**КРИПТОГРАФІЯ**

**КОМП’ЮТЕРНИЙ ПРАКТИКУМ №1**

Експериментальна оцінка ентропії на символ джерела відкритого тексту

**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.

Виконали: Бондаренко Олексій, Кригін Дмитро. ФБ-03

Варіант: 2

**Порядок виконання роботи**

0. Уважно прочитати методичні вказівки до виконання комп’ютерного практикуму.

1. Написати програми для підрахунку частот букв і частот біграм в тексті, а також підрахунку H1 та H2 за безпосереднім означенням. Підрахувати частоти букв та біграм, а також значення H1 та H2 на довільно обраному тексті російською мовою достатньої довжини (щонайменше 1Мб), де імовірності замінити відповідними частотами. Також одержати значення H1 та H2 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.

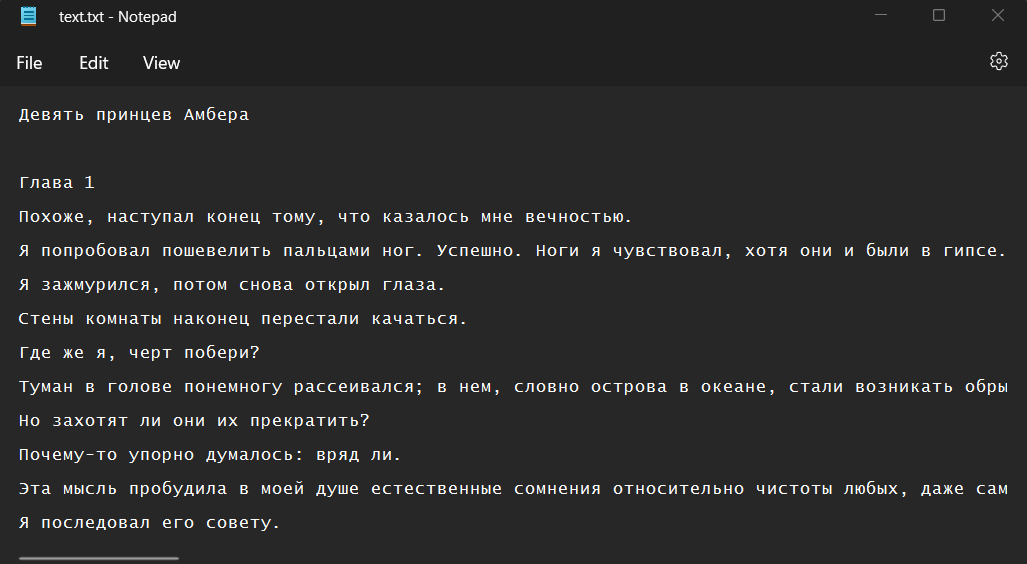
**Хід роботи:**

0. Уважно прочитати методичні вказівки до виконання комп’ютерного практикуму.

Done

1. Написати програми для підрахунку частот букв і частот біграм в тексті, а також підрахунку H1 та H2 за безпосереднім означенням. Підрахувати частоти букв та біграм, а також значення H1 та H2 на довільно обраному тексті російською мовою достатньої довжини (щонайменше 1Мб), де імовірності замінити відповідними частотами. Також одержати значення H1 та H2 на тому ж тексті, в якому вилучено всі пробіли.

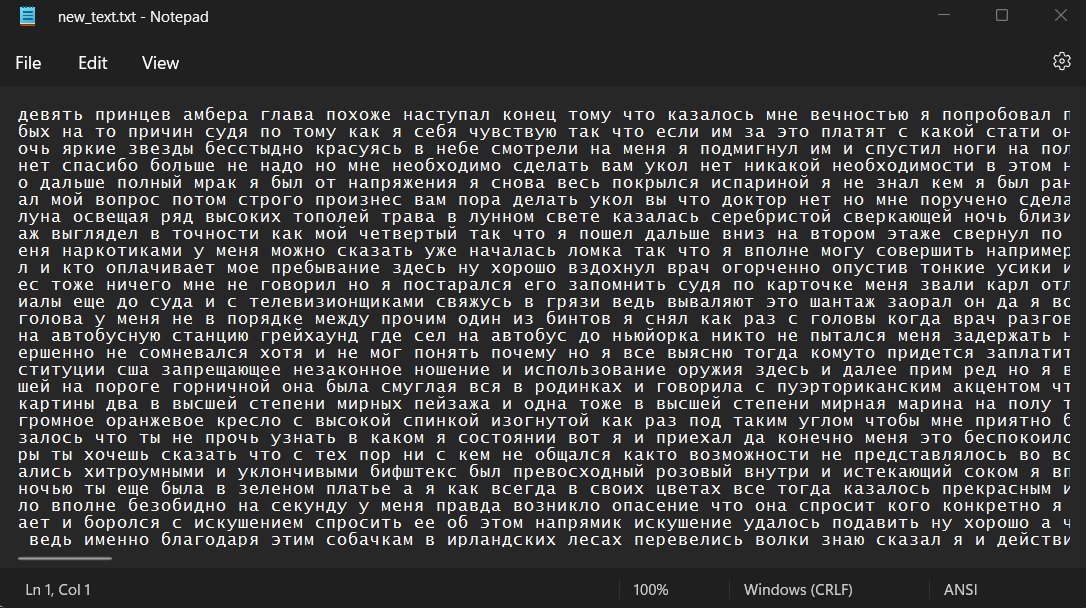
**Обраний текст – Роджер Желязни – Хроніки Амбера (Том 1) (1.5 Мб) – text.txt**

****

**Програма-фільтр, для відкидання зайвих символів (filter.py)**

# By Bondarenko and Kryhin  
  
FROM\_FILE = 'text.txt'  
TO\_FILE = 'new\_text.txt'  
alphabet = 'абвгдежзийклмнопрстуфхцчшщыьэюя '  
  
with open(FROM\_FILE, 'r') as f:  
 text = f.read().lower()  
  
result = ''  
for i in text:  
 if i in alphabet:  
 result += i  
 if i == '\n':  
 result += ' '  
 if i == 'ъ':  
 result += 'ь'  
  
with open(TO\_FILE, 'w') as nf:  
 nf.write(" ".join(result.split()))

**OUTPUT(new\_text.txt):**



**Програма-фільтр, для відкидання зайвих символів та пробілів (text\_without\_spaces.py)**

# By Bondarenko and Kryhin  
  
FROM\_FILE = 'text.txt'  
TO\_FILE = 'text\_no\_spaces.txt'  
  
with open(FROM\_FILE, 'r') as f:  
 text = f.read().lower()  
  
alphabet = 'абвгдежзийклмнопрстуфхцчшщыьэюя'  
result = ''  
for i in text:  
 if i in alphabet:  
 result += i  
 if i == 'ъ':  
 result += 'ь'  
  
with open(TO\_FILE, 'w') as nf:  
 nf.write(result)

**OUTPUT(text\_no\_spaces.txt):**



**Підрахуємо частоту букв у тексті з пробілами та H1 для букв (h1\_spaces.py):**

# By Bondarenko and Kryhin  
  
from pprint import pprint  
from math import log  
  
FROM\_FILE = 'new\_text.txt'  
alphabet = 'абвгдежзийклмнопрстуфхцчшщыьэюя '  
  
with open(FROM\_FILE, 'r') as f:  
 text = f.read().lower()  
length = len(text)  
  
letters = dict()  
  
for i in alphabet:  
 letters[i] = text.count(i)  
  
pprint(sorted(letters.items(), key=lambda item: item[1], reverse=True), sort\_dicts=False) # sorted output  
  
h1 = 0  
for i in alphabet:  
 if letters[i] == 0:  
 continue  
 h1 -= (letters[i] / length) \* log(letters[i] / length, 2)  
  
print(f"H1: {h1}")

OUTPUT (h1\_spaces.txt):

[(' ', 249090),

('о', 142771),

('е', 106547),

('а', 95744),

('н', 83826),

('и', 76681),

('т', 76038),

('с', 65814),

('л', 63783),

('р', 54907),

('в', 53318),

('м', 42498),

('к', 39760),

('д', 38053),

('п', 36142),

('у', 34269),

('я', 33405),

('ь', 26042),

('б', 23523),

('ы', 23271),

('з', 21778),

('г', 20487),

('ч', 18572),

('ж', 12885),

('й', 12559),

('ш', 10157),

('х', 9349),

('ю', 6444),

('э', 4673),

('щ', 4335),

('ц', 3278),

('ф', 1273)]

**H1: 4.360843323137924**

**Підрахуємо частоту букв у тексті без пробілів та H1 для букв (h1\_no\_spaces.py):**

# By Bondarenko and Kryhin  
  
from pprint import pprint  
from math import log  
  
FROM\_FILE = 'text\_no\_spaces.txt'  
alphabet = 'абвгдежзийклмнопрстуфхцчшщыьэюя'  
  
with open(FROM\_FILE, 'r') as f:  
 text = f.read().lower()  
length = len(text)  
  
letters = dict()  
  
for i in alphabet:  
 letters[i] = text.count(i)  
  
pprint(sorted(letters.items(), key=lambda item: item[1], reverse=True), sort\_dicts=False) # sorted output  
  
h1 = 0  
for i in alphabet:  
 if letters[i] == 0:  
 continue  
 h1 -= (letters[i] / length) \* log(letters[i] / length, 2)  
  
print(f"H1: {h1}")

OUTPUT(h1\_no\_spaces.txt):

[('о', 142771),

('е', 106547),

('а', 95744),

('н', 83826),

('и', 76681),

('т', 76038),

('с', 65814),

('л', 63783),

('р', 54907),

('в', 53318),

('м', 42498),

('к', 39760),

('д', 38053),

('п', 36142),

('у', 34269),

('я', 33405),

('ь', 26042),

('б', 23523),

('ы', 23271),

('з', 21778),

('г', 20487),

('ч', 18572),

('ж', 12885),

('й', 12559),

('ш', 10157),

('х', 9349),

('ю', 6444),

('э', 4673),

('щ', 4335),

('ц', 3278),

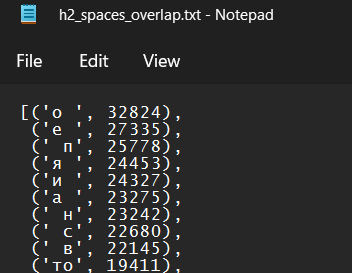
('ф', 1273)]

**H1: 4.453920329839864**

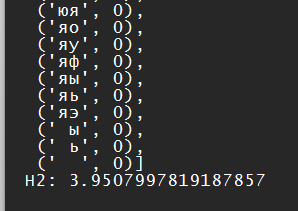
**Підрахуємо частоту біграм (з перетином) у тексті з пробілами та H2 для біграм (h2\_spaces\_overlap.py):**

# By Bondarenko and Kryhin  
  
from pprint import pprint  
from math import log  
  
FROM\_FILE = 'new\_text.txt'  
alphabet = 'абвгдежзийклмнопрстуфхцчшщыьэюя '  
  
with open(FROM\_FILE, 'r') as f:  
 text = f.read().lower()  
  
length = len(text) - 1 # determined experimentally (number of bigrams)  
bigrams = dict() # numbers of each bigram  
  
for i in alphabet:  
 for j in alphabet:  
 bigrams[i + j] = 0  
  
for i in bigrams.keys():  
 bigrams[i] = text.count(i)  
  
pprint(sorted(bigrams.items(), key=lambda item: item[1], reverse=True), sort\_dicts=False) # sorted output  
  
h2 = 0  
for i in bigrams:  
 if bigrams[i] == 0:  
 continue  
 h2 -= (bigrams[i] / length) \* log(bigrams[i] / length, 2)  
  
print(f"H2: {h2 / 2}")

OUTPUT(h2\_spaces\_overlap.txt):



…

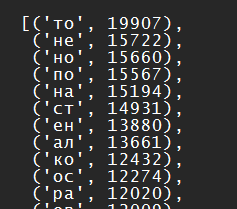


**H2: 3.9507997819187857**

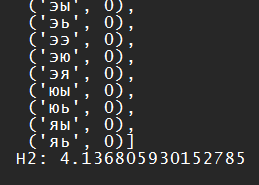
**Підрахуємо частоту біграм (з перетином) у тексті без пробілів та H2 для біграм (h2\_no\_spaces\_overlap.py):**

# By Bondarenko and Kryhin  
  
from pprint import pprint  
from math import log  
  
FROM\_FILE = 'text\_no\_spaces.txt'  
alphabet = 'абвгдежзийклмнопрстуфхцчшщыьэюя'  
  
with open(FROM\_FILE, 'r') as f:  
 text = f.read().lower()  
  
length = len(text) - 1 # determined experimentally (number of bigrams)  
bigrams = dict() # numbers of each bigram  
  
for i in alphabet:  
 for j in alphabet:  
 bigrams[i + j] = 0  
  
for i in bigrams.keys():  
 bigrams[i] = text.count(i)  
  
pprint(sorted(bigrams.items(), key=lambda item: item[1], reverse=True), sort\_dicts=False) # sorted output  
  
h2 = 0  
for i in bigrams:  
 if bigrams[i] == 0:  
 continue  
 h2 -= (bigrams[i] / length) \* log(bigrams[i] / length, 2)  
  
print(f"H2: {h2 / 2}")

OUTPUT(h2\_no\_spaces\_overlap.txt):



…

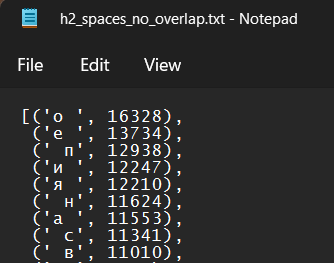


**H2: 4.136805930152785**

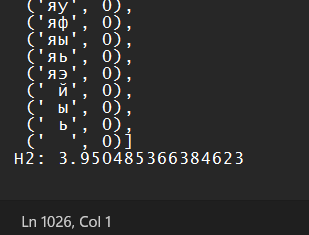
**Підрахуємо частоту біграм (без перетину) у тексті з пробілами та H2 для біграм (h2\_spaces\_no\_overlap.py):**

# By Bondarenko and Kryhin  
  
from pprint import pprint  
from math import log  
  
FROM\_FILE = 'new\_text.txt'  
alphabet = 'абвгдежзийклмнопрстуфхцчшщыьэюя '  
  
with open(FROM\_FILE, 'r') as f:  
 text = f.read().lower()  
  
text\_length = len(text)  
length = text\_length // 2 # determined experimentally (number of bigrams without overlap)  
bigrams = dict()  
  
for i in alphabet:  
 for j in alphabet:  
 bigrams[i + j] = 0  
  
i = 0  
while i < text\_length:  
 if i == text\_length - 2: # save from "index out of range"  
 break  
 bigrams[text[i:i+2]] += 1  
 i += 2  
  
pprint(sorted(bigrams.items(), key=lambda item: item[1], reverse=True), sort\_dicts=False)  
  
h2 = 0  
for i in bigrams.keys():  
 if bigrams[i] == 0:  
 continue  
 h2 -= (bigrams[i] / length) \* log(bigrams[i] / length, 2)  
  
print(f"H2: {h2 / 2}")

OUTPUT(h2\_spaces\_no\_overlap.txt):



…

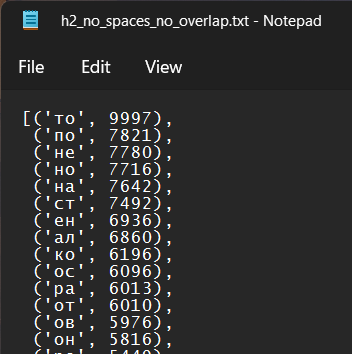


**H2: 3.950485366384623**

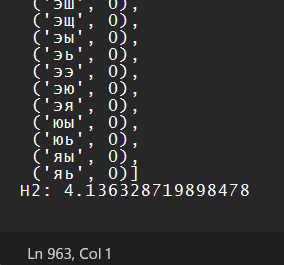
**Підрахуємо частоту біграм (без перетину) у тексті без пробілів та H2 для біграм (h2\_no\_spaces\_no\_overlap.py):**

# By Bondarenko and Kryhin  
  
from pprint import pprint  
from math import log  
  
FROM\_FILE = 'text\_no\_spaces.txt'  
alphabet = 'абвгдежзийклмнопрстуфхцчшщыьэюя'  
  
with open(FROM\_FILE, 'r') as f:  
 text = f.read().lower()  
  
text\_length = len(text)  
length = text\_length // 2 # determined experimentally (number of bigrams without overlap)  
bigrams = dict()  
  
for i in alphabet:  
 for j in alphabet:  
 bigrams[i + j] = 0  
  
i = 0  
while i < text\_length:  
 if i == text\_length - 2: # save from "index out of range"  
 break  
 bigrams[text[i:i+2]] += 1  
 i += 2  
  
pprint(sorted(bigrams.items(), key=lambda item: item[1], reverse=True), sort\_dicts=False)  
  
h2 = 0  
for i in bigrams.keys():  
 if bigrams[i] == 0:  
 continue  
 h2 -= (bigrams[i] / length) \* log(bigrams[i] / length, 2)  
  
print(f"H2: {h2 / 2}")

OUTPUT(h2\_no\_spaces\_no\_overlap.txt):



…



**H2: 4.136328719898478**

**Note:**

Файл new\_text.txt створюється програмою filter.txt

Файл text\_no\_spaces.txt створюється програмою text\_without\_spaces.py

Файли .txt у кодуванні ANSI (можуть виникати проблеми на інших пристроях)

**Проблеми та шляхи їх вирішення:**

Особливо проблем не виникало, лише була помилка, коли ми неправильно рахували ентропію через що, ділили не на кількість біграм, а на довжину тексту.

Інших проблем не було, код і алгоритм легкий.

Отже, отримали такі значення ентропії:

H1: 4.360843323137924

H1: 4.453920329839864

H2: 3.9507997819187857

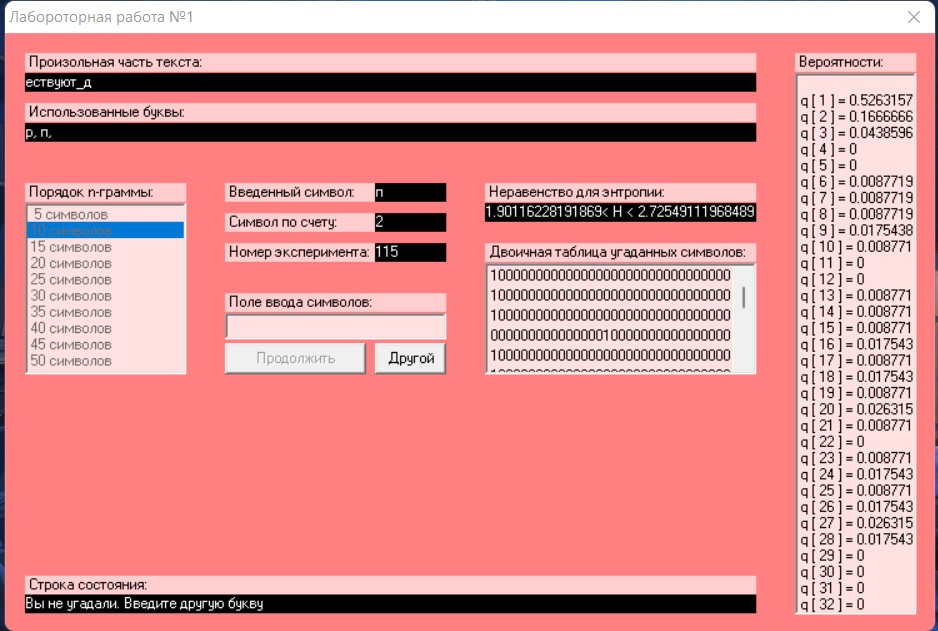
H2: 4.136805930152785

H2: 3.950485366384623

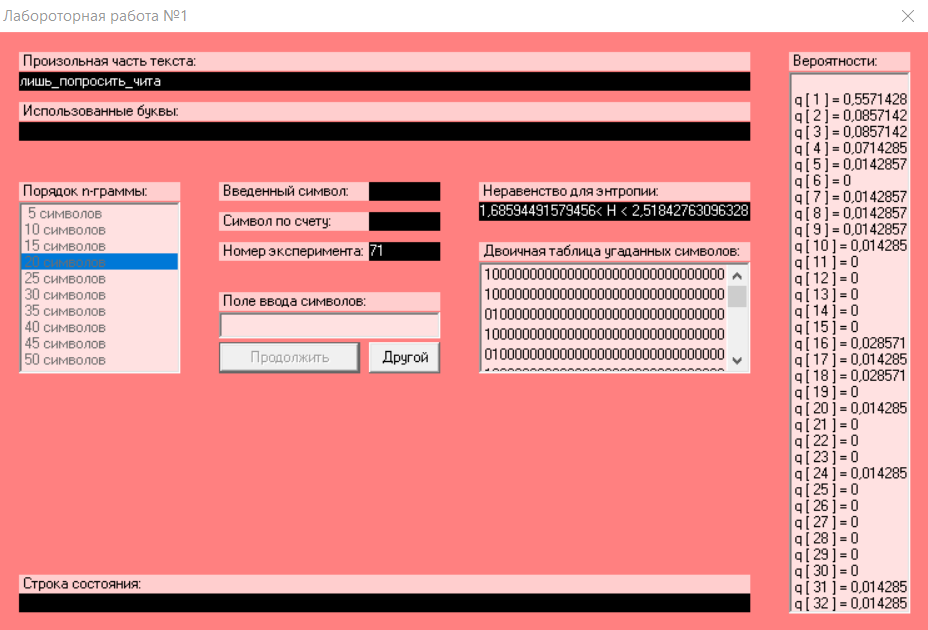
H2: 4.136328719898478

**2. За допомогою програми CoolPinkProgram оцінити значення H(10), H(20), H(30).**

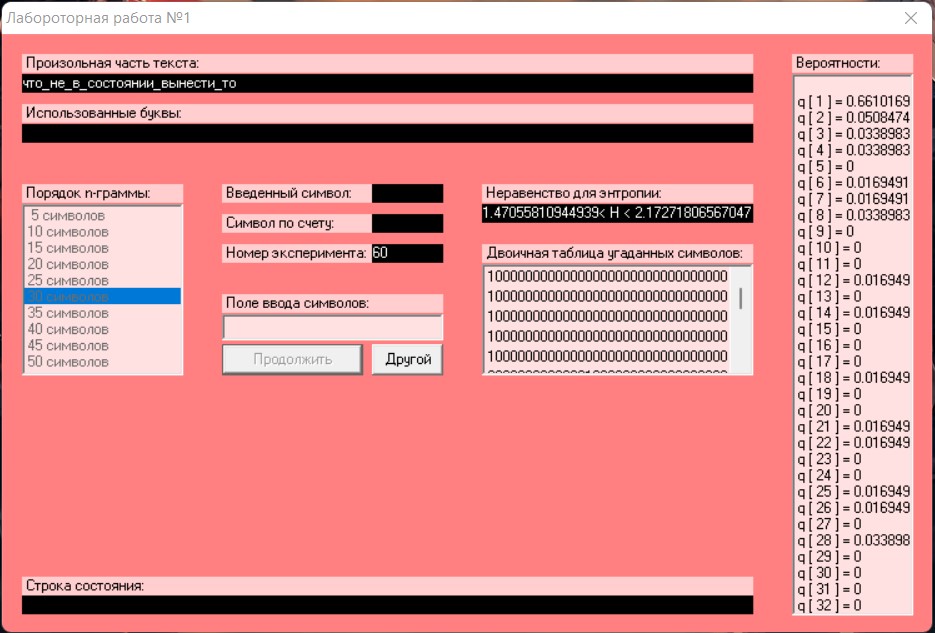
**H(10)**



**H(20)**



**H(30)**



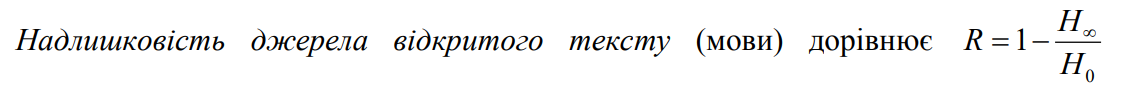
Отже, значення:

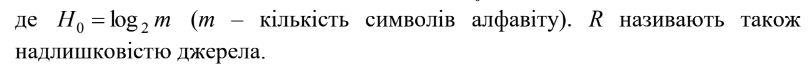
1.90116228191869 < H(10) < 2.72549111968489

1.68594491579456 < H(20) < 2.51842763096328

1.47055810944939 < H(30) < 2.17271806567047

3. Використовуючи отримані значення ентропії, оцінити надлишковість російської мови в різних моделях джерела.





**TEXT:**

У файлі TEXT маємо такий алфавіт:

alphabet = 'абвгдежзийклмнопрстуфхцчшщыьэюя '

Отже, m = 32

H0 = log232 = 5

**Тоді, значення надлишковості російської мови за цим джерелом (TEXT):**

H(10): 45.49% < R < 61.97%

H(20): 49.63% < R < 66.28%

H(30): 56.54% < R < 70.59%

**Порахуємо надлишковість для тексту (Хроніки Амбера):**

H1: 4.360843323137924 (h1: spaces)

H0 = log232 = 5

R = 12.78%

H1: 4.453920329839864 (h1: no\_spaces)

H0 = log231 = 4.9542

R = 10.09%

H2: 3.9507997819187857 (h2: spaces + overlap)

H0 = log232 = 5

R = 20.98%

H2: 4.136805930152785 (h2: no spaces + overlap)

H0 = log231 = 4.9542

R = 16.5%

H2: 3.950485366384623 (h2: spaces + overlap)

H0 = log232 = 5

R = 21%

H2: 4.136328719898478 (h2: no spaces + no overlap)

H0 = log231 = 4.9542

R = 16.5%

Висновок:

Під час виконання даної лабораторної роботи, ми засвоїли поняття ентропії та надлишковості. Мовою Python написали кілька програм, які рахують частоту входження кожної букви в текст, та частоту біграм, а також Н1 та Н2. За отриманими значеннями розрахували ентропії H1 та H2. Застосувавши програму CoolPinkProgram.exe, знайшли межі умовної ентропії джерела. В решті решт, оцінили надлишковість російської мови в різних моделях джерела та отримали такі значення:

H(10): 45.49% < R < 61.97%

H(20): 49.63% < R < 66.28%

H(30): 56.54% < R < 70.59%

R = 12.78%

R = 10.09%

R = 20.98%

R = 16.5%

R = 21%

R = 16.5%