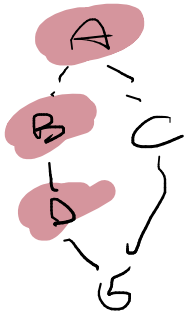
Quesito 2

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

```
class A {
 public:
  void f(int) {cout << "A::f(int) "; f(true);}
  virtual void f(bool) {cout <<"A::f(bool) ";}
  virtual A* f(Z) {cout <<"A::f(Z) "; f(2); return this;}
  A() {cout <<"A() "; }
};
```

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

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




```
class D: public B {
public:
  virtual void f(bool) const {cout <<"D::f(bool) ";}
  B* f(Z) {cout << "D::f(Z) "; return this;}
  ~D() {cout <<"~D ";}
  D() {cout <<"D() "; }
};
```

```
class E: public D, public C {
 public:
  void f(bool){cout<< "E::f(bool) ";}
  E* f(Z){cout <<"E::f(Z) "; return this;}
  E() {cout <<"E() "; }
  ~E() {cout <<"~E ";}
};
```

```
E* puntE = new E;
```

A B C D E

```
D* puntD = new D;
```

A B D

```
C* puntC = new C;
```

⟶ Stampava SOLO A() C()

```
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; B *pb1=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**.

```
E* puntE = new E;       .....
D* puntD = new D;       .....
pa3->f(3);              .....
pa4->f(3);              .....
pb1->f(true);           .....
pa4->f(true);           .....
pa2->f(Z(2));           .....
pa4->f(Z(2));           .....
pb->f(3);               .....
pc->f(3);               .....
(pa4->f(Z(3)))->f(4);   .....
(pc->f(Z(3)))->f(4);    .....
delete pa4;             .....
delete pd;              .....
```

Quesito 2

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

class A {
 public:
  void f(int) {cout << "A::f(int) "; f(true);}
  virtual void f(bool) {cout <<"A::f(bool) ";}
  virtual A* f(Z) {cout <<"A::f(Z) "; f(2); return this;}
  A() {cout <<"A() "; }
};

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

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

class D: public B {
public:
  virtual void f(bool) const {cout <<"D::f(bool) ";}
  B* f(Z) {cout << "D::f(Z) "; return this;}
  ~D() {cout <<"~D ";}
  D() {cout <<"D() "; }
};

class E: public D, public C {
 public:
  void f(bool){cout<< "E::f(bool) ";}
  E* f(Z){cout <<"E::f(Z) "; return this;}
  E() {cout <<"E() "; }
  ~E() {cout <<"~E ";}
};

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; B *pb1=pe;
```

pa3→f(3);

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**.

```
E* puntE = new E;       .....

D* puntD = new D;       .....

pa3->f(3);              .....

pa4->f(3);              .....

pb1->f(true);           .....

pa4->f(true);           .....

pa2->f(Z(2));           .....

pa4->f(Z(2));           .....

pb->f(3);               .....

pc->f(3);               .....

(pa4->f(Z(3)))->f(4);   .....

(pc->f(Z(3)))->f(4);    .....

delete pa4;             .....

delete pd;              .....
```

Quesito 2

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

class A {
 public:
  void f(int) {cout << "A::f(int) "; f(true);}
  virtual void f(bool) {cout << "A::f(bool) ";}
  virtual A* f(Z) {cout <<"A::f(Z) "; f(2); return this;}
  A() {cout <<"A() "; }
};

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

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

class D: public B {
public:
  virtual void f(bool) const {cout <<"D::f(bool) ";}
  B* f(Z) {cout << "D::f(Z) "; return this;}
  ~D() {cout <<"~D ";}
  D() {cout <<"D() "; }
};

class E: public D, public C {
 public:
  void f(bool){cout<< "E::f(bool) ";}
  E* f(Z){cout <<"E::f(Z) "; return this;}
  E() {cout <<"E() "; }
  ~E() {cout <<"~E ";}
};

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; B *pb1=pe;
```

pa4→f(3)

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**.

```
E* puntE = new E;       .....

D* puntD = new D;       .....

pa3->f(3);              .....

pa4->f(3);              .....

pb1->f(true);           .....

pa4->f(true);           .....

pa2->f(Z(2));           .....

pa4->f(Z(2));           .....

pb->f(3);               .....

pc->f(3);               .....

(pa4->f(Z(3)))->f(4);   .....

(pc->f(Z(3)))->f(4);    .....

delete pa4;             .....

delete pd;              .....
```

Quesito 2

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

class A {
 public:
  void f(int) {cout << "A::f(int) "; f(true);}
  virtual void f(bool) {cout <<"A::f(bool) ";}
  virtual A* f(Z) {cout <<"A::f(Z) "; f(2); return this;}
  A() {cout <<"A() "; }
};

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

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

class D: public B {
public:
  virtual void f(bool) const {cout <<"D::f(bool) ";}
  B* f(Z) {cout << "D::f(Z) "; return this;}
  ~D() {cout <<"~D ";}
  D() {cout <<"D() "; }
};

class E: public D, public C {
 public:
  void f(bool){cout<< "E::f(bool) ";}
  E* f(Z){cout <<"E::f(Z) "; return this;}
  E() {cout <<"E() "; }
  ~E() {cout <<"~E ";}
};

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; B *pb1=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**.

```
E* puntE = new E;       .....

D* puntD = new D;       .....

pa3->f(3);              .....

pa4->f(3);              .....

pb1->f(true);           .....

pa4->f(true);           .....

pa2->f(Z(2));           .....

pa4->f(Z(2));           .....

pb->f(3);               .....

pc->f(3);               .....

(pa4->f(Z(3)))->f(4);   .....

(pc->f(Z(3)))->f(4);    .....

delete pa4;             .....

delete pd;              .....
```

Quesito 2

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

```
class A {
 public:
  void f(int) {cout << "A::f(int) "; f(true);}
  virtual void f(bool) {cout <<"A::f(bool) ";}
  virtual A* f(Z) {cout <<"A::f(Z) "; f(2); return this;}
  A() {cout <<"A() "; }
};
```

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

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

```
class D: public B {
public:
  virtual void f(bool) const {cout <<"D::f(bool) ";}
  B* f(Z) {cout << "D::f(Z) "; return this;}
  ~D() {cout <<"~D ";}
  D() {cout <<"D() "; }
};
```

```
class E: public D, public C {
 public:
  void f(bool){cout<< "E::f(bool) ";}
  E* f(Z){cout <<"E::f(Z) "; return this;}
  E() {cout <<"E() "; }
  ~E() {cout <<"~E ";}
};
```

```
pa4→f(true);
```


```
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; B *pb1=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**.

```
E* puntE = new E;       .....

D* puntD = new D;       .....

pa3->f(3);              .....

pa4->f(3);              .....

pb1->f(true);           .....

pa4->f(true);           .....

pa2->f(Z(2));           .....

pa4->f(Z(2));           .....

pb->f(3);               .....

pc->f(3);               .....

(pa4->f(Z(3)))->f(4);   .....

(pc->f(Z(3)))->f(4);    .....

delete pa4;             .....

delete pd;              .....
```

Quesito 2

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

```
class A {
 public:
  void f(int) {cout << "A::f(int) "; f(true);}
  virtual void f(bool) {cout <<"A::f(bool) ";}
  virtual A* f(Z) {cout <<"A::f(Z) "; f(2); return this;}
  A() {cout <<"A() "; }
};
```

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

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

```
class D: public B {
public:
  virtual void f(bool) const {cout <<"D::f(bool) ";}
  B* f(Z) {cout << "D::f(Z) "; return this;}
  ~D() {cout <<"~D ";}
  D() {cout <<"D() "; }
};
```

```
class E: public D, public C {
 public:
  void f(bool){cout<< "E::f(bool) ";}
  E* f(Z){cout <<"E::f(Z) "; return this;}
  E() {cout <<"E() "; }
  ~E() {cout <<"~E ";}
};
```

```
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; B *pb1=pe;
```

pa2→f(Z(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**.

```
E* puntE = new E;       .....

D* puntD = new D;       .....

pa3->f(3);              .....

pa4->f(3);              .....

pb1->f(true);           .....

pa4->f(true);           .....

pa2->f(Z(2));           .....

pa4->f(Z(2));           .....

pb->f(3);               .....

pc->f(3);               .....

(pa4->f(Z(3)))->f(4);   .....

(pc->f(Z(3)))->f(4);    .....

delete pa4;             .....

delete pd;              .....
```

Quesito 2

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

```
class A {
 public:
  void f(int) {cout << "A::f(int) "; f(true);}
  virtual void f(bool) {cout <<"A::f(bool) ";}
  virtual A* f(Z) {cout <<"A::f(Z) "; f(2); return this;}
  A() {cout <<"A() "; }
};
```

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

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

```
class D: public B {
public:
  virtual void f(bool) const {cout <<"D::f(bool) ";}
  B* f(Z) {cout << "D::f(Z) "; return this;}
  ~D() {cout <<"~D ";}
  D() {cout <<"D() "; }
};
```

```
class E: public D, public C {
 public:
  void f(bool){cout<< "E::f(bool) ";}
  E* f(Z){cout <<"E::f(Z) "; return this;}
  E() {cout <<"E() "; }
  ~E() {cout <<"~E ";}
};
```

pa4→f(Z(2));


```
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; B *pb1=pe;
```

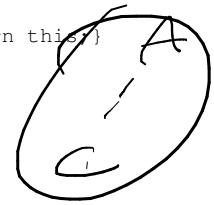
Le precedenti definizioni compilano correttamente. Per ognuna delle seguenti istruzioni scrivere nell'apposito spazio:

- **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**.

```
E* puntE = new E;       .....

D* puntD = new D;       .....

pa3->f(3);              .....

pa4->f(3);              .....

pb1->f(true);           .....

pa4->f(true);           .....

pa2->f(Z(2));           .....

pa4->f(Z(2));           .....

pb->f(3);               .....

pc->f(3);               .....

(pa4->f(Z(3)))->f(4);   .....

(pc->f(Z(3)))->f(4);    .....

delete pa4;             .....

delete pd;              .....
```

Quesito 2

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

class A {
 public:
  void f(int) {cout << "A::f(int) "; f(true);}
  virtual void f(bool) {cout <<"A::f(bool) ";}
  virtual A* f(Z) {cout <<"A::f(Z) "; f(2); return this;}
  A() {cout <<"A() "; }
};

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

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

class D: public B {
public:
  virtual void f(bool) const {cout <<"D::f(bool) ";}
  B* f(Z) {cout << "D::f(Z) "; return this;}
  ~D() {cout <<"~D ";}
  D() {cout <<"D() "; }
};

class E: public D, public C {
 public:
  void f(bool){cout<< "E::f(bool) ";}
  E* f(Z){cout <<"E::f(Z) "; return this;}
  E() {cout <<"E() "; }
  ~E() {cout <<"~E ";}
};

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; B *pb1=pe;
```




```
pb→f(3);

pc→f(3);
```

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**.

```
E* puntE = new E;       .....

D* puntD = new D;       .....

pa3->f(3);              .....

pa4->f(3);              .....

pb1->f(true);           .....

pa4->f(true);           .....

pa2->f(Z(2));           .....

pa4->f(Z(2));           .....

pb->f(3);               .....

pc->f(3);               .....

(pa4->f(Z(3)))->f(4);   .....

(pc->f(Z(3)))->f(4);    .....

delete pa4;             .....

delete pd;              .....
```

Quesito 2

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

class A {
 public:
  void f(int) {cout << "A::f(int) "; f(true);}
  virtual void f(bool) {cout <<"A::f(bool) ";}
  virtual A* f(Z) {cout <<"A::f(Z) "; f(2); return this;}
  A() {cout <<"A() "; }
};

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

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

class D: public B {
public:
  virtual void f(bool) const {cout <<"D::f(bool) ";}
  B* f(Z) {cout << "D::f(Z) "; return this;}
  ~D() {cout <<"~D ";}
  D() {cout <<"D() "; }
};

class E: public D, public C {
 public:
  void f(bool){cout<< "E::f(bool) ";}
  E* f(Z){cout <<"E::f(Z) "; return this;}
  E() {cout <<"E() "; }
  ~E() {cout <<"~E ";}
};

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; B *pb1=pe;
```

(pa4→f(Z(3)))→f(4);

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**.

```
E* puntE = new E;       .....

D* puntD = new D;       .....

pa3->f(3);              .....

pa4->f(3);              .....

pb1->f(true);           .....

pa4->f(true);           .....

pa2->f(Z(2));           .....

pa4->f(Z(2));           .....

pb->f(3);               .....

pc->f(3);               .....

(pa4->f(Z(3)))->f(4);   .....

(pc->f(Z(3)))->f(4);    .....

delete pa4;             .....

delete pd;              .....
```

Quesito 2

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


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

class D: public B {
public:
  virtual void f(bool) const {cout <<"D::f(bool) ";}
  B* f(Z) {cout << "D::f(Z) "; return this;}
  ~D() {cout <<"~D ";}
  D() {cout <<"D() "; }
};

class E: public D, public C {
 public:
  void f(bool){cout<< "E::f(bool) ";}
  E* f(Z){cout <<"E::f(Z) "; return this;}
  E() {cout <<"E() "; }
  ~E() {cout <<"~E ";}
};
```

```
class A {
 public:
  void f(int) {cout << "A::f(int) "; f(true);}
  virtual void f(bool) {cout <<"A::f(bool) ";}
  virtual A* f(Z) {cout <<"A::f(Z) "; f(2); return this;}
  A() {cout <<"A() "; }
};

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

NON TRACCIATO

delete pa4; → NESSUNA STAMPA

delete pd; → ~D

BASTA CANCELLARE TS = NO STATICO

```
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; B *pb1=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**.

```
E* puntE = new E;       .....

D* puntD = new D;       .....

pa3->f(3);              .....

pa4->f(3);              .....

pb1->f(true);           .....

pa4->f(true);           .....

pa2->f(Z(2));           .....

pa4->f(Z(2));           .....

pb->f(3);               .....

pc->f(3);               .....

(pa4->f(Z(3)))->f(4);   .....

(pc->f(Z(3)))->f(4);    .....

delete pa4;             .....

delete pd;              .....
```