**Національний Технічний Університет України**

**«Київський Політехнічний Інститут»**

**Факультет інформатики та обчислювальної техніки**

**Кафедра обчислювальної техніки**

**Лабораторна робота №4**

**Перевірка гіпотез за критерієм χ2**

|  |  |
| --- | --- |
| **Прийняв**  **Доц. Марковський О.П.**  **«\_\_»\_\_\_\_\_\_\_\_ 2014 р.** | **Виконала Студентка 2-ого курсу ФІОТ**  **групи ІО-32**  **Шапран К.О.** |

**Варіант: сума 25-Сімпсонів**

**Лістинг**

**import java.util.Arrays;**

**import java.util.Random;**

**import org.apache.commons.math3.special.\*;**

**public class Main {**

**public static void main(String[] args){**

**int length1 = 1000;**

**int ni = 10;//number of intervals**

**double m = 3;**

**double a = 2;**

**double[] arr = new double[length1];**

**for (int i = 0; i < arr.length; i++){**

**arr[i] = *getSequence*(m,a);**

**}**

**Arrays.*sort*(arr);**

**System.*out*.print("Length of sequence: "+ arr.length + "; Number of intervals: " + ni);**

**double me = *getMeanValue*(arr);**

**double sigma = *getStandardDeviation*(arr,me);**

**double[] limits = *getLimits*(arr,ni,100);**

**int[] mi = *getMi*(arr,limits);**

**double[] pi = *getProbabilities*(limits,ni,me,sigma);**

**int summi = 0;**

**double sumProb = 0;**

**double chiSquared1 = *getChiSquared*(mi,pi,length1);**

**System.*out*.print("\nChiSquared "+ chiSquared1);**

**System.*out*.println("\nProbability " + *getProbability*(ni,chiSquared1));**

**System.*out*.println();**

**int length2 = 10;**

**int l = 3;**

**double[] sequence = {76.1, 75, 71.3, 73.2, 72, 72.8, 74.5, 75.6, 77.9, 69.9};**

**Arrays.*sort*(sequence);**

**for (int i = 0; i< sequence.length;i++){**

**System.*out*.print(sequence[i] + " ");**

**}**

**double me2 = *getMeanValue*(sequence);**

**double sigma2 = *getStandardDeviation*(sequence,me2);**

**double[] limits2 = *getLimits*(sequence,l,4);**

**int[] mi2 = *getMi*(sequence,limits2);**

**double[] pi2 = *getProbabilities*(limits2,l,me2,sigma2);**

**double chiSquared2 = *getChiSquared*(mi2,pi2,length2);**

**System.*out*.print("\nChiSquared "+chiSquared2);**

**System.*out*.println("\nProbability " + *getProbability*(l,chiSquared2));**

**}**

**private static double getSequence(double m, double a){**

**Random rnd = new Random();**

**double r,k,s;**

**double sum = 0;**

**for (int g = 0; g < 25; g++){**

**r = rnd.nextDouble();**

**if (r < 0.5){**

**k = (double) (a \* Math.*sqrt*(r/ 0.5));**

**s = m - a + k;**

**}**

**else {**

**k = (double) (a \* Math.*sqrt*((1-r)/0.5));**

**s = m + a - k;**

**}**

**sum += s;**

**}**

**return sum;**

**}**

**private static double getMeanValue (double[] arr){**

**double sum = 0;**

**for (int i = 0; i < arr.length; i++){**

**sum+= arr[i];**

**}**

**return sum/arr.length;**

**}**

**private static double getStandardDeviation (double[] arr, double meanVal){**

**double variance = 0;**

**for (int i = 0; i < arr.length; i++)**

**variance +=(double) (Math.*pow*((arr[i] - meanVal), 2));**

**double v = variance/arr.length;**

**return Math.*sqrt*(v);**

**}**

**private static double[] getLimits(double[] arr, int numberOfIntervals, int st){**

**Arrays.*sort*(arr);**

**double [] limits = new double [numberOfIntervals+1];**

**limits[0] = arr[0];**

**int step = st-1;**

**int s = step;**

**for(int i = 1; i < limits.length-1; i++){**

**limits[i] = (double) arr[s];**

**s+=step;**

**}**

**limits[limits.length-1] = arr[arr.length-1];**

**return limits;**

**}**

**private static int[] getMi(double[] arr, double[] limits){**

**int [] numberOfValues = new int[limits.length-1];**

**int k = 0;**

**for(int i = 0; i < arr.length; i++){**

**if(arr[i] >= limits[k] && arr[i] <= limits[k+1])**

**numberOfValues[k]++;**

**else {**

**k++;**

**numberOfValues[k]++;**

**}**

**}**

**return numberOfValues;**

**}**

**private static double[] getProbabilities(double[] limits, int numberOfIntervals, double m, double s){**

**double[] probability = new double[numberOfIntervals];**

**probability[0]=1.0/2\*Erf.*erf*((limits[1]-m)/(s\*Math.*sqrt*(2)))+0.5;**

**for (int i = 1; i < limits.length-2; i++){**

**probability[i] = 1.0/2\*Erf.*erf*((limits[i] - m)/(s\*Math.*sqrt*(2))) - 1.0/2\*Erf.*erf*((limits[i-1] - m)/(s\*Math.*sqrt*(2)));**

**}**

**probability[limits.length-2]=0.5 - 1.0/2\*Erf.*erf*((limits[limits.length-2]-m)/(s\*Math.*sqrt*(2))) ;**

**return probability;**

**}**

**private static double getChiSquared(int[] mi, double[] pi, int n){**

**double chiSquared = 0;**

**for(int i = 0; i < mi.length; i++){**

**double r = n\*pi[i];**

**chiSquared += Math.*pow*(mi[i]-r,2)/r;**

**}**

**return chiSquared;**

**}**

**private static double getProbability(int numberOfIntervals, double chiSquared){**

**int l = numberOfIntervals - 1;**

**double probability = 0;**

**int k = 0;**

**double chi = 0;**

**double[][] table = {{0, 0.02, 0.04, 0.103, 0.211, 0.446, 0.713, 1.386, 2.41, 3.22, 4.6, 5.99, 7.82, 9.21, 13.82},**

**{0, 2.09 , 2.53, 3.32, 4.17, 5.38, 6.39, 8.34, 10.66, 12.24, 14.68, 16.92, 19.68, 21.7, 27.9}};**

**double[] prob = { 1, 0.99, 0.98, 0.95, 0.9, 0.8, 0.7, 0.5, 0.3, 0.2, 0.1, 0.05, 0.02, 0.01, 0.001};**

**if(l == 9){**

**for(int i = 0; i < table[1].length-1; i++)**

**if(chiSquared >= table[1][i] && chiSquared <= table[1][i+1]){**

**chi = table[1][i+1];**

**probability = prob[i+1];**

**}**

**}else**

**for(int i = 0; i < table[0].length-1; i++)**

**if(chiSquared >= table[0][i] && chiSquared <= table[0][i+1]){**

**chi = table[0][i+1];**

**probability = prob[i+1];**

**}**

**return probability;**

**}**

**}**

**Результат**

