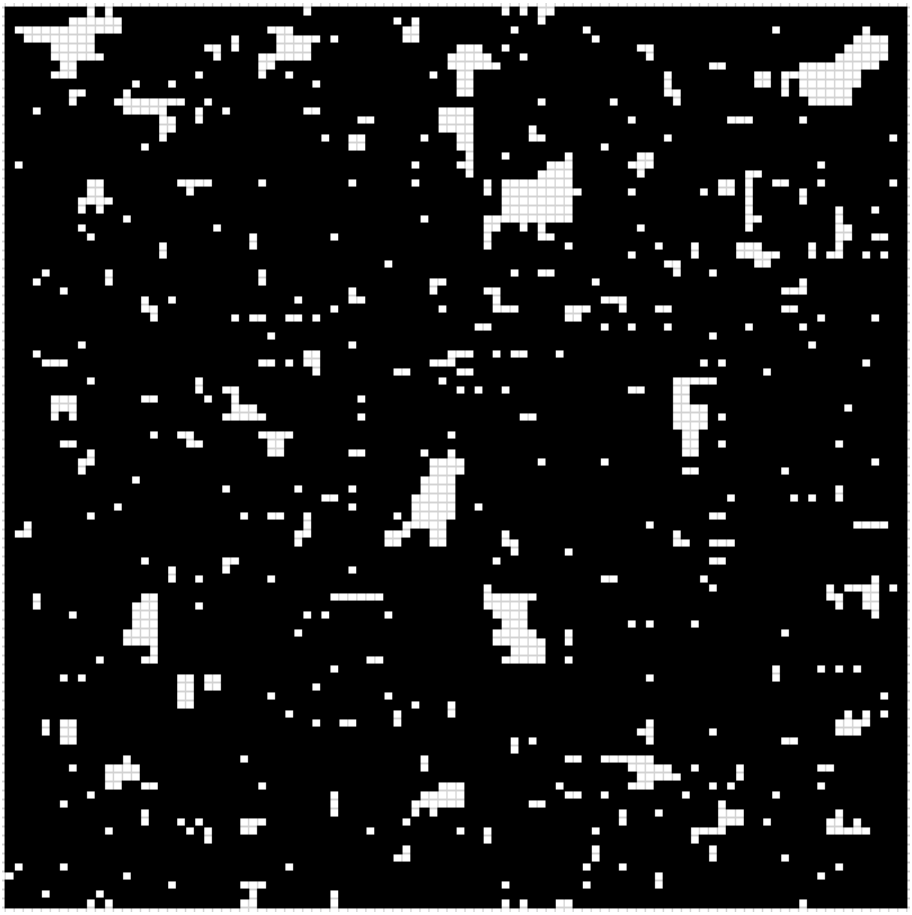
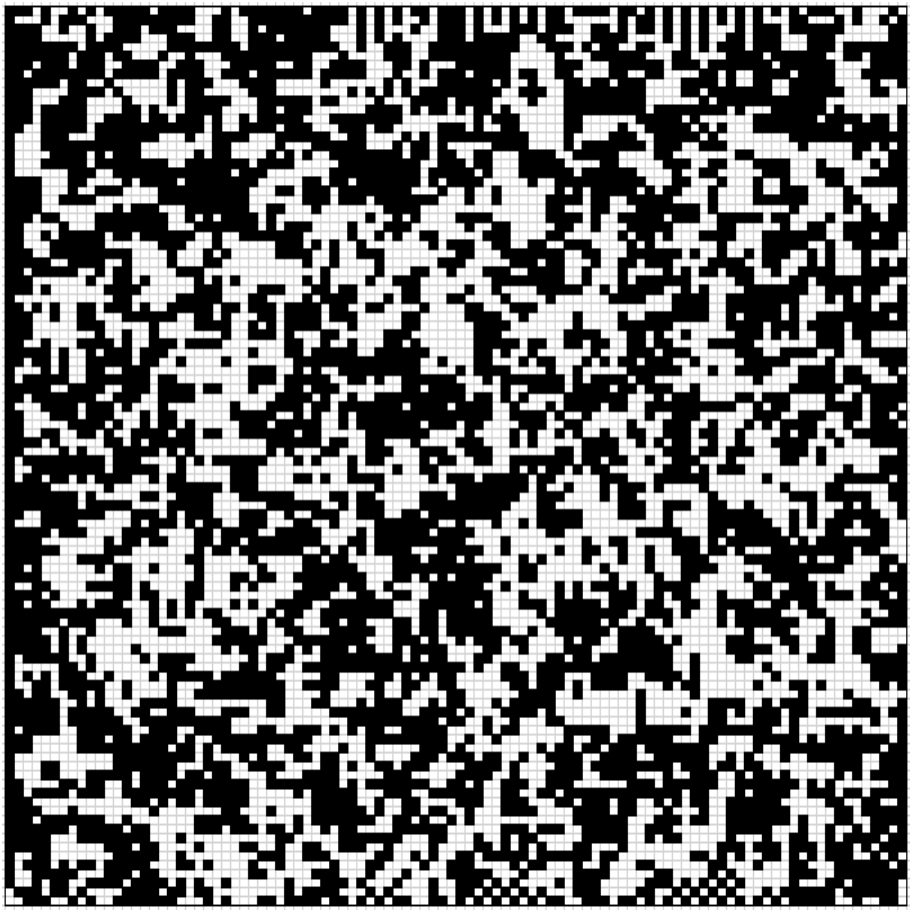
Przykładowe konfiguracje dla układu 100x100

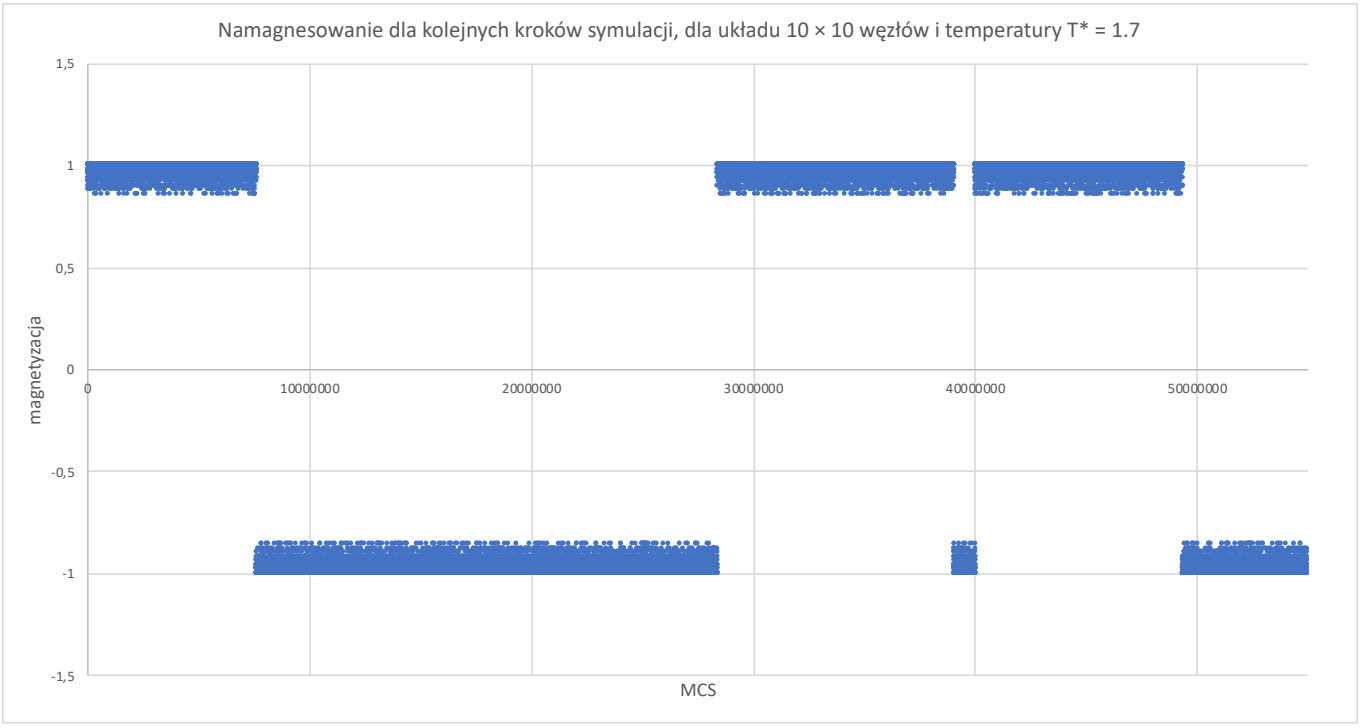
1. T*=1,2

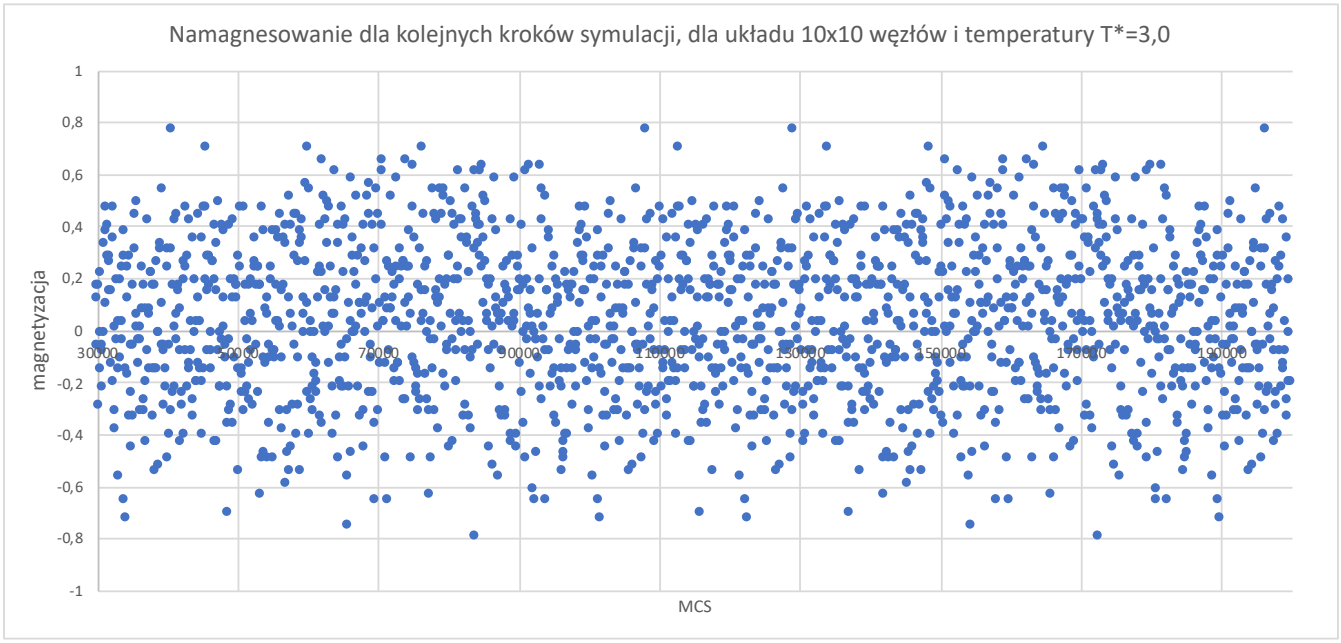
2. T*=2,26



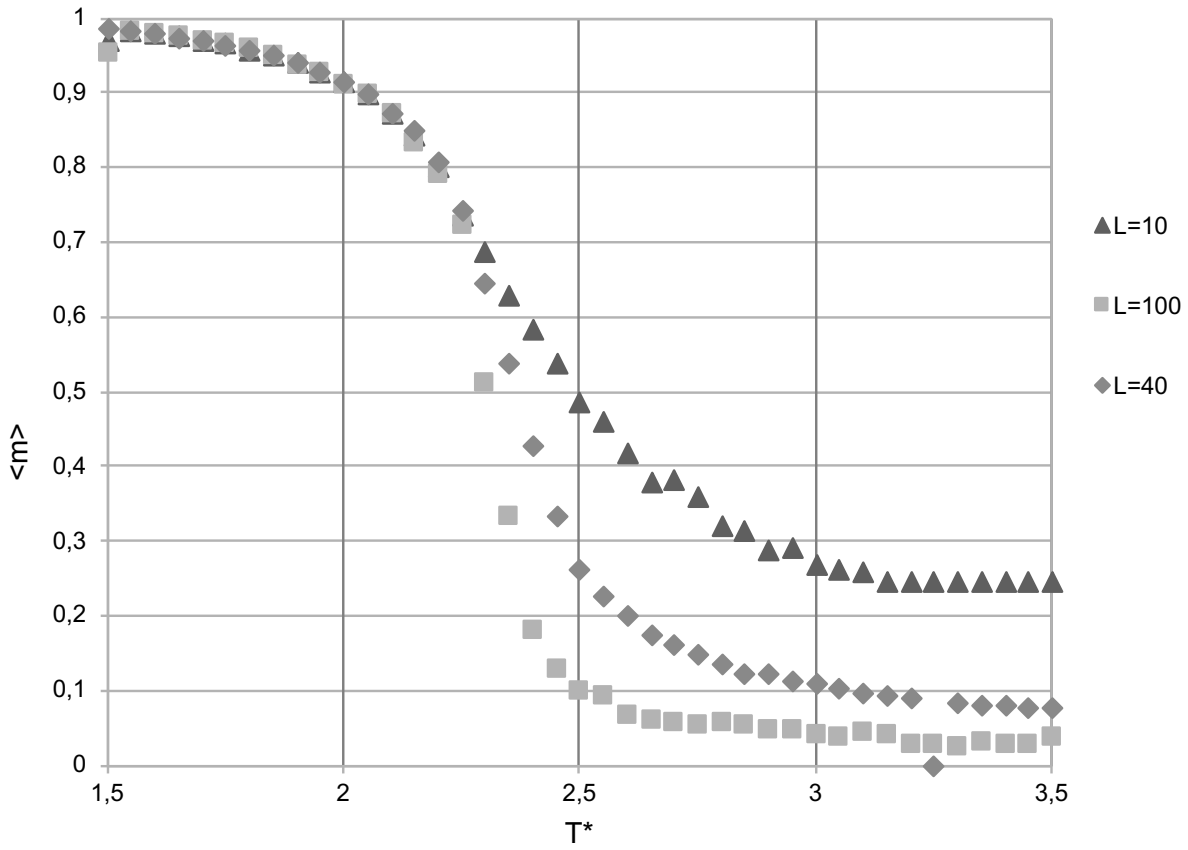
3. T*=4,15

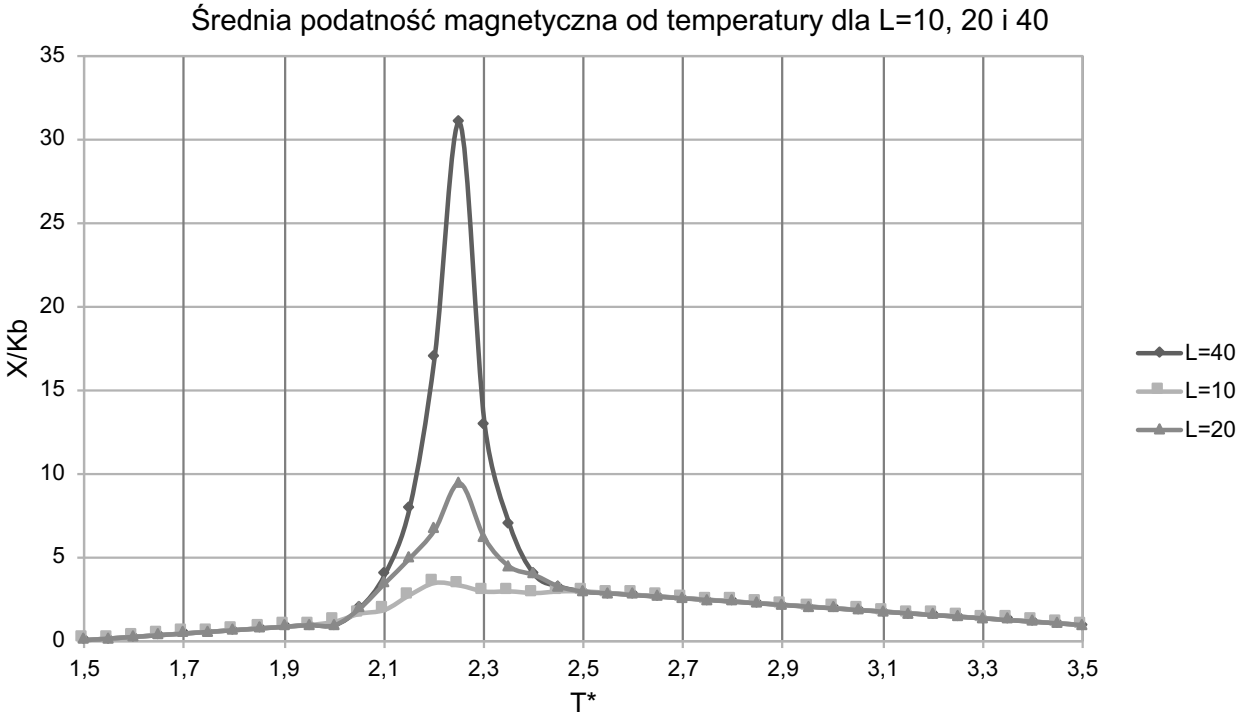


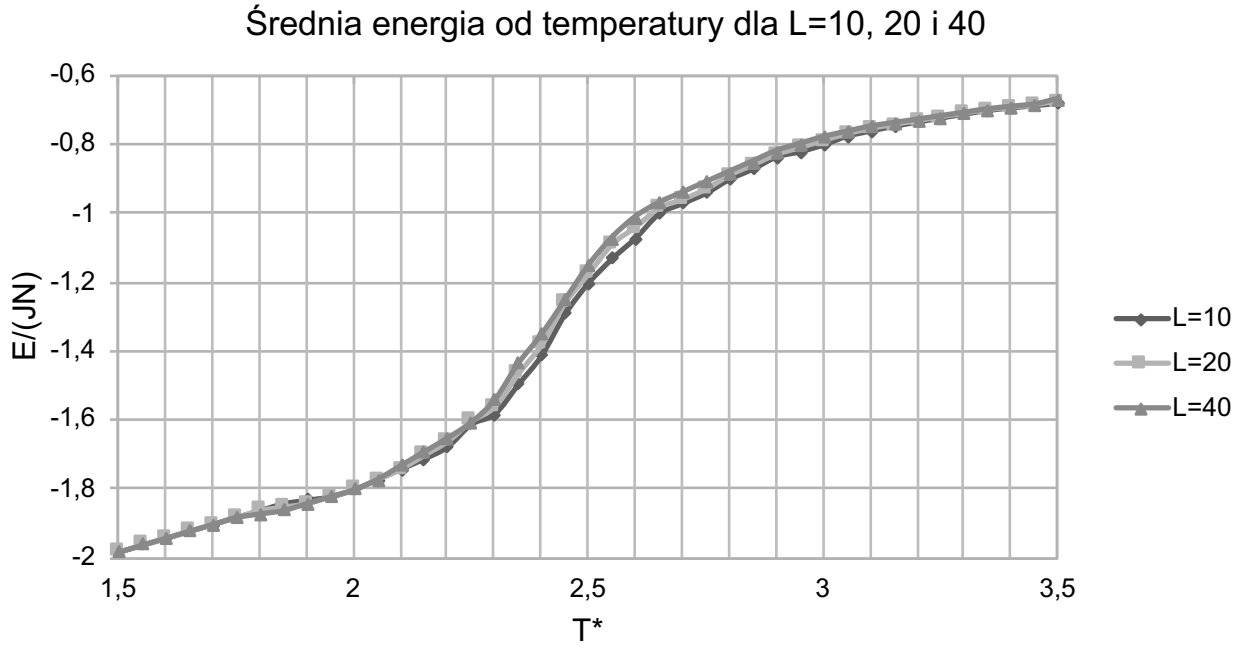

Namagnesowanie dla kolejnych kroków symulacji, dla układu 10 × 10 węzłów i temperatury T* = 1.7
1,5
1
0,5
0
-0,5
-1
-1,5
magnetyzacja
0
10000000
20000000
30000000
40000000
50000000
MCS



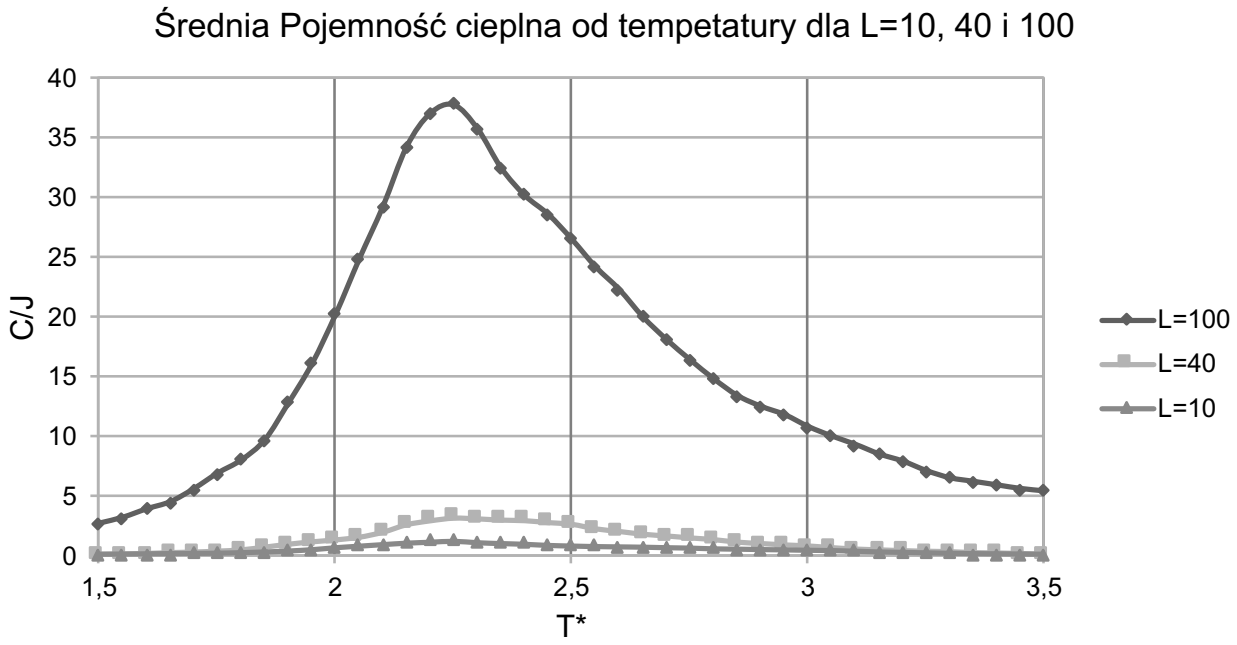
Średnia magnetyzacja w zależności od temperatury dla L=10, 40 i 100

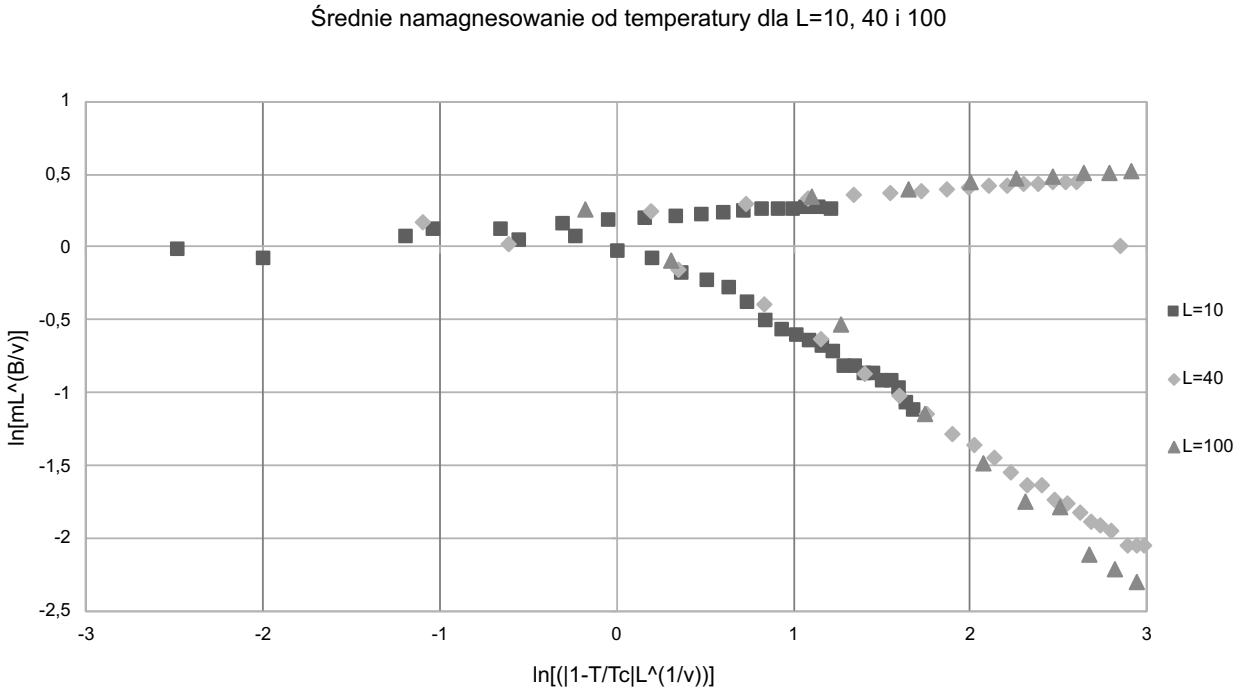


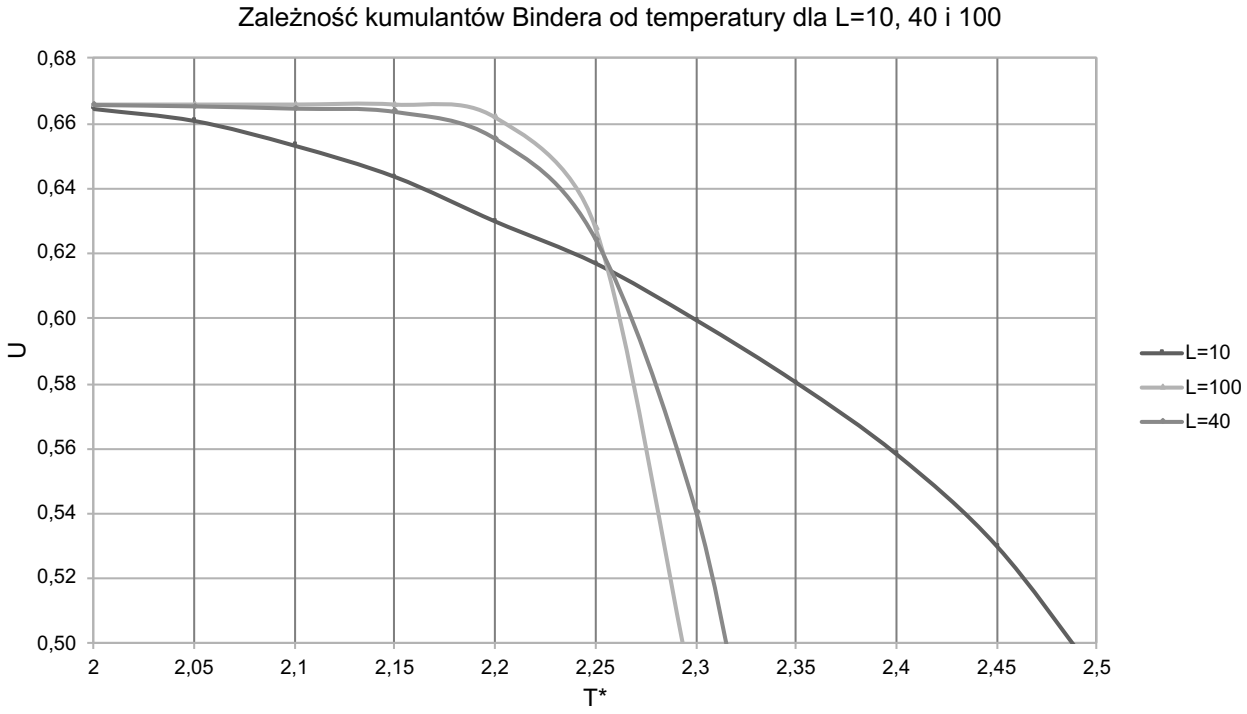




Dla niskich temperatur średnia energia dąży do wartości E=-2, co oznacza, że wszystkie domeny mają ten sam zwrot wektora pola magnetycznego.


Średnia Pojemność cieplna od tempetatury dla L=10, 40 i 100
C/J
T*
40
35
30
25
20
15
10
5
0
1,5
2
2,5
3
3,5
L=100
L=40
L=10


Średnie namagnesowanie od temperatury dla L=10, 40 i 100
ln[mL^(B/v)]
ln[(|1-T/Tc|L^(1/v))]
1
0,5
0
-0,5
-1
-1,5
-2
-2,5
-3
-2
-1
0
1
2
3
L=10
L=40
L=100



Miejsce przecięcia znajduje się w okolicach temperatury T*=2,255, więc jest to temperatura krytyczna.

Mateusz Bulanda - Gorol

```
//
//  main.cpp
//  Model_Isinga
//
//  Created by Mateusz on 04/05/2019.
//  Copyright © 2019 Mateusz Bulanda - Gorol. All rights reserved.
//

#include <iostream>
#include <random>
#include <chrono>
#include <math.h>
#include <cstdlib>
#include <fstream>
#include <stdlib.h>

#define L 40



using namespace std;

int spin[L][L];

//int spinS[L][L][4];

int energy;
int energiaU;

//double czynnikBoltzmanna [5];

void energia( int i, int j);
void pokaz();
double namagnesowanie();
void U ();




int main()
{
    //mt19937
generator(chrono::high_resolution_clock::now().time_since_epoch().co
unt());

  //  std::uniform_real_distribution<double> distribution(0.0, 1.0);

    srand(time(NULL));

    fstream plik("namagnesowanie.txt", ios::out | ios::trunc);
    fstream plik_1("energia.txt", ios::out | ios::trunc);

    plik.open("namagnesowanie.txt");
    plik_1.open("energia.txt");
```

```
    double Tz = 1.7;

    double exp8 = exp((-8)/(Tz));
    double exp4 = exp((-4)/(Tz));
    double exp0 = exp((0)/(Tz));

    for (int i=0; i<L; i++)
    {
        for (int j=0; j<L; j++)
        {
            /*
            int xl=i-1;
            int xp=i+1;
            int yd=j-1;
            int yg=j+1;
             */

            /*if (xl<0)
            {
                xl=L+xl;
            }
            if (xp<0)
            {
                xp=L+xp;
            }
            if (yd<0)
            {
                yd=L+yd;
            }
            if (yg<0)
            {
                yg=L+yg;
            }*/

            int s=1;
            int q=rand()%2;
           // if (distribution(generator) > 0.5)
            if (q==1)
            {
                s=-1;
            }

            spin[i][j]=s;

            /*
            spinS[i][j][0]=xl%L;
            spinS[i][j][1]=xp%L;
            spinS[i][j][2]=yd%L;
            spinS[i][j][3]=yg%L;
             */

            //cout<<spinS[i][j][0]<<"\t"<<spinS[i][j]
[1]<<"\t"<<spinS[i][j][2]<<"\t"<<spinS[i][j][3]<<endl;
```

```
        }
    }


    int mcs = 0;
    int limit = 3000000;
    int kontrola = 1000;
    for (int t=1.5; t<3,5; t=t+0,05)
    {
    while (mcs < limit)
    {
        mcs++;
        for (int i=0; i<L; i++)
        {
            for (int j=0; j<L; j++)
            {
                energia(i, j);
                int delta_E = energy;
                if (delta_E < 0)
                {
                    spin[i][j] *= (-1);
                }
                else
                {
                    double w;

                    if (delta_E == 8)
                    {
                        w = exp8;
                    }
                    else
                    {
                        if (delta_E == 4)
                        {
                            w = exp4;
                        }
                        else
                        {
                            w = exp0;
                        }
                    }


                    double losowa = (((double)rand())/
(RAND_MAX)); //distribution(generator);

                    if (w>=losowa)
                    {
                        spin[i][j] *= (-1);
                    }
                }
            }
        }
        if (mcs % kontrola == 0 && mcs >= 30000)
```

```
        {
            //pokaz();
            U ();
            float M=namagnesowanie();
            plik<<M<<endl;
            plik_1<<energiaU<<endl;
        }
    }
}

    plik.close();
    plik_1.close();
    return 0;

}




void pokaz(){

    for(int i=0;i<L;i++)
    {

        for(int j=0;j<L;j++)
        {
            if(spin[i][j]==1)
            {
                cout<<"■"<<" ";
            }
            else
            {
                cout<<" "<<" ";
            }

        }
        cout<<endl;
    }

}
double namagnesowanie()
{

    float suma =0;

    for(int i=0;i<L;i++)
    {

        for(int j=0;j<L;j++){

            suma+=spin[i][j];
        }
```

```
    }

    //cout<< suma/L/L<<endl;
    return suma;
}

void energia( int i, int j)
{

    /*
    int gornyN =spinS[i][j][3];
    int dolnyN =spinS[i][j][2];
    int lewyN =spinS[i][j][0];
    int prawyN = spinS[i][j][1];
     */

    //cout<<"pkt=("<<i<<","<<j<<"):
"<<"górny:"<<gornyN<<"dolny:"<<dolnyN<<"lewy:"<<lewyN<<"prawy:"<<pra
wyN<<endl<<endl;

    //int energy = (2) * (spin[i][j]) * (spin[lewyN][j] +
spin[prawyN][j] + spin[i][gornyN] + spin[i][dolnyN]);

    energy = 2 * (spin[i][j]) * (spin[((i-1)%L)][j] +
spin[((i+1)%L)][j] + spin[i][((j-1)%L)] + spin[i][((j+1)%L)]);

    //cout<<energy<<endl;

    //return energy;
}


void U ()
{
    energiaU=0;
    for (int i=0; i<L; i++)
    {
        for (int j=0; j<L; j++)
        {
            energiaU = energiaU + ((spin[i][j]) * (spin[((i-1)%L)]
[j] + spin[((i+1)%L)][j] + spin[i][((j-1)%L)] + spin[i]
[((j+1)%L)]));
        }
    }
}
```