

↓
2 (p1 → n()) → g(). → n()

Esercizio 1 Esercizio Cosa Stampa

```

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

class D: public B {
protected:
    void h() {cout <<"D::h ";}
public:
    virtual void g() {cout <<"D::g ";}
    void k() const {cout <<"D::k "; k();}
    void m() {cout <<"D::m "; g(); h();}
};

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

class C: public B {
public:
    virtual void g() const {cout <<"C::g ";}
    void k() {cout <<"C::k "; B::n();}
    virtual void m() {cout <<"C::m "; g(); h();}
    B* n() {cout <<"C::n "; return this;}
};

class E: public C {
protected:
    void h() {cout <<"E::h ";}
public:
    void m() {cout <<"E::m "; g(); h();}
    C* n() {cout <<"E::n "; return this;}
};

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

```

S5

CONST-CAST

ALLORA

S1

Le precedenti definizioni compilano correttamente. Per ognuno dei seguenti 14 statement in tabella con **numerazione da 01 a 14**, scrivere **chiaramente nel foglio 14 risposte con numerazione da 01 a 14** e per ciascuna risposta:

- **NON COMPILA** se la compilazione dell'istruzione provoca un errore;
- **UNDEFINED BEHAVIOUR** se l'istruzione compila correttamente ma la sua esecuzione provoca un undefined behaviour o un errore a run-time;
- se l'istruzione compila correttamente e non provoca errori a run-time allora si scriva **chiaramente** la stampa che l'esecuzione produce in output su cout; se non provoca alcuna stampa allora si scriva **NESSUNA STAMPA**.

```

01: p1->g();
02: (p1->n())->g();
03: p2->f();
04: p2->m();
05: (static_cast<D*>(p2))->k();
06: p3->k();
07: p3->f();
08: (p3->n())->m();
09: (p3->n())->n()->g();
10: (static_cast<C*>(p3->n()))->g();
11: (p4->n())->m();
12: p5->g();
13: p5->k();
14: (dynamic_cast<C*>(p5))->m();

```

// SOLUZIONE

```

01: B::g
02: NON COMPILA
03: B::f C::g B::h
04: B::m C::g B::h
05: UNDEFINED BEHAVIOUR
06: B::k D::h B::m B::g D::h
07: B::f B::g D::h
08: B::n B::m B::g D::h
09: B::n B::n B::g
10: B::n B::g
11: E::n B::m C::g E::h
12: C::g
13: C::k B::n
14: E::m C::g E::h

```

Esercizio Cosa Stampa

5 (static_cast<D*> (p2)) -> k();

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

```
class D: public B {
protected:
    void h() {cout <<"D::h ";}
public:
    virtual void g() {cout <<"D::g ";}
    void k() const {cout <<"D::k "; k();}
    void m() {cout <<"D::m "; g(); h();}
};
```

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

```
class E: public C {
protected:
    void h() {cout <<"E::h ";}
public:
    void m() {cout <<"E::m "; g(); h();}
    C* n() {cout <<"E::n "; return this;}
};
```

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

SCHEGGIATO! STACK OVERFLOW = UB

Le precedenti definizioni compilano correttamente. Per ognuno dei seguenti 14 statement in tabella con **numerazione da 01 a 14**, scrivere **chiaramente nel foglio 14 risposte con numerazione da 01 a 14** e per ciascuna risposta:

- **NON COMPILA** se la compilazione dell'istruzione provoca un errore;
- **UNDEFINED BEHAVIOUR** se l'istruzione compila correttamente ma la sua esecuzione provoca un undefined behaviour o un errore a run-time;
- se l'istruzione compila correttamente e non provoca errori a run-time allora si scriva **chiaramente** la stampa che l'esecuzione produce in output su cout; se non provoca alcuna stampa allora si scriva **NESSUNA STAMPA**.

```
01: p1->g();
02: (p1->n())->g();
03: p2->f();
04: p2->m();
05: (static_cast<D*>(p2))->k();
06: p3->k();
07: p3->f();
08: (p3->n())->m();
09: (p3->n())->n()->g();
10: (static_cast<C*>(p3->n()))->g();
11: (p4->n())->m();
12: p5->g();
13: p5->k();
14: (dynamic_cast<C*>(p5))->m();
```

// SOLUZIONE

```
01: B::g
02: NON COMPILA
03: B::f C::g B::h
04: B::m C::g B::h
05: UNDEFINED BEHAVIOUR
06: B::k D::h B::m B::g D::h
07: B::f B::g D::h
08: B::n B::m B::g D::h
09: B::n B::n B::g
10: B::n B::g
11: E::n B::m C::g E::h
12: C::g
13: C::k B::n
14: E::m C::g E::h
```

Esercizio Cosa Stampa

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

```
class D: public B {
protected:
    void h() {cout<<"D::h ";}
public:
    virtual void g() {cout<<"D::g ";}
    void k() const {cout<<"D::k "; k();}
    void m() {cout<<"D::m "; g(); h();}
};
```

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

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

```
class E: public C {
protected:
    void h() {cout<<"E::h ";}
public:
    void m() {cout<<"E::m "; g(); h();}
    C* n() {cout<<"E::n "; return this;}
};
```

Le precedenti definizioni compilano correttamente. Per ognuno dei seguenti 14 statement in tabella con **numerazione da 01 a 14**, scrivere **chiaramente nel foglio 14 risposte con numerazione da 01 a 14** e per ciascuna risposta:

- **NON COMPILA** se la compilazione dell'istruzione provoca un errore;
- **UNDEFINED BEHAVIOUR** se l'istruzione compila correttamente ma la sua esecuzione provoca un undefined behaviour o un errore a run-time;
- se l'istruzione compila correttamente e non provoca errori a run-time allora si scriva **chiaramente** la stampa che l'esecuzione produce in output su `cout`; se non provoca alcuna stampa allora si scriva **NESSUNA STAMPA**.

```
01: p1->g();
02: (p1->n())->g();
03: p2->f();
04: p2->m();
05: (static_cast<D*>(p2))->k();
06: p3->k();
07: p3->f();
08: (p3->n())->m();
09: (p3->n())->n()->g();
10: (static_cast<C*>(p3->n()))->g();
11: (p4->n())->m();
12: p5->g();
13: p5->k();
14: (dynamic_cast<C*>(p5))->m();
```

// SOLUZIONE

```
01: B::g
02: NON COMPILA
03: B::f C::g B::h
04: B::m C::g B::h
05: UNDEFINED BEHAVIOUR
06: B::k D::h B::m B::g D::h
07: B::f B::g D::h
08: B::n B::m B::g D::h
09: B::n B::n B::g
10: B::n B::g
11: E::n B::m C::g E::h
12: C::g
13: C::k B::n
14: E::m C::g E::h
```

Esercizio Cosa Stampa

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

class D: public B {
protected:
    void h() {cout << "D::h ";}
public:
    virtual void g() {cout << "D::g ";}
    void k() const {cout << "D::k "; k();}
    void m() {cout << "D::m "; g(); h();}
};
```

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

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

```
class E: public C {
protected:
    void h() {cout << "E::h ";}
public:
    void m() {cout << "E::m "; g(); h();}
    C* n() {cout << "E::n "; return this;}
};
```

Le precedenti definizioni compilano correttamente. Per ognuno dei seguenti 14 statement in tabella con **numerazione da 01 a 14**, scrivere **chiaramente nel foglio 14 risposte con numerazione da 01 a 14** e per ciascuna risposta:

- **NON COMPILA** se la compilazione dell'istruzione provoca un errore;
- **UNDEFINED BEHAVIOUR** se l'istruzione compila correttamente ma la sua esecuzione provoca un undefined behaviour o un errore a run-time;
- se l'istruzione compila correttamente e non provoca errori a run-time allora si scriva **chiaramente** la stampa che l'esecuzione produce in output su `cout`; se non provoca alcuna stampa allora si scriva **NESSUNA STAMPA**.

```
01: p1->g();
02: (p1->n())->g();
03: p2->f();
04: p2->m();
05: (static_cast<D*>(p2))->k();
06: p3->k();
07: p3->f();
08: (p3->n())->m();
09: (p3->n())->n()->g();
10: (static_cast<C*>(p3->n()))->g();
11: (p4->n())->m();
12: p5->g();
13: p5->k();
14: (dynamic_cast<C*>(p5))->m();
```

// SOLUZIONE

```
01: B::g
02: NON COMPILA
03: B::f C::g B::h
04: B::m C::g B::h
05: UNDEFINED BEHAVIOUR
06: B::k D::h B::m B::g D::h
07: B::f B::g D::h
08: B::n B::m B::g D::h
09: B::n B::n B::g
10: B::n B::g
11: E::n B::m C::g E::h
12: C::g
13: C::k B::n
14: E::m C::g E::h
```

Esercizio Cosa Stampa

P3 → KC,

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

```
class D: public B {
protected:
    void h() {cout <<"D::h ";}
public:
    virtual void g() {cout <<"D::g ";}
    void k() const {cout <<"D::k "; k();}
    void m() {cout <<"D::m "; g(); h();}
};
```

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

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

```
class E: public C {
protected:
    void h() {cout <<"E::h ";}
public:
    void m() {cout <<"E::m "; g(); h();}
    C* n() {cout <<"E::n "; return this;}
};
```



Le precedenti definizioni compilano correttamente. Per ognuno dei seguenti 14 statement in tabella con **numerazione da 01 a 14**, scrivere **chiaramente nel foglio 14 risposte con numerazione da 01 a 14** e per ciascuna risposta:

- **NON COMPILA** se la compilazione dell'istruzione provoca un errore;
- **UNDEFINED BEHAVIOUR** se l'istruzione compila correttamente ma la sua esecuzione provoca un undefined behaviour o un errore a run-time;
- se l'istruzione compila correttamente e non provoca errori a run-time allora si scriva **chiaramente** la stampa che l'esecuzione produce in output su cout; se non provoca alcuna stampa allora si scriva **NESSUNA STAMPA**.

```
01: p1->g();
02: (p1->n())->g();
03: p2->f();
04: p2->m();
05: (static_cast<D*>(p2))->k();
06: p3->k();
07: p3->f();
08: (p3->n())->m();
09: (p3->n())->n()->g();
10: (static_cast<C*>(p3->n()))->g();
11: (p4->n())->m();
12: p5->g();
13: p5->k();
14: (dynamic_cast<C*>(p5))->m();
```

// SOLUZIONE

```
01: B::g
02: NON COMPILA
03: B::f C::g B::h
04: B::m C::g B::h
05: UNDEFINED BEHAVIOUR
06: B::k D::h B::m B::g D::h
07: B::f B::g D::h
08: B::n B::m B::g D::h
09: B::n B::n B::g
10: B::n B::g
11: E::n B::m C::g E::h
12: C::g
13: C::k B::n
14: E::m C::g E::h
```

Esercizio Cosa Stampa

p3 → f()

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

class D: public B {
protected:
    void h() {cout <<"D::h ";}
public:
    virtual void g() {cout <<"D::g ";}
    void k() const {cout <<"D::k "; k();}
    void m() {cout <<"D::m "; g(); h();}
};

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

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

class E: public C {
protected:
    void h() {cout <<"E::h ";}
public:
    void m() {cout <<"E::m "; g(); h();}
    C* n() {cout <<"E::n "; return this;}
};
```

Le precedenti definizioni compilano correttamente. Per ognuno dei seguenti 14 statement in tabella con **numerazione da 01 a 14**, scrivere **chiaramente nel foglio 14 risposte con numerazione da 01 a 14** e per ciascuna risposta:

- **NON COMPILA** se la compilazione dell'istruzione provoca un errore;
- **UNDEFINED BEHAVIOUR** se l'istruzione compila correttamente ma la sua esecuzione provoca un undefined behaviour o un errore a run-time;
- se l'istruzione compila correttamente e non provoca errori a run-time allora si scriva **chiaramente** la stampa che l'esecuzione produce in output su `cout`; se non provoca alcuna stampa allora si scriva **NESSUNA STAMPA**.

```
01: p1->g();
02: (p1->n())->g();
03: p2->f();
04: p2->m();
05: (static_cast<D*>(p2))->k();
06: p3->k();
07: p3->f();
08: (p3->n())->m();
09: (p3->n())->n()->g();
10: (static_cast<C*>(p3->n()))->g();
11: (p4->n())->m();
12: p5->g();
13: p5->k();
14: (dynamic_cast<C*>(p5))->m();
```

// SOLUZIONE

```
01: B::g
02: NON COMPILA
03: B::f C::g B::h
04: B::m C::g B::h
05: UNDEFINED BEHAVIOUR
06: B::k D::h B::m B::g D::h
07: B::f B::g D::h
08: B::n B::m B::g D::h
09: B::n B::n B::g
10: B::n B::g
11: E::n B::m C::g E::h
12: C::g
13: C::k B::n
14: E::m C::g E::h
```

$(P3 \rightarrow NC) \rightarrow NC$;

Esercizio Cosa Stampa

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

```
class D: public B {
protected:
    void h() {cout <<"D::h ";}
public:
    virtual void g() {cout <<"D::g ";}
    void k() const {cout <<"D::k "; k();}
    void m() {cout <<"D::m "; g(); h();}
};
```

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

```
class E: public C {
protected:
    void h() {cout <<"E::h ";}
public:
    void m() {cout <<"E::m "; g(); h();}
    C* n() {cout <<"E::n "; return this;}
};
```

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

Le precedenti definizioni compilano correttamente. Per ognuno dei seguenti 14 statement in tabella con **numerazione da 01 a 14**, scrivere **chiaramente nel foglio 14 risposte con numerazione da 01 a 14** e per ciascuna risposta:

- **NON COMPILA** se la compilazione dell'istruzione provoca un errore;
- **UNDEFINED BEHAVIOUR** se l'istruzione compila correttamente ma la sua esecuzione provoca un undefined behaviour o un errore a run-time;
- se l'istruzione compila correttamente e non provoca errori a run-time allora si scriva **chiaramente** la stampa che l'esecuzione produce in output su cout; se non provoca alcuna stampa allora si scriva **NESSUNA STAMPA**.

```
01: p1->g();
02: (p1->n())->g();
03: p2->f();
04: p2->m();
05: (static_cast<D*>(p2))->k();
06: p3->k();
07: p3->f();
08: (p3->n())->m();
09: (p3->n())->n()->g();
10: (static_cast<C*>(p3->n()))->g();
11: (p4->n())->m();
12: p5->g();
13: p5->k();
14: (dynamic_cast<C*>(p5))->m();
```

// SOLUZIONE

```
01: B::g
02: NON COMPILA
03: B::f C::g B::h
04: B::m C::g B::h
05: UNDEFINED BEHAVIOUR
06: B::k D::h B::m B::g D::h
07: B::f B::g D::h
08: B::n B::m B::g D::h
09: B::n B::n B::g
10: B::n B::g
11: E::n B::m C::g E::h
12: C::g
13: C::k B::n
14: E::m C::g E::h
```

Esercizio Cosa Stampa

$(P3 \rightarrow NC) \rightarrow NC \rightarrow GC$

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

class D: public B {
protected:
    void h() {cout <<"D::h ";}
public:
    virtual void g() {cout <<"D::g ";}
    void k() const {cout <<"D::k "; k();}
    void m() {cout <<"D::m "; g(); h();}
};

class C: public B {
public:
    virtual void g() const {cout <<"C::g ";}
    void k() {cout <<"C::k "; B::n();}
    virtual void m() {cout <<"C::m "; g(); h();}
    B* n() {cout <<"C::n "; return this;}
};

class E: public C {
protected:
    void h() {cout <<"E::h ";}
public:
    void m() {cout <<"E::m "; g(); h();}
    C* n() {cout <<"E::n "; return this;}
};

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

Le precedenti definizioni compilano correttamente. Per ognuno dei seguenti 14 statement in tabella con **numerazione da 01 a 14**, scrivere **chiaramente nel foglio 14 risposte con numerazione da 01 a 14** e per ciascuna risposta:

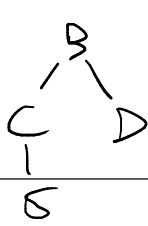
- **NON COMPILA** se la compilazione dell'istruzione provoca un errore;
- **UNDEFINED BEHAVIOUR** se l'istruzione compila correttamente ma la sua esecuzione provoca un undefined behaviour o un errore a run-time;
- se l'istruzione compila correttamente e non provoca errori a run-time allora si scriva **chiaramente** la stampa che l'esecuzione produce in output su `cout`; se non provoca alcuna stampa allora si scriva **NESSUNA STAMPA**.

```
01: p1->g();
02: (p1->n())->g();
03: p2->f();
04: p2->m();
05: (static_cast<D*>(p2))->k();
06: p3->k();
07: p3->f();
08: (p3->n())->m();
09: (p3->n())->n()->g();
10: (static_cast<C*>(p3->n()))->g();
11: (p4->n())->m();
12: p5->g();
13: p5->k();
14: (dynamic_cast<C*>(p5))->m();
```

// SOLUZIONE

```
01: B::g
02: NON COMPILA
03: B::f C::g B::h
04: B::m C::g B::h
05: UNDEFINED BEHAVIOUR
06: B::k D::h B::m B::g D::h
07: B::f B::g D::h
08: B::n B::m B::g D::h
09: B::n B::n B::g
10: B::n B::g
11: E::n B::m C::g E::h
12: C::g
13: C::k B::n
14: E::m C::g E::h
```


Esercizio Cosa Stampa



COMPILA SOLO
CON STATIC-CAST
[B::N / B::G]
(static_cast<C*>(p3->n()))->g();

```

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

```

class D: public B {
protected:
    void h() {cout <<"D::h ";}
public:
    virtual void g() {cout <<"D::g ";}
    void k() const {cout <<"D::k "; k();}
    void m() {cout <<"D::m "; g(); h();}
};
  
```

```

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

```

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

```

class E: public C {
protected:
    void h() {cout <<"E::h ";}
public:
    void m() {cout <<"E::m "; g(); h();}
    C* n() {cout <<"E::n "; return this;}
};
  
```

DYNAMIC-CAST (C*)

(p3->n())->g();
B::N / ERROR

Le precedenti definizioni compilano correttamente. Per ognuno dei seguenti 14 statement in tabella con **numerazione da 01 a 14**, scrivere **chiaramente nel foglio 14 risposte con numerazione da 01 a 14** e per ciascuna risposta:

- **NON COMPILA** se la compilazione dell'istruzione provoca un errore;
- **UNDEFINED BEHAVIOUR** se l'istruzione compila correttamente ma la sua esecuzione provoca un undefined behaviour o un errore a run-time;
- se l'istruzione compila correttamente e non provoca errori a run-time allora si scriva **chiaramente** la stampa che l'esecuzione produce in output su cout; se non provoca alcuna stampa allora si scriva **NESSUNA STAMPA**.

```

01: p1->g();
02: (p1->n())->g();
03: p2->f();
04: p2->m();
05: (static_cast<D*>(p2))->k();
06: p3->k();
07: p3->f();
08: (p3->n())->m();
09: (p3->n())->n()->g();
10: (static_cast<C*>(p3->n()))->g();
11: (p4->n())->m();
12: p5->g();
13: p5->k();
14: (dynamic_cast<C*>(p5))->m();
  
```

// SOLUZIONE

```

01: B::g
02: NON COMPILA
03: B::f C::g B::h
04: B::m C::g B::h
05: UNDEFINED BEHAVIOUR
06: B::k D::h B::m B::g D::h
07: B::f B::g D::h
08: B::n B::m B::g D::h
09: B::n B::n B::g
10: B::n B::g
11: E::n B::m C::g E::h
12: C::g
13: C::k B::n
14: E::m C::g E::h
  
```

Esercizio Cosa Stampa

$(p4 \rightarrow n()) \rightarrow m()$

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

```
class D: public B {
protected:
    void h() {cout <<"D::h ";}
public:
    virtual void g() {cout <<"D::g ";}
    void k() const {cout <<"D::k "; k();}
    void m() {cout <<"D::m "; g(); h();}
};
```

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

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

```
class E: public C {
protected:
    void h() {cout <<"E::h ";}
public:
    void m() {cout <<"E::m "; g(); h();}
    C* n() {cout <<"E::n "; return this;}
};
```

Le precedenti definizioni compilano correttamente. Per ognuno dei seguenti 14 statement in tabella con **numerazione da 01 a 14**, scrivere **chiaramente nel foglio 14 risposte con numerazione da 01 a 14** e per ciascuna risposta:

- **NON COMPILA** se la compilazione dell'istruzione provoca un errore;
- **UNDEFINED BEHAVIOUR** se l'istruzione compila correttamente ma la sua esecuzione provoca un undefined behaviour o un errore a run-time;
- se l'istruzione compila correttamente e non provoca errori a run-time allora si scriva **chiaramente** la stampa che l'esecuzione produce in output su `cout`; se non provoca alcuna stampa allora si scriva **NESSUNA STAMPA**.

```
01: p1->g();
02: (p1->n())->g();
03: p2->f();
04: p2->m();
05: (static_cast<D*>(p2))->k();
06: p3->k();
07: p3->f();
08: (p3->n())->m();
09: (p3->n())->n()->g();
10: (static_cast<C*>(p3->n()))->g();
11: (p4->n())->m();
12: p5->g();
13: p5->k();
14: (dynamic_cast<C*>(p5))->m();
```

// SOLUZIONE

```
01: B::g
02: NON COMPILA
03: B::f C::g B::h
04: B::m C::g B::h
05: UNDEFINED BEHAVIOUR
06: B::k D::h B::m B::g D::h
07: B::f B::g D::h
08: B::n B::m B::g D::h
09: B::n B::n B::g
10: B::n B::g
11: E::n B::m C::g E::h
12: C::g
13: C::k B::n
14: E::m C::g E::h
```

Esercizio Cosa Stampa

PS → G(),

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

```
class D: public B {
protected:
    void h() {cout <<"D::h ";}
public:
    virtual void g() {cout <<"D::g ";}
    void k() const {cout <<"D::k "; k();}
    void m() {cout <<"D::m "; g(); h();}
};
```

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

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

```
class E: public C {
protected:
    void h() {cout <<"E::h ";}
public:
    void m() {cout <<"E::m "; g(); h();}
    C* n() {cout <<"E::n "; return this;}
};
```

Le precedenti definizioni compilano correttamente. Per ognuno dei seguenti 14 statement in tabella con **numerazione da 01 a 14**, scrivere **chiaramente nel foglio 14 risposte con numerazione da 01 a 14** e per ciascuna risposta:

- **NON COMPILA** se la compilazione dell'istruzione provoca un errore;
- **UNDEFINED BEHAVIOUR** se l'istruzione compila correttamente ma la sua esecuzione provoca un undefined behaviour o un errore a run-time;
- se l'istruzione compila correttamente e non provoca errori a run-time allora si scriva **chiaramente** la stampa che l'esecuzione produce in output su `cout`; se non provoca alcuna stampa allora si scriva **NESSUNA STAMPA**.

```
01: p1->g();
02: (p1->n())->g();
03: p2->f();
04: p2->m();
05: (static_cast<D*>(p2))->k();
06: p3->k();
07: p3->f();
08: (p3->n())->m();
09: (p3->n())->n()->g();
10: (static_cast<C*>(p3->n()))->g();
11: (p4->n())->m();
12: p5->g();
13: p5->k();
14: (dynamic_cast<C*>(p5))->m();
```

// SOLUZIONE

```
01: B::g
02: NON COMPILA
03: B::f C::g B::h
04: B::m C::g B::h
05: UNDEFINED BEHAVIOUR
06: B::k D::h B::m B::g D::h
07: B::f B::g D::h
08: B::n B::m B::g D::h
09: B::n B::n B::g
10: B::n B::g
11: E::n B::m C::g E::h
12: C::g
13: C::k B::n
14: E::m C::g E::h
```

Esercizio Cosa Stampa

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

```
class D: public B {
protected:
    void h() {cout <<"D::h ";}
public:
    virtual void g() {cout <<"D::g ";}
    void k() const {cout <<"D::k "; k();}
    void m() {cout <<"D::m "; g(); h();}
};
```

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

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

```
class E: public C {
protected:
    void h() {cout <<"E::h ";}
public:
    void m() {cout <<"E::m "; g(); h();}
    C* n() {cout <<"E::n "; return this;}
};
```

Le precedenti definizioni compilano correttamente. Per ognuno dei seguenti 14 statement in tabella con **numerazione da 01 a 14**, scrivere **chiaramente nel foglio 14 risposte con numerazione da 01 a 14** e per ciascuna risposta:

- **NON COMPILA** se la compilazione dell'istruzione provoca un errore;
- **UNDEFINED BEHAVIOUR** se l'istruzione compila correttamente ma la sua esecuzione provoca un undefined behaviour o un errore a run-time;
- se l'istruzione compila correttamente e non provoca errori a run-time allora si scriva **chiaramente** la stampa che l'esecuzione produce in output su `cout`; se non provoca alcuna stampa allora si scriva **NESSUNA STAMPA**.

```
01: p1->g();
02: (p1->n())->g();
03: p2->f();
04: p2->m();
05: (static_cast<D*>(p2))->k();
06: p3->k();
07: p3->f();
08: (p3->n())->m();
09: (p3->n())->n()->g();
10: (static_cast<C*>(p3->n()))->g();
11: (p4->n())->m();
12: p5->g();
13: p5->k();
14: (dynamic_cast<C*>(p5))->m();
```

// SOLUZIONE

```
01: B::g
02: NON COMPILA
03: B::f C::g B::h
04: B::m C::g B::h
05: UNDEFINED BEHAVIOUR
06: B::k D::h B::m B::g D::h
07: B::f B::g D::h
08: B::n B::m B::g D::h
09: B::n B::n B::g
10: B::n B::g
11: E::n B::m C::g E::h
12: C::g
13: C::k B::n
14: E::m C::g E::h
```

(dynamic_cast<C*>(p5))->m();

Esercizio Cosa Stampa

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

class D: public B {
protected:
    void h() {cout <<"D::h ";}
public:
    virtual void g() {cout <<"D::g ";}
    void k() const {cout <<"D::k "; k();}
    void m() {cout <<"D::m "; g(); h();}
};

class C: public B {
public:
    virtual void g() const {cout <<"C::g ";}
    void k() {cout <<"C::k "; B::n();}
    virtual void m() {cout <<"C::m "; g(); h();}
    B* n() {cout <<"C::n "; return this;}
};

class E: public C {
protected:
    void h() {cout <<"E::h ";}
public:
    void m() {cout <<"E::m "; g(); h();}
    C* n() {cout <<"E::n "; return this;}
};

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

Le precedenti definizioni compilano correttamente. Per ognuno dei seguenti 14 statement in tabella con **numerazione da 01 a 14**, scrivere **chiaramente nel foglio 14 risposte con numerazione da 01 a 14** e per ciascuna risposta:

- **NON COMPILA** se la compilazione dell'istruzione provoca un errore;
- **UNDEFINED BEHAVIOUR** se l'istruzione compila correttamente ma la sua esecuzione provoca un undefined behaviour o un errore a run-time;
- se l'istruzione compila correttamente e non provoca errori a run-time allora si scriva **chiaramente** la stampa che l'esecuzione produce in output su cout; se non provoca alcuna stampa allora si scriva **NESSUNA STAMPA**.

```
01: p1->g();
02: (p1->n())->g();
03: p2->f();
04: p2->m();
05: (static_cast<D*>(p2))->k();
06: p3->k();
07: p3->f();
08: (p3->n())->m();
09: (p3->n())->n()->g();
10: (static_cast<C*>(p3->n()))->g();
11: (p4->n())->m();
12: p5->g();
13: p5->k();
14: (dynamic_cast<C*>(p5))->m();
```

// SOLUZIONE

```
01: B::g
02: NON COMPILA
03: B::f C::g B::h
04: B::m C::g B::h
05: UNDEFINED BEHAVIOUR
06: B::k D::h B::m B::g D::h
07: B::f B::g D::h
08: B::n B::m B::g D::h
09: B::n B::n B::g
10: B::n B::g
11: E::n B::m C::g E::h
12: C::g
13: C::k B::n
14: E::m C::g E::h
```