НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ   
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»   
ФІЗИКО-ТЕХНІЧНИЙ ІНСТИТУТ

CИМЕТРИЧНА КРИПТОГРАФІЯ

КОМП’ЮТЕРНИЙ ПРАКТИКУМ №1

Експериментальна оцінка ентропії на символ джерела

відкритого тексту

Виконали:  
студенти групи ФІ-52  
Кратт Ярослав

Бурлака Марія  
Прийняв:   
Деркач О.Г.

**Мета роботи**

Засвоєння понять ентропії на символ джерела та його надлишковості, вивчення та порівняння різних моделей джерела відкритого тексту для наближеного визначення ентропії, набуття практичних навичок щодо оцінки ентропії на символ джерела.

**Постановка задачі**

Підрахувати частоти букв та біграм, а також підрахувати *H₁* та *H₂* в довільно обраному тексті розміром більше 1Мб. Одержати значення *H₁* та *H₂* на тому ж тексті, в якому вилучено всі пробіли. Використовуючи значення ентропії, отримані з програми CoolPinkProgram, оцінити надлишковість російської мови в різних моделях джерела.

**Хід роботи та труднощі**

Роботу було почато з написання програм для виконання завдань. Для реалізації було обрано мову Java, оскільки вона є більш зручною та гнучкою, ніж С++ (суб'єктивно). В реалізації було неуважно використано одну зі структур даних, а саме TreeSet (деякі частоти співпадали(вони були ключами в TeeSet), що неможливо в TreeSet). Це призвело до перенаписання всього майже з нуля. Друга проблема, з якою ми зіштовхнулись, це log*₂* (0) при обрахунку *H₁* та *H₂.* Зрозуміло, що з доданками, які містять log*₂* (0), коректного результату отримати неможливо. Тому в сумі доданки, в яких був log*₂* (0), прирівнювались до нуля.

**Лістинг коду**

**public class CryptoLab1 {**

**private final static char alphabet1[] ={'а','б','в','г','д','е','ё','ж','з','и','й','к','л','м','н','о','п','р','с','т','у','ф','х','ц','ч','ш','щ','ъ','ы','ь','э','ю','я',' '};**

**private final static char alphabet2[] ={'а','б','в','г','д','е','ё','ж','з','и','й','к','л','м','н','о','п','р','с','т','у','ф','х','ц','ч','ш','щ','ъ','ы','ь','э','ю','я'};**

**public static void main(String[] args) throws UnsupportedEncodingException, IOException {**

**//Filter f=new Filter("text1.txt","filtered.txt");**

**//f.removeSpaces("filtered.txt","filteredWitoutSpaces.txt");**

**Frequency fr1=new Frequency();**

**fr1.symbolFreq("filtered.txt", 33);**

**fr1.printTable(fr1.bigramFreq1("filtered.txt"),alphabet1);**

**fr1.printTable(fr1.bigramFreq2("filtered.txt"),alphabet1);**

**System.out.println("\\n Для текста без пробелов:");**

**fr1.symbolFreq("filteredWitoutSpaces.txt",32);**

**fr1.printTable(fr1.bigramFreq1("filteredWitoutSpaces.txt"),alphabet2);**

**fr1.printTable(fr1.bigramFreq2("filteredWitoutSpaces.txt"),alphabet2);**

**}**

**}**

**/**

**public class Filter {**

**public Filter(String inFileName, String outFileName) throws FileNotFoundException, UnsupportedEncodingException, IOException{**

**String path="D://STUDY//крипта//cryptoLab1//"+inFileName;**

**BufferedReader br = new BufferedReader(new InputStreamReader(new FileInputStream(path),"CP1251"));**

**FileWriter writer = new FileWriter("D://STUDY//крипта//cryptoLab1//"+outFileName, true);**

**String line,bigLine="";**

**while((line = br.readLine()) != null){**

**bigLine=bigLine+line;**

**}**

**bigLine=bigLine.replaceAll("[^А-Яа-я]", " ");**

**bigLine=bigLine.replaceAll("\\s+", " ");**

**bigLine=bigLine.toLowerCase();**

**bigLine=bigLine+"\n";**

**writer.write(bigLine);**

**writer.close();**

**}**

**void removeSpaces(String inFileName, String outFileName) throws FileNotFoundException, UnsupportedEncodingException, IOException**

**{**

**String path="D://STUDY//крипта//cryptoLab1//"+inFileName;**

**BufferedReader br = new BufferedReader(new InputStreamReader(new FileInputStream(path),"UTF8"));**

**FileWriter public class Filter {**

**public Filter(String inFileName, String outFileName) throws FileNotFoundException, UnsupportedEncodingException, IOException{**

**String path="D://STUDY//крипта//cryptoLab1//"+inFileName;**

**BufferedReader br = new BufferedReader(new InputStreamReader(new FileInputStream(path),"CP1251"));**

**FileWriter writer = new FileWriter("D://STUDY//крипта//cryptoLab1//"+outFileName, true);**

**String line,bigLine="";**

**while((line = br.readLine()) != null){**

**bigLine=bigLine+line;**

**}**

**bigLine=bigLine.replaceAll("[^А-Яа-я]", " ");**

**bigLine=bigLine.replaceAll("\\s+", " ");**

**bigLine=bigLine.toLowerCase();**

**bigLine=bigLine+"\n";**

**writer.write(bigLine);**

**writer.close();**

**}**

**void removeSpaces(String inFileName, String outFileName) throws FileNotFoundException, UnsupportedEncodingException, IOException**

**{**

**String path="D://STUDY//крипта//cryptoLab1//"+inFileName;**

**BufferedReader br = new BufferedReader(new InputStreamReader(new FileInputStream(path),"UTF8"));**

**FileWriter writer = new FileWriter("D://STUDY//крипта//cryptoLab1//"+outFileName, true);**

**String line,bigLine="";**

**while((line = br.readLine()) != null){**

**bigLine=bigLine+line;**

**}**

**bigLine=bigLine.replaceAll(" ", "");**

**bigLine=bigLine+"\n";**

**writer.write(bigLine);**

**writer.close();**

**}**

**}writer = new FileWriter("D://STUDY//крипта//cryptoLab1//"+outFileName, true);**

**String line,bigLine="";**

**while((line = br.readLine()) != null){**

**bigLine=bigLine+line;**

**}**

**bigLine=bigLine.replaceAll(" ", "");**

**bigLine=bigLine+"\n";**

**writer.write(bigLine);**

**writer.close();**

**}**

**}**

**public class Frequency {**

**private String path,line;**

**private BufferedReader br;**

**private char arr[];**

**private char alphabet[];**

**private Result[] res;**

**private int resLength;**

**public Frequency() throws FileNotFoundException, UnsupportedEncodingException**

**{**

**int k=0;**

**resLength=0;**

**}**

**void symbolFreq(String fileName,int alphabetSize) throws IOException**

**{ path = "D://STUDY//крипта//cryptoLab1//"+ fileName;**

**br = new BufferedReader(new InputStreamReader(new FileInputStream(path),"UTF8"));**

**TreeMap<Character,Integer> mapOfFrequency =new TreeMap<Character, Integer>();**

**while((line=br.readLine())!=null)**

**{**

**arr=line.toCharArray();**

**for(char symbol :arr)**

**{**

**if(mapOfFrequency.get(symbol)==null)**

**{**

**mapOfFrequency.put(symbol, 1);**

**}**

**else**

**{**

**mapOfFrequency.put(symbol,mapOfFrequency.get(symbol)+1);**

**}**

**}**

**}**

**TreeMap<Integer,Character> sorted =new TreeMap< Integer, Character>();**

**for(Map.Entry<Character,Integer> item : mapOfFrequency.entrySet()){**

**sorted.put(item.getValue(), item.getKey());**

**}**

**resLength=0;**

**SymbolResult res []=new SymbolResult [alphabetSize];**

**int i=0; int sum=0; double H1=0; double temp;**

**for(Map.Entry<Integer,Character> item : sorted.entrySet()) //для красивого вывода перегоняем всё в массивчик результирующего класса, с которым, в теории, удобнее работать**

**{**

**res[i]=new SymbolResult( item.getKey(),item.getValue());**

**sum=sum+ item.getKey();**

**// System.out.println(i+" "+res[i].getCount()+" "+ res[i].getBigramma());**

**i++;**

**}**

**for(int j=res.length-1; j>=0;j--)**

**{**

**temp=(double)res[j].getCount()/sum;**

**H1= (H1+temp\*log2(temp));**

**System.out.printf("%.4f",((float)res[j].getCount()/sum)\*100);**

**System.out.println(" "+res[j].getCharacter());**

**}**

**H1=-H1;**

**System.out.println("H1="+H1);**

**}**

**static double log2(double x )**

**{**

**double res = 0;**

**if(x!=0)**

**{**

**res= (log(x)/log(2));**

**return res;**

**}**

**else**

**return 0;**

**}**

**Result[] bigramFreq1(String fileName) throws IOException**

**{**

**path = "D://STUDY//крипта//cryptoLab1//"+fileName;**

**br = new BufferedReader(new InputStreamReader(new FileInputStream(path),"UTF8"));**

**String allText="";**

**while((line=br.readLine())!=null)**

**{**

**allText=allText+line;**

**}**

**arr= allText.toCharArray();**

**String tempStr;**

**TreeMap<String,Integer> map=new TreeMap<String,Integer>();**

**for(int i=1; i<arr.length; i++)**

**{**

**tempStr="";**

**tempStr= Character.toString(arr[i-1])+ Character.toString(arr[i]);**

**if(map.get(tempStr)==null)**

**{**

**map.put(tempStr,1);**

**}**

**else**

**{**

**map.put(tempStr,map.get(tempStr)+1);**

**}**

**}**

**int i=0;**

**resLength=0;**

**Result res []=new Result [1156];**

**for(Map.Entry<String,Integer> item : map.entrySet()) //для красивого вывода перегоняем всё в массивчик результирующего класса, с которым, в теории, удобнее работать**

**{**

**res[i]=new Result( item.getValue(),item.getKey());**

**// System.out.println(i+" "+res[i].getCount()+" "+ res[i].getBigramma());**

**resLength++;**

**i++;**

**}**

**System.out.println("Пары букв, которые пересекаются");**

**return res;**

**}**

**Result[] bigramFreq2(String fileName) throws IOException**

**{**

**path = "D://STUDY//крипта//cryptoLab1//"+ fileName;**

**br = new BufferedReader(new InputStreamReader(new FileInputStream(path),"UTF8"));**

**String allText="";**

**while((line=br.readLine())!=null)**

**{**

**allText=allText+line;**

**}**

**arr= allText.toCharArray();**

**String tempStr;**

**TreeMap<String,Integer> map=new TreeMap<String,Integer>();**

**for(int i=1; i<arr.length; i=i+2)**

**{**

**tempStr="";**

**tempStr= Character.toString(arr[i-1])+ Character.toString(arr[i]); //засовуем биграммы и кол-во их вхождений в TreeMap, где ключ это биграмма**

**if(map.get(tempStr)==null)**

**{**

**map.put(tempStr,1);**

**}**

**else**

**{**

**map.put(tempStr,map.get(tempStr)+1);**

**}**

**}**

**int i=0;**

**resLength=0;**

**Result res []=new Result [1156];**

**for(Map.Entry<String,Integer> item : map.entrySet()) //для красивого вывода перегоняем всё в массивчик результирующего класса, с которым, в теории, удобнее работать**

**{**

**res[i]=new Result( item.getValue(),item.getKey());**

**// System.out.println(i+" "+res[i].getCount()+" "+ res[i].getBigramma());**

**resLength++;**

**i++;**

**}**

**/\*int k=0;**

**for(Map.Entry<Integer,String> item : sorted.entrySet()){**

**k++;**

**System.out.print(k+" ");**

**System.out.printf("Ключ: %d Значение: %s \n", item.getKey(), item.getValue());**

**}\*/System.out.println("Пары букв, которые не пересекаются");**

**return res;**

**}**

**void printTable(Result [] res, char [] alphabet )**

**{ int table[][]=new int [alphabet.length][alphabet.length];**

**int sum=0;**

**for(int i=0; i<resLength; i++)**

**{**

**sum=sum+res[i].getCount();**

**}**

**System.out.println("Всего биграмм "+sum);**

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

**{**

**for(int j=0; j<alphabet.length; j++)**

**{**

**for(int z=0; z<resLength; z++)**

**{ //System.out.println("i"+i+" j"+j+" z"+z);**

**if(res[z].getBigramma().toCharArray()[0]==alphabet[i]&&res[z].getBigramma().toCharArray()[1]==alphabet[j])**

**{**

**table[i][j]=res[z].getCount();**

**}**

**}**

**}**

**}**

**System.out.print("\t");**

**double H2=0,temp=0;**

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

**System.out.print(alphabet[i]+ "\t");**

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

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

**{**

**System.out.print(alphabet[i]+ "\t");**

**for(int j=0; j<alphabet.length; j++)**

**{**

**temp=(double)table[i][j]/sum;**

**H2= H2+temp\*log2(temp);**

**// System.out.print(H2+" ");**

**System.out.printf("%.4f",(((float)temp))); System.out.print("\t");**

**}**

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

**}**

**H2=-H2;**

**System.out.println("H2="+H2);**

**}**

**}**

**public class Result {**

**private final int count;**

**private final String bigramma;**

**Result(int count, String bigramma)**

**{**

**this.count=count;**

**this.bigramma=bigramma;**

**}**

**String getBigramma() {return bigramma;}**

**int getCount() {return count;}**

**}**

**public class SymbolResult {**

**private final int count;**

**private final Character symbol;**

**SymbolResult(int count, Character symbol)**

**{**

**this.count=count;**

**this.symbol=symbol;**

**}**

**Character getCharacter() {return symbol;}**

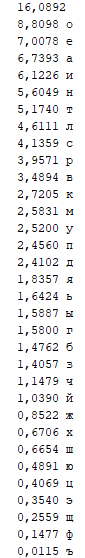
**int getCount() {return count;}**

**}**

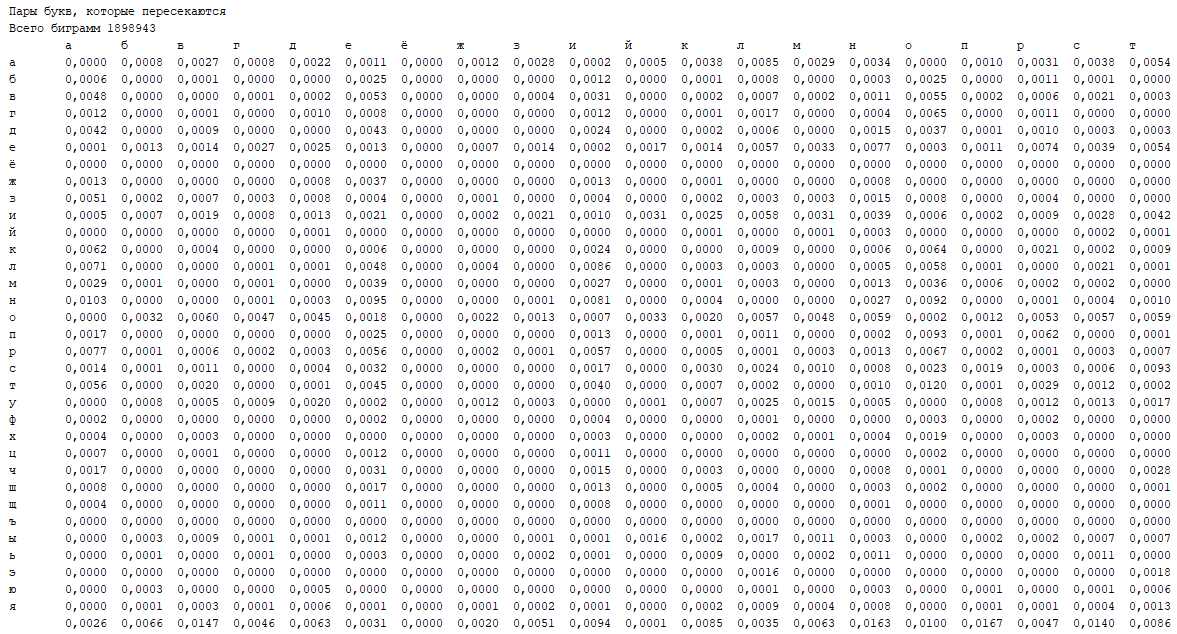
**Скріншоти та результати**

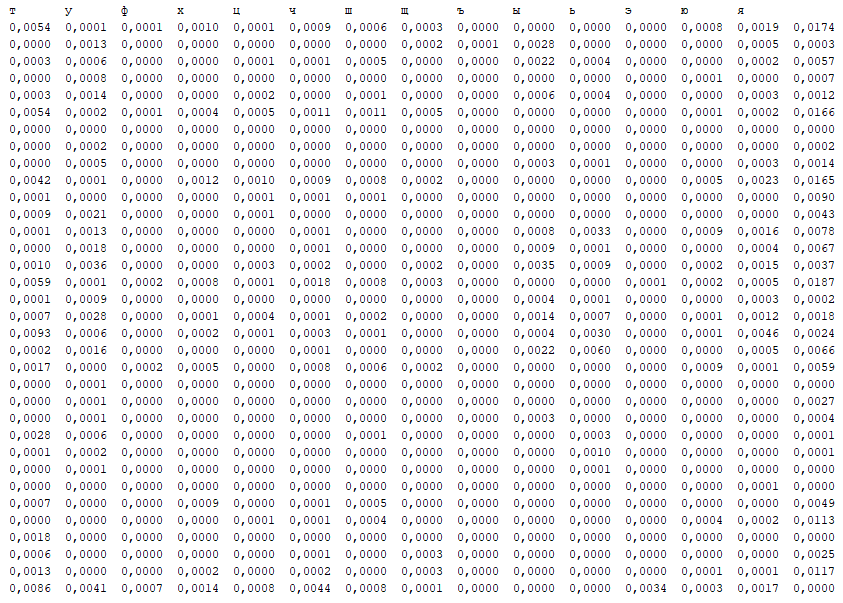
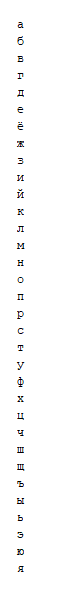
**Для відфільтрованого тексту з пробілами:**

**Частота символів:**

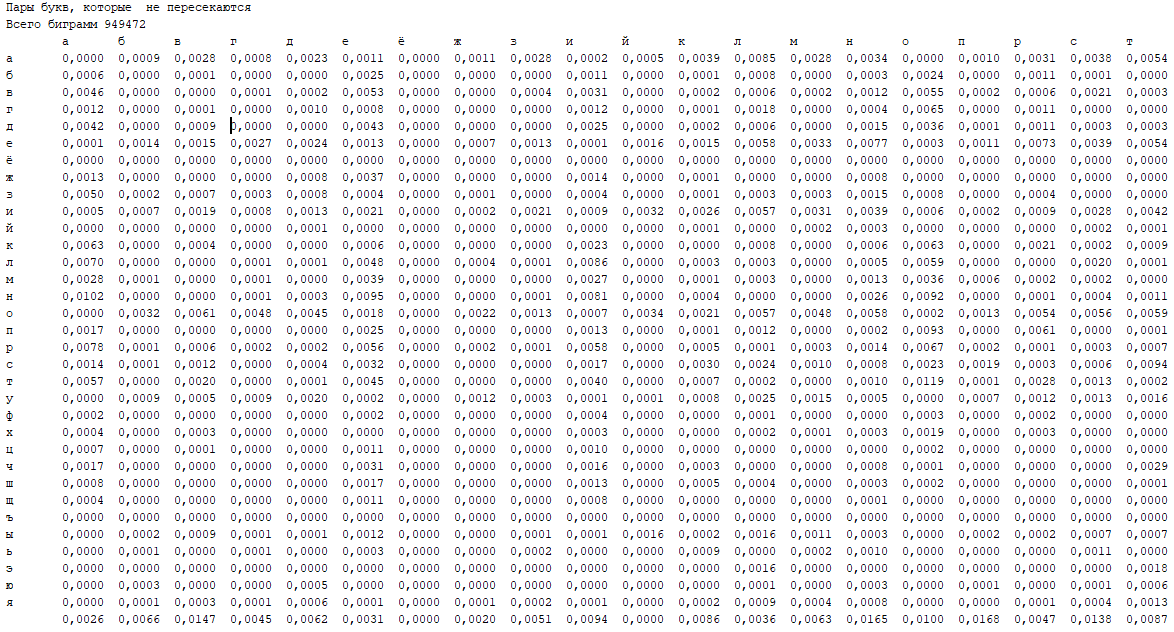


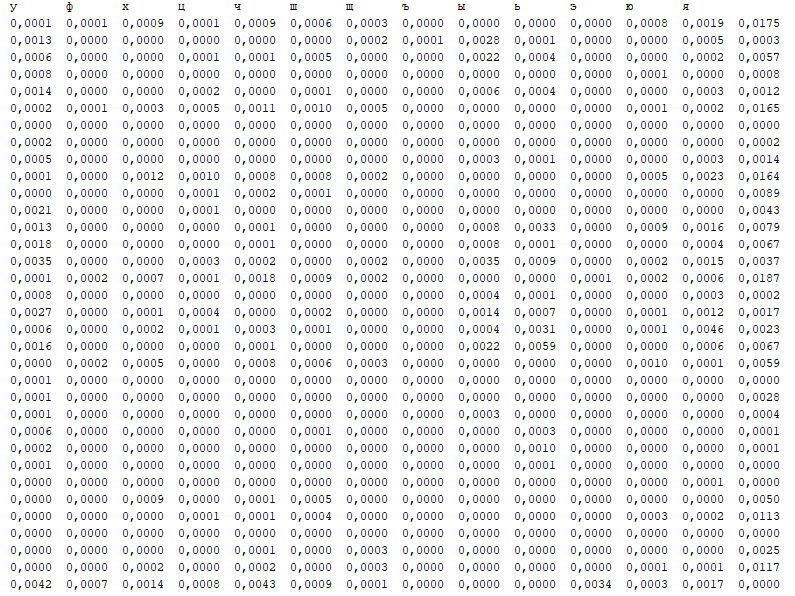
***H₁* =4.387750326017841**





***H₂* =7.989682583253761**

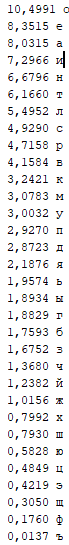


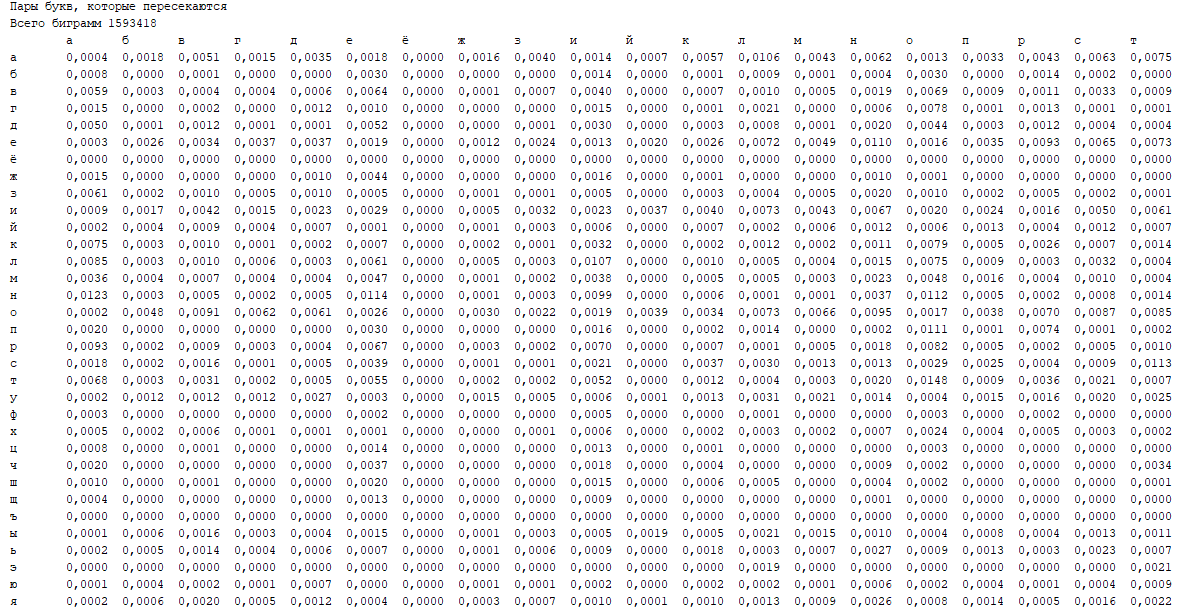
***H₂*=7.98886313586187**

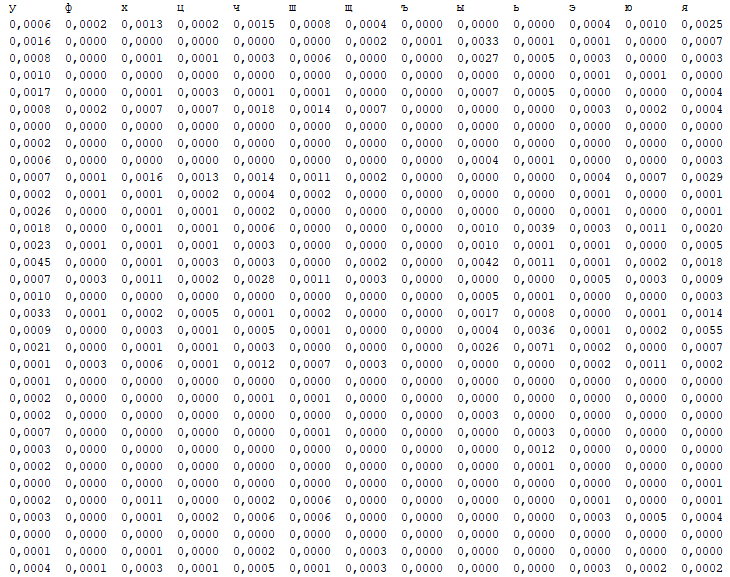
**Для відфільтрованого тексту без пробілів:**

**Частота символів:**

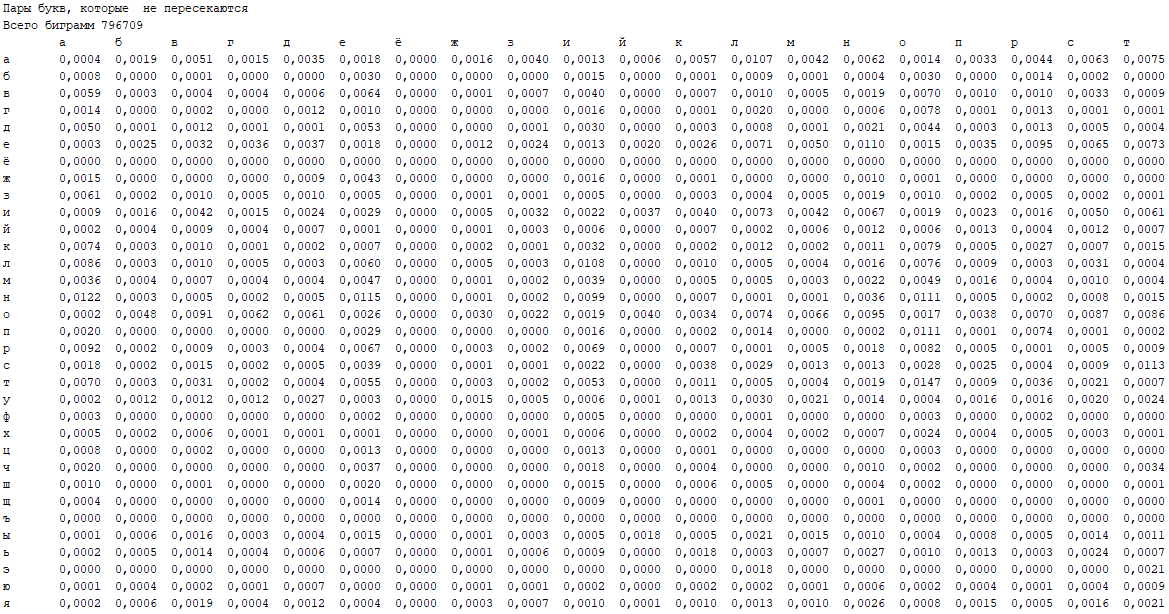


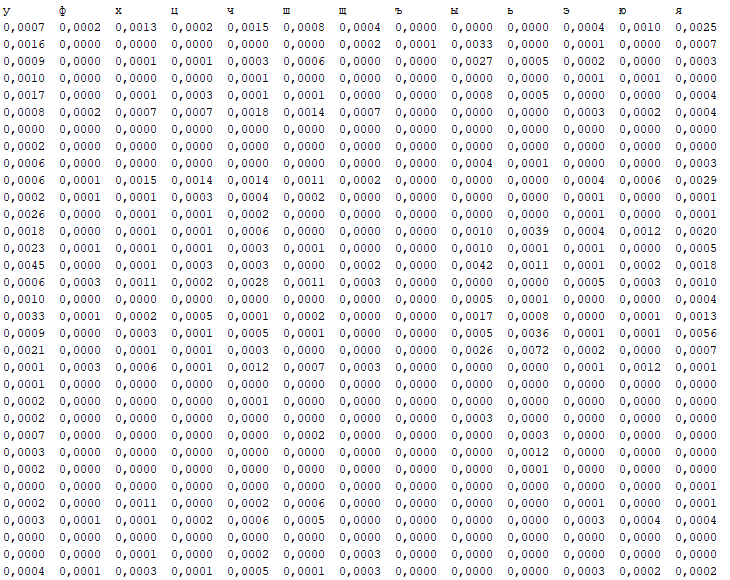
***H₁*=4.470594012197176**





***H₂*=** **8.33336221604355**





***H₂*=** **8.331785639222467**

(Скріншотів наведено не буде, оскільки розділ «Оформлення звіту» знаходиться після розділів «Методичні вказівки» та «Порядок роботи» )

***1.906869<H(10)<2.684623***

***1.698086<H (20)<2.436634***

***1.830808<H(30)<2.584585***

**Оцінка надлишковості R російської мови у різних моделях відкритого тексту**

R обчислюємо за формулою:



Hmax = log2 M, а M – число букв в алфавіті

Було отримано такі значення:

***0.46368<R(10)<0.61862***

***0.51267<R (20)<0.66666***

***0.48308<R(30)<0.63383***