Applied Programming
Submission 3

In this assignment you are going to extend an existing Matrix and Vector implementation using Templates. Currently the Matrix and Vector code uses *double* arrays to store the entries, and in this assignment you will extend it to handle any primitive type (int, float, double, etc.) by the use of templates. The Matrix and Vector header files are uploaded along this description.

The handin format is the same as usual.

What to hand in

You should extend the implementation in the *Vector.hpp* and *Matrix.hpp* - note that you should **not** create corresponding .cpp files for these, but just modify the code inside the .hpp files. This means that the .zip file you are submitting should just contain those two header files.

How to get started

We recommend that you start templating the Vector class, and get that to work before moving on to the Matrix class. Start by modifying the function declarations, and then inspect each function to see where you have to change something (e.g. adding $< T >$ or changing double to T, where T is the typename). Before checking your solution in the code checker you should get the following code snippet to compile:

```
// Get this to work as a first step
#include "Vector.hpp"
#include "Matrix.hpp"

int main(int argc, char const *argv[])
{

    Vector<float> v(2);
    v(1) = 3.14;

    Matrix<int> m(2,2);
    m(2,2) = 13;

    return 0;
}
```

The most difficult part of the assignment is probably getting the friend functions to work as you will likely get errors like "...undefined reference...". Try to consult the book or search the internet for how to avoid this kind of error (`http://stackoverflow.com/questions/4660123/overloading-friend-operator-for-template-class`).