```
Esercizio 2
```

```
class A {
                                                                                                   class B: public A {
protected:
                                                                                                   public:
                                                                                                      virtual void g() const override { cout <<" B::g "; }
virtual void m() { cout <<" B::m "; g(); j(); }
void k() { cout <<" B::k "; A::n(); }</pre>
   virtual void j() { cout<<" A::j "; }</pre>
public:
  virtual void g() const { cout <<" A::g "; }
virtual void f() { cout <<" A::f "; g(); j(); }</pre>
                                                                                                      A* n() override { cout <<" B::n "; return this; }
  void m() { cout << "A::m "; g(); j(); }
virtual void k() { cout << "A::k "; j(); m(); }
virtual A* n() { cout << "A::n "; return this; }</pre>
class C: public A {
                                                                                                   class D: public B {
private:
                                                                                                   protected:
   void j() { cout <<" C::j "; }</pre>
                                                                                                      void j() { cout <<" D::j "; }</pre>
public:
                                                                                                   public:
                                                                                                     B* n() final { cout <<" D::n "; return this; }
void m() { cout <<" D::m "; g(); j(); }</pre>
  virtual void g() { cout <<" C::g "; }
void m() { cout <<" C::m "; g(); j(); }
void k() const { cout <<" C::k "; k(); }</pre>
}:
A* p1 = new D(); A* p2 = new B(); A* p3 = new C(); B* p4 = new D(); const A* p5 = new C();
```

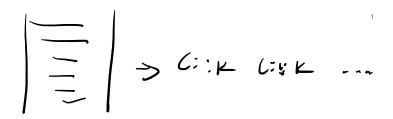
 $(static_cast< B*>(p3\rightarrow n()))\rightarrow g());$

A::M

Esercizio 2

```
class A {
                                                                                         class B: public A {
protected:
                                                                                        public:
                                                                                           virtual void g() const override { cout <<" B::g "; }
virtual void m() { cout <<" B::m "; g(); j(); }
void k() { cout <<" B::k "; A::n(); }</pre>
   virtual void j() { cout<<" A::j "; }</pre>
  virtual void g() const { cout <<" A::g "; }</pre>
  virtual void f() { cout <<" A::f "; g(); j(); }</pre>
                                                                                           A* n() override { cout <<" B::n "; return this; }
  void m() { cout <<" A::m "; g(); j(); }
virtual void k() { cout <<" A::k "; j(); m(); }
virtual A* n() { cout <<" A::n "; return this; }</pre>
class C: public A {
                                                                                        class D: public B {
private:
                                                                                        protected:
  void j() { cout <<" C::j "; }</pre>
                                                                                           void j() { cout <<" D::j "; }</pre>
                                                                                         public:
                                                                                           B* n() final { cout <<" D::n "; return this; }
void m() { cout <<" D::m "; g(); j(); }</pre>
  virtual void g() { cout <<" C::g "; }
void m() { cout <<" C::m "; g(); j(); }</pre>
  void k() const { cout << " C::k "; k(); }</pre>
A* p1 = new D(); A* p2 = new B(); A* p3 = new C(); B* p4 = new D(); const A* p5 = new C();
```

 $static_cast<C*>(p2))\rightarrow k()$ 1 STACK OVERSION



Esercizio 2

```
class B: public A {
class A {
                                                                                                                                                                                                                       2
protected:
                                                                                                                         public:
                                                                                                                             DDIC:
virtual void g() const override { cout <<" B::q "; }
virtual void m() { cout <<" B::m "; g(); j(); }
void k() { cout <<" B::k "; A::n(); }
A* n() override { cout <<" B::n "; return this; }</pre>
    virtual void j() { cout<<" A::j "; }
public:
   ublic:
    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; }
}</pre>
class C: public A {
                                                                                                                         class D: public B {
                                                                                                                         protected:
private:
   void j() { cout <<" C::j "; }</pre>
                                                                                                                             void j() { cout <<" D::j "; }</pre>
public:
                                                                                                                         public:
                                                                                                                            ublic:
B* n() final { cout <<" D::n "; return this; }
void m() { cout <<" D::m "; g(); j(); }</pre>
   virtual void g() { cout <<" C::g "; }
void m() { cout <<" C::m "; g(); j(); }
void k() const { cout <<" C::k "; k(); }</pre>
A* p1 = new D(); A* p2 = new B(); A* p3 = new C(); B* p4 = new D(); const A* p5 = new C();
```

J SO NO N REFORM A 2 THIS

)→m() CONVOINT SUBITO! $(dynamic_cast<B*>(p1))\rightarrow m()$ CATIGIA CSIA CON STATIC CHE CON DYNAMIC) IL TIPO ASX!

A P1 = WW D ().

(B P1 = NOW DC):

Esercizio 2

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

class C: public A {
    protected:
    void j() { cout <<" C::j "; }
    public:
    virtual void g() { cout <<" C::g "; }
    void m() { cout <<" C::m "; g(); j(); }
    void m() { cout <<" C::m "; g(); j(); }
    void m() { cout <<" C::m "; g(); j(); }
    void m() { cout <<" C::m "; g(); j(); }
    void m() { cout <<" C::m "; g(); j(); }
    void m() { cout <<" C::m "; g(); j(); }
    void m() { cout <<" C::m "; g(); j(); }
    void m() { cout <<" C::m "; g(); j(); }
    void m() { cout <<" C::m "; g(); j(); }
    void m() { cout <<" C::m "; g(); j(); }
    void m() { cout <<" C::m "; g(); j(); }
    void m() { cout <<" D::m "; g(); j(); }
};

A* pl = new D(); A* p2 = new B(); A* p3 = new C(); B* p4 = new D(); const A* p5 = new C();</pre>
```

 $(p5\rightarrow n())\rightarrow g();$

NON COMPILA