

ITEM 8:

Date:

Destructors should never emit exceptions.

```
class DbConnection {  
public:  
    static DbConnection create();  
    void close();  
};
```

To ensure client don't forget to call close on DbConnection, make a resource managing class DbConn which calls close in its destructor.

```
class DbConn  
{ public:  
    ~DbConn()  
    { db.close();  
private:  
    DbConnection db; // composition  
};
```

If some exception comes while closing Connection in DbConn's destructor then it should not emit exception, it should either terminate program.

```
try { db.close(); }  
catch (...) {  
    std::abort();  
}
```


or swallow exceptn.

```
DbConn::~DbConn()
{
    try { db.close(); }
    catch(...) { make log entry }
}
```

Better strategy is to design DbConn's Interface to allow client, react to problem

Make close() funcn of DbConn.

```
class DbConn
{
    void close()
    {
        db.close();
        closed = true;
    }
    ~DbConn()
    {
        if (!closed) {
            try { db.close(); }
            catch(...) { make log }
        }
    }
private:
    DbConnect db;
    bool closed;
};
```