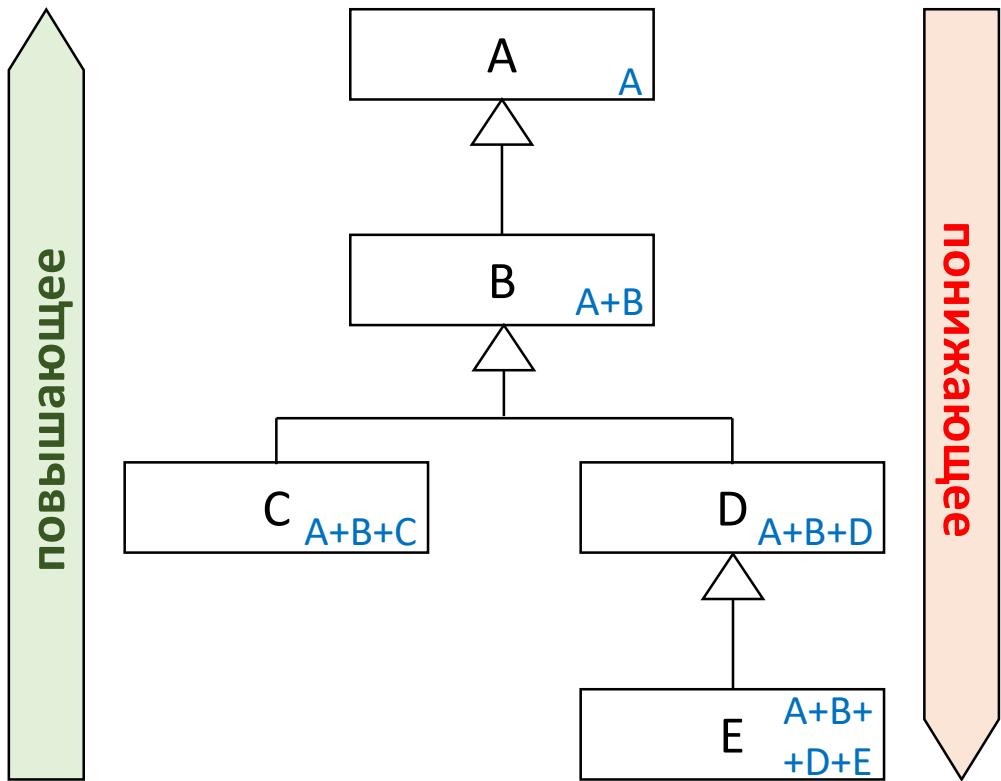


# Объектно-ориентированное программирование

Наследование и полиморфизм

Гришмановский Павел Валерьевич,  
кафедра автоматики и компьютерных систем, Политехнический институт, СурГУ

# Повышающее и понижающее преобразования



При повышающем и понижающем преобразованиях изменяется только трактовка указателя или ссылки, но над объектом никаких действий не выполняется

**Повышающим** преобразованием называют приведение типа указателя или ссылки на объект **от потомка к предку** (вверх)

- является безопасным, выполняется автоматически (неявно)

**Понижающим** преобразованием называют приведение типа указателя или ссылки на объект **от предка к потомку** (вниз)

- является потенциально опасным, автоматически не выполняется (только явно)

# Виртуальные методы

- **virtual <прототип>;** – виртуальный метод
- **override (C++11)**
- **virtual <прототип> =0 [const];** – абстрактный метод
- абстрактный класс

# Виртуальные методы

```
class A
{
public:
    void Attach();
    virtual void Out();
    virtual double Calc() = 0;
    virtual int State();
    A();
    virtual ~A();
};

class B : public A
{
public:
    void Attach();
    void Detach(int chain);
    virtual void Out();
    virtual double Calc();
    virtual int State(int chain);
    B();
    ~B();
};
```

A::VMT – virtual methods table of class A

#	Signature	Address
0	Out()	abcd0100
1	Calc()	0
2	State()	abcd0200
3	~A()	abcd0400

B::VMT – virtual methods table of class B

#	Signature	Address
0	Out()	abcd0700
1	Calc()	abcd0800
2	State()	abcd0200
3	~B()	abcd0b00

CS – code segment

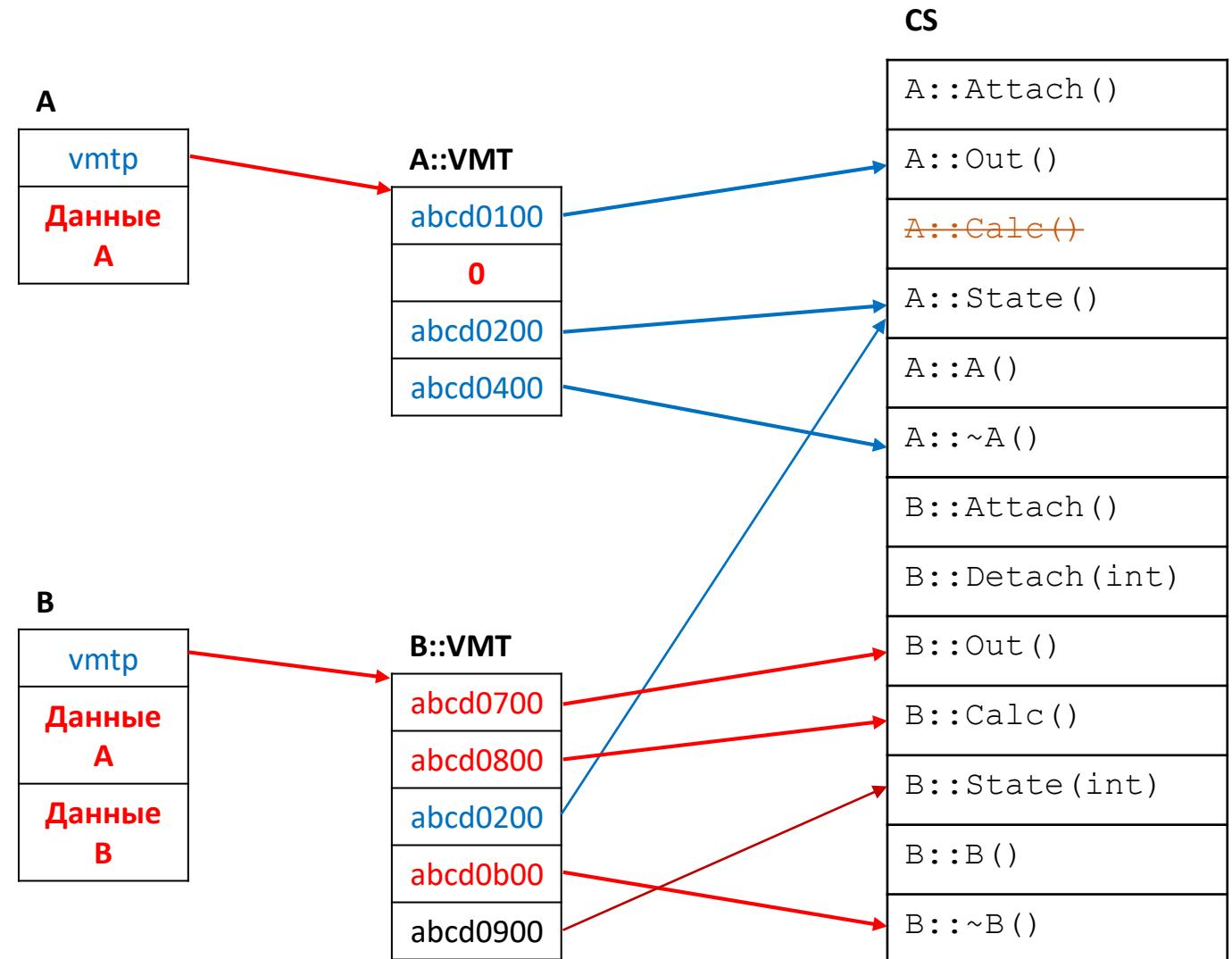
A::Attach()
A::Out()
A::Calc()
A::State()
A::A()
A::~A()
B::Attach()
B::Detach(int)
B::Out()
B::Calc()
B::State(int)
B::B()
B::~B()



# Виртуальные методы

```
class A
{
public:
    void Attach();
    virtual void Out();
    virtual double Calc() = 0;
    virtual int State();
    A();
    virtual ~A();
};

class B : public A
{
public:
    void Attach();
    void Detach(int chain);
    virtual void Out();
    virtual double Calc();
    virtual int State(int chain);
    B();
    ~B();
};
```



# Виртуальные методы

```
class A
{
public:
    void Attach();
    virtual void Out();
    virtual double Calc() = 0;
    virtual int State();
    A();
    virtual ~A();
};

class B : public A
{
public:
    void Attach();
    void Detach(int chain);
    virtual void Out();
    virtual double Calc();
    virtual int State(int chain);
    B();
    ~B();
};
```

