Date:.....

ITEM 8:

Destructors should never emit exceptns.

```
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 exceptn comes while closing Connection in DbConn's destructor then it should not emit exceptn, it should either terminate program.

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

Date:.....

or swallow exception.

```
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() func^n of DbConn.

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