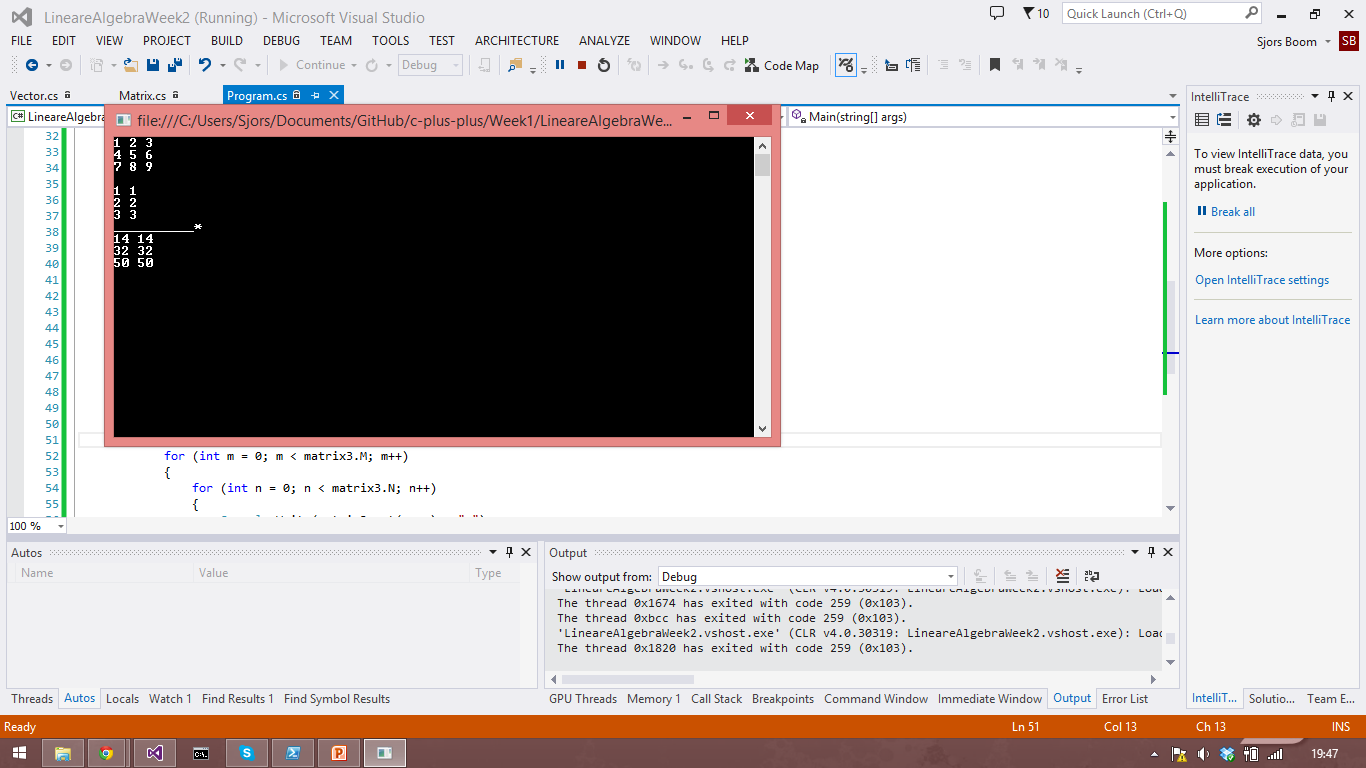
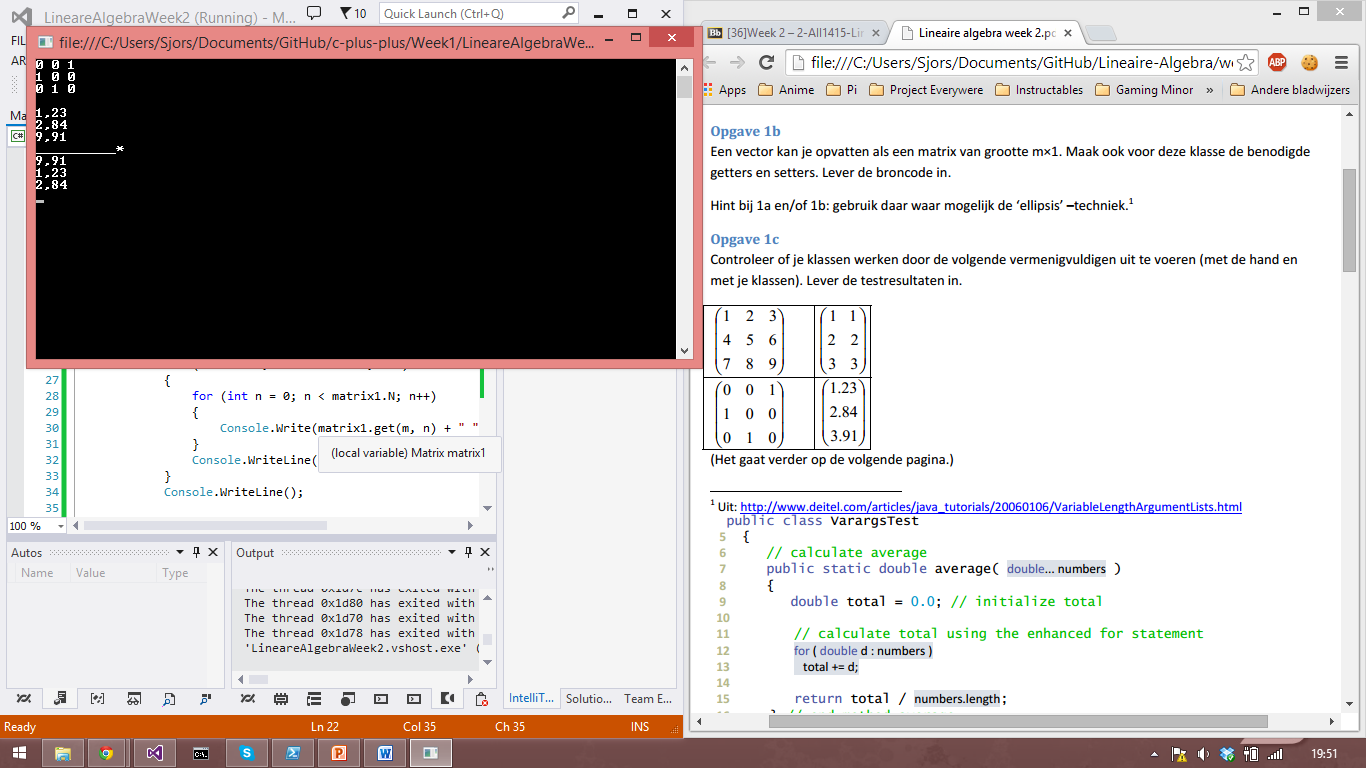
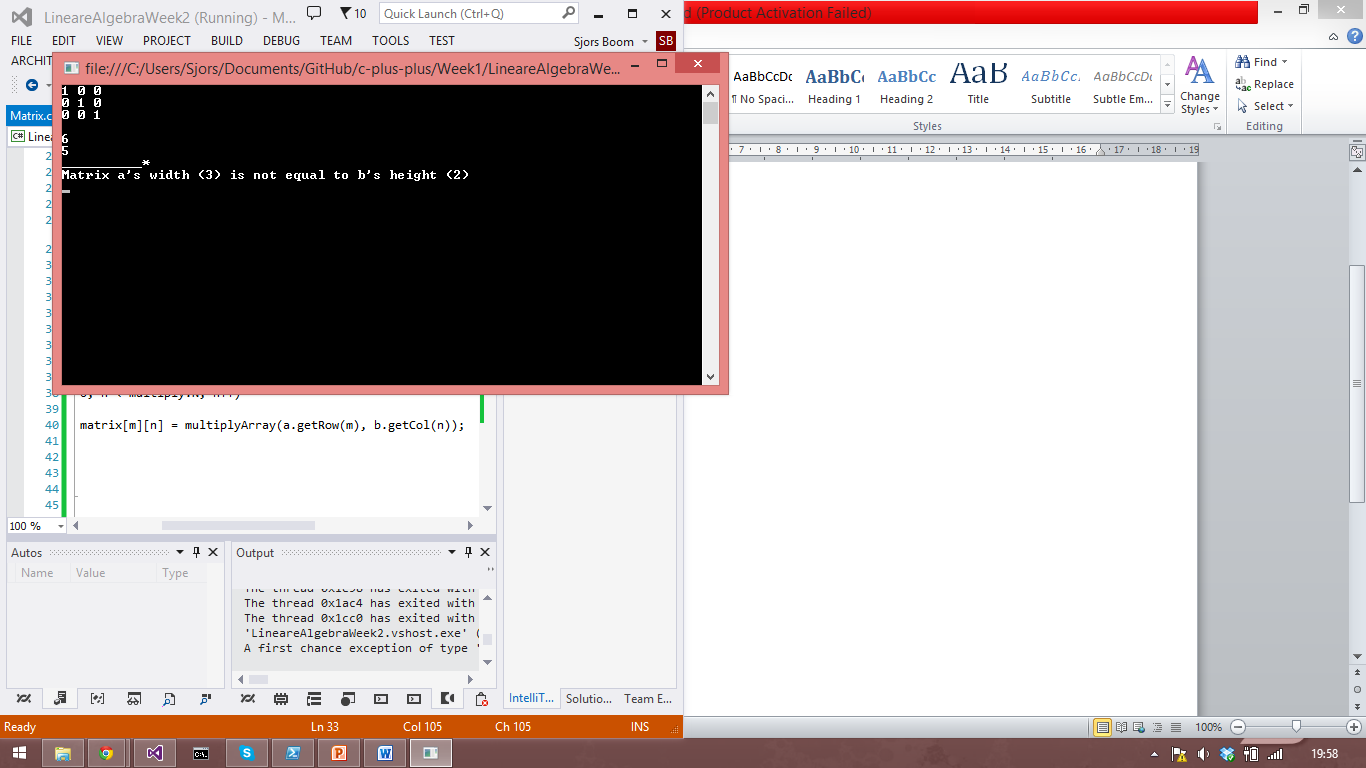
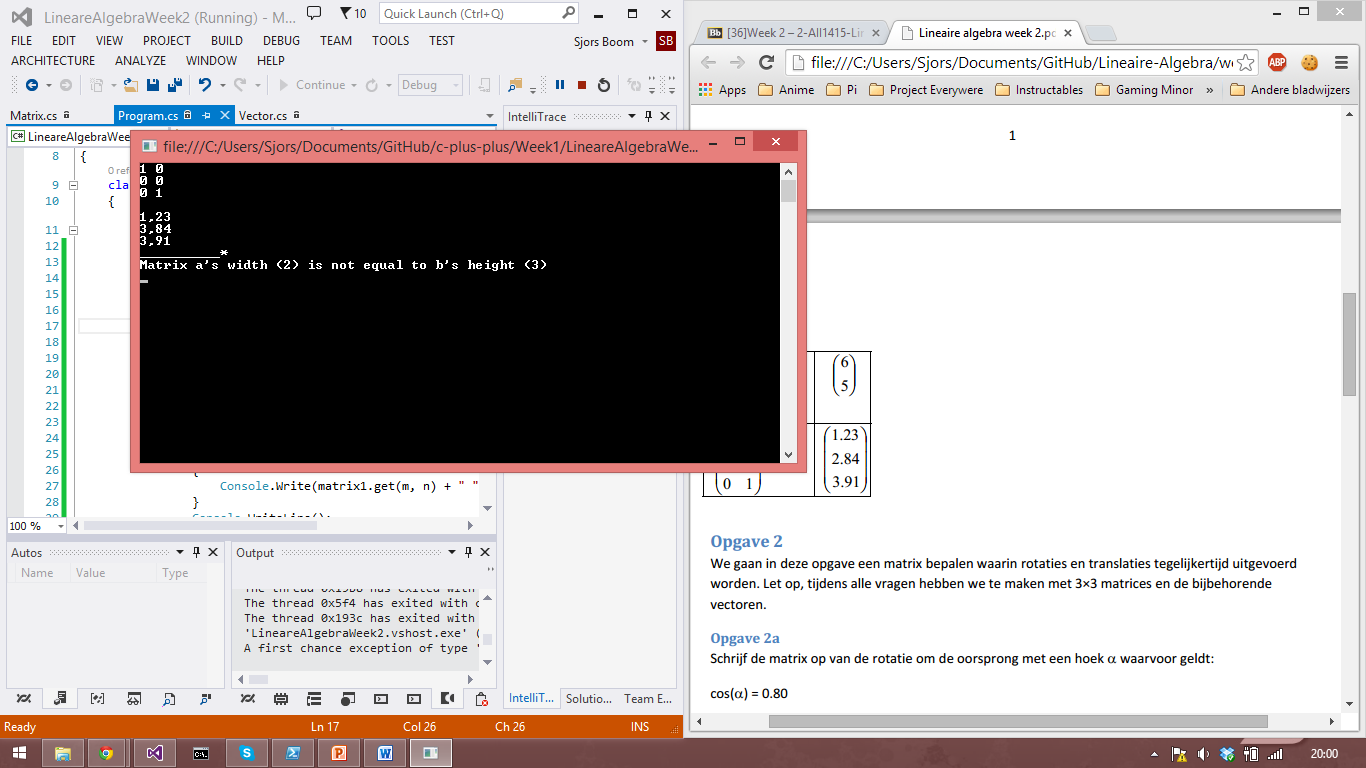
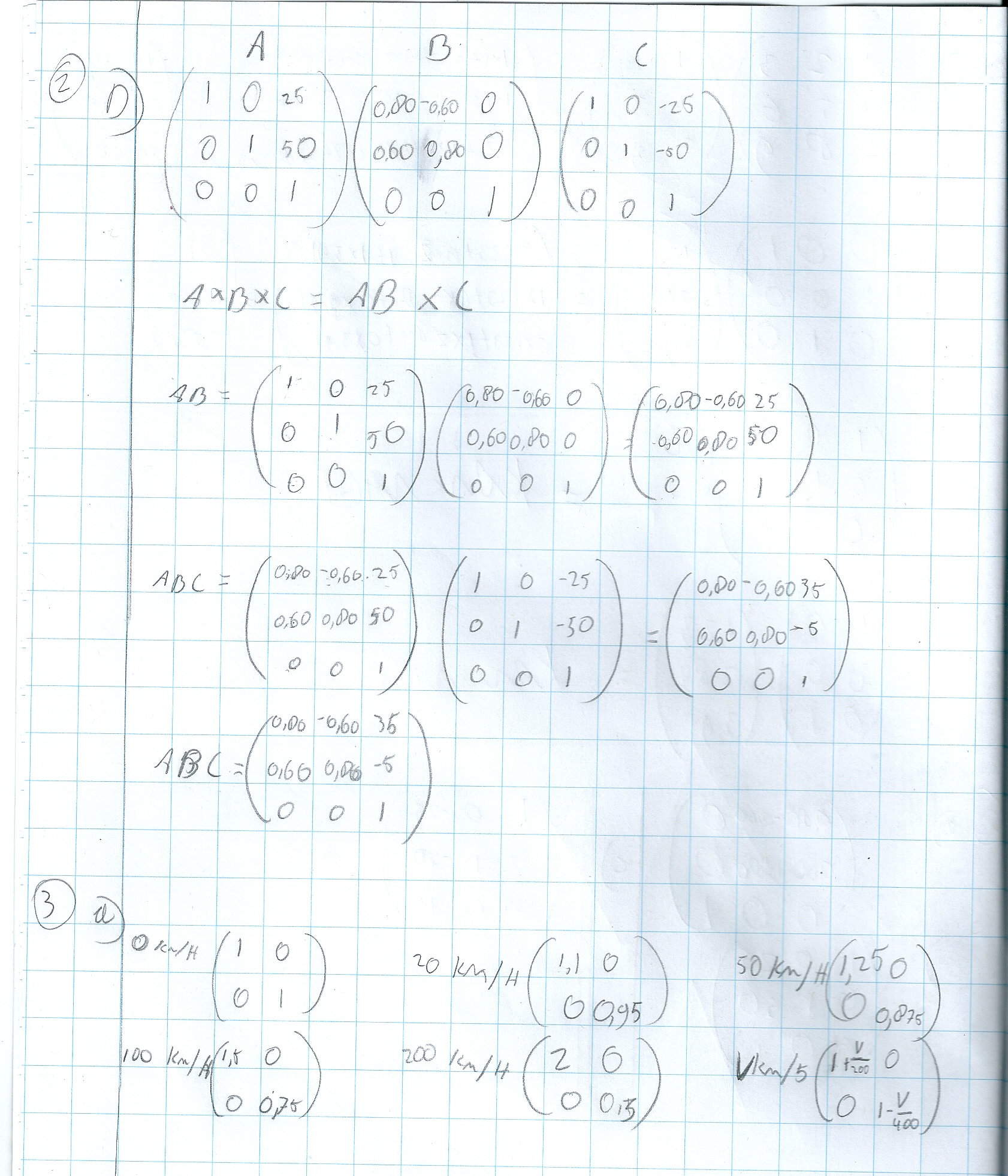
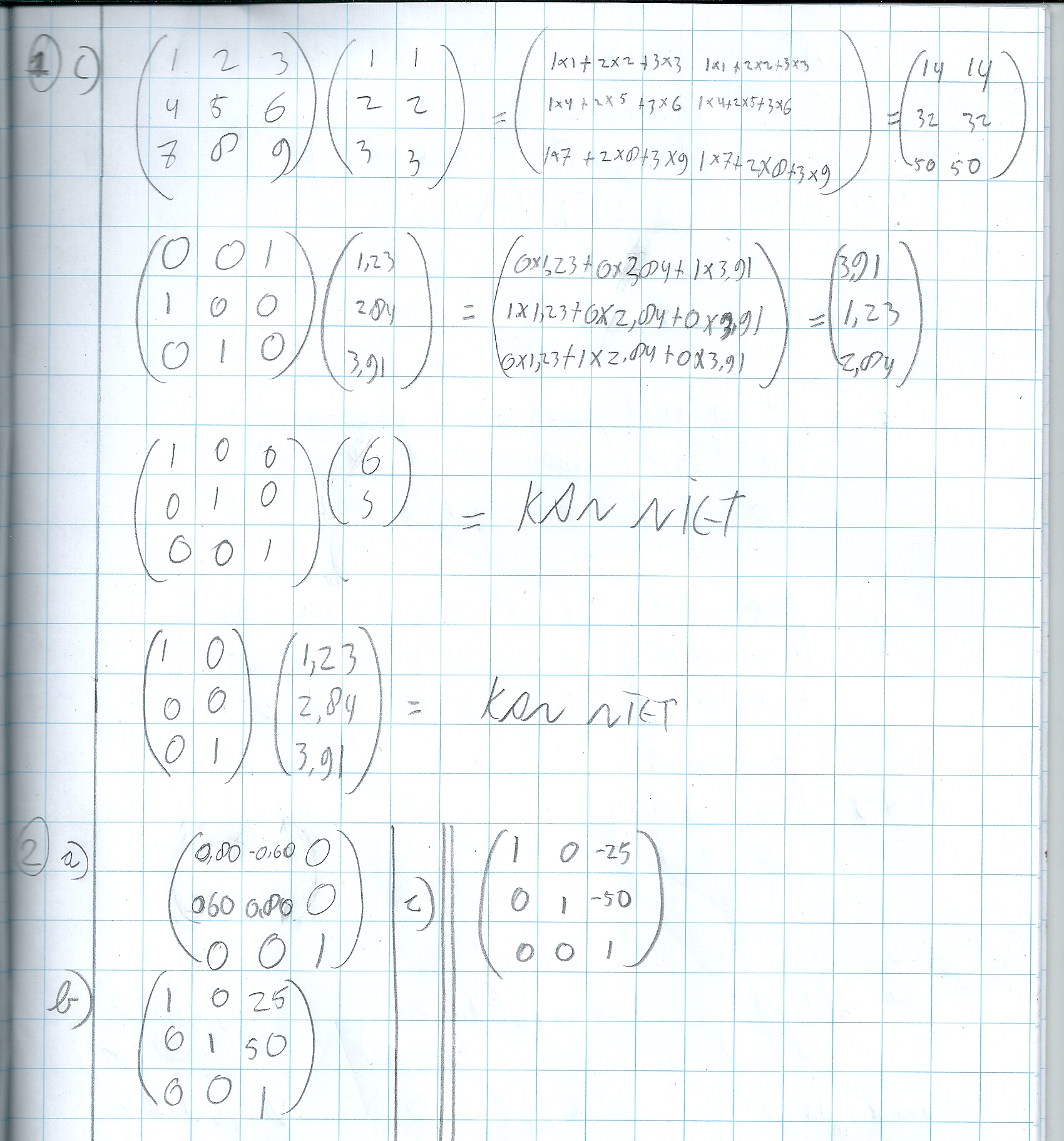
# Opgave 1

## C







class Matrix

{

double[][] matrix;

public int M { get; private set; }

public int N { get; private set; }

public Matrix(int m, int n)

{

M = m;

N = n;

matrix = new double[M][];

for (int i = 0; i < m; i++)

{

matrix[i] = new double[N];

}

}

public static Matrix operator \*(Matrix a, Matrix b)

{

if (a.N != b.M)

{

throw new Exception("Matrix a's width (" + a.N + ") is not equal to b's

height (" + b.M + ")");

}

Matrix multiply = new Matrix(a.M, b.N);

for (int m = 0; m < multiply.M; m++)

{

for (int n = 0; n < multiply.N; n++)

{

multiply.matrix[m][n] = multiplyArray(a.getRow(m), b.getCol(n));

}

}

return multiply;

}

private static double multiplyArray(double[] row, double[] col)

{

double result = 0;

for (int i = 0; i < row.Length; i++)

{

result += row[i] \* col[i];

}

return result;

}

public double[] getCol(int n)

{

double[] col = new double[M];

for (int i = 0; i < M; i++)

{

col[i] = matrix[i][n];

}

return col;

}

public double[] getRow(int m)

{

return matrix[m];

}

public double get(int m, int n)

{

return matrix[m][n];

}

public void set(int m, int n, double value)

{

matrix[m][n] = value;

}

public static Matrix scale(double velocity)

{

Matrix scale = new Matrix(2, 2);

scale.set(0, 0, 1 + velocity / 200);

scale.set(1, 1, 1 - velocity / 400);

return scale;

}

}

class Vector : Matrix

{

public Vector(int m)

: base(m, 1)

{

}

public void set(int m, double value)

{

base.set(m, 1, value);

}

public double get(int m)

{

return base.get(m, 1);

}

}

class Program

{

static void Main(string[] args)

{

Matrix matrix1 = new Matrix(3, 3);

Matrix matrix2 = new Matrix(3, 3);

Matrix matrix3 = new Matrix(3, 3);

matrix1.set(1, 1, 1);

matrix1.set(2, 2, 1);

matrix1.set(1, 3, 25);

matrix1.set(2, 3, 50);

matrix1.set(3, 3, 1);

matrix2.set(1, 1, 0.80);

matrix2.set(1, 2, -0.60);

matrix2.set(2, 1, 0.60);

matrix2.set(2, 2, 0.80);

matrix2.set(3, 3, 1);

matrix3.set(1, 1, 1);

matrix3.set(2, 2, 1);

matrix3.set(1, 3, -25);

matrix3.set(2, 3, -50);

matrix3.set(3, 3, 1);

for (int m = 1; m <= matrix1.M; m++)

{

for (int n = 1; n <= matrix1.N; n++)

{

Console.Write(matrix1.get(m, n) + " ");

}

Console.WriteLine();

}

Console.WriteLine();

for (int m = 1; m <= matrix2.M; m++)

{

for (int n = 1; n <= matrix2.N; n++)

{

Console.Write(matrix2.get(m, n) + " ");

}

Console.WriteLine();

}

Console.WriteLine();

for (int m = 1; m <= matrix3.M; m++)

{

for (int n = 1; n <= matrix3.N; n++)

{

Console.Write(matrix3.get(m, n) + " ");

}

Console.WriteLine();

}

Console.WriteLine("\_\_\_\_\_\_\_\_\_\_\*");

try

{

Matrix matrix4 = matrix1 \* matrix2 \* matrix3;

for (int m = 1; m <= matrix4.M; m++)

{

for (int n = 1; n <= matrix4.N; n++)

{

Console.Write(matrix4.get(m, n) + " ");

}

Console.WriteLine();

}

}

catch (Exception e)

{

Console.WriteLine(e.Message);

}

Console.ReadLine();

}

}