

SVEUČILIŠTE U RIJECI

**TEHNIČKI FAKULTET**

Preddiplomski sveučilišni studij računarstva

Izborni projekt

Programiranje II

**N-GRAMI SLOVA HRVATSKE WIKIPEDIJE**

Mentor: prof.dr.sc Ivo Ipšić

Rijeka, veljača 2022.

Jan Šubarić  
0069087527

# Sadržaj:

<b>1. UVOD.....</b>	<b>1</b>
1.1 UVOD U PROBLEMSKI ZADATAK .....	1
<b>2. RAZRADA PROBLEMA .....</b>	<b>2</b>
2.1. BASH SKRIPTA .....	2
2.2. FUNKCIJA U C PROGRAMSKOM JEZIKU .....	4
2.2.1 Programske knjižice i definiranje simboličkih konstanti.....	5
2.2.2 Main funkcija.....	5
2.2.3 Definiranje struktura .....	6
2.2.4 Funkcija za čitanje iz datoteke .....	6
2.2.5 Funkcija za ispis n-grama .....	7
2.2.6 Funkcija print_bigram_to_quintgram.....	8
2.3. POKRETANJE PROGRAMA.....	11
2.4. REZULTATI PROGRAMA .....	12
<b>3. ZAKLJUČAK .....</b>	<b>16</b>
<b>LITERATURA.....</b>	<b>17</b>

# 1. UVOD

## 1.1 Uvod u problemski zadatak

Za izradu izbornog projekta, na trećoj akademskoj godini, odabrao sam kolegij Programiranje II s kojim sam se upoznao u drugom semestru prve akademske godine. Profesor dr.sc. Ivo Ipšić, voditelj kolegija Programiranje II, zadao mi je zadatak za izborni projekt. Zadatak za izborni projekt je pokrio cjelokupno gradivo kolegija Programiranje II, najviše područje vezano uz znakovne nizove. Osim gradiva s kolegija Programiranje II, zadatak je obuhvatio i pisanje bash skripte s kojom sam se upoznao unutar kolegija Operacijski sustavi. Naslov zadatka izbornog projekta je glasio "N-grami slova hrvatske Wikipedije". Unutar zadatka, tražilo se pisanje bash skripte za preuzimanje članaka s hrvatske Wikipedije te ih je bilo potrebno pretvoriti u text format. Profesor je objasnio da se prilikom pisanja bash skripte i pretvaranja html formata u text format mogu koristiti rani alati u operativnom sustavu Linux. Neki od tih alata su wget, curl, pandoc te razni parseri prilikom rada s html i xml zapisima. Glavni dio zadatka odnosio se na implementaciju funkcije unutar C programskog jezika. Trebalo je napisati funkciju koja će iz zapisa tekstualnog formata odrediti statistiku pojavljivanja pojedinih kombinacija slova i izračunati njihovu vjerojatnost. Nakon navedenog izračuna, funkcija je trebala ispisati sve bigrame, trigrame, kvadgrame i kvintgrame pojavljivanja tih kombinacija slova te uz to i izračun njihove vjerojatnosti.

U nastavku ovog dokumenta detaljno su opisani koraci izrade i većina funkcija programa.

## 2. RAZRADA PROBLEMA

### 2.1. Bash skripta

Prije početka pisanja bash skripte, proučio sam alate koji su bili predloženi za korištenje prilikom parsiranja html i xml zapisa. Odlučio sam koristiti wget alat za preuzimanje članaka Wikipedije jer je zadovoljavao potrebe za rješavanje zadanog problema. Instalirao sam wget alat te provjerio njegovu verziju putem terminala u operativnom sustavu Linux. Također, provjerio sam i verzije drugih potrebnih alata kao što su 7z, lynx, perl, tr i tar. Nakon pripreme i provjere alata, započeo sam s pisanjem same skripte. Odlučio sam se na preuzimanje dump verzije članaka s hrvatske Wikipedije kako bi se izbjeglo opterećenje servera Wikipedije i "blacklistanje". Preuzimanje sam izvršio pomoću wget alata. Nakon preuzimanja, koje je zahtijevalo dulji vremenski period zbog količine podataka, raspakirao sam preuzeti sadržaj. Nakon toga mogao sam krenuti s parsiranjem sadržaja te sam tekst iz html datoteka spojio u jednu tekstualnu datoteku. Također, prilikom pisanja u tekstualnu datoteku, sva slova pretvorio sam u mala i očistio datoteku od nepotrebnih znakova.

```
#!/usr/bin/env/ bash

# korišteni alati:
# wget --version
# 7z --version
# lynx --version
# perl --version
# tr --version
# tar --version

# preuzimanje dumpa kako bi se izbjeglo preoptecenje servera i blaclistanje
echo 'Downloading hr.wiki dump...'
wget https://dumps.wikimedia.org/other/static_html_dumps/current/hr/wikipedia-hr-html.tar.7z

# extract preuzetog dumpa
echo 'Extracting hr.wiki dump...'
mkdir hr_wiki
7z x -so wikipedia-hr-html.tar.7z | tar xf - -C ./hr_wiki
```

Slika 2.1: Primjer koda za preuzimanje sadržaja s Wikipedije unutar bash skripte

Na slici 2.1 vidljiv je početak koda unutar bash skripte. Prva linija `#!/usr/bin/env/ bash` omogućuje fleksibilnost pokretanja skripte i na drugim operativnim sustavima s različitim lokacijama path-a. Pomoću komentara naveden je popis alata koji su korišteni unutar bash skripte. Korištenje `echo` funkcije omogućuje ispis poruke na sistemskoj konzoli. Uporaba `wget` komande s URL linkom omogućuje preuzimanje sadržaja dump-a hrvatske Wikipedije. Prilikom pokretanja wget alata i početka preuzimanja, wget prikazuje loading bar koji označava postotak skinutog sadržaja unutar systemske konzole. Također, vidljivi su ime datoteke koja se preuzima, njena veličina i brzina preuzimanja. Nakon što se preuzeo željeni sadržaj, pomoću komande `mkdir` stvoren je direktorij `hr_wiki` unutar kojeg će biti spremljen raspakiran sadržaj preuzetog sadržaja s Wikipedije. Sadržaj se raspakirao pomoću alata `7z` i tar pomoću naredbe spojene znakom `|` (engl. pipeline) `7z x -so wikipedia-hr-html.tar.7z | tar xf - -C ./hr_wiki`.

```
FNAME="combined.txt"

# parsiranje i spajanje teksta iz html datoteka u jednu datoteku
# konverzija slova u mala slova
shopt -s globstar
for page in **/*.html; do
    echo "Parsing: ${page}"
    lynx -dump -nolist ${page} | tr '[:upper:]' '[:lower:]' >> ${FNAME}
done

echo 'Removing chars different from [a-z]...'
# ascii a-z znakove - u jednom redu
perl -pi -e 's/[^a-z]+/ /g' ${FNAME}

# uklanjanje svih rijeci koje sadrže slova koja nisu dio hrvatske abecede
sed -i 's/[^ ]*x[^ ]*//ig' ${FNAME}
sed -i 's/[^ ]*y[^ ]*//ig' ${FNAME}
sed -i 's/[^ ]*w[^ ]*//ig' ${FNAME}
sed -i 's/[^ ]*q[^ ]*//ig' ${FNAME}
```

Ln 2, Col 1 | 100% | Unix (LF) | UTF-8

Slika 2.2: Prikaz ostatka koda unutar bash skripte

Pomoću varijable `FNAME`, postavlja se ime datoteke u koju će se spremati obrađeni tekst. U ovom slučaju ime datoteke je `combined.txt`. Komanda `shopt -s globstar` služi za aktiviranje opcije `globstar` koja omogućuje korištenje zamjenskih znakova za kasnije pretraživanje svih naziva datoteka i direktorija pomoću uzorka `**`. Zatim pomoću `for` petlje za svaki html dokument radimo parsiranje i spajanje svih tekstova u jednu tekstualnu datoteku. Komanda `lynx -dump -nolist ${page} | tr '[:upper:]' '[:lower:]' >> ${FNAME}` služi za parsiranje html dokumenta bez html oznaka, a komanda `tr` istovremeno pretvara sva velika slova u mala te se sve zapisuje u jednu tekstualnu datoteku `combined.txt`. Nakon toga komandom `perl -pi -e 's/[^a-z]+/ /g' ${FNAME}` omogućuje uklanjanje svih znakova koji nisu dio standardne abecede uz pomoć opcije `substitute` `s`. Opcija `g` omogućuje označavanje svih stringova koji sadrže znakove koji su različiti od znakova abecede. Opcija `-pi` komande `perl` omogućuje stvaranje petlje za ispisivanje preuređene linije u tekstualnu datoteku bez stvaranja kopija originala, dok opcija `-e` daje mogućnost da se `perl` pozove samo jednom unutar programa. Posljednje četiri komande služe za uklanjanje svih riječi koje u sebi sadrže znakove koji nisu dio hrvatske abecede, kao npr. `x`, `y`, `w`, `q`.

Komanda `sed -i 's/[^ ]*x[^ ]*//ig' ${FNAME}` uklanja sve riječi koje sadrže slovo `x` na početku, u sredini ili na kraju riječi te se zbog opcije `ig` odnosi i na sve sljedeće znakovne nizove, a ne samo na prvi. Ostale komande funkcioniraju na isti način.

## 2.2. Funkcija u C programskom jeziku

Nakon što sam završio s pisanjem bash skripte, krenuo sam rješavati glavni dio zadatka. Kao razvojnu okolinu za pisanje koda unutar C programskog jezika odabrao sam softversku aplikaciju Sublime. Sublime je razvojna okolina koju sam upoznao prilikom pohađanja kolegija Programiranje I i II. Prije samog pisanja koda, razmišljao sam o problemu i mogućnostima njegovog rješavanja. Nakon što sam razradio način na koji mogu riješiti problem, započeo sam s njegovom implementacijom unutar C programskog jezika.

Kao najvažnije dijelove koda, izdvojio bih određene funkcije. Funkcija čitanja iz datoteke bila je potrebna kako bih unutar buffera zapisao sve ono što se u tekstualnoj datoteci nalazi te kako bismo s istim tim podacima mogli nastaviti raditi. Unutar te funkcije, alocirana je memorija za buffer te su korištene brojne funkcije vezane uz datoteku, kao što su otvaranje, čitanje i zatvaranje datoteke. Funkcije za ispih pojedinih n-grama bile su od izuzetne važnosti za ispih svih kombinacija slova i njihovih vjerojatnosti. Također, jedna od najvažnijih funkcija je i ona koja se koristila za brojanje ponavljanja određenih kombinacija slova te računanja njihove vjerojatnosti. Unutar te funkcije, korištene su brojne naredbe za alociranje memorije prilikom računanja broja n-grama, strukture za upis dobivenih rezultata, funkcije za ispis te isto tako oslobađanje te iste memorije nakon ispisa.

Kako bih olakšao pisanje koda, koristio sam strukture bigrama, trigrama, kvadgrama i kvintgrama gdje sam zapisao sve njihove moguće permutacije. Također, definirao sam i određene konstante za dužinu pojedinih n-grama kako ih ne bih morao unositi ručno. U glavnom programu definirao sam pokazivač na datoteku pročitane pomoću funkcije te pozvao funkciju "print\_bigram\_to\_quintgram(hr\_wiki)" koja je ispisala sve kombinacije bigrama, trigrama, kvadgrama i kvintgrama te vjerojatnosti njihove pojave. Također, nakon završetka izvođenja funkcije, oslobodio sam memoriju. Prilikom pisanja koda, uključene su i knjižice koje su bile neophodne za njegovo izvođenje.

U nastavku su opisane sve pojedinosti određenih funkcija i ostale komponente koje su korištene prilikom stvaranja programa.

### 2.2.1 Programske knjižice i definiranje simboličkih konstanti

Na slici 2.5 naveden je početak koda unutar datoteke `main.c`. Prve četiri linije odnose se na unose programskih knjižica neophodnih za izvršavanje cjelokupnog programa. Nakon toga se pomoću `#define` definiraju određene simboličke konstante za lakše razumijevanje koda i snalaženje unutar koda.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>

#define FNAME "combined.txt"

#define BIGRAM_LEN      2
#define TRIGRAM_LEN     3
#define QUADGRAM_LEN    4
#define QUINTGRAM_LEN   5
```

Slika 2.5: Programske knjižice i definiranje simboličkih konstanti

### 2.2.2 Main funkcija

Na slici 2.4 prikazan je kod unutar glavne `main` funkcije programa. U prvoj liniji `main` funkcije se deklarira i inicijalizira pokazivač na polje znakova koji se spremaju u memoriju prilikom čitanja iz tekstualne datoteke uz pomoću funkcije `read_file(FNAME)`. Ispod deklaracije pokazivača, poziva se najvažnija funkcija `print_bigram_to_quintgram(hr_wiki)` koja će biti objašnjena u nastavku ovog dokumenta. Nakon uspješnog izvršavanja navedene funkcije, oslobađa se memorija zauzeta znakovima iz tekstualne datoteke s komandom `free(hr_wiki)`. Komanda `return EXIT_SUCCESS` označava uspješan završetak cijelog programa.

```
int main()
{
    char *hr_wiki = read_file(FNAME);

    print_bigram_to_quintgram(hr_wiki);

    free(hr_wiki);

    return EXIT_SUCCESS;
}
```

Ln 1, Col 1

100%

Unix (LF)

UTF-8

Slika 1.4: Prikaz `main` funkcije unutar programa

### 2.2.3 Definiranje struktura

Na slici 2.6 prikazano je definiranje struktura potrebnih za izradu programa. Definiraju se strukture za bigrame, trigrame, kvadgrame i kvintgrame. Svaka struktura n-grama sadrži polje veličine n za spremanje svih mogućih permutacija te jednu varijablu za spremanje broja n-grama. Na slici su prikazane strukture za bigram i trigram. Na isti način izrađene su i strukture za kvadgrame i kvintgrame.

```
struct bigram {
    /**2D" polje za bigrame*/
    unsigned long long (*perms)[ALPHABET_LEN];
    /* broj bigrama*/
    unsigned long long n;
};

struct trigram {
    /**3D" polje za trigrame*/
    unsigned long long (*perms)[ALPHABET_LEN][ALPHABET_LEN];
    unsigned long long n;
};
```

Slika 2.6: Definiranje struktura n-grama

### 2.2.4 Funkcija za čitanje iz datoteke

Na slici 2.7 prikazana je funkcija za čitanje sadržaja iz tekstualne datoteke. Ovo je jedna od najvažnijih funkcija unutar cijelog programa. Potpis funkcije je *char\* read\_file(char \*filename)*. Funkcija vraća pokazivač na polje znakova u main funkciju gdje je i pozvana, dok kao argument prima ime datoteke koje je deklarirano pomoću makroa. Unutar funkcije se deklarira pokazivač buffer tipa char i postavlja se na NULL vrijednost. Buffer će služiti kao spremnik za spremanje znakova pročitanih iz tekstualne datoteke. Deklariraju se varijable *string\_size* i *read\_size* za unos veličina znakovnih nizova. Potrebno je deklarirati i pokazivač na otvorenu datoteku *\*handler*. Pomoću *\*handlera* otvaramo željenu datoteku, u ovom slučaju *combined.txt*, za čitanje pomoću naredbe *fopen(filename, "r")*. Unutar if petlje, koja se izvršava dok se ne dođe do kraja datoteke, zapisuju se znakovi iz datoteke u buffer.

Funkcija *fseek(handler, 0, SEEK\_END)* postavlja pokazivač na otvorenu datoteku na kraj datoteke. Unutar varijable *string\_size* zapisuje se trenutni položaj unutar datoteke, odnosno broj znakova do kraja datoteke. To je implementirano pomoću *string\_size = ftell(handler)*. Pomoću funkcije *rewind(handler)* vraćamo pokazivač na početak datoteke. Pomoću funkcije *malloc* alocira se memorija unutar buffera potrebna za unos svih znakova iz datoteke. Funkcija *fread(buffer, sizeof(char), string\_size, handler)* čita iz datoteke na koju pokazuje handler, odnosno iz *combined.txt*, i smješta znakove u polje na koje pokazuje buffer. Također, funkcija vraća i broj pročitanih objekata te ih sprema u varijablu *read\_size*. Na kraj buffera postavlja se null terminator. Ako je broj znakova (*string\_size*) iz datoteke različit od broja pročitanih znakova (*read\_size*), došlo je do greške prilikom čitanja iz datoteke te se oslobađa memorija na koju pokazuje buffer i buffer se postavlja na NULL vrijednost. Po završetku if uvjeta, datoteka se zatvara pomoću funkcije *fclose(handler)* koja kao argument prima pokazivač na datoteku. Na samom kraju, vraća se vrijednost buffera s naredbom *return buffer* u main funkciju od kuda je funkcija i pozvana.



```

/*
 *
 * funkcija za citanje iz datoteke
 *
 */
char* read_file(char *filename)
{
    char *buffer = NULL;
    unsigned long long string_size, read_size;
    FILE *handler = fopen(filename, "r");

    if (handler) {
        fseek(handler, 0, SEEK_END);
        string_size = ftell(handler);
        rewind(handler);

        buffer = (char*) malloc(sizeof(char) * (string_size + 1) );

        read_size = fread(buffer, sizeof(char), string_size, handler);

        buffer[string_size] = '\0';

        if (string_size != read_size) {
            free(buffer);
            buffer = NULL;
        }

        fclose(handler);
    }

    return buffer;
}

```

Slika 2.7: Funkcija za čitanje iz datoteke

### 2.2.5 Funkcija za ispis n-grama

U nastavku, na slici 2.8 je prikazana funkcija za ispis bigrama i trigrama. Funkcije za ispis kvadgrama i kvintgrama funkcioniraju na sličan način. U nastavku je objašnjena funkcija za ispis bigrama. Potpis funkcije za ispis bigrama je *void print\_bigram(struct bigram bi)*. Funkcija je tipa void te ne vraća vrijednost, a kao argument prima strukturu bigram, definiranu ranije u kodu. Sama uloga funkcije je ispis bigrama. Na početku funkcije ispisuje se "BIGRAM:" pomoću naredbe *puts("BIGRAM:");*. Uz pomoć ugniježdene for petlje, ispisuju se sve permutacije bigrama. Vanjska petlja služi za zapisivanje prvog znaka bigrama, dok se unutarnja petlja koristi za zapis drugog znaka bigrama. Unutar unutarnje petlje poziva se funkcija *printf("%c%c: %.4f ", i+'a', j+'a', bi.perms[i][j] / (double)bi.n)* koja ispisuje određeni bigram tako što se varijabla *i* i *j* inkrementira slovom 'a' da bi se dobila ASCII vrijednost i decimalni broj zaokružen na četiri decimale koji predstavlja vjerojatnost pojave tog bigrama. Navedeni izraz u printf funkciji *bi.perms[i][j] / (double)bi.n* računa vjerojatnost. Pomoću naredbe *putchar('\n')*, ispisuje se novi red nakon završetka unutarnje for petlje. Petlje se izvršavaju sve dok je kontrolna varijabla manja od veličine simboličke konstante ALPHABET\_LEN koja označava broj slova između 'a' i 'z'.

Ispisi ostalih n-grama funkcioniraju na sličan način, no kod svakog višeg n-grama dodaje se jedna dodatna unutarnja petlja.

```

void print_bigram(struct bigram bi)
{
    puts("BIGRAM:");
    for (size_t i = 0; i < ALPHABET_LEN; i++) {
        for (size_t j = 0; j < ALPHABET_LEN; j++)
            printf("%c%c: %.4f ", i+'a', j+'a', bi.perms[i][j] / (double)bi.n);
        putchar('\n');
    }
}

void print_trigram(struct trigram tri)
{
    puts("TRIGRAM:");
    for (size_t i = 0; i < ALPHABET_LEN; i++) {
        for (size_t j = 0; j < ALPHABET_LEN; j++) {
            for (size_t k = 0; k < ALPHABET_LEN; k++)
                printf("%c%c%c: %.4f ", i+'a', j+'a', k+'a', tri.perms[i][j][k] /
                    (double)tri.n);
            putchar('\n');
        }
        putchar('\n');
    }
}

```

Slika 2.8: Ispis n-grama

### 2.2.6 Funkcija `print_bigram_to_quintgram`

Na slici 2.9 vidljiv je prvi dio funkcije `print_bigram_to_quintgram`. Funkcija ima potpis `void print_bigram_to_quintgram(char *buf)`. Funkcija ne vraća podatak jer je tipa `void`, a kao argument prima pokazivač na polje znakova, u ovom slučaju na zapis iz pročitane datoteke. Funkcija se poziva unutar `main` funkcije. U prvom dijelu funkcije alokira se memorija za svaki pojedini n-gram spremljen u obliku strukture. Primjer alokacije objašnjen je na primjeru bigrama, dok je za sve ostale n-grame objašnjenje slično.

Unutar varijable `bi_sz` sprema se veličina memorije zauzeta od svih mogućih permutacija bigrama pomoću naredbe `sizeof(unsigned long long[ALPHABET_LEN][ALPHABET_LEN])`. Nakon toga se za varijablu `bi.perms` alokira memorija s naredbom `malloc(bi_sz)` koja kao argument prima prethodno zadanu veličinu memorije broja svih mogućih permutacija bigrama. Nakon toga se prethodno zauzeta memorija postavlja na 0 pomoću komande `memset(bi.perms, 0, bi_sz)`.

Prethodno navedeno vrijedi i za ostale n-grame, samo se svaki viši n-gram razlikuje u jednoj dodatnoj dimenziji polja.

```

/*
 *
 * racunanje broja kombinacija i ispis
 *
 */
void print_bigram_to_quintgram(char *buf)
{
    struct bigram bi = {};
    /* alociranje memorije */
    unsigned long long bi_sz = sizeof(unsigned long long[ALPHABET_LEN][ALPHABET_LEN]);
    bi.perms = malloc(bi_sz);
    memset(bi.perms, 0, bi_sz);

    struct trigram tri = {};
    unsigned long long tri_sz = sizeof(unsigned long long[ALPHABET_LEN][ALPHABET_LEN][ALPHABET_LEN]);
    tri.perms = malloc(tri_sz);
    memset(tri.perms, 0, tri_sz);

    struct quadgram quad = {};
    unsigned long long quad_sz = sizeof(unsigned long long[ALPHABET_LEN][ALPHABET_LEN][ALPHABET_LEN][ALPHABET_LEN]);
    quad.perms = malloc(quad_sz);
    memset(quad.perms, 0, quad_sz);

    struct quintgram quint = {};
    unsigned long long quint_sz = sizeof(unsigned long long[ALPHABET_LEN][ALPHABET_LEN][ALPHABET_LEN][ALPHABET_LEN][ALPHABET_LEN]);
    quint.perms = malloc(quint_sz);
    memset(quint.perms, 0, quint_sz);

    unsigned long long buflen = strlen(buf);

```

Slika 2.9: Alociranje memorije n-grama unutar funkcije

Na slici 2.10 vidljiv je dio funkcije za brojanje broja svih permutacija n-grama koje se pojavljuju unutar tekstualne datoteke combined.txt. Deklarirana je varijabla *buflen* i inicijalizirana je na duljinu danog znakovnog niza. To je učinjeno pomoću naredbe *strlen(buf)* koja kao argument prima pokazivač na znakovni niz, a vraća dužinu istog tog znakovnog niza.

Zadana je for petlja koja se izvodi sve dok je varijabla *i* manja od varijable *buflen*, odnosno dok se ne dođe do kraja znakovnog niza. Unutar for petlje stoje četiri if uvjetne naredbe za svaki n-gram posebno. Svaka if uvjetna naredba služi za izračun broja određene permutacije određenog n-grama.

U primjeru će biti objašnjen if uvjet za izračun broja određenog bigrama. If uvjet za izračun bigrama je napisan u ovom obliku *if (i + BIGRAM\_LEN - 1 < buflen && islower(buf[i]) && islower(buf[i+1]))*. Prvi if uvjet provjerava je li veličina zadana ovim izrazom  $(i + \text{BIGRAM\_LEN} - 1)$  manja od veličine znakovnog niza, odnosno od varijable *buflen*. U slučaju da veličina nije manja, program ne ulazi u if naredbu, što znači da je došao do kraja znakovnog niza i da ne može više pročitati niti jedan bigram. Kao napomenu treba navesti da se u izrazu oduzima jedinica zbog null terminatora, varijabla *i* se koristi za šetnju kroz znakovni niz, dok se *BIGRAM\_LEN* koristi za određivanje veličine n-grama, u ovom slučaju bigrama (2). Ostala dva uvjeta unutar if naredbe provjeravaju jesu li znakovi na pozicijama *buf[i]* i *buf[i+1]* mala slova. Sva tri uvjeta moraju biti ispunjena kako bi program ušao u if naredbu, u protivnom se ispunjava uvjetna naredba *else* i program nastavlja s izvođenjem. Ako su if uvjeti ispunjeni, inkrementira se određena permutacija bigrama i povećava se ukupan broj bigrama unutar strukture *bigram*. Struktura bigrama je deklarirana i njena je memorija alocirana u prethodnom koraku.

Sličan opis se odnosi i na izračun ostalih n-grama.

```

unsigned long long buflen = strlen(buf);

for (unsigned long long i = 0; i < buflen; i++) {

    if (i + BIGRAM_LEN - 1 < buflen
        && islower(buf[i]) && islower(buf[i+1])) {
        bi.perms[buf[i]-'a'][buf[i+1]-'a']++;
        bi.n++;
    }
    else
        continue;

    if (i + TRIGRAM_LEN - 1 < buflen
        && islower(buf[i+2])) {
        tri.perms[buf[i]-'a'][buf[i+1]-'a'][buf[i+2]-'a']++;
        tri.n++;
    }
    else
        continue;

    if (i + QUADGRAM_LEN - 1 < buflen
        && islower(buf[i+3])) {
        quad.perms[buf[i]-'a'][buf[i+1]-'a'][buf[i+2]-'a'][buf[i+3]-'a']++;
        quad.n++;
    }
    else
        continue;

    if (i + QUINTGRAM_LEN - 1 < buflen
        && islower(buf[i+4])) {
        quint.perms[buf[i]-'a'][buf[i+1]-'a'][buf[i+2]-'a'][buf[i+3]-'a'][buf[i+4]-'a']++;
        quint.n++;
    }
}

```

Slika 2.10: Računanje broja n-grama unutar funkcije

Na slici 2.11 prikazan je kraj funkcije *print\_bigram\_to\_quintgram*. Na kraju se pozivaju prethodno objašnjene funkcije za ispis pojedinih n-grama koje kao argument primaju strukturu za određeni n-gram. Nakon ispisa svih bigrama, trigrama, kvadgrama i kvintgrama, oslobađa se memorija. Memorija se oslobađa s naredbom *free(bi.perms)* koja prima pokazivač na alociranu memoriju za permutacije unutar strukture.

```

print_bigram(bi);
print_trigram(tri);
print_quadgram(quad);
print_quintgram(quint);

/* oslobađanje memorije */
free(bi.perms);
free(tri.perms);
free(quad.perms);
free(quint.perms);
}

```

Slika 2.11: Ispisivanje n-grama i oslobađanje memorije na kraju funkcije

### 2.3. Pokretanje programa

Prije pokretanja bash skripte potrebno je provjeriti verzije korištenih alata navedenih na početku skripte.

Pokretanje skripte s naredbom u terminalu: *bash get\_hr\_wiki\_data.sh*.

Prije samog pokretanja C programa, potrebno ga je buildati s naredbom: *gcc main.c*.

U slučaju da se program ne builda s prethodnom naredbom, potrebno je instalirati gdb te buildati s naredbom: *gcc -ggdb main.c*. Nakon toga unesti naredbu: *gdb ./a.out* i onda naredbu *run* unutar debuggera.

Može se koristiti dodatno naredba za ispis standardnog izlaza u datoteku: *./a.out > rez.txt*.

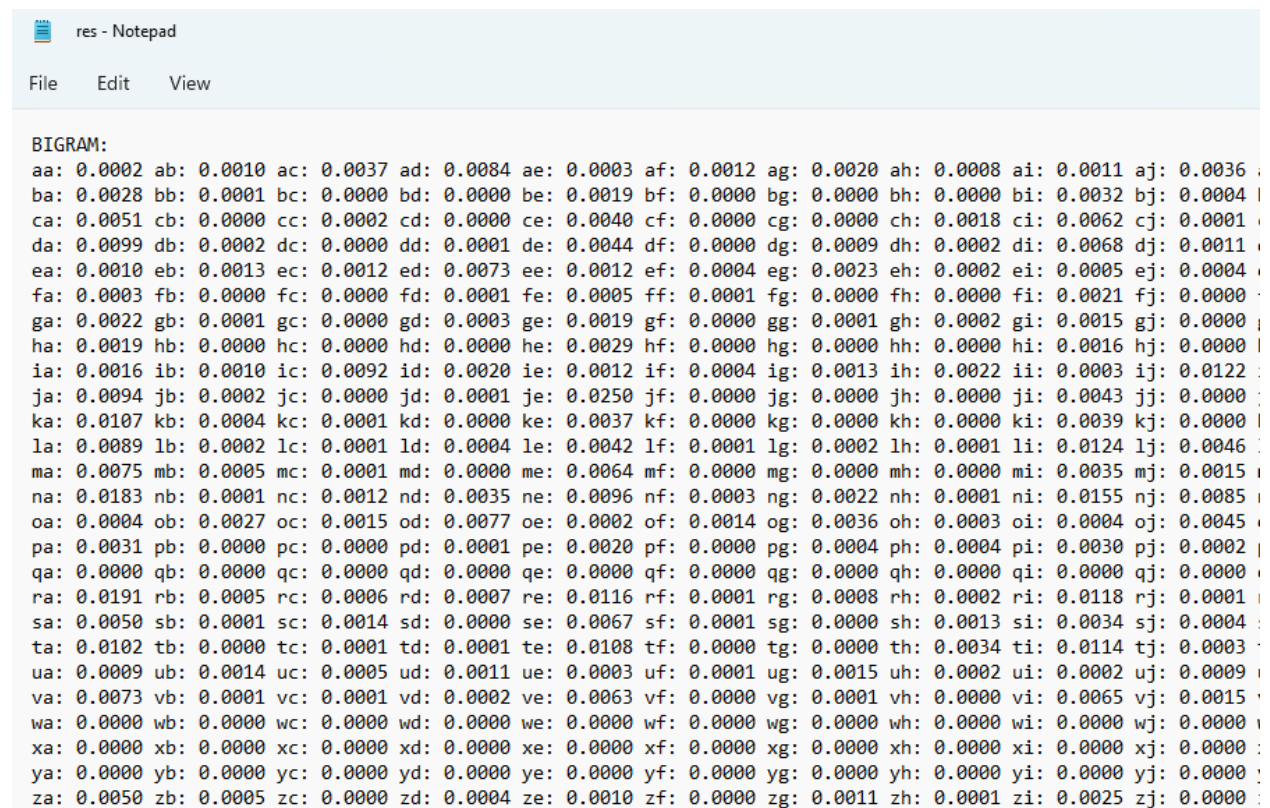
## 2.4. Rezultati programa

Kao rezultat programa, dobiva se ispis svih permutacija bigrama, trigrama, kvadgrama i kvintgrama i broj njihove vjerojatnosti pojavljivanja unutar tekstualne datoteke combined.txt koja je nastala parsiranjem i modificiranjem članaka s hrvatske Wikipedije unutar bash skripte.

Zbog boljeg prikaza rezultata, rezultati su ispisani u tekstualnu datoteku rez.txt pomoću naredbe unutar konzole za ispis standardnog izlaza u vanjsku datoteku: `./a.out > rez.txt`.

Na slikama 2.12, 2.13, 2.14, 2.15 vidljiv je dio ispisa rezultata nekih bigrama, trigrama, kvadgrama i kvintgrama s njihovim vjerojatnostima pojavljivanja.

U tekstu je pronađeno ukupno 484 različitih bigrama.



```
res - Notepad
File Edit View

BIGRAM:
aa: 0.0002 ab: 0.0010 ac: 0.0037 ad: 0.0084 ae: 0.0003 af: 0.0012 ag: 0.0020 ah: 0.0008 ai: 0.0011 aj: 0.0036
ba: 0.0028 bb: 0.0001 bc: 0.0000 bd: 0.0000 be: 0.0019 bf: 0.0000 bg: 0.0000 bh: 0.0000 bi: 0.0032 bj: 0.0004
ca: 0.0051 cb: 0.0000 cc: 0.0002 cd: 0.0000 ce: 0.0040 cf: 0.0000 cg: 0.0000 ch: 0.0018 ci: 0.0062 cj: 0.0001
da: 0.0099 db: 0.0002 dc: 0.0000 dd: 0.0001 de: 0.0044 df: 0.0000 dg: 0.0009 dh: 0.0002 di: 0.0068 dj: 0.0011
ea: 0.0010 eb: 0.0013 ec: 0.0012 ed: 0.0073 ee: 0.0012 ef: 0.0004 eg: 0.0023 eh: 0.0002 ei: 0.0005 ej: 0.0004
fa: 0.0003 fb: 0.0000 fc: 0.0000 fd: 0.0001 fe: 0.0005 ff: 0.0001 fg: 0.0000 fh: 0.0000 fi: 0.0021 fj: 0.0000
ga: 0.0022 gb: 0.0001 gc: 0.0000 gd: 0.0003 ge: 0.0019 gf: 0.0000 gg: 0.0001 gh: 0.0002 gi: 0.0015 gj: 0.0000
ha: 0.0019 hb: 0.0000 hc: 0.0000 hd: 0.0000 he: 0.0029 hf: 0.0000 hg: 0.0000 hh: 0.0000 hi: 0.0016 hj: 0.0000
ia: 0.0016 ib: 0.0010 ic: 0.0092 id: 0.0020 ie: 0.0012 if: 0.0004 ig: 0.0013 ih: 0.0022 ii: 0.0003 ij: 0.0122
ja: 0.0094 jb: 0.0002 jc: 0.0000 jd: 0.0001 je: 0.0250 jf: 0.0000 jg: 0.0000 jh: 0.0000 ji: 0.0043 jj: 0.0000
ka: 0.0107 kb: 0.0004 kc: 0.0001 kd: 0.0000 ke: 0.0037 kf: 0.0000 kg: 0.0000 kh: 0.0000 ki: 0.0039 kj: 0.0000
la: 0.0089 lb: 0.0002 lc: 0.0001 ld: 0.0004 le: 0.0042 lf: 0.0001 lg: 0.0002 lh: 0.0001 li: 0.0124 lj: 0.0046
ma: 0.0075 mb: 0.0005 mc: 0.0001 md: 0.0000 me: 0.0064 mf: 0.0000 mg: 0.0000 mh: 0.0000 mi: 0.0035 mj: 0.0015
na: 0.0183 nb: 0.0001 nc: 0.0012 nd: 0.0035 ne: 0.0096 nf: 0.0003 ng: 0.0022 nh: 0.0001 ni: 0.0155 nj: 0.0085
oa: 0.0004 ob: 0.0027 oc: 0.0015 od: 0.0077 oe: 0.0002 of: 0.0014 og: 0.0036 oh: 0.0003 oi: 0.0004 oj: 0.0045
pa: 0.0031 pb: 0.0000 pc: 0.0000 pd: 0.0001 pe: 0.0020 pf: 0.0000 pg: 0.0004 ph: 0.0004 pi: 0.0030 pj: 0.0002
qa: 0.0000 qb: 0.0000 qc: 0.0000 qd: 0.0000 qe: 0.0000 qf: 0.0000 qg: 0.0000 qh: 0.0000 qi: 0.0000 qj: 0.0000
ra: 0.0191 rb: 0.0005 rc: 0.0006 rd: 0.0007 re: 0.0116 rf: 0.0001 rg: 0.0008 rh: 0.0002 ri: 0.0118 rj: 0.0001
sa: 0.0050 sb: 0.0001 sc: 0.0014 sd: 0.0000 se: 0.0067 sf: 0.0001 sg: 0.0000 sh: 0.0013 si: 0.0034 sj: 0.0004
ta: 0.0102 tb: 0.0000 tc: 0.0001 td: 0.0001 te: 0.0108 tf: 0.0000 tg: 0.0000 th: 0.0034 ti: 0.0114 tj: 0.0003
ua: 0.0009 ub: 0.0014 uc: 0.0005 ud: 0.0011 ue: 0.0003 uf: 0.0001 ug: 0.0015 uh: 0.0002 ui: 0.0002 uj: 0.0009
va: 0.0073 vb: 0.0001 vc: 0.0001 vd: 0.0002 ve: 0.0063 vf: 0.0000 vg: 0.0001 vh: 0.0000 vi: 0.0065 vj: 0.0015
wa: 0.0000 wb: 0.0000 wc: 0.0000 wd: 0.0000 we: 0.0000 wf: 0.0000 wg: 0.0000 wh: 0.0000 wi: 0.0000 wj: 0.0000
xa: 0.0000 xb: 0.0000 xc: 0.0000 xd: 0.0000 xe: 0.0000 xf: 0.0000 xg: 0.0000 xh: 0.0000 xi: 0.0000 xj: 0.0000
ya: 0.0000 yb: 0.0000 yc: 0.0000 yd: 0.0000 ye: 0.0000 yf: 0.0000 yg: 0.0000 yh: 0.0000 yi: 0.0000 yj: 0.0000
za: 0.0050 zb: 0.0005 zc: 0.0000 zd: 0.0004 ze: 0.0010 zf: 0.0000 zg: 0.0011 zh: 0.0001 zi: 0.0025 zj: 0.0000
```

Slika 2.12: Ispis bigrama i njihovih vjerojatnosti

U tekstu je pronađeno ukupno 9929 različitih trigrami.

```
TRIGRAM:|
aaa: 0.0000 aab: 0.0000 aac: 0.0000 aad: 0.0000 aae: 0.0000 aaf: 0.0000 aag: 0.0000 aah: 0.0000 aai: 0.0000
aba: 0.0001 abb: 0.0000 abc: 0.0000 abd: 0.0000 abe: 0.0001 abf: 0.0000 abg: 0.0000 abh: 0.0000 abi: 0.0002
aca: 0.0002 acb: 0.0000 acc: 0.0001 acd: 0.0000 ace: 0.0001 acf: 0.0000 acg: 0.0000 ach: 0.0001 aci: 0.0032
ada: 0.0017 adb: 0.0000 adc: 0.0000 add: 0.0000 ade: 0.0004 adf: 0.0000 adg: 0.0000 adh: 0.0000 adi: 0.0007
aea: 0.0000 aeb: 0.0000 aec: 0.0000 aed: 0.0000 aee: 0.0000 aef: 0.0000 aeg: 0.0000 aeh: 0.0000 aei: 0.0001
afa: 0.0000 afb: 0.0000 afc: 0.0000 afd: 0.0000 afe: 0.0000 aff: 0.0000 afg: 0.0000 afh: 0.0000 afi: 0.0011
aga: 0.0002 agb: 0.0000 agc: 0.0000 agd: 0.0000 age: 0.0010 agf: 0.0000 agg: 0.0000 agh: 0.0000 agi: 0.0001
aha: 0.0005 ahb: 0.0000 ahc: 0.0000 ahd: 0.0000 ahe: 0.0001 ahf: 0.0000 ahg: 0.0000 ahh: 0.0000 ahi: 0.0000
aia: 0.0000 aib: 0.0000 aic: 0.0000 aid: 0.0000 aie: 0.0000 aif: 0.0000 aig: 0.0000 aih: 0.0000 aii: 0.0000
aja: 0.0004 ajb: 0.0001 ajc: 0.0000 ajd: 0.0000 aje: 0.0007 ajf: 0.0000 ajg: 0.0000 ajh: 0.0000 aji: 0.0008
aka: 0.0006 akb: 0.0000 akc: 0.0001 akd: 0.0000 ake: 0.0003 akf: 0.0000 akg: 0.0000 akh: 0.0000 aki: 0.0001
ala: 0.0014 alb: 0.0002 alc: 0.0000 ald: 0.0001 ale: 0.0008 alf: 0.0000 alg: 0.0000 alh: 0.0002 ali: 0.0022
ama: 0.0010 amb: 0.0001 amc: 0.0000 amd: 0.0000 ame: 0.0007 amf: 0.0000 amg: 0.0000 amh: 0.0000 ami: 0.0003
ana: 0.0026 anb: 0.0000 anc: 0.0007 and: 0.0019 ane: 0.0004 anf: 0.0000 ang: 0.0002 anh: 0.0000 ani: 0.0044
aoa: 0.0000 aob: 0.0000 aoc: 0.0000 aod: 0.0000 aoe: 0.0000 aof: 0.0000 aog: 0.0000 aoh: 0.0000 aoi: 0.0000
apa: 0.0004 apb: 0.0000 apc: 0.0000 apd: 0.0000 ape: 0.0001 apf: 0.0000 apg: 0.0000 aph: 0.0001 api: 0.0004
aqa: 0.0000 aqb: 0.0000 aqc: 0.0000 aqd: 0.0000 aqe: 0.0000 aqf: 0.0000 aqg: 0.0000 aqh: 0.0000 aqi: 0.0000
ara: 0.0013 arb: 0.0001 arc: 0.0002 ard: 0.0004 are: 0.0006 arf: 0.0000 arg: 0.0001 arh: 0.0001 ari: 0.0013
asa: 0.0007 asb: 0.0000 asc: 0.0000 asd: 0.0000 ase: 0.0004 asf: 0.0000 asg: 0.0000 ash: 0.0002 asi: 0.0002
ata: 0.0009 atb: 0.0000 atc: 0.0001 atd: 0.0000 ate: 0.0022 atf: 0.0000 atg: 0.0000 ath: 0.0001 ati: 0.0044
aua: 0.0000 aub: 0.0000 auc: 0.0000 aud: 0.0001 aue: 0.0000 auf: 0.0000 aug: 0.0001 auh: 0.0000 auj: 0.0000
ava: 0.0016 avb: 0.0000 avc: 0.0000 avd: 0.0000 ave: 0.0007 avf: 0.0000 avg: 0.0000 avh: 0.0000 avi: 0.0014
awa: 0.0000 awb: 0.0000 awc: 0.0000 awd: 0.0000 awe: 0.0000 awf: 0.0000 awg: 0.0000 awh: 0.0000 awi: 0.0000
axa: 0.0000 axb: 0.0000 axc: 0.0000 axd: 0.0000 axe: 0.0000 axf: 0.0000 axg: 0.0000 axh: 0.0000 axi: 0.0000
aya: 0.0000 ayb: 0.0000 ayc: 0.0000 ayd: 0.0000 aye: 0.0000 ayf: 0.0000 ayg: 0.0000 ayh: 0.0000 ayi: 0.0000
aza: 0.0004 azb: 0.0002 azc: 0.0000 azd: 0.0001 aze: 0.0001 azf: 0.0000 azg: 0.0012 azh: 0.0000 azi: 0.0009

baa: 0.0000 bab: 0.0000 bac: 0.0002 bad: 0.0000 bae: 0.0000 baf: 0.0000 bag: 0.0000 bah: 0.0004 bai: 0.0000
bba: 0.0000 bbb: 0.0000 bbc: 0.0000 bbd: 0.0000 bbe: 0.0000 bbf: 0.0000 bbg: 0.0000 bbh: 0.0000 bbi: 0.0000
bca: 0.0000 bcb: 0.0000 bcc: 0.0000 bcd: 0.0000 bce: 0.0000 bcf: 0.0000 bcg: 0.0000 bch: 0.0000 bci: 0.0000
bda: 0.0000 bdb: 0.0000 bdc: 0.0000 bdd: 0.0000 bde: 0.0000 bdf: 0.0000 bdg: 0.0000 bdh: 0.0000 bdi: 0.0000
bea: 0.0001 beb: 0.0000 bec: 0.0001 bed: 0.0000 bee: 0.0001 bef: 0.0000 beg: 0.0000 beh: 0.0000 bei: 0.0000
bfa: 0.0000 bfb: 0.0000 bfc: 0.0000 bfd: 0.0000 bfe: 0.0000 bff: 0.0000 bfg: 0.0000 bfh: 0.0000 bfi: 0.0000
```

Slika 2.13: Ispis trigrami i njihovih vjerojatnosti



U tekstu je pronađeno ukupno 90415 različitih kvadograma.

QUADGRAM:

aaaa: 0.00001389	aaab: 0.00000002	aaac: 0.00000000	aaad: 0.00000001	aaae: 0.00000000	aaaf: 0.00000109	aaag: 0.00000001
aaba: 0.00000045	aabb: 0.00000002	aabc: 0.00000002	aabd: 0.00000001	aabe: 0.00000039	aabf: 0.00000000	aabg: 0.00000000
aaca: 0.00000025	aacb: 0.00000000	aacc: 0.00000003	aacd: 0.00000000	aace: 0.00000008	aacf: 0.00000001	aacg: 0.00000000
aada: 0.00000029	aadb: 0.00000000	aadc: 0.00000000	aadd: 0.00000000	aade: 0.00000051	aadf: 0.00000000	aadg: 0.00000000
aaea: 0.00000000	aaeb: 0.00000000	aaec: 0.00000001	aaed: 0.00000000	aaee: 0.00000003	aaef: 0.00000000	aaeg: 0.00000000
aafa: 0.00000038	aafb: 0.00000003	aafc: 0.00000000	aafd: 0.00000000	aafe: 0.00000001	aaff: 0.00000035	aafg: 0.00000000
aaga: 0.00000033	aagb: 0.00000000	aagc: 0.00000000	aagd: 0.00000002	aage: 0.00000089	aagf: 0.00000000	aagg: 0.00000001
aaha: 0.00000007	aahb: 0.00000006	aahc: 0.00000001	aahd: 0.00000034	aahe: 0.00000007	aahf: 0.00000000	aahg: 0.00000000
aaia: 0.00000000	aaib: 0.00000000	aaic: 0.00000000	aaid: 0.00000000	aaie: 0.00000000	aaif: 0.00000000	aaig: 0.00000000
aaja: 0.00000010	aajb: 0.00000000	aajc: 0.00000000	aajd: 0.00000001	aaje: 0.00000002	aajf: 0.00000000	aaig: 0.00000000
aaka: 0.00000016	aakb: 0.00000001	aakc: 0.00000001	aakd: 0.00000000	aake: 0.00000031	aakf: 0.00000000	aakg: 0.00000000
aala: 0.00000097	aalb: 0.00000078	aalc: 0.00000001	aald: 0.00000007	aale: 0.00000049	aalf: 0.00000010	aalg: 0.00000003
aama: 0.00000029	aamb: 0.00000001	aamc: 0.00000000	aamd: 0.00000001	aame: 0.00000008	aamf: 0.00000000	aamg: 0.00000000
aana: 0.00000088	aanb: 0.00000002	aanc: 0.00000022	aand: 0.00000071	aane: 0.00000032	aanf: 0.00000000	aang: 0.00000024
aaoa: 0.00000001	aaob: 0.00000000	aaoc: 0.00000000	aaod: 0.00000000	aaoe: 0.00000000	aaof: 0.00000000	aaog: 0.00000000
aapa: 0.00000006	aapb: 0.00000000	aapc: 0.00000000	aapd: 0.00000007	aape: 0.00000005	aapf: 0.00000000	aapg: 0.00000000
aaqa: 0.00000000	aaqb: 0.00000000	aaqc: 0.00000000	aaqd: 0.00000000	aaqe: 0.00000000	aaqf: 0.00000000	aaqg: 0.00000000
aara: 0.00000218	aarb: 0.00000030	aarc: 0.00000022	aard: 0.00000326	aare: 0.00000167	aarf: 0.00000000	aarg: 0.00000088
aasa: 0.00000023	aasb: 0.00000003	aasc: 0.00000011	aasd: 0.00000001	aase: 0.00000108	aasf: 0.00000000	aasg: 0.00000002
aata: 0.00000027	aatb: 0.00000004	aatc: 0.00000003	aatd: 0.00000001	aate: 0.00000055	aatf: 0.00000000	aatg: 0.00000002
aaua: 0.00000000	aaub: 0.00000000	aauc: 0.00000001	aaud: 0.00000002	aaue: 0.00000000	aauf: 0.00000000	aaug: 0.00000002
aava: 0.00000013	aavb: 0.00000000	aavc: 0.00000000	aavd: 0.00000000	aave: 0.00000100	aavf: 0.00000000	aavg: 0.00000000
aawa: 0.00000000	aawb: 0.00000000	aawc: 0.00000000	aawd: 0.00000000	aawe: 0.00000000	aawf: 0.00000000	aawg: 0.00000000
aaxa: 0.00000000	aaxb: 0.00000000	aaxc: 0.00000000	aaxd: 0.00000000	aaxe: 0.00000000	aaxf: 0.00000000	aaxg: 0.00000000
aaya: 0.00000000	aayb: 0.00000000	aayc: 0.00000000	aayd: 0.00000000	aaye: 0.00000000	aayf: 0.00000000	aayg: 0.00000000
aaza: 0.00000000	aazb: 0.00000000	aazc: 0.00000000	aazd: 0.00000000	aaze: 0.00000005	aazf: 0.00000000	aazg: 0.00000000
abaa: 0.00000028	abab: 0.00000098	abac: 0.00000707	abad: 0.00000411	abae: 0.00000035	abaf: 0.00000000	abag: 0.00000065
abba: 0.00000483	abbb: 0.00000003	abbc: 0.00000001	abbd: 0.00000001	abbe: 0.00000269	abbf: 0.00000000	abbg: 0.00000000
abca: 0.00000021	abcb: 0.00000005	abcc: 0.00000005	abcd: 0.00000007	abce: 0.00000007	abcf: 0.00000000	abcg: 0.00000002
abda: 0.00000043	abdb: 0.00000000	abdc: 0.00000003	abdd: 0.00000000	abde: 0.00000452	abdf: 0.00000000	abdg: 0.00000000
abea: 0.00000079	abeb: 0.00000061	abec: 0.00004067	abed: 0.00000110	abee: 0.00000061	abef: 0.00000002	abeg: 0.00000032
abfa: 0.00000004	abfb: 0.00000000	abfc: 0.00000000	abfd: 0.00000000	abfe: 0.00000013	abff: 0.00000000	abfg: 0.00000000
abga: 0.00000004	abgb: 0.00000000	abgc: 0.00000000	abgd: 0.00000000	abge: 0.00000071	abgf: 0.00000000	abgg: 0.00000000
abha: 0.00000754	abhb: 0.00000000	abhc: 0.00000006	abhd: 0.00000000	abhe: 0.00000005	abhf: 0.00000000	abhg: 0.00000000
abia: 0.00000472	abib: 0.00000052	abic: 0.00000382	abid: 0.00000209	abie: 0.00000390	abif: 0.00000027	abig: 0.00000066
abja: 0.00000066	abjb: 0.00000000	abjc: 0.00000000	abjd: 0.00000000	abje: 0.00000033	abjf: 0.00000000	abjg: 0.00000000
abka: 0.00000015	abkb: 0.00000000	abkc: 0.00000003	abkd: 0.00000000	abke: 0.00000000	abkf: 0.00000000	abkg: 0.00000000
abla: 0.00001720	ablb: 0.00000000	ablc: 0.00000001	abld: 0.00000000	able: 0.00008251	ablf: 0.00000002	ablg: 0.00000000
abma: 0.00000009	abmb: 0.00000006	abmc: 0.00000000	abmd: 0.00000000	abme: 0.00000011	abmf: 0.00000000	abmg: 0.00000000
abna: 0.00000031	abnb: 0.00000000	abnc: 0.00000000	abnd: 0.00000000	abne: 0.00000019	abnf: 0.00000011	abng: 0.00000000

Slika 2.14: Ispis kvadograma i njihovih vjerojatnosti



QUINTGRAM:									
aaaaa: 0.00000437	aaaab: 0.00000000	aaaac: 0.00000000	aaaad: 0.00000000	aaaae: 0.00000000	aaaaf: 0.00000000	aaaag: 0.00000000	aaaah: 0.00000000	aaaai: 0.00000000	aaaaj: 0.00000000
aaaab: 0.00000001	aaaab: 0.00000000	aaabc: 0.00000000	aaabd: 0.00000000	aaabe: 0.00000000	aaabf: 0.00000000	aaabg: 0.00000000	aaabh: 0.00000000	aaabi: 0.00000000	aaabaj: 0.00000000
aaaac: 0.00000000	aaacb: 0.00000000	aaacc: 0.00000000	aaacd: 0.00000000	aaace: 0.00000000	aaacf: 0.00000000	aaacg: 0.00000000	aaach: 0.00000000	aaaci: 0.00000000	aaaja: 0.00000000
aaaad: 0.00000000	aaadb: 0.00000000	aaadc: 0.00000000	aaadd: 0.00000000	aaade: 0.00000001	aaadf: 0.00000000	aaadg: 0.00000000	aaadh: 0.00000000	aaadi: 0.00000000	aaaja: 0.00000000
aaaee: 0.00000000	aaae: 0.00000000	aaaec: 0.00000000	aaaed: 0.00000000	aaaea: 0.00000000	aaae: 0.00000000	aaae: 0.00000000	aaaih: 0.00000000	aaai: 0.00000000	aaaja: 0.00000000
aaafa: 0.00000000	aaafb: 0.00000000	aaafc: 0.00000000	aaafd: 0.00000000	aaafe: 0.00000000	aaaff: 0.00000000	aaafg: 0.00000000	aaafh: 0.00000000	aaafi: 0.00000000	aaaja: 0.00000000
aagga: 0.00000000	aaggb: 0.00000000	aaggc: 0.00000000	aaggd: 0.00000000	aagee: 0.00000000	aage: 0.00000000	aage: 0.00000000	aagh: 0.00000000	aagi: 0.00000000	aagja: 0.00000000
aaaha: 0.00000000	aaahb: 0.00000000	aaahc: 0.00000000	aaahd: 0.00000000	aaah: 0.00000000	aaah: 0.00000000	aaah: 0.00000000	aaah: 0.00000000	aaah: 0.00000000	aaah: 0.00000000
aaail: 0.00000000	aaail: 0.00000000	aaail: 0.00000000	aaail: 0.00000000	aaail: 0.00000000	aaail: 0.00000000	aaail: 0.00000000	aaail: 0.00000000	aaail: 0.00000000	aaail: 0.00000000
aaaja: 0.00000000	aaajb: 0.00000000	aaajc: 0.00000000	aaajd: 0.00000001	aaaje: 0.00000000	aaajf: 0.00000000	aaajg: 0.00000000	aaajh: 0.00000000	aaaji: 0.00000000	aaaja: 0.00000000
aaaka: 0.00000000	aaakb: 0.00000000	aaakc: 0.00000000	aaakd: 0.00000000	aaake: 0.00000000	aaakf: 0.00000000	aaakg: 0.00000000	aaakh: 0.00000000	aaaki: 0.00000000	aaaja: 0.00000000
aaala: 0.00000005	aaalb: 0.00000000	aalal: 0.00000000	aalld: 0.00000000	aaale: 0.00000000	aallf: 0.00000000	aallg: 0.00000000	aalhh: 0.00000000	aalhi: 0.00000000	aaaja: 0.00000000
aaama: 0.00000000	aaamb: 0.00000000	aaamc: 0.00000000	aaamd: 0.00000000	aaame: 0.00000000	aaamf: 0.00000000	aaamg: 0.00000000	aaamh: 0.00000000	aaami: 0.00000000	aaaja: 0.00000000
aaana: 0.00000000	aanab: 0.00000000	aanac: 0.00000000	aanad: 0.00000000	aanne: 0.00000000	aanfn: 0.00000000	aanng: 0.00000000	aanhn: 0.00000000	aanii: 0.00000000	aaaja: 0.00000000
aaaoa: 0.00000000	aaao: 0.00000000	aaao: 0.00000000	aaao: 0.00000000	aaao: 0.00000000	aaao: 0.00000000	aaao: 0.00000000	aaao: 0.00000000	aaao: 0.00000000	aaao: 0.00000000
aaapa: 0.00000000	aaapb: 0.00000000	aaapc: 0.00000000	aaapd: 0.00000000	aaape: 0.00000000	aaapf: 0.00000000	aaapg: 0.00000000	aaaph: 0.00000000	aaapi: 0.00000000	aaapa: 0.00000000
aaaga: 0.00000000	aaagb: 0.00000000	aaagc: 0.00000000	aaagd: 0.00000000	aaage: 0.00000000	aaagf: 0.00000000	aaagg: 0.00000000	aaagh: 0.00000000	aaagi: 0.00000000	aaaga: 0.00000000
aaara: 0.00000004	aaarb: 0.00000000	aaarc: 0.00000000	aaard: 0.00000000	aaare: 0.00000003	aaraf: 0.00000000	aaarg: 0.00000007	aaarh: 0.00000000	aaari: 0.00000000	aaara: 0.00000000
aaasa: 0.00000000	aaasb: 0.00000000	aaasc: 0.00000000	aaasd: 0.00000000	aaase: 0.00000000	aaasf: 0.00000000	aaasg: 0.00000000	aaash: 0.00000000	aaasi: 0.00000000	aaasa: 0.00000000
aaata: 0.00000000	aatat: 0.00000000	aatc: 0.00000000	aatd: 0.00000000	aatte: 0.00000000	aatff: 0.00000000	aatgt: 0.00000000	aatht: 0.00000000	aatii: 0.00000000	aaata: 0.00000000
aaaua: 0.00000000	aaaub: 0.00000000	aaauc: 0.00000000	aaaud: 0.00000000	aaau: 0.0					

15

### 3. ZAKLJUČAK

Smatram da je kolegij Izborni projekt bio vrlo interesantan i koristan za unapređivanje mog dosadašnjeg znanja iz kolegija Programiranje II. Problemski zadatak, koji mi je zadao profesor dr.sc. Ivo Ipšić, zahtijevao je dosta logičkog razmišljanja i kontinuiranog rada na samom zadatku. U početku je bilo teško shvatiti samu problematiku zadatka, ali nakon nekoliko pokušaja i neuspjelih implementacija, uspio sam implementirati ispravno rješenje. Prilikom rješavanja zadatka, proširio sam svoje znanje iz područja znakovnih nizova, upravljanja memorijom te sam prvi put koristio alate kao što su wget prilikom pisanja bash skripte. Svidjelo mi se to što je studentu pružena sloboda prilikom rješavanja zadatka te sam siguran da se zadatak mogao riješiti i na mnoge druge načine koje bih u budućnosti želio implementirati.

Unutar tekstualne datoteke combined.txt pronađeno je:

1. Bigrama: 484
2. Trigrama: 9929
3. Kvadgrama: 90415
4. Kvintgrama: 363971

Veličina tekstualne datoteke combined.txt koja sadrži tekst hrvatske Wikipedije iznosi 282.7 MB i u sebi sadrži ukupno 282728837 znakova.

# LITERATURA

- [1] M. Jurak: Programski jezik C, predavanja, ak. g. 2003/04
- [2] <https://linuxize.com/post/wget-command-examples/>
- [3] <https://linux.die.net/man/1/lynx>
- [4] <https://www.linuxjournal.com/content/globstar-new-bash-globbing-option>
- [5] [http://redhat-linux4u.blogspot.com/2018/03/sed\\_24.html](http://redhat-linux4u.blogspot.com/2018/03/sed_24.html)
- [6] <https://perldoc.perl.org>
- [7] <https://stackoverflow.com/questions/6302025/perl-flags-pe-pi-p-w-d-i-t>