```cpp
#include<iostream>
using namespace std;
class base
{
        protected:
                int base_number;
        public:
                base(int=0);
                base(const base&);
                ~base();
                int get_base_number()const;
};
inline base::base(int x) : base_number(x)
{
        cout << "In base constructor" << endl;
}
inline base::base(const base& b) : base_number(b.base_number)
{
        cout << "In base copy constructor" << endl;
}
inline base::~base()
{
        cout << "In base destructor" << endl;
}
inline int base::get_base_number()const
{
        return base_number;
}
class derived : public base
{
                int derived_number;
        public:
                derived(int = 0, int = 0);
                derived(const derived&);
                ~derived();
                int get_derived_number()const;
};
inline derived::derived(int a, int b) : base(a), derived_number(b)
{
        cout << "In derived constructor" << endl;
}
inline derived::derived(const derived& d) : base(d),derived_number(d.derived_number)
{
        cout << "In derived copy constructor" << endl;
}
inline derived::~derived()
{
        cout << "In derived destructor" << endl;
}
inline int derived::get_derived_number()const
{
        return derived_number;
}
int main()
{
        derived d1(1,2);
        derived d2 = d1;
        cout << "d2 = (" << d2.get_base_number()  << "," << d2.get_derived_number() << ")" << endl;
        return 0;
}
```