

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

# КРИПТОГРАФІЯ

## КОМП'ЮТЕРНИЙ ПРАКТИКУМ №1

### Експериментальна оцінка ентропії на символ джерела відкритого тексту

Підготували студенти групи ФБ-23  
Марченко Родіон та Лотиш Андрій

Київ, Жовтень 2024

## 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.

## Порядок виконання роботи:

1. Написати програми для підрахунку частот букв і частот біграм в тексті, а також підрахунку  $H_1$  та  $H_2$  за безпосереднім означенням. Підрахувати частоти букв та біграм, а також значення  $H_1$  та  $H_2$  на довільно обраному тексті російською мовою достатньої довжини (щонайменше 1Мб), де імовірності замінити відповідними частотами. Також одержати значення  $H_1$  та  $H_2$  на тому ж тексті, в якому вилучено всі пробіли.
2. За допомогою програми CoolPinkProgram оцінити значення  $H^{(10)}$ ,  $H^{(20)}$ ,  $H^{(30)}$ .
3. Використовуючи отримані значення ентропії, оцінити надлишковість російської мови в різних моделях джерела.

### 1. Напишу необхідну програму для підрахунку на мові Python.

Дана програма приймає текст російською (в коді є підтримка і для української) мови у форматі .txt, назву папки для зберігання результатів та параметр наявності пробілів у підрахунку.

Реалізовано функціонал для:

- ◆ Попередньої очистки файлу від символів, що не входять у базовий алфавіт мови.
- ◆ Обрахунку частот букв та їх імовірностей, а також частот й імовірностей біграм з кроком 1 та 2.
- ◆ Обрахунку питомої ентропії на символ джерела  $H_1$  та  $H_2$ .
- ◆ Виводу результатів у консоль та окремі таблиці формату .csv.

### Початковий код програми (додатково надається у вигляді скриптового файлу):

```
# Marchenko Rodion Cryptography lab №1

import math
import os.path
import sys
import pandas as pd

BOLD = "\033[1m"
END = "\033[0m"
YELLOW = "\033[1;33m"

#This function turns a raw .TXT text file into a sequence of space-separated lowercase words
def PreprocessText(AllowedChars, InputFileName, OutputFileName, AllowNewLines = True):
    FormerChar = " "
    if os.path.isfile(InputFileName):
        with open(InputFileName, "r") as InputFile:
            with open(OutputFileName, "w") as OutputFile:
                Notfirst = True
                while True:
                    char = InputFile.read(1).lower()
                    if (AllowNewLines == False and char == "\n"): #Process newlines
                        char = " "
                    elif (char == "ё"): #Normalize characters
                        char = "e"
                    elif (char == "ё"):
                        char = "ё"
                    if (char in AllowedChars):
                        if ((char != " ") or (char == " " and FormerChar != " ")):
                            #Multiple spaces in a row prevention
                            OutputFile.write(char)
                            FormerChar = char
                    if not char:
                        break

                OutputFile.close()
            InputFile.close()
```

```

#This function calculates the number of occurrences and frequency in text of single
letters from CharArray
def CalculateSingleLetterFrequency(InputFileName, CharArray):

    ResultDict = {}
    Sum = 0
    for i in range(0, len(CharArray)):
        ResultDict.update({CharArray[i]: [0,0]})

    if (os.path.isfile(InputFileName)):
        with open(InputFileName, "r") as InputFile:
            while True:
                char = InputFile.read(1).lower()
                if (char in CharArray):
                    ResultDict.update({char: [ResultDict[char][0] + 1,0]})
                    Sum = Sum + 1

                if(Sum % 10 == 0):
                    print("Processing char № "+YELLOW+BOLD+str(Sum)+END+END, end='\r')

                if not char:
                    print("\n")
                    print(YELLOW+BOLD+"Processing of single characters
                    completed!" +END+END, end='\r')
                    break

            InputFile.close()
            for key in ResultDict.keys():
                Probability = round(ResultDict[key][0] / Sum, 8)
                ResultDict.update({str(key) : [ResultDict[key][0], Probability]})

            print("\n"+BOLD+"TOTAL:", Sum, "characters\n"+END)
    return ResultDict

```

```

#This function calculates the number of occurrences and frequency in text of bigrams of
letters from CharArray
# DoublePass = True - runs two passes with offset of 1 for higher accuracy of bigram
frequencies
def CalculateBigramFrequency(InputFileName, CharArray, DoublePass = True):

    ResultDict = {}
    Sum = 0
    for i in range(0, len(CharArray)):
        for j in range(0, len(AllowedChars)):
            ResultDict.update({AllowedChars[i]+AllowedChars[j]: [0,0]})

    if (os.path.isfile(InputFileName)):
        with open(InputFileName, "r") as InputFile:
            for i in range(0,2):
                InputFile.seek(0)
                if (i == 1):
                    InputFile.read(1)
                while True:
                    char = InputFile.read(2).lower()
                    #print(char)
                    if ((len(char) == 2) and (char[0] in CharArray) and (char[1] in
                    CharArray)):
                        ResultDict.update({char: [ResultDict[char][0] + 1,0]})
                        Sum = Sum + 1

                    if(Sum % 10 == 0):
                        print("Processing char № "+YELLOW+BOLD+str(Sum)+END+END,
                        end='\r')

                    if not char:
                        print(YELLOW+BOLD+"Processing of bigrams completed!" +END+END)
                        break

            if (DoublePass == False):
                break

```

```

        InputFile.close()
        for key in ResultDict.keys():
            Probability = round(ResultDict[key][0] / Sum, 8)
            ResultDict.update({str(key) : [ResultDict[key][0], Probability]})

        print("\n"+BOLD+"TOTAL:", Sum, "bigrams\n"+END)
    return ResultDict

#This function calculates the entropy of n-grams for given occurrence frequency dict with
structure of { N-gram:[NofOccurences, Frequency] }
def CalculateEntropy(FrequencyDict):
    EntropyOfN = 0
    for key, value in FrequencyDict.items():
        if(value[1] != 0):
            EntropyOfN = EntropyOfN + (value[1] * math.log(value[1], 2))

    EntropyOfN = -EntropyOfN / 2
    return EntropyOfN

#This function converts the frequency and probability data of bigrams from dict into 2d
Pandas dataframe
def FrequencyDict2dToDataframe(FrequencyDict, CharArray, Freq = True):
    if(Freq == False):
        val = 1
        fill = 0.0
    else:
        val = 0
        fill = 0

    df = pd.DataFrame(columns = CharArray)
    for i in range(0, len(CharArray)):
        df.loc[len(df)] = [fill]*len(CharArray)
    df.index = CharArray

    for key, value in FrequencyDict.items():
        if(value[val] != 0):
            df.at[key[0], key[1]] = value[val]
    return df

#This function converts frequency and probability data from dict into Pandas dataframe
and .CSV file
def FrequencyDictToCSV(OutputFile, FrequencyDictSingleChar, FrequencyDictBigram,
FrequencyDictBigramDouble, CharArray):
    pd.set_option("display.precision", 8)
    df11 = FrequencyDict2dToDataframe(FrequencyDictBigram, CharArray, True)
    df12 = FrequencyDict2dToDataframe(FrequencyDictBigram, CharArray, False)

    df21 = FrequencyDict2dToDataframe(FrequencyDictBigramDouble, CharArray, True)
    df22 = FrequencyDict2dToDataframe(FrequencyDictBigramDouble, CharArray, False)

    df3 = pd.DataFrame(columns = ["Frequency", "Probability"])
    for i in range(0, len(CharArray)):
        df3.loc[len(df3)] = [0, 0.0]
        df3 = df3.astype({"Frequency": "int", "Probability": "float"})
    df3.index = CharArray
    for key, value in FrequencyDictSingleChar.items():
        if(value[1] != 0):
            df3.at[key, "Frequency"] = value[0]
            df3.at[key, "Probability"] = value[1]

    if (OutputFile[-4:] == ".csv"):
        Name = OutputFile[:-4]
    else:
        Name = OutputFile

    print(BOLD+"Frequency of bigrams in text (single pass with step = 2):
\n"+END, df11, "\n")
    df11.to_csv(Name+"-Bg-Freq-Singlepass.csv")
    print(BOLD+"Probability of bigrams in text (single pass with step = 2):
\n"+END, df12, "\n")
    df12.to_csv(Name+"-Bg-Pro-Singlepass.csv")

    print(BOLD+"Frequency of bigrams in text (double pass with overlap):
\n"+END, df21, "\n")
    df21.to_csv(Name+"-Bg-Freq-Doublepass.csv")
    print(BOLD+"Probability of bigrams in text (double pass with overlap):
\n"+END, df22, "\n")
    df22.to_csv(Name+"-Bg-Pro-Doublepass.csv")
    print(BOLD+"Frequency and probability of single letters in text:
\n"+END, df3, "\n")
    df3.to_csv(Name+"-SingleLetters.csv")

```



```

#Driver code:
if (len(sys.argv) == 4):
    source = sys.argv[1]
    workdir = sys.argv[2]

    Exit = False
elif(len(sys.argv) != 4 or (sys.argv[0] == "-h")):
    print("Usage: Crypto-lab1.py <source text> <workdir> <count spaces {T/F}>\n")
    Exit = True

if (Exit == False):
    print(BOLD+"="*82)
    print("Text processing and letter frequency, probability and entropy
    calculation program.")
    print(("="*82)+END+"\n")

    if ((os.path.exists(workdir)) and (os.path.isfile(source))):

        #AllowedChars = ["a", "б", "в", "г", "д", "е", "є", "ж", "з", "и",
        "і", "й", "к", "л", "м", "н", "о", "п", "р", "с", "т", "у", "ф", "х",
        "ц", "ч", "ш", "щ", "ь", "ы", "ь", "э", "ю", "я"]
        AllowedChars = ["a", "б", "в", "г", "д", "е", "ж", "з", "и", "й", "к",
        "л", "м", "н", "о", "п", "р", "с", "т", "у", "ф", "х", "ц", "ч", "ш",
        "щ", "ы", "ь", "э", "ю", "я"]
        if ((sys.argv[3] == "T") or (sys.argv[3] == "t") or (sys.argv[3] ==
        "True") or (sys.argv[3] == "true")):
            AllowedChars.append(" ")

        PreprocessText(AllowedChars, source, workdir+"/out.txt", False)

        P1 = CalculateSingleLetterFrequency(workdir+"/out.txt", AllowedChars)
        P2 = CalculateBigramFrequency(workdir+"/out.txt", AllowedChars, False)
        P3 = CalculateBigramFrequency(workdir+"/out.txt", AllowedChars, True)

        FrequencyDictToCSV(workdir+"/crypto_results.csv", P1, P2, P3, AllowedChars)
        print(BOLD+"\nEntropies on the symbol of source:\n"+END)
        print("H2(single pass) =",CalculateEntropy(P2))
        print("H2(double pass) =",CalculateEntropy(P3))
        print("H1 =",CalculateEntropy(P1))
        print("\n")
    else:
        print("ERROR! File or directory does not exist!")

```

## Демонстрація роботи:

В якості джерела тексту застосую текст книги Ф. Достоевського “Злочин і кара” у форматі .txt (2.1Mb, текст додаю до звіту окремим файлом).

Виклик для тексту зі збереженням пробілів.

```

rodion@rodion-Vivobook:~$ python3 Crypto-lab1.1.py Dostoevskiy.txt CRYPT-LAB1-2 True
=====
Text processing and letter frequency, probability and entropy calculation program.
=====

Processing char № 1110960

Processing of single characters completed!
TOTAL: 1110960 characters

Processing of bigrams completed!

TOTAL: 555480 bigrams

Processing of bigrams completed!
Processing of bigrams completed!

TOTAL: 1110959 bigrams

Entropies on the symbol of source:

H2(single pass) = 3.9506137785792954
H2(double pass) = 3.950776721281697
H1 = 2.179022434940531

```



## Frequency of bigrams in text (single pass with step = 2):

	а	б	в	г	д	е	ж	...	щ	ы	ь	э	ю	я	
а	14	268	1490	343	995	680	884	...	124	0	0	4	477	1051	9251
б	322	1	30	0	14	1156	2	...	139	1991	91	0	3	323	250
в	3195	1	15	9	412	2723	0	...	7	1422	79	0	0	111	3298
г	531	0	3	5	579	124	0	...	0	0	0	0	2	0	580
д	2809	14	494	10	17	2365	14	...	1	244	568	0	4	235	506
е	15	687	711	1716	1515	938	438	...	509	0	0	0	123	113	10223
ж	656	14	0	2	398	2477	6	...	0	0	28	0	1	0	369
з	2503	72	499	164	363	113	22	...	0	149	77	0	2	247	711
и	60	357	1385	280	943	1049	114	...	77	0	0	0	170	721	9695
й	2	11	0	1	133	3	0	...	0	0	0	0	0	0	3483
к	3710	0	130	0	1	227	12	...	0	0	0	0	0	0	2541
л	2890	13	7	78	9	2147	161	...	0	293	2584	0	437	688	3677
м	1663	3	5	38	2	2127	0	...	1	465	47	0	2	207	4180
н	5460	4	13	21	162	5469	2	...	45	1526	617	0	122	912	2300
о	4	1983	4352	2383	2724	1059	1078	...	132	0	0	10	311	353	12787
п	457	2	0	0	0	1546	0	...	0	112	88	0	0	287	37
р	4216	71	200	98	172	3060	144	...	7	471	482	0	103	560	410
с	955	46	779	11	107	2380	12	...	0	115	1616	0	129	1904	2025
т	3081	3	1278	6	44	3223	0	...	16	715	3506	7	43	219	3216
у	37	433	340	792	1077	106	890	...	159	0	0	0	481	33	3607
ф	128	0	0	0	0	29	0	...	0	14	57	0	0	0	16
х	213	0	106	0	0	172	0	...	0	0	0	0	0	0	1642
ц	278	0	8	0	0	421	0	...	0	99	0	0	0	0	201
ч	1327	0	2	0	0	2176	0	...	0	0	127	0	0	0	252
ш	466	0	4	0	0	1144	0	...	0	0	307	0	0	0	32
щ	200	0	0	0	0	803	0	...	0	0	20	0	0	0	4
ы	0	81	432	63	88	408	10	...	8	0	0	0	0	2	2420
ь	0	45	20	76	14	249	0	...	21	0	0	0	172	326	6348
э	0	0	1	3	1	0	0	...	0	0	0	3	0	0	6
ю	0	155	3	1	165	0	10	...	119	0	0	0	19	0	1625
я	0	14	152	40	275	50	48	...	79	0	0	0	55	54	6361
	1388	3692	9069	1652	4463	2289	1213	...	31	0	0	1518	31	1369	0

[32 rows x 32 columns]

## Probability of bigrams in text (single pass with step = 2):

	а	б	в	...	ю	я	
а	0.00002520	0.00048247	0.00268236	...	0.00085872	0.00189206	0.01665406
б	0.00057968	0.00000180	0.00005401	...	0.00000540	0.00058148	0.00045006
в	0.00575178	0.00000180	0.00002700	...	0.00000000	0.00019983	0.00593721
г	0.00095593	0.00000000	0.00000540	...	0.00000360	0.00000000	0.00104414
д	0.00505689	0.00002520	0.00088932	...	0.00000720	0.00042306	0.00091092
е	0.00002700	0.00123677	0.00127997	...	0.00022143	0.00020343	0.01840390
ж	0.00118096	0.00002520	0.00000000	...	0.00000180	0.00000000	0.00066429
з	0.00450601	0.00012962	0.00089832	...	0.00000360	0.00044466	0.00127997
и	0.00010801	0.00064269	0.00249334	...	0.00030604	0.00129798	0.01745337
й	0.00000360	0.00001980	0.00000000	...	0.00000000	0.00000000	0.00627025
к	0.00667891	0.00000000	0.00023403	...	0.00000000	0.00000000	0.00457442
л	0.00520271	0.00002340	0.00001260	...	0.00078671	0.00123857	0.00661950
м	0.00299381	0.00000540	0.00000900	...	0.00000360	0.00037265	0.00752502
н	0.00982934	0.00000720	0.00002340	...	0.00021963	0.00164182	0.00414056
о	0.00000720	0.00356989	0.00783467	...	0.00055988	0.00063549	0.02301973
п	0.00082271	0.00000360	0.00000000	...	0.00000000	0.00051667	0.00006661
р	0.00758983	0.00012782	0.00036005	...	0.00018543	0.00100814	0.00073810
с	0.00171923	0.00008281	0.00140239	...	0.00023223	0.00342767	0.00364550
т	0.00554655	0.00000540	0.00230071	...	0.00007741	0.00039425	0.00578959
у	0.00006661	0.00077951	0.00061208	...	0.00086592	0.00005941	0.00649348
ф	0.00023043	0.00000000	0.00000000	...	0.00000000	0.00000000	0.00002880
х	0.00038345	0.00000000	0.00019083	...	0.00000000	0.00000000	0.00295600
ц	0.00050047	0.00000000	0.00001440	...	0.00000000	0.00000000	0.00036185
ч	0.00238892	0.00000000	0.00000360	...	0.00000000	0.00000000	0.00045366
ш	0.00083891	0.00000000	0.00000720	...	0.00000000	0.00000000	0.00005761
щ	0.00036005	0.00000000	0.00000000	...	0.00000000	0.00000000	0.00000720
ы	0.00000000	0.00014582	0.00077771	...	0.00000000	0.00000360	0.00435659
ь	0.00000000	0.00008101	0.00003600	...	0.00030964	0.00058688	0.01142795
э	0.00000000	0.00000000	0.00000180	...	0.00000000	0.00000000	0.00001080
ю	0.00000000	0.00027904	0.00000540	...	0.00003420	0.00000000	0.00292540
я	0.00000000	0.00002520	0.00027364	...	0.00009901	0.00009721	0.01145136
	0.00249874	0.00664650	0.01632642	...	0.00005581	0.00246454	0.00000000

[32 rows x 32 columns]



## Frequency of bigrams in text (double pass with overlap):

	а	б	в	г	д	е	...	ы	ь	э	ю	я	
а	31	557	2985	702	1996	1348	...	0	0	4	949	2125	18520
б	666	1	60	0	24	2296	...	3942	199	0	8	649	486
в	6258	3	28	21	884	5564	...	2914	166	0	0	219	6583
г	1083	0	7	7	1178	267	...	0	0	0	3	0	1167
д	5572	29	964	18	41	4774	...	480	1109	0	9	456	1005
е	29	1408	1387	3594	3093	1885	...	0	0	0	251	240	20764
ж	1304	24	0	4	832	4753	...	0	54	0	2	0	730
з	4943	139	960	357	758	213	...	310	150	0	2	445	1348
и	124	698	2733	581	1876	2063	...	0	0	1	374	1457	19411
й	2	20	0	1	279	4	...	0	0	0	0	0	6974
к	7493	0	261	0	2	428	...	0	0	0	0	0	5183
л	5793	27	14	139	21	4353	...	622	5205	0	843	1403	7363
м	3340	13	8	79	3	4259	...	953	84	1	3	392	8370
н	11050	10	21	37	328	10881	...	2993	1264	0	227	1875	4579
о	6	3895	8818	4709	5389	2075	...	0	0	17	626	701	25671
п	863	4	0	1	0	3112	...	235	166	0	0	590	62
р	8353	151	422	200	330	5944	...	941	998	0	204	1069	791
с	1901	96	1509	31	217	4745	...	229	3243	0	267	3865	4029
т	6255	12	2551	10	108	6435	...	1469	7040	10	87	433	6339
у	64	854	700	1560	2125	209	...	0	0	1	871	66	7338
ф	245	0	0	0	0	68	...	23	107	0	0	0	36
х	463	0	195	0	0	325	...	0	0	2	0	0	3328
ц	557	0	25	0	0	867	...	186	0	0	0	0	400
ч	2737	0	2	0	0	4336	...	0	257	0	0	0	494
ш	951	0	8	0	0	2290	...	0	638	0	0	0	62
щ	421	0	1	0	0	1594	...	0	41	0	0	0	6
ы	0	154	855	116	168	857	...	0	0	0	0	3	4756
ь	0	94	39	142	20	489	...	0	0	0	358	670	12645
э	0	0	4	8	6	0	...	0	0	13	0	0	13
ю	0	330	4	4	337	0	...	0	0	0	41	0	3306
я	0	17	289	66	532	102	...	0	0	0	108	93	12615
	2797	7344	18314	3440	8895	4455	...	0	0	3159	63	2765	0

[32 rows x 32 columns]

## Probability of bigrams in text (double pass with overlap):

	а	б	в	...	ю	я	
а	0.00002790	0.00050137	0.00268687	...	0.00085422	0.00191276	0.01667028
б	0.00059948	0.00000090	0.00005401	...	0.00000720	0.00058418	0.00043746
в	0.00563297	0.00000270	0.00002520	...	0.00000000	0.00019713	0.00592551
г	0.00097483	0.00000000	0.00000630	...	0.00000270	0.00000000	0.00105044
д	0.00501549	0.00002610	0.00086772	...	0.00000810	0.00041046	0.00090462
е	0.00002610	0.00126737	0.00124847	...	0.00022593	0.00021603	0.01869016
ж	0.00117376	0.00002160	0.00000000	...	0.00000180	0.00000000	0.00065709
з	0.00444931	0.00012512	0.00086412	...	0.00000180	0.00040055	0.00121337
и	0.00011162	0.00062829	0.00246004	...	0.00033665	0.00131148	0.01747229
й	0.00000180	0.00001800	0.00000000	...	0.00000000	0.00000000	0.00627746
к	0.00674462	0.00000000	0.00023493	...	0.00000000	0.00000000	0.0466534
л	0.00521441	0.00002430	0.00001260	...	0.00075880	0.00126287	0.00662761
м	0.00300641	0.00001170	0.00000720	...	0.00000270	0.00035285	0.00753403
н	0.00994636	0.00000900	0.00001890	...	0.00020433	0.00168773	0.00412166
о	0.00000540	0.00350598	0.00793729	...	0.00056348	0.00063099	0.02310706
п	0.00077681	0.00000360	0.00000000	...	0.00000000	0.00053107	0.00005581
р	0.00751873	0.00013592	0.00037985	...	0.00018363	0.00096223	0.00071200
с	0.00171113	0.00008641	0.00135829	...	0.00024033	0.00347898	0.00362660
т	0.00563027	0.00001080	0.00229621	...	0.00007831	0.00038975	0.00570588
у	0.00005761	0.00076871	0.00063009	...	0.00078401	0.00005941	0.00660510
ф	0.00022053	0.00000000	0.00000000	...	0.00000000	0.00000000	0.00003240
х	0.00041676	0.00000000	0.00017552	...	0.00000000	0.00000000	0.00299561
ц	0.00050137	0.00000000	0.00002250	...	0.00000000	0.00000000	0.00036005
ч	0.00246364	0.00000000	0.00000180	...	0.00000000	0.00000000	0.00044466
ш	0.00085602	0.00000000	0.00000720	...	0.00000000	0.00000000	0.00005581
щ	0.00037895	0.00000000	0.00000090	...	0.00000000	0.00000000	0.00000540
ы	0.00000000	0.00013862	0.00076961	...	0.00000000	0.00000270	0.00428099
ь	0.00000000	0.00008461	0.00003510	...	0.00032224	0.00060308	0.01138206
э	0.00000000	0.00000000	0.00000360	...	0.00000000	0.00000000	0.00001170
ю	0.00000000	0.00029704	0.00000360	...	0.00003691	0.00000000	0.00297581
я	0.00000000	0.00001530	0.00026014	...	0.00009721	0.00008371	0.01135505
	0.00251764	0.00661050	0.01648486	...	0.00005671	0.00248884	0.00000000

[32 rows x 32 columns]

### Frequency and probability of single letters in text:

	Frequency	Probability
а	73301	0.06597987
б	15880	0.01429394
в	43164	0.03885288
г	15827	0.01424624
д	29442	0.02650140
е	80991	0.07290181
ж	10364	0.00932887
з	14223	0.01280244
и	61339	0.05521261
й	9548	0.00859437
к	30634	0.02757435
л	42108	0.03790235
м	29112	0.02620436
н	60588	0.05453662
о	106176	0.09557140
п	25477	0.02293242
р	39292	0.03536761
с	49720	0.04475409
т	59545	0.05359779
у	27044	0.02434291
ф	1204	0.00108375
х	8081	0.00727389
ц	2709	0.00243843
ч	16473	0.01482772
ш	7378	0.00664110
щ	2928	0.00263556
ы	15297	0.01376917
ь	20721	0.01865144
э	3208	0.00288759
ю	5296	0.00476705
я	19516	0.01756679
	184374	0.16595917

Виклик для тексту без збереження пробілів.

```
rodion@rodion-Vivobook:~$ python3 Crypto-lab1.1.py Dostoevskiy.txt CRYPT-LAB1-3 False
```

```
=====
Text processing and letter frequency, probability and entropy calculation program.
=====
```

```
Processing char № 926580
```

```
Processing of single characters completed!
```

```
TOTAL: 926586 characters
```

```
Processing of bigrams completed!
```

```
TOTAL: 463293 bigrams
```

```
Processing of bigrams completed!
```

```
Processing of bigrams completed!
```

```
TOTAL: 926585 bigrams
```

```
Entropies on the symbol of source:
```

```
H2(single pass) = 4.128233254678958
```

```
H2(double pass) = 4.127843733528513
```

```
H1 = 2.2239126204293846
```



# Frequency of bigrams in text (single pass with step = 2):

	а	б	в	г	д	е	...	щ	ы	ь	э	ю	я
а	140	605	2418	505	1433	905	...	147	0	0	154	495	1215
б	348	2	44	0	14	1109	...	118	1921	91	49	4	347
в	3208	92	218	129	579	2920	...	5	1463	79	128	2	118
г	541	17	74	15	610	136	...	0	0	0	3	2	4
д	2723	33	528	12	42	2378	...	1	238	532	13	4	213
е	135	1177	1836	1985	2067	1154	...	506	0	0	236	125	220
ж	671	27	29	5	447	2328	...	0	0	29	19	1	8
з	2413	87	559	198	434	113	...	2	166	83	9	1	233
и	202	750	2386	486	1462	1222	...	87	0	0	113	204	870
й	47	104	278	109	339	46	...	2	0	0	39	6	39
к	3808	288	363	34	114	246	...	2	0	0	50	1	60
л	2960	124	338	140	142	2398	...	1	323	2622	42	435	741
м	1671	135	397	131	236	2180	...	2	494	45	59	4	277
н	5551	118	281	55	222	5494	...	49	1510	626	22	112	933
о	126	2650	5825	2570	3281	1359	...	132	0	0	214	338	622
п	461	5	0	5	2	1554	...	0	108	83	0	0	304
р	4250	98	239	106	196	2924	...	5	481	494	7	104	547
с	989	119	892	38	216	2451	...	1	120	1580	40	136	1927
т	3211	201	1520	51	223	3289	...	15	720	3478	65	41	262
у	128	506	733	828	1249	158	...	166	0	0	56	415	104
ф	132	0	0	0	2	29	...	0	11	55	0	0	0
х	241	51	238	52	109	196	...	1	0	0	24	0	17
ц	282	4	34	2	7	462	...	0	88	0	2	1	4
ч	1419	6	32	7	12	2158	...	0	0	127	3	0	4
ш	476	3	10	2	2	1137	...	0	0	310	0	0	1
щ	216	0	1	0	0	765	...	0	0	19	0	0	0
ы	20	162	686	95	219	494	...	11	0	0	54	1	24
ь	131	260	620	156	283	458	...	14	0	0	105	161	455
э	1	1	4	7	4	0	...	0	0	0	9	0	0
ю	43	199	131	31	260	23	...	129	0	0	17	24	34
я	145	210	873	129	596	223	...	83	0	0	89	50	118

[31 rows x 31 columns]

## Probability of bigrams in text (single pass with step = 2):

	а	б	в	...	э	ю	я
а	0.00030218	0.00130587	0.00521916	...	0.00033240	0.00106844	0.00262253
б	0.00075114	0.00000432	0.00009497	...	0.00010576	0.00000863	0.00074899
в	0.00692434	0.00019858	0.00047054	...	0.00027628	0.00000432	0.00025470
г	0.00116773	0.00003669	0.00015973	...	0.00000648	0.00000432	0.00000863
д	0.00587749	0.00007123	0.00113967	...	0.00002806	0.00000863	0.00045975
е	0.00029139	0.00254051	0.00396293	...	0.00050940	0.00026981	0.00047486
ж	0.00144833	0.00005828	0.00006260	...	0.00004101	0.00000216	0.00001727
з	0.00520837	0.00018779	0.00120658	...	0.00001943	0.00000216	0.00050292
и	0.00043601	0.00161885	0.00515009	...	0.00024391	0.00044033	0.00187786
й	0.00010145	0.00022448	0.00060005	...	0.00008418	0.00001295	0.00008418
к	0.00821942	0.00062164	0.00078352	...	0.00010792	0.00000216	0.00012951
л	0.00638905	0.00026765	0.00072956	...	0.00009066	0.00093893	0.00159942
м	0.00360679	0.00029139	0.00085691	...	0.00012735	0.00000863	0.00059789
н	0.01198162	0.00025470	0.00060653	...	0.00004749	0.00024175	0.00201384
о	0.00027197	0.00571992	0.01257304	...	0.00046191	0.00072956	0.00134256
п	0.00099505	0.00001079	0.00000000	...	0.00000000	0.00000000	0.00065617
р	0.00917346	0.00021153	0.00051587	...	0.00001511	0.00022448	0.00118068
с	0.00213472	0.00025686	0.00192535	...	0.00008634	0.00029355	0.00415935
т	0.00693082	0.00043385	0.00328086	...	0.00014030	0.00008850	0.00056552
у	0.00027628	0.00109218	0.00158215	...	0.00012087	0.00089576	0.00022448
ф	0.00028492	0.00000000	0.00000000	...	0.00000000	0.00000000	0.00000000
х	0.00052019	0.00011008	0.00051371	...	0.00005180	0.00000000	0.00003669
ц	0.00060869	0.00000863	0.00007339	...	0.00000432	0.00000216	0.00000863
ч	0.00306286	0.00001295	0.00006907	...	0.00000648	0.00000000	0.00000863
ш	0.00102743	0.00000648	0.00002158	...	0.00000000	0.00000000	0.00000216
щ	0.00046623	0.00000000	0.00000216	...	0.00000000	0.00000000	0.00000000
ы	0.00004317	0.00034967	0.00148070	...	0.00011656	0.00000216	0.00005180
ь	0.00028276	0.00056120	0.00133825	...	0.00022664	0.00034751	0.00098210
э	0.00000216	0.00000216	0.00000863	...	0.00001943	0.00000000	0.00000000
ю	0.00009281	0.00042953	0.00028276	...	0.00003669	0.00005180	0.00007339
я	0.00031298	0.00045328	0.00188434	...	0.00019210	0.00010792	0.00025470

[31 rows x 31 columns]



## Frequency of bigrams in text (double pass with overlap):

	а	б	в	г	д	е	...	щ	ы	ь	э	ю	я
а	294	1172	4825	1028	2882	1869	...	290	0	0	316	951	2430
б	670	5	83	0	26	2311	...	252	3942	199	93	8	684
в	6337	191	440	289	1183	5731	...	14	2914	166	295	4	256
г	1100	25	150	28	1232	286	...	0	0	0	8	4	11
д	5592	62	1057	28	90	4801	...	2	480	1109	22	9	465
е	273	2310	3617	4003	4135	2268	...	1008	0	0	420	261	479
ж	1310	55	51	9	855	4765	...	0	0	54	38	2	27
з	4959	166	1088	390	844	240	...	3	310	150	25	2	453
и	363	1491	4777	922	2896	2478	...	160	0	0	232	383	1730
й	106	225	557	225	659	93	...	3	0	0	91	10	66
к	7545	540	715	83	228	508	...	4	0	0	97	1	104
л	5942	255	688	263	300	4694	...	3	622	5205	97	843	1529
м	3456	282	800	258	451	4403	...	4	953	84	115	7	533
н	11103	244	571	111	494	10966	...	93	2993	1264	49	227	1916
о	238	5222	11716	5159	6557	2785	...	255	0	0	399	633	1213
п	865	8	2	8	4	3113	...	0	235	166	0	0	590
р	8373	181	484	210	356	5953	...	15	941	998	8	204	1079
с	1969	232	1830	96	428	4809	...	4	229	3243	69	270	3895
т	6445	383	3135	102	432	6621	...	32	1469	7040	130	88	527
у	246	1032	1419	1671	2483	327	...	335	0	0	103	873	210
ф	246	1	2	0	2	68	...	0	23	107	0	0	0
х	506	127	474	108	215	396	...	2	0	0	56	1	34
ц	568	9	61	6	14	882	...	0	186	0	6	1	8
ч	2742	20	61	12	25	4343	...	0	0	257	5	0	10
ш	952	4	15	2	4	2291	...	0	0	638	0	0	2
щ	422	0	1	0	0	1597	...	0	0	41	0	0	0
ы	53	346	1325	189	407	993	...	14	0	0	90	1	54
ь	257	472	1259	291	575	920	...	33	0	0	211	361	922
э	1	1	6	9	7	0	...	0	0	0	13	0	0
ю	77	401	279	61	503	57	...	257	0	0	50	42	56
я	291	418	1676	266	1155	423	...	145	0	0	170	110	233

[31 rows x 31 columns]

## Probability of bigrams in text (double pass with overlap):

	а	б	в	...	э	ю	я
а	0.00031729	0.00126486	0.00520729	...	0.00034104	0.00102635	0.00262253
б	0.00072309	0.00000540	0.00008958	...	0.00010037	0.00000863	0.00073819
в	0.00683909	0.00020613	0.00047486	...	0.00031837	0.00000432	0.00027628
г	0.00118715	0.00002698	0.00016188	...	0.00000863	0.00000432	0.00001187
д	0.00603506	0.00006691	0.00114075	...	0.00002374	0.00000971	0.00050184
е	0.00029463	0.00249303	0.00390358	...	0.00045328	0.00028168	0.00051695
ж	0.00141379	0.00005936	0.00005504	...	0.00004101	0.00000216	0.00002914
з	0.00535191	0.00017915	0.00117420	...	0.00002698	0.00000216	0.00048889
и	0.00039176	0.00160913	0.00515549	...	0.00025038	0.00041335	0.00186707
й	0.00011440	0.00024283	0.00060113	...	0.00009821	0.00001079	0.00007123
к	0.00814280	0.00058279	0.00077165	...	0.00010469	0.00000108	0.00011224
л	0.00641280	0.00027520	0.00074251	...	0.00010469	0.00090979	0.00165015
м	0.00372983	0.00030434	0.00086339	...	0.00012411	0.00000755	0.00057523
н	0.01198271	0.00026333	0.00061624	...	0.00005288	0.00024499	0.00206781
о	0.00025686	0.00563575	0.01264428	...	0.00043061	0.00068315	0.00130911
п	0.00093354	0.00000863	0.00000216	...	0.00000000	0.00000000	0.00063675
р	0.00903641	0.00019534	0.00052235	...	0.00000863	0.00022016	0.00116449
с	0.00212501	0.00025038	0.00197499	...	0.00007447	0.00029139	0.00420361
т	0.00695565	0.00041335	0.00338339	...	0.00014030	0.00009497	0.00056876
у	0.00026549	0.00111377	0.00153143	...	0.00011116	0.00094217	0.00022664
ф	0.00026549	0.00000108	0.00000216	...	0.00000000	0.00000000	0.00000000
х	0.00054609	0.00013706	0.00051156	...	0.00006044	0.00000108	0.00003669
ц	0.00061300	0.00000971	0.00006583	...	0.00000648	0.00000108	0.00000863
ч	0.00295925	0.00002158	0.00006583	...	0.00000540	0.00000000	0.00001079
ш	0.00102743	0.00000432	0.00001619	...	0.00000000	0.00000000	0.00000216
щ	0.00045544	0.00000000	0.00000108	...	0.00000000	0.00000000	0.00000000
ы	0.00005720	0.00037341	0.00142998	...	0.00009713	0.00000108	0.00005828
ь	0.00027736	0.00050940	0.00135875	...	0.00022772	0.00038960	0.00099505
э	0.00000108	0.00000108	0.00000648	...	0.00001403	0.00000000	0.00000000
ю	0.00008310	0.00043277	0.00030111	...	0.00005396	0.00004533	0.00006044
я	0.00031406	0.00045112	0.00180879	...	0.00018347	0.00011872	0.00025146

[31 rows x 31 columns]



Frequency and probability of single letters in text:		
	Frequency	Probability
а	73301	0.07910869
б	15880	0.01713818
в	43164	0.04658391
г	15827	0.01708098
д	29442	0.03177471
е	80991	0.08740797
ж	10364	0.01118515
з	14223	0.01534990
и	61339	0.06619893
й	9548	0.01030449
к	30634	0.03306115
л	42108	0.04544424
м	29112	0.03141856
н	60588	0.06538843
о	106176	0.11458839
п	25477	0.02749556
р	39292	0.04240513
с	49720	0.05365935
т	59545	0.06426279
у	27044	0.02918671
ф	1204	0.00129939
х	8081	0.00872126
ц	2709	0.00292364
ч	16473	0.01777817
ш	7378	0.00796256
щ	2928	0.00315999
ы	15297	0.01650899
ь	20721	0.02236274
э	3208	0.00346217
ю	5296	0.00571561
я	19516	0.02106227

**2.** За допомогою CoolPinkProgram.exe знайдемо умовні ентропії джерела для 10, 20, 30 символів.

### Оцінки $H^{(10)}$ , $H^{(20)}$ , $H^{(30)}$ у CoolPinkProgram:

 $H^{(10)}:$ [illegible]

$$\text{H}^{(20)}.$$
[illegible]
$$\text{H}^{(30)}:$$
[illegible]



## Результати:

### Результати аналізу умовної ентропії і надлишковості мови:

Експеримент	Ентропія	Надлишковість
$H^{(10)}$	$1,67035730421506 < H < 2,44914397406541$	$0,510171205186918 < R < 0,665928539156988$
$H^{(20)}$	$1,41288986068504 < H < 2,28046658275398$	$0,543906683449204 < R < 0,717422027862992$
$H^{(30)}$	$1,55451374163455 < H < 2,44786020017394$	$0,510427959965212 < R < 0,68909725167309$

### Результати аналізу питомої ентропії на символ джерела:

Текст зі збереженням пробілів:

Експеримент	Ентропія	Надлишковість
$H_1$	$H = 2.179022434940531$	$R = 0.564195513$
$H_{2(\text{step} = 2)}$	$H = 3.9506137785792954$	$R = 0.209877244$
$H_{2(\text{step} = 1)}$	$H = 3.950776721281697$	$R = 0.209844656$

Текст без збереження пробілів:

Експеримент	Ентропія	Надлишковість
$H_1$	$H = 2.2239126204293846$	$R = 0.551105269$
$H_{2(\text{step} = 2)}$	$H = 4.128233254678958$	$R = 0.166719888$
$H_{2(\text{step} = 1)}$	$H = 4.127843733528513$	$R = 0.166798513$

Надлишковість вираховується за формулою  $1 - (H_n / \log_2(m))$ , де  $m$  - кількість букв у алфавіті:  $m = 32$  з пробілом,  $m = 31$  без.

## Частоти та імовірності символів та біграм для джерела:

Вихідний текст: Ф.Достоевський „Злочин і кара”. **Через великий візуальний розмір даних, частоти та імовірності додаю у додаткових файлах таблиць:**

Текст зі збереженням пробілів:

Експеримент	Частоти	Імовірності
$H_1$	Spaces-Singlechar.csv	
$H_{2(\text{step} = 2)}$	Spaces-Doublepass-freq.csv	Spaces-Doublepass-pro.csv
$H_{2(\text{step} = 1)}$	Spaces-Singlepass-freq.csv	Spaces-Singlepass-pro.csv

Текст без збереження пробілів:

Експеримент	Частоти	Імовірності
$H_1$	NoSpaces-Singlechar.csv	
$H_{2(\text{step} = 2)}$	NoSpaces-Doublepass-freq.csv	NoSpaces-Doublepass-pro.csv
$H_{2(\text{step} = 1)}$	NoSpaces-Singlepass-freq.csv	NoSpaces-Singlepass-pro.csv

## Висновки:

В цій роботі ми в деталях ознайомилися з базовим поняттям ентропії тексту, питомою та умовною ентропіями джерела. В цій роботі ми розраховували питому ентропію на символ джерела для окремих символів та біграм російської мови на основі великого тексту, визначили частоти та імовірності появи символів алфавіту у тексті та, обрахували приблизне значення умовної ентропії та приблизну надлишковість мови. Для цього було застосоване надане у роботі ПЗ, а також написана власна програма для аналізу на мові Python.