Exercises week 1

Klaas Isaac Bijlsma
s2394480

David Vroom
s2309939

November 18, 2017

Exercise 1

Attain some familiarity with the way functions are selected from namespaces

We used the following code,

Exercise 2

ziet ie dit?

Exercise 3

Learn to implement index operators

The Matrix class that is used here, is derived from the solutions of excercise 64. We used the following code,

matrix/matrix.h

```
#ifndef INCLUDED_MATRIX_
#define INCLUDED_MATRIX_

#include <iosfwd>
#include <initializer_list>

class Matrix
{
    size_t d_nRows = 0;
    size_t d_nCols = 0;
    double *d_data = 0;                     // in fact R x C matrix

    public:
        typedef std::initializer_list<
            std::initializer_list<double>> IniList;

        Matrix() = default;
        Matrix(size_t nRows, size_t nCols);         // 1
        Matrix(Matrix const &other);                // 2
        Matrix(Matrix &&tmp);                       // 3
        Matrix(IniList inilist);                    // 4

        ~Matrix();

        Matrix &operator=(Matrix const &rhs);
        Matrix &operator=(Matrix &&tmp);

        size_t nRows() const;
        size_t nCols() const;
        size_t size() const;            // nRows * nCols

        static Matrix identity(size_t dim);
```

```
33
34          Matrix &tr();                   // transpose (must be square)
35          Matrix transpose() const;       // any dim.
36
37          void swap(Matrix &other);
38
39              // exercise 3
40              // ==========
41          double *operator[](size_t index);
42          double *operator[](size_t index) const;
43
44
45  private:
46          double &el(size_t row, size_t col) const;
47          void transpose(double *dest) const;
48
49              // exercise 3
50              // ==========               // private backdoor
51          double *operatorIndex(size_t index) const;
52  };
53
54  inline size_t Matrix::nCols() const
55  {
56      return d_nCols;
57  }
58
59  inline size_t Matrix::nRows() const
60  {
61      return d_nRows;
62  }
63
64  inline size_t Matrix::size() const
65  {
66      return d_nRows * d_nCols;
67  }
68
69  inline double &Matrix::el(size_t row, size_t col) const
70  {
71      return d_data[row * d_nCols + col];
72  }
73
```

```
74      // exercise 3
75      // ==========
76  inline double *Matrix::operatorIndex(size_t index) const
77  {
78      return d_data + index * d_nCols;
79  }
80
81  inline double *Matrix::operator[](size_t index)
82  {
83      return operatorIndex(index);
84  }
85
86  inline double *Matrix::operator[](size_t index) const
87  {
88      return operatorIndex(index);
89  }
90
91  #endif
```

Exercise 4

Exercise 5

Exercise 6

Exercise 7

Exercise 8

Exercise 9

Exercise 10