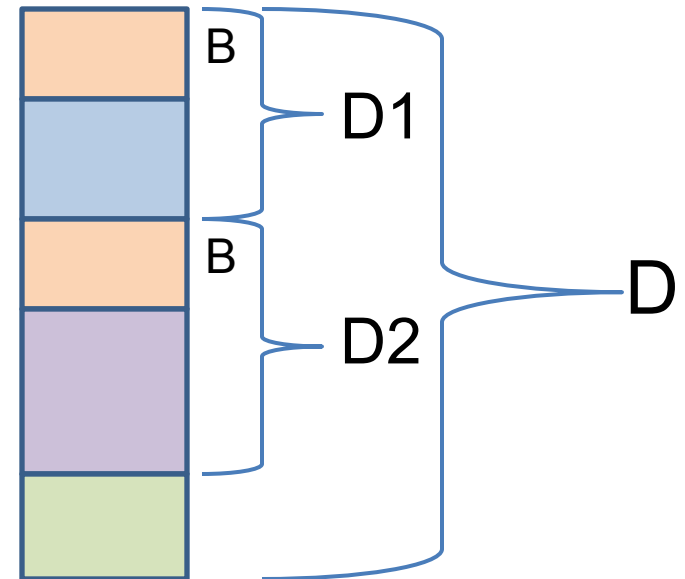
```
p1->f( . . . ); p2->f( . . . );
```


13.5. Сложная иерархия классов

Иерархия классов



Состояние класса



13.5. Сложная иерархия классов

(продолжение)

```
class B {  
private:  
    int x[10];  
public:  
    . . .  
};
```

13.5. Сложная иерархия классов

(продолжение)

```
class D1: public B {  
private:  
    int y, w[3];  
public:  
    . . .  
};
```

```
class D2: public B {  
private:  
    int y;  
public:  
    . . .  
};
```

13.5. Сложная иерархия классов

(продолжение)

```
class D12: public D1, public D2 {  
private:  
    int z;  
public:  
    D12(...): D1(...), D2(...) { ... }  
    ...  
};
```

13.5. Сложная иерархия классов

(продолжение)

```
int main()
{
    B a;
    cout << "sizeof(a) = " << sizeof(a) <<
endl;
```

sizeof(a) = 40

13.5. Сложная иерархия классов

(продолжение)

```
D1 a1;  
cout << "sizeof(a1) = " << sizeof(a1) <<  
endl;
```

```
sizeof(a1) = 56
```

13.5. Сложная иерархия классов

(продолжение)

```
D2 a2;  
cout << "sizeof(a2) = " << sizeof(a2) <<  
endl;
```

`sizeof(a2) = 44`

13.5. Сложная иерархия классов

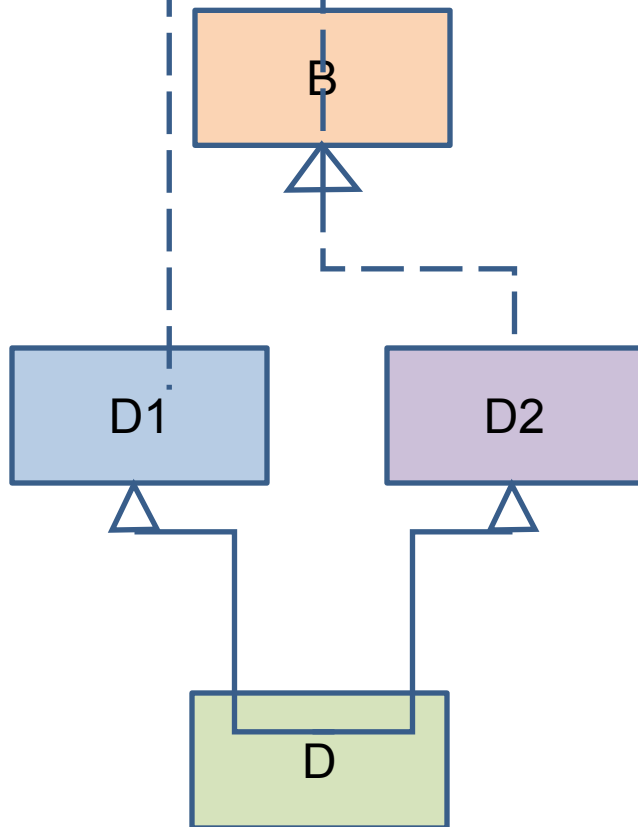
(продолжение)

```
D12 a12;  
cout << "sizeof(a12) = " << sizeof(a12) <<  
endl;
```

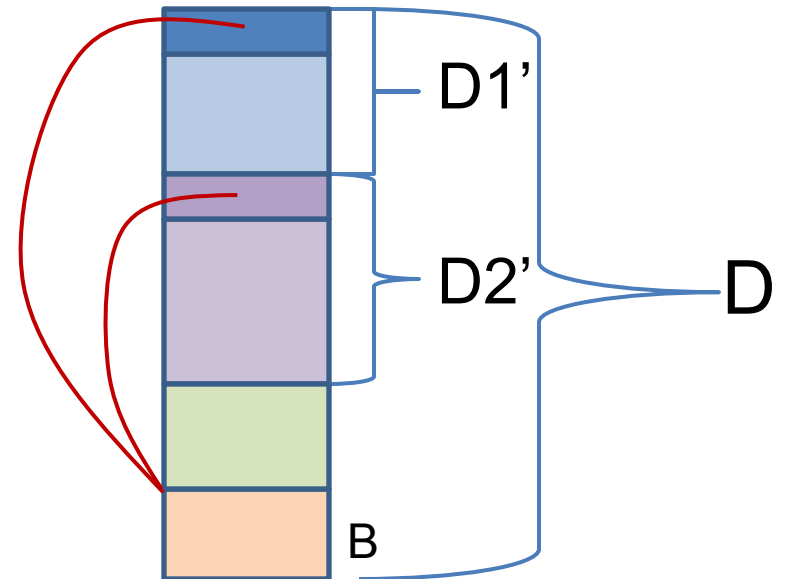
`sizeof(a12) = 104`

13.6. Виртуальный базовый класс

Иерархия классов



Состояние класса



13.6. Виртуальный базовый класс

(продолжение)

```
class B {  
private:  
    int x[10];  
public:  
    . . .  
};
```

13.6. Виртуальный базовый класс (продолжение)

```
class D1:  
    virtual public B {  
private:  
    int y, w[3];  
public:  
    . . .  
};
```

```
class D2:  
    virtual public B {  
private:  
    int y;  
public:  
    . . .  
};
```

13.6. Виртуальный базовый класс (продолжение)

```
class D12: public D1, public D2 {  
private:  
    int z;  
public:  
    D12(...): D1(...), D2(...), B(...) { ... }  
    ...  
};
```

13.6. Виртуальный базовый класс (продолжение)

```
int main()
{
    B a;
    cout << "sizeof(a) = " << sizeof(a) <<
endl;
```

sizeof(a) = 40

13.6. Виртуальный базовый класс (продолжение)

```
D1 a1;  
cout << "sizeof(a1) = " << sizeof(a1) <<  
endl;
```

```
sizeof(a1) = 60
```

13.6. Виртуальный базовый класс (продолжение)

```
D2 a2;  
cout << "sizeof(a2) = " << sizeof(a2) <<  
endl;
```

`sizeof(a2) = 48`

13.6. Виртуальный базовый класс

(продолжение)

```
D12 a12;  
cout << "sizeof(a12) = " << sizeof(a12) <<  
endl;
```

`sizeof(a12) = 72`