```java
package matice;


public class barierka {
	private int bariera = 0; // pocet vlaken narazejicich do bariery

	private double suma = 0; // suma prumeru

	private int synchronizace;



	barierka(int synchronizace){
		this.synchronizace = synchronizace;
	}


	synchronized public void pricti(Pracant p) throws InterruptedException{

		if(p.getVelikost() % synchronizace == 0){

			bariera++;

			if(bariera == p.getPocetVlaken()){

				suma += p.getPrumer();

				bariera = 0;

				notify();

				System.out.println("Synchronizace je dokoncena, probouzi
                                    se ostatni vlakna.");
			}
			else{

				suma += p.getPrumer();

				try {

					wait(100);

				} catch (InterruptedException e) {
					// TODO Auto-generated catch block
					e.printStackTrace();
				}
			}
		}
	}

	public double getSuma(){

		return suma;
	}
}
```

```
public class Matice {

	final static int SIZE = 100;

	final static String FILE_NAME = "matrixM.dat";

	public static void main(String[] args) {

		// TODO Auto-generated method stub

		Matrix m1 = new Matrix(FILE_NAME, SIZE);

		m1.start();
		try {


			m1.join();


		} catch (InterruptedException e) {

			// TODO Auto-generated catch block
			e.printStackTrace();
		}
		System.out.println("Prumer v cele matici je: " + m1.prumHodnota());

		System.out.println("Suma: "+m1.getSuma());
	}
}
```

```
public class Pracant extends Thread{

	private String jmeno;

	private Matrix farmar;

	private double prumer;

	private int radek;

	private int velikost = 0;

	private int [][] matice;

	private int pocet_radek;

	private barierka bar;
```

```
public Pracant(String jmeno, int radek, Matrix farmar, int velikost,
                        barierka bar, int pocet_radek){

      this.jmeno = jmeno;

      this.radek = radek;

      this.farmar = farmar;

      this.velikost = velikost;

      this.bar = bar;

      this.pocet_radek = pocet_radek;
}


public void run() {

      int i;

      int suma = 0;

      matice = farmar.getMatice();

      int count = 0;

      int jakyRadek;

      while(count < pocet_radek){

            jakyRadek = farmar.getCisloRadku();

            for(i = 0; i < velikost; i++){

                  suma += matice[jakyRadek][i];
            }

             prumer = suma/velikost;

             farmar.prumery[radek] = prumer;

             farmar.prumery(jakyRadek, prumer);

            farmar.addSuma(suma);

            count++;

            try {

                  bar.pricti(this);

            } catch (InterruptedException e) {

                  // TODO Auto-generated catch block
                  e.printStackTrace();
            }

            System.out.println(jmeno + " ma prumer z " + jakyRadek + ".
                                    radku " + prumer);
```

```
                    //vynulování sumy pro dalsi chod vlaken
                    suma = 0;
                }

        }


        public double getPrumer(){
                return this.prumer;
        }


        public int getVelikost(){
                return this.velikost;
        }


        public int getPocetVlaken(){
                return this.velikost;
        }
}
```

```
package matice;

import java.io.BufferedReader;
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.IOException;


class Matrix extends Thread{

        private int size;

        private int matice[][];

        private BufferedReader in;

        private Pracant pracant[];

        public double prumery[];

        public int pocet_radek;

        public int pocet_pracantu = 4;

        public int citac = -1;

        public int suma = 0;
```

```
Matrix (String file, int size) {

	FileReader fr;

	try {
		fr = new FileReader(file);

		in = new BufferedReader(fr);

	} catch (FileNotFoundException e) {

		// TODO Auto-generated catch block
		e.printStackTrace();
	}

	this.size = size;

	matice = new int[size][size];

	this.pocet_radek = size/pocet_pracantu;
}

synchronized public void prumery(int index_v_poli, double hodnota){

	prumery[index_v_poli] = hodnota;
}

synchronized public void addSuma(int hodnota){

	this.suma += hodnota;
}

public int getSuma(){

	return this.suma;
}

synchronized public int getCisloRadku(){

	citac++;

	return citac;
}

public void run() {

// nacteni matice

	int i;
	int j;
```

```
        String line = null;

        String[] cisla = null;

        for (i = 0; i < size; i++) {

                try {

                        line = in.readLine();
                } catch (IOException e) {

                        // TODO Auto-generated catch block
                        e.printStackTrace();
                }

                cisla = line.split("\t");

                for(j = 0; j <size; j++) {

                        matice[i][j] = Integer.parseInt(cisla[j]);
                }
        }
// konec nacteni matice

        System.out.println("Vytvarim pracovniky:");

        barierka bar = new barierka(size);

        pracant = new Pracant[pocet_pracantu];

        prumery = new double[size];

        int radek;


        for(radek = 0; radek < pocet_pracantu; radek++){

                pracant[radek] = new Pracant("Pracant " + radek, radek, this, size, bar,
                                                        pocet_radek);

                pracant[radek].start();

                //prumery[radek] = pracant[radek].getPrumer();
        }

// farmar ceka na dopracovani pracantu

        int k;

        for(k = 0; k < pocet_pracantu; k++){
```

```
                try {

                        pracant[k].join();

                } catch (InterruptedException e) {

                        // TODO Auto-generated catch block
                        e.printStackTrace();
                }
        }

        try {

                in.close();


        } catch (IOException e) {

                // TODO Auto-generated catch block
                e.printStackTrace();
        }

}

public int [][] getMatice(){

        return matice;
}

public int minHodnota() {

        int i;
        int j;

        int min = matice[0][0];

        for (i = 0; i < size; i++) {

                for (j = 0 ; j < size; j ++) {

                        if (min > matice[i][j]) min = matice[i][j];
                }
        }

        return (min);
}

public int maxHodnota() {
        int i;
        int j;

        int max = matice[0][0];
```

```
            for (i = 0; i < size; i++) {

                for (j = 0 ; j < size; j ++) {

                    if (max < matice[i][j]) max = matice[i][j];
                }
            }

            return (max);
        }

        public double prumHodnota() {

            int i;

            double prumer;


            int suma = 0;

            for(i = 0; i < size; i++) {

                    suma += prumery[i];
            }

            //prumer = suma / (size * size);

            prumer = suma / size;

            return prumer;
        }

}
```
