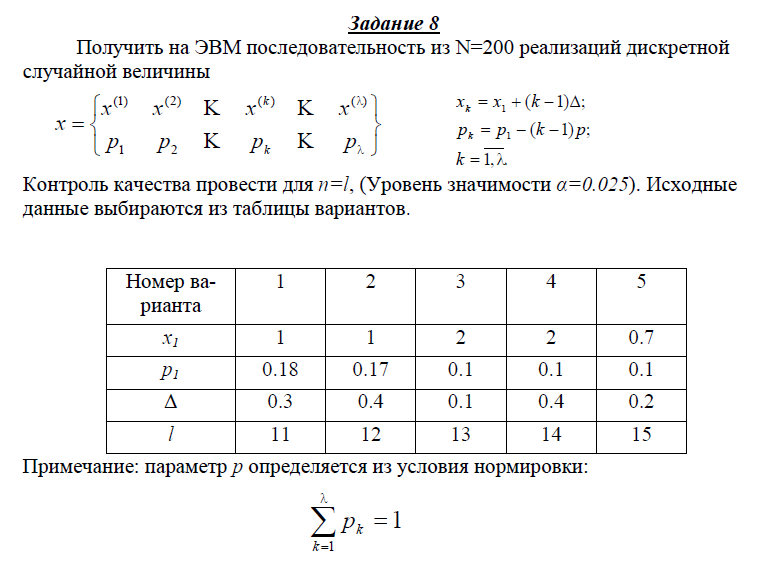
МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ  
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ  
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
САМАРСКИЙ УНИВЕРСИТЕТ

Отчёт по лабораторному практикуму  
по курсу «Моделирование информационно-вычислительных систем»

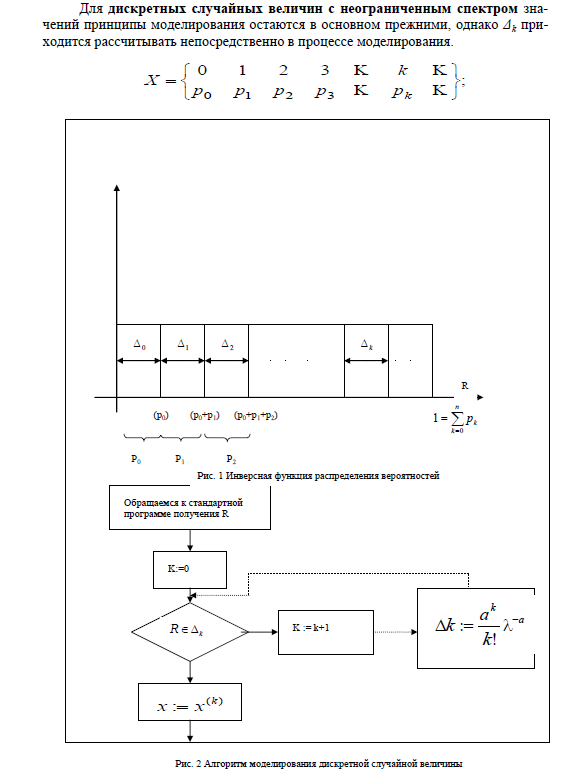
Вариант №8

Выполнил:  
Власов Георгий  
гр.6303  
  
Проверила:  
Симонова Е.В.

Самара 2019



***Метод получения случайной величины***



***Листингпрограммы***

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Laba\_1

{

class RandomValue

{

private List<double> listX;

private List<double> listP;

private List<double> listIntervalP;

private List<double> listOfSelection;

private double valueOfP;

private int[] arrayOfX;

private int intervalAmount;

private int countOfElements;

private double lengthOfInterval;

public List<double> ListP {

get => listP; set => listP = value;

}

public List<double> ListX {

get => listX; set => listX = value;

}

public List<double> Selection {

get => listOfSelection; set => listOfSelection = value;

}

public List<double> ListDelP {

get => listIntervalP; set => listIntervalP = value;

}

public int[] ArrayOfX {

get => arrayOfX; set => arrayOfX = value;

}

public double ValueOfP {

get => valueOfP; set => valueOfP = value;

}

public RandomValue(double FirstElementX, double FirstElementP, double LengthOfInterval, int IntervalAmount, int CountOfElements)

{

listP = new List<double>() { FirstElementP };

listX = new List<double>() { FirstElementX };

listIntervalP = new List<double>();

countOfElements = CountOfElements;

intervalAmount = IntervalAmount;

lengthOfInterval = LengthOfInterval;

valueOfP = CalculateP(IntervalAmount);

CalculateXPLengthOfInterval();

CalculateOfInterval();

}

private double CalculateXk(double K)

{

return listX[0] + (K - 1) \* lengthOfInterval;

}

private double CalculatePk(double K)

{

return listP[0] - (K - 1) \* valueOfP;

}

private double CalculateP(int countOfInterval)

{

double summa = 0;

for (int i = 2; i <= countOfInterval; i++)

summa += (i - 1);

return (countOfInterval \* listP[0] - 1.0) / summa;

}

public void CalculateXPLengthOfInterval()

{

ListDelP.Add(0);

ListDelP.Add(listP[0]);

for (int k = 2; k <= intervalAmount; k++)

{

listX.Add(CalculateXk(k));

listP.Add(CalculatePk(k));

ListDelP.Add(ListDelP[ListDelP.Count - 1] + listP[listP.Count - 1]);

}

}

public void CalculateOfInterval()

{

Random random = new Random();

arrayOfX = new int[ListX.Count];

Selection = new List<double>();

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

{

listOfSelection.Add(random.NextDouble());

for (int j = 0; j < ListP.Count; j++)

{

if (listOfSelection[listOfSelection.Count - 1] < ListDelP[j + 1])

{

arrayOfX[j]++;

break;

}

}

}

}

public string CalculateHiSquare()

{

double hiSquare = intervalAmount / (double)countOfElements;

double summaOfCount = 0;

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

{

summaOfCount += ArrayOfX[i] \* ArrayOfX[i];

}

hiSquare = hiSquare \* summaOfCount - countOfElements;

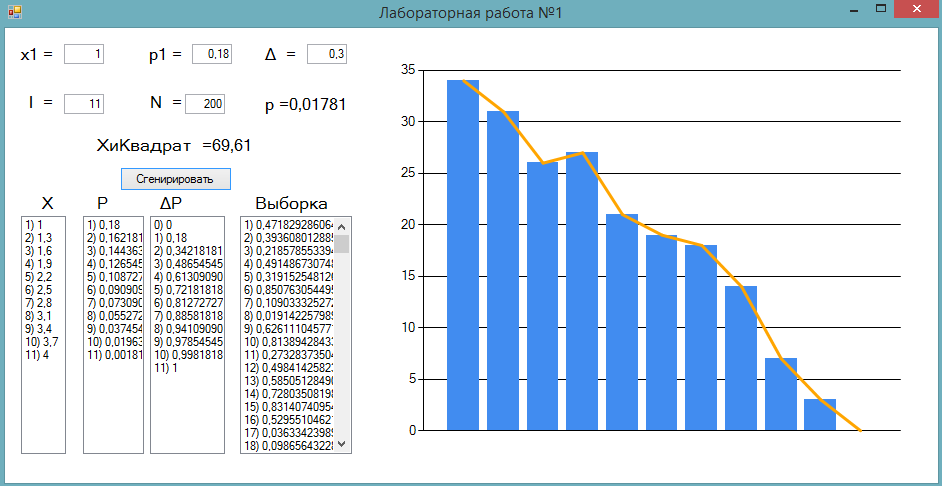
return hiSquare.ToString();

}

}

}

***Результат выполнения:***

******

***Вывод:***

На основании сравнения хи-квадрат с порогом *y0*,можно сделать вывод о том, что проверяемая гипотеза не может быть отвергнута.