CME213: Parallel Computing using MPI, OpenMP and CUDA
Homework 0: Prerequisites **Chi Zhang**
Stanford University SUID: 06116342
Spring 2018 Date: April 11, 2018

Problem 1

Running on a Macbook (early 2015 model) with 2.9GHz Intel Core i5 processor, the time of swapping data in two arrays is measured as $9036.08\mu s$ while swapping two pointers is $0.064\mu s$. It is obvious that swapping data takes a considerably larger amount of time compared to simply swapping pointers. In terms of algorithmic analysis, swapping data is $O(n)$ operation while swapping pointers is $O(1)$.

Problem 2

(a) Refer to code for details. The method `l0_norm()` uses `std::count_if()`. Also, please note it is assumed that the size of lower triangular matrices falls into the range `[1, INT_MAX]`.

(b) Since the base class `Matrix` is a *pure abstract* class, it only provides interfaces for derived classes. In my design, the interfaces are

```
virtual int l0_norm() = 0;
virtual size_t size() = 0;
virtual T &operator ()(const int i, const int j) = 0;
```

By implementing the three interfaces and a constructor, `MatrixLt` class supports three operations:

- construct objects with specified size
- read and write data from/to lower triangular elements (upper triangular elements are zeros, which users cannot touch) using `()` operator
- obtain the number of non-zero values using `l0_norm()`
- obtain the size of the matrix using `size()`

Although there are much more operations a fully-functional matrix class should support, such as `+, -` operators, the operations implemented in my `MatrixLt` class meet all specifications of this problem.

Test cases include mainly two aspects: **initialization** and **read/write data**.

- initialize `MatrixLt` objects with different data types and sizes (empty matrices are NOT allowed) given it is a template class
- try accessing matrix elements with invalid indices, such as upper triangular elements
- write data to lower triangular elements and read data
- test whether `l0_norm()` works properly

(c) Refer to `main_q2.cpp` for test implementations. My class passed all my tests.

Problem 3

This is a C++ runtime polymorphism problem. The key here is declaring the container as a vector of base class type `Matrix`, and pushing objects of derived classes into it.

Problem 4

On my Macbook, the number of points in the range `[2, 10]` is 23. On Stanford cardinal machine, the number of points in the range `[2, 10]` is 19.