

```

class A { public: virtual ~A(){} };
class B: public A {};
class C: public A {};
class D: virtual public B {};
class E: virtual public B {};
class F: public D, public E {};

void f(A* p) throw(E*) {
    if(!dynamic_cast<D*>(p)) throw new E();
}

char fun(A* p, B& r) {
    if(typeid(*p)==typeid(r)) return '1';
    try{
        f(&r);
    } catch(B*) {return '2';}
    E* punt = dynamic_cast<F*>(&r);
    try{
        B& ref = dynamic_cast<B*>(*p);
    }
    catch(bad_cast) {
        if(!punt) return '3';
        return '4';
    }
    if(punt) return '5';
    return '6';
}

```

```

int main(){
    A a; B b; C c; D d; E e; F f;
}

```

Definire opportunamente le chiamate in tale main() usando gli oggetti a, b, c, d, e, f locali al main() in modo tale che l'esecuzione del main() provochi la stampa 123456 su cout.

1 → FUN (?, ?) (A, A)
 2 → FUN (D, B) (A, B)
 3 → FUN (D, B) (D, D)
 4 → FUN (A, F) (A, B)
 5 → FUN (B, F) (B, F)
 6 → FUN (B, B) (B, B)

IN F → CAST SUCCESS (A ≤ D)
 P, D PER B (B ≤ B)

REF, R & B
 P, D PER B (B ≤ B)

F(B, B)
 4
 6

NO BAD_CAST → SI, NO!