

Esercizio 2

```
class A {
protected:
    virtual void j() { cout<<" A::j "; }
public:
    virtual void g() const { cout <<" A::g "; }
    virtual void f() { cout <<" A::f "; g(); j(); }
    void m() { cout <<" A::m "; g(); j(); }
    virtual void k() { cout <<" A::k "; j(); m(); }
    virtual A* n() { cout <<" A::n "; return this; }
};
```

```
class C: public A {
private:
    void j() { cout <<" C::j "; }
public:
    virtual void g() { cout <<" C::g "; }
    void m() { cout <<" C::m "; g(); j(); }
    void k() const { cout <<" C::k "; k(); }
};
```

```
class B: public A {
public:
    virtual void g() const override { cout <<" B::g "; }
    virtual void m() { cout <<" B::m "; g(); j(); }
    void k() { cout <<" B::k "; A::n(); }
    A* n() override { cout <<" B::n "; return this; }
};
```

```
class D: public B {
protected:
    void j() { cout <<" D::j "; }
public:
    B* n() final { cout <<" D::n "; return this; }
    void m() { cout <<" D::m "; g(); j(); }
};
```

A* p1 = new D(); A* p2 = new B(); A* p3 = new C(); B* p4 = new D(); const A* p5 = new C();

(static_cast<B*>(p3->n()))->g();

A::m A::g

Esercizio 2

```
class A {
protected:
    virtual void j() { cout<<" A::j "; }
public:
    virtual void g() const { cout <<" A::g "; }
    virtual void f() { cout <<" A::f "; g(); j(); }
    void m() { cout <<" A::m "; g(); j(); }
    virtual void k() { cout <<" A::k "; j(); m(); }
    virtual A* n() { cout <<" A::n "; return this; }
};
```

```
class C: public A {
private:
    void j() { cout <<" C::j "; }
public:
    virtual void g() { cout <<" C::g "; }
    void m() { cout <<" C::m "; g(); j(); }
    void k() const { cout <<" C::k "; k(); }
};
```

```
class B: public A {
public:
    virtual void g() const override { cout <<" B::g "; }
    virtual void m() { cout <<" B::m "; g(); j(); }
    void k() { cout <<" B::k "; A::n(); }
    A* n() override { cout <<" B::n "; return this; }
};
```

```
class D: public B {
protected:
    void j() { cout <<" D::j "; }
public:
    B* n() final { cout <<" D::n "; return this; }
    void m() { cout <<" D::m "; g(); j(); }
};
```

A* p1 = new D(); A* p2 = new B(); A* p3 = new C(); B* p4 = new D(); const A* p5 = new C();

static_cast<C*>(p2)->k()

STACK OVERFLOW

C::k C::k ...

Esercizio 2

```
class A {
protected:
    virtual void j() { cout<<" A::j "; }
public:
    virtual void g() const { cout <<" A::g "; }
    virtual void f() { cout <<" A::f "; g(); j(); }
    void m() { cout <<" A::m "; g(); j(); }
    virtual void k() { cout <<" A::k "; j(); m(); }
    virtual A* n() { cout <<" A::n "; return this; }
};
```

```
class C: public A {
private:
    void j() { cout <<" C::j "; }
public:
    virtual void g() { cout <<" C::g "; }
    void m() { cout <<" C::m "; g(); j(); }
    void k() const { cout <<" C::k "; k(); }
};
```

A* p1 = new D(); A* p2 = new B(); A* p3 = new C(); B* p4 = new D(); const A* p5 = new C();

```
class B: public A {
public:
    virtual void g() const override { cout <<" B::g "; }
    virtual void m() { cout <<" B::m "; g(); j(); }
    void k() { cout <<" B::k "; A::n(); }
    A* n() override { cout <<" B::n "; return this; }
};
```

```
class D: public B {
protected:
    void j() { cout <<" D::j "; }
public:
    B* n() final { cout <<" D::n "; return this; }
    void m() { cout <<" D::m "; g(); j(); }
};
```

(dynamic_cast<B*>(p1))->m()

✓ SO NO N RIGIAN A THIS, CONVERSION SUBITO!

A p1 = new D();
[B p1 = new D();]

CATTURA (SIA CON STATIC CHE CON DYNAMIC) IL TIPO A SX!

Esercizio 2

```
class A {
protected:
    virtual void j() { cout<<" A::j "; }
public:
    virtual void g() const { cout <<" A::g "; }
    virtual void f() { cout <<" A::f "; g(); j(); }
    void m() { cout <<" A::m "; g(); j(); }
    virtual void k() { cout <<" A::k "; j(); m(); }
    virtual A* n() { cout <<" A::n "; return this; }
};
```

```
class C: public A {
private:
    void j() { cout <<" C::j "; }
public:
    virtual void g() { cout <<" C::g "; }
    void m() { cout <<" C::m "; g(); j(); }
    void k() const { cout <<" C::k "; k(); }
};
```

```
class B: public A {
public:
    virtual void g() const override { cout <<" B::g "; }
    virtual void m() { cout <<" B::m "; g(); j(); }
    void k() { cout <<" B::k "; A::n(); }
    A* n() override { cout <<" B::n "; return this; }
};
```

```
class D: public B {
protected:
    void j() { cout <<" D::j "; }
public:
    B* n() final { cout <<" D::n "; return this; }
    void m() { cout <<" D::m "; g(); j(); }
};
```

```
A* p1 = new D(); A* p2 = new B(); A* p3 = new C(); B* p4 = new D(); const A* p5 = new C();
```

(p5→n())→g();

NON COMPILA