## Esercizio Gerarchia

Definire una unica gerarchia di classi che includa:

- (1) una unica classe base polimorfa A alla radice della gerarchia;
- (2) una classe derivata astratta B;
- (3) una sottoclasse C di B che sia concreta;
- (4) una classe □ che non permetta la costruzione pubblica dei suoi oggetti, ma solamente la costruzione di oggetti di □ che siano sottooggetti;
- (5) una classe E definita mediante derivazione multipla a diamante con base virtuale, che abbia D come supertipo, e con l'assegnazione ridefinita pubblicamente con comportamento identico a quello dell'assegnazione standard di E.

```
class C: public B{
// Gerarchia 1
                                                                           // concreto = utilizzo il metodo astratto
                                                                           // e gli "do un senso" = lo uso
class A{
                                                                           virtual double calcola(){
        virtual ~A(){};
                                                                                    return 42;
};
                                                                           }
                                                                  };
class B: public A{
        // astratto = virtuale puro
                                                                  class D: virtual public B, virtual public C{
        // classe astratta = non poter instanziar oggetti
                                                                           private:
        virtual double calcola() = 0;
                                                                                    int x;
};
                                                                           protected:
                                                                                    D(int x1): x(x1) {};
                        class E: public D{
                                                                  };
                               private:
                                      int i:
                                      double* d;
                               public:
                                      E& operator=(const E& e){
                                             D::operator=(e);
                                             i = e.i;
                                             d = e.d;
                                             return *this;
                                      }
                        };
```

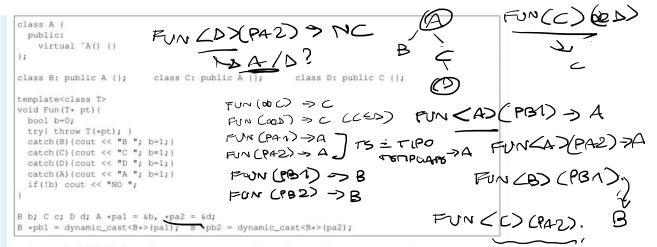
## Esercizio 2

Scrivere un programma che consista esattamente di tre classi A, B e C, dove B è un sottotipo di A, mentre C non è in relazione di subtyping nè con A nè con B, che dimostri in un metodo di C un tipico esempio di un uso giustificato e necessario della conversione di tipo dynamic\_cast per effettuare type downcasting. A questo fine, si usino il minor numero possibile di metodi.

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

class C{
    // Giusta
    A* a;
    C* c = dynamic_cast<C*>(a);

    // Caso da coprire
    B* b = dynamic_cast<B*>(a);
    C* c = dynamic_cast<C*>(b);
};
```



Le precedenti definizioni compilano senza provocare errori (con gli opportuni #include e using).

Per ognuna delle seguenti 12 istruzioni di invocazione della funzione Fun della tabella scrivere chiaramente nel foglio 12 righe con numerazione da 01 a 12 e per ciascuna riga:

AON COMPLIA

- · 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 la stampa che l'esecuzione produce in output su cout; se non provoca alcuna stampa allora si scriva NESSUNA STAMPA.

01:	Fun (&c);		
02:	Fun (6d);		
03:	Fun(pal);		
04:	Fun (pa2);		- 1/1
05;	Fun (pb1); FUN (FUN(A) (PA2)).	_	-> A/A
06:	Fun(pb2);		
07:	Fun <a>(pb1);</a>		
08;	Fun <a>(pa2);</a>		
09:	Fun <b>(pb1);</b>		
10:	Fun <c>(pa2);</c>		
11:	Fun <c>(6d);</c>		
12:	Fun <d>(pa2);</d>		