static_cast und dynamic_cast

```
#include <iostream>
using namespace std;
class B {
    protected: int b;
    public:    B(int b=0):b(b){}
               virtual int get_b(){ return b; }
};

class D: public B { int d;
public: D(int b=0, int d=0):B(b),d(d){}
        int get_d(){ return d; }
};

void main(){
    B *pbd = new D(1,2);
    B *pbb = new B(5);
    D *pdd = new D(3,4);
//  D *pd2 =(D *)(pbd); //D *pd2 = static_cast<D*>(pbd)

    D *pd2 = static_cast<D*>(pbd); //unsafe, result true
    cout<<"b = "<<pd2->get_b()<<"  "<<"  d (true) = "
        <<pd2->get_d()<<endl;

    B *pb2 = static_cast<B *>(pdd);//safe upcast
    cout<<"b = "<<pb2->get_b()<<endl;

    D *pd3 = static_cast<D*>(pbb); //unsafe, result false
    cout<<"b = "<<pd3->get_b()<<"  "<<"  d (false) = "
        <<pd3->get_d()<<endl;

    pd3 = dynamic_cast<D *>(pbb);
    if(!pd3){ cout<<"pd3 == 0 (unzulaessiger Downcast)\n";
              cout<<"pbb->b = "<<pbb->get_b()<<endl; }
    else
        cout<<"b = "<<pd3->get_b()<<"  "<<"  d = "
            <<pd3->get_d()<<endl;

    try { *pd3 = dynamic_cast<D &>(*pbb);
          cout<<"b = "<<pd3->get_b()<<"  "<<"  d = " }
    catch(...){ cout<<"*pd3 unzulaessig !\n";
                cout<<"pbb->b = "<<pbb->get_b()<<endl;}
    delete pbd; pbd=0; delete pbb; pbb=0; delete pdd; pdd=0;
}
b = 1    d (true)  = 2
b = 3
b = 5    d (false) = -33686019
pd3 == 0 (unzulaeesiger Downcast)
pbb->b = 5
*pd3 unzulaessig !
pbb->b = 5
```