private - Ableitung

```
#include <iostream>
using namespace std;

class base{ public:    void pubB();
            private:   void privB();
            protected: void protB();
};

class derived:private base{ public: base::pubB;
                         // public:  using base::pubB; //alt.
                            public:  void pubD();
                            private: void privD();
};

void out(base &b){cout<<"out"<<endl;
                  b.pubB();
                //  b.privB();
                //  b.protB();
};

void base::pubB(){ cout<<"base pubB()"<<endl;    }

void base::privB(){ cout<<"base privB()"<<endl;  }

void base::protB(){ cout<<" base protB()"<<endl; }

void derived::pubD(){ cout<<"derived pubD()"<<endl; }

void derived::privD(){ cout<<"derived privD()"<<endl; }

void main(){
        derived d1;
        d1.pubB();
        d1.pubD();
     // base b5=d1;              // Error, kein Zugriff !
     // base b6=(base)d1;        // Error, kein Zugriff !
     // base   *b1=&d1;          // Error, kein Zugriff !
     // base   &b2=d1;           // Error, kein Zugriff !
        base   *b3=(base *)&d1;
        base   &b4=(base &)d1;
     // out(d1);                 // Error, kein Zugriff !
        out((base &)d1);
        cin.ignore();
}
```