

Esercizio 2

```
class A {
    bool x;
public:
    virtual ~A() = default;
};
```

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

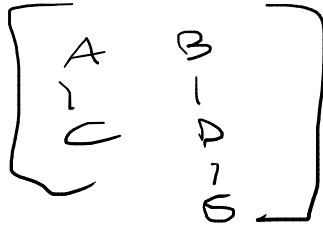
```
class C: public A {};
```

```
class D: public B {
public:
    void f() const { cout << "D::f "; }
};
```

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

```
template<class T>
void Fun(const T& ref) {
    try{ throw ref; }
    catch(const C& c) {cout << "C ";}
    catch(const E& e) {cout << "E "; e.f();}
    catch(const B& b) {cout << "B "; b.f();}
    catch(const A& a) {cout << "A ";}
    catch(const D& d) {cout << "D ";}
    catch(...) {cout << "GEN ";}
```

```
C c; D d; E e; A& a1 = c; B& b1 = d; B& b2 = e; D& d1 = e; D* pd = dynamic_cast<E*>(&b2);
```



Fun(c); cout << endl; $\rightarrow A$

Fun(d) $\rightarrow B$ D::f

Fun(e) $\rightarrow E$ E::f

Fun(a1) $\rightarrow A$ A1 = A1C

Fun(d) (*pd); cout << endl;

$\rightarrow B$ D & PD = NON E () ; $\rightarrow B$ E::f

Fun<E>(*pd); cout << endl;

PD $\rightarrow D$ / E NON COMPILA

Le precedenti definizioni compilano senza provocare errori (con gli opportuni #include e using). Per ognuna delle seguenti istruzioni di invocazione della funzione Fun 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 cout; se non provoca alcuna stampa allora si scriva **NESSUNA STAMPA**.

Fun(c);	
Fun(d);	
Fun(e);	TEMPLATES \rightarrow MATCH CON TIPO STATICO PIU' T
Fun(a1);	
Fun(b1);	
Fun(d1);	T = TYPE NAME (GENERICO)
Fun(*pd);	
Fun<D>(*pd);	
Fun<D>(e);	
Fun<E>(*pd);	Fun<T>(...)
Fun<E>(e);	TS! TS(TD) E / E / E::f
Fun<E>(d1);	
Fun<A>(c);	

\rightarrow Fun(Fun<E>(e))