

Esercizio 3

```

class Z {
public: Z(int x) {}
};

class A {
public:
void f(int){cout << "A::f(int) ";}
virtual void f(bool){cout <<"A::f(bool) ";}
virtual void f(Z){cout <<"A::f(Z) ";}
};

class B: virtual public A {
public:
void f(const bool&){cout<< "B::f(const bool&) ";}
void f(const int&){cout<< "B::f(const int&) ";}
};

class C: virtual public A {
public:
virtual void f(Z){cout <<"C::f(Z) ";}
};

class D: public B, public C {
public:
virtual void f(int*){cout<< "D::f(int*) ";}
void f(int&){cout <<"D::f(int&) ";}
};

class E: public D {
public:
void f(Z){cout <<"E::f(Z) ";}
};

B* pb=new B; C* pc = new C; D* pd = new D; E* pe = new E; A *pa1=pb, *pa2=pc, *pa3=pd, *pa4=pe; C *pc1=pe;

```

↑ 1 → int const = 1

(dynamic_cast<B*>(pa1))→f(1);

B::f(const int &);

Le precedenti definizioni compilano correttamente. Per ognuna delle seguenti istruzioni scrivere nell'apposito spazio:

- **NON COMPILA** se la compilazione dell'istruzione provoca un errore;
- **ERRORE RUN-TIME** se l'istruzione compila correttamente ma la sua esecuzione provoca un errore a run-time;
- se l'istruzione compila correttamente e non provoca errori a run-time allora si scriva la stampa che l'esecuzione produce in output su `std::cout`; se non provoca alcuna stampa allora si scriva **NESSUNA STAMPA**.

(dynamic_cast<B*>(pa1))→f(1);
(dynamic_cast<B*>(pa1))→f(true);
pa1→f(true);
pa2→f(1);
(dynamic_cast<C*>(pa2))→f(1);
(dynamic_cast<E*>(pa2))→f(1);
(dynamic_cast<C*>(pa3))→f(0);
(dynamic_cast<D*>(pa3))→f(0);
pa4→f(1);
(dynamic_cast<C*>(pa4))→f(1);
pc1→f(1);
(static_cast<E*>(pc1))→f(1);
(static_cast<A*>(pc1))→f(1);

Esercizio 3

```
class Z {
public: Z(int x) {}
};

class A {
public:
    void f(int){cout << "A::f(int) ";}
    virtual void f(bool){cout <<"A::f(bool) ";}
    virtual void f(Z){cout <<"A::f(Z) ";}
};

class B: virtual public A {
public:
    void f(const bool&){cout<< "B::f(const bool&) ";}
    void f(const int&){cout<< "B::f(const int&) ";}
};

class C: virtual public A {
public:
    virtual void f(Z){cout <<"C::f(Z) ";}
};

class D: public B, public C {
public:
    virtual void f(int*){cout<< "D::f(int*) ";}
    void f(int&){cout <<"D::f(int&) ";}
};

class E: public D {
public:
    void f(Z){cout <<"E::f(Z) ";}
};

B* pb=new B; C* pc = new C; D* pd = new D; E* pe = new E; A *pa1=pb, *pa2=pc, *pa3=pd, *pa4=pe; C *pc1=pe;
```

STAMPA

(dynamic_cast<B*>(pa1))→f(true);

Le precedenti definizioni compilano correttamente. Per ognuna delle seguenti istruzioni scrivere nell'apposito spazio:

- **NON COMPILA** se la compilazione dell'istruzione provoca un errore;
- **ERRORE RUN-TIME** se l'istruzione compila correttamente ma la sua esecuzione provoca un errore a run-time;
- se l'istruzione compila correttamente e non provoca errori a run-time allora si scriva la stampa che l'esecuzione produce in output su `std::cout`; se non provoca alcuna stampa allora si scriva **NESSUNA STAMPA**.

<code>(dynamic_cast<B*>(pa1))→f(1);</code>
<code>(dynamic_cast<B*>(pa1))→f(true);</code>
<code>pa1→f(true);</code>
<code>pa2→f(1);</code>
<code>(dynamic_cast<C*>(pa2))→f(1);</code>
<code>(dynamic_cast<E*>(pa2))→f(1);</code>
<code>(dynamic_cast<C*>(pa3))→f(0);</code>
<code>(dynamic_cast<D*>(pa3))→f(0);</code>
<code>pa4→f(1);</code>
<code>(dynamic_cast<C*>(pa4))→f(1);</code>
<code>pc1→f(1);</code>
<code>(static_cast<E*>(pc1))→f(1);</code>
<code>(static_cast<A*>(pc1))→f(1);</code>

Esercizio 3

```

class Z {
public: Z(int x) {}
};

class A {
public:
    void f(int){cout << "A::f(int) ";}
    virtual void f(bool){cout <<"A::f(bool) ";}
    virtual void f(Z){cout <<"A::f(Z) ";}
};

class B: virtual public A {
public:
    void f(const bool&){cout<< "B::f(const bool&) ";}
    void f(const int&){cout<< "B::f(const int&) ";}
};

class C: virtual public A {
public:
    virtual void f(Z){cout <<"C::f(Z) ";}
};

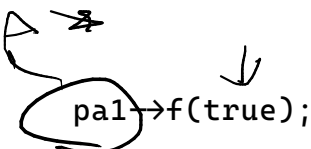
class D: public B, public C {
public:
    virtual void f(int*){cout<< "D::f(int*) ";}
    void f(int&){cout <<"D::f(int&) ";}
};

class E: public D {
public:
    void f(Z){cout <<"E::f(Z) ";}
};

```

F
 $f(\text{bool}) \neq \text{CONST bool}$
 NO DYN-CAST

→ RUNTIME IN A C?

CONST bool


B* pb=new B; C* pc = new C; D* pd = new D; E* pe = new E; A *pa1=pb, *pa2=pc, *pa3=pd, *pa4=pe; C *pc1=pe;

Le precedenti definizioni compilano correttamente. Per ognuna delle seguenti istruzioni scrivere nell'apposito spazio:

- **NON COMPILA** se la compilazione dell'istruzione provoca un errore;
- **ERRORE RUN-TIME** se l'istruzione compila correttamente ma la sua esecuzione provoca un errore a run-time;
- se l'istruzione compila correttamente e non provoca errori a run-time allora si scriva la stampa che l'esecuzione produce in output su `std::cout`; se non provoca alcuna stampa allora si scriva **NESSUNA STAMPA**.

<code>(dynamic_cast<B*>(pa1))->f(1);</code>
<code>(dynamic_cast<B*>(pa1))->f(true);</code>
<code>pa1->f(true);</code>
<code>pa2->f(1);</code>
<code>(dynamic_cast<C*>(pa2))->f(1);</code>
<code>(dynamic_cast<E*>(pa2))->f(1);</code>
<code>(dynamic_cast<C*>(pa3))->f(0);</code>
<code>(dynamic_cast<D*>(pa3))->f(0);</code>
<code>pa4->f(1);</code>
<code>(dynamic_cast<C*>(pa4))->f(1);</code>
<code>pc1->f(1);</code>
<code>(static_cast<E*>(pc1))->f(1);</code>
<code>(static_cast<A*>(pc1))->f(1);</code>

Esercizio 3

```
class Z {
public: Z(int x) {}
};

class A {
public:
    void f(int){cout << "A::f(int) ";}
    virtual void f(bool){cout <<"A::f(bool) ";}
    virtual void f(Z){cout <<"A::f(Z) ";}
};

class B: virtual public A {
public:
    void f(const bool&){cout<< "B::f(const bool&) ";}
    void f(const int&){cout<< "B::f(const int&) ";}
};

class C: virtual public A {
public:
    virtual void f(Z){cout <<"C::f(Z) ";}
};

class D: public B, public C {
public:
    virtual void f(int*){cout<< "D::f(int*) ";}
    void f(int&){cout <<"D::f(int&) ";}
};

class E: public D {
public:
    void f(Z){cout <<"E::f(Z) ";}
};

B* pb=new B; C* pc = new C; D* pd = new D; E* pe = new E; A *pa1=pb, *pa2=pc, *pa3=pd, *pa4=pe; C *pc1=pe;
```

pa2→f(1);

Le precedenti definizioni compilano correttamente. Per ognuna delle seguenti istruzioni scrivere nell'apposito spazio:

- **NON COMPILA** se la compilazione dell'istruzione provoca un errore;
- **ERRORE RUN-TIME** se l'istruzione compila correttamente ma la sua esecuzione provoca un errore a run-time;
- se l'istruzione compila correttamente e non provoca errori a run-time allora si scriva la stampa che l'esecuzione produce in output su `std::cout`; se non provoca alcuna stampa allora si scriva **NESSUNA STAMPA**.

<code>(dynamic_cast<B*>(pa1))->f(1);</code>
<code>(dynamic_cast<B*>(pa1))->f(true);</code>
<code>pa1->f(true);</code>
<code>pa2->f(1);</code>
<code>(dynamic_cast<C*>(pa2))->f(1);</code>
<code>(dynamic_cast<E*>(pa2))->f(1);</code>
<code>(dynamic_cast<C*>(pa3))->f(0);</code>
<code>(dynamic_cast<D*>(pa3))->f(0);</code>
<code>pa4->f(1);</code>
<code>(dynamic_cast<C*>(pa4))->f(1);</code>
<code>pc1->f(1);</code>
<code>(static_cast<E*>(pc1))->f(1);</code>
<code>(static_cast<A*>(pc1))->f(1);</code>

Esercizio 3

```

class Z {
public: Z(int x) {}
};

class A {
public:
    void f(int){cout << "A::f(int) ";}
    virtual void f(bool){cout <<"A::f(bool) ";}
    virtual void f(Z){cout <<"A::f(Z) ";}
};

class B: virtual public A {
public:
    void f(const bool&){cout<< "B::f(const bool&) ";}
    void f(const int&){cout<< "B::f(const int&) ";}
};

class C: virtual public A {
public:
    virtual void f(Z){cout <<"C::f(Z) ";}
};

class D: public B, public C {
public:
    virtual void f(int*){cout<< "D::f(int*) ";}
    void f(int&){cout <<"D::f(int&) ";}
};

class E: public D {
public:
    void f(Z){cout <<"E::f(Z) ";}
};

```

```

B* pb=new B; C* pc = new C; D* pd = new D; E* pe = new E; A *pa1=pb, *pa2=pc, *pa3=pd, *pa4=pe; C *pc1=pe;

```

ERT
↓
(dynamic_cast<E*>(pa2))→f(1);

≠ (Z(1)).

A & PA 2 = PC → E & PA 2.

Le precedenti definizioni compilano correttamente. Per ognuna delle seguenti istruzioni scrivere nell'apposito spazio:

- **NON COMPILA** se la compilazione dell'istruzione provoca un errore;
- **ERRORE RUN-TIME** se l'istruzione compila correttamente ma la sua esecuzione provoca un errore a run-time;
- se l'istruzione compila correttamente e non provoca errori a run-time allora si scriva la stampa che l'esecuzione produce in output su `std::cout`; se non provoca alcuna stampa allora si scriva **NESSUNA STAMPA**.

(dynamic_cast<B*>(pa1))→f(1);
(dynamic_cast<B*>(pa1))→f(true);
pa1→f(true);
pa2→f(1);
(dynamic_cast<C*>(pa2))→f(1);
(dynamic_cast<E*>(pa2))→f(1);
(dynamic_cast<C*>(pa3))→f(0);
(dynamic_cast<D*>(pa3))→f(0);
pa4→f(1);
(dynamic_cast<C*>(pa4))→f(1);
pc1→f(1);
(static_cast<E*>(pc1))→f(1);
(static_cast<A*>(pc1))→f(1);

Esercizio 3

```

class Z {
public: Z(int x) {}
};

class A {
public:
    void f(int){cout << "A::f(int) ";}
    virtual void f(bool){cout <<"A::f(bool) ";}
    virtual void f(Z){cout <<"A::f(Z) ";}
};

class B: virtual public A {
public:
    void f(const bool&){cout<< "B::f(const bool&) ";}
    void f(const int&){cout<< "B::f(const int&) ";}
};

class C: virtual public A {
public:
    virtual void f(Z){cout <<"C::f(Z) ";}
};

class D: public B, public C {
public:
    virtual void f(int*){cout<< "D::f(int*) ";}
    void f(int&){cout <<"D::f(int&) ";}
};

class E: public D {
public:
    void f(Z){cout <<"E::f(Z) ";}
};

B* pb=new B; C* pc = new C; D* pd = new D; E* pe = new E; A *pa1=pb, *pa2=pc, *pa3=pd, *pa4=pe; C *pc1=pe;

```

① → const int
 ② → null ptr

(dynamic_cast<C*>(pa3))→f(0);

Le precedenti definizioni compilano correttamente. Per ognuna delle seguenti istruzioni scrivere nell'apposito spazio:

- **NON COMPILA** se la compilazione dell'istruzione provoca un errore;
- **ERRORE RUN-TIME** se l'istruzione compila correttamente ma la sua esecuzione provoca un errore a run-time;
- se l'istruzione compila correttamente e non provoca errori a run-time allora si scriva la stampa che l'esecuzione produce in output su `std::cout`; se non provoca alcuna stampa allora si scriva **NESSUNA STAMPA**.

(dynamic_cast<B*>(pa1))→f(1);
(dynamic_cast<B*>(pa1))→f(true);
pa1→f(true);
pa2→f(1);
(dynamic_cast<C*>(pa2))→f(1);
(dynamic_cast<E*>(pa2))→f(1);
(dynamic_cast<C*>(pa3))→f(0);
(dynamic_cast<D*>(pa3))→f(0);
pa4→f(1);
(dynamic_cast<C*>(pa4))→f(1);
pc1→f(1);
(static_cast<E*>(pc1))→f(1);
(static_cast<A*>(pc1))→f(1);

Esercizio 3

```
class Z {
public: Z(int x) {}
};

class A {
public:
    void f(int){cout << "A::f(int) ";}
    virtual void f(bool){cout <<"A::f(bool) ";}
    virtual void f(Z){cout <<"A::f(Z) ";}
};

class B: virtual public A {
public:
    void f(const bool&){cout<< "B::f(const bool&) ";}
    void f(const int&){cout<< "B::f(const int&) ";}
};

class C: virtual public A {
public:
    virtual void f(Z){cout <<"C::f(Z) ";}
};

class D: public B, public C {
public:
    virtual void f(int*){cout<< "D::f(int*) ";}
    void f(int&){cout <<"D::f(int&) ";}
};

class E: public D {
public:
    void f(Z){cout <<"E::f(Z) ";}
};

B* pb=new B; C* pc = new C; D* pd = new D; E* pe = new E; A *pa1=pb, *pa2=pc, *pa3=pd, *pa4=pe; C *pc1=pe;
```

Handwritten notes:

- $\text{D} = \text{INT} \rightarrow$
- $(\text{dynamic_cast}\langle \text{D}^* \rangle (\text{pa3})) \rightarrow \text{f}(0);$
- $\text{D} = \text{f}(\text{INT}) \rightarrow$

Le precedenti definizioni compilano correttamente. Per ognuna delle seguenti istruzioni scrivere nell'apposito spazio:

- **NON COMPILA** se la compilazione dell'istruzione provoca un errore;
- **ERRORE RUN-TIME** se l'istruzione compila correttamente ma la sua esecuzione provoca un errore a run-time;
- se l'istruzione compila correttamente e non provoca errori a run-time allora si scriva la stampa che l'esecuzione produce in output su `std::cout`; se non provoca alcuna stampa allora si scriva **NESSUNA STAMPA**.

<code>(dynamic_cast<B*>(pa1))->f(1);</code>
<code>(dynamic_cast<B*>(pa1))->f(true);</code>
<code>pa1->f(true);</code>
<code>pa2->f(1);</code>
<code>(dynamic_cast<C*>(pa2))->f(1);</code>
<code>(dynamic_cast<E*>(pa2))->f(1);</code>
<code>(dynamic_cast<C*>(pa3))->f(0);</code>
<code>(dynamic_cast<D*>(pa3))->f(0);</code>
<code>pa4->f(1);</code>
<code>(dynamic_cast<C*>(pa4))->f(1);</code>
<code>pc1->f(1);</code>
<code>(static_cast<E*>(pc1))->f(1);</code>
<code>(static_cast<A*>(pc1))->f(1);</code>

Esercizio 3

```
class Z {
public: Z(int x) {}
};
```

```
class A {
public:
void f(int){cout << "A::f(int) ";}
virtual void f(bool){cout << "A::f(bool) ";}
virtual void f(Z){cout << "A::f(Z) ";}
};
```

```
class B: virtual public A {
public:
void f(const bool&){cout<< "B::f(const bool&) ";}
void f(const int&){cout<< "B::f(const int&) ";}
};
```

```
class C: virtual public A {
public:
virtual void f(Z){cout << "C::f(Z) ";}
};
```

```
class D: public B, public C {
public:
virtual void f(int*){cout<< "D::f(int*) ";}
void f(int&){cout << "D::f(int&) ";}
};
```

```
class E: public D {
public:
void f(Z){cout << "E::f(Z) ";}
};
```

```
B* pb=new B; C* pc = new C; D* pd = new D; E* pe = new E; A *pa1=pb, *pa2=pc, *pa3=pd, *pa4=pe; C *pc1=pe;
```

STATIC_CAST(A*) (PC1) → f(1)

STATIC (Y)

DYNAMIC_CAST(B*) (PA2) → f(1)

(static_cast<E*>(pc1))→f(1); →

B::f(1)

C * PC1 = B;



[B * PC1 = B;]

Le precedenti definizioni compilano correttamente. Per ognuna delle seguenti istruzioni scrivere nell'apposito spazio:

- **NON COMPILA** se la compilazione dell'istruzione provoca un errore;
- **ERRORE RUN-TIME** se l'istruzione compila correttamente ma la sua esecuzione provoca un errore a run-time;
- se l'istruzione compila correttamente e non provoca errori a run-time allora si scriva la stampa che l'esecuzione produce in output su `std::cout`; se non provoca alcuna stampa allora si scriva **NESSUNA STAMPA**.

<code>(dynamic_cast<B*>(pa1))->f(1);</code>
<code>(dynamic_cast<B*>(pa1))->f(true);</code>
<code>pa1->f(true);</code>
<code>pa2->f(1);</code>
<code>(dynamic_cast<C*>(pa2))->f(1);</code>
<code>(dynamic_cast<E*>(pa2))->f(1);</code>
<code>(dynamic_cast<C*>(pa3))->f(0);</code>
<code>(dynamic_cast<D*>(pa3))->f(0);</code>
<code>pa4->f(1);</code>
<code>(dynamic_cast<C*>(pa4))->f(1);</code>
<code>pc1->f(1);</code>
<code>(static_cast<E*>(pc1))->f(1);</code>
<code>(static_cast<A*>(pc1))->f(1);</code>