OOP Homework 1

设计一个保存整数矩阵的 `Matrix` 类。下面有一些基本要求：

- `mat.nRow()`，`mat.nCol()` 返回矩阵的行数和列数
- 能使用 `mat(i, j)` 的方式访问到第 `i` 行第 `j` 列的元素
- 能使用 `mat[i][j]` 的方式访问到第 `i` 行第 `j` 列的元素
- 能使用 `mat1 += mat2` 的方式把 `mat2` 加到 `mat1` 上
- 能使用 `mat4 = mat1 + mat2 + mat3` 的方式把矩阵相加
- 能使用 `mat1 == mat2` 判断两矩阵是否相等
- 能使用流操作符把矩阵输出 `out << mat`
- 能使用流操作符把矩阵输入 `in >> mat`
- 用流操作符把矩阵输出再用流操作符输入，应该得到原矩阵
- 对超出边界的下标访问需要简单处理
- 自己管理内存，不允许使用任何STL容器
- 用户使用你的 `Matrix` 类的时候应该能够像使用STL容器一样，不用担心内存问题，也就是，你得实现好析构函数

还有一些注意事项：

- 注意不要有内存泄露，提交作业后我们会用 `valgrind` 检验
- 保存一个 `10000*10000` 的稠密矩阵不应该崩溃（如果你有 `400MB` 空余内存的话）
- 要正确实现构造函数、复制构造函数、赋值操作符以及析构函数
- 对一个 `const` 对象，如 `const Matrix mat`，那么 `mat[i][j] = x;` 不应该能够修改 `mat` 的值
- `mat[i][j]` 应该能检查下标越界
- 更多的要求见 `matrix_test.cc` 中的测试
    - 所有 `test??()` 函数都必须 `PASS`
    - 所有 `ce??()` 函数都不应该通过编译

此次作业旨在让同学们掌握类以及一些常用成员和非成员函数的实现方法，同学们在满足要求的情况下可以自由发挥，希望同学们的实现都能尽量靠近 Best Practice。

- **Deadline:** 2015年12月06日 23:59:59
- **提交形式:** 一个 `matrix.h` 即可
- **提交方式:** 发邮件至 `i@abcdabcd987.com`