Московский авиационный институт
(Национальный исследовательский университет)

Институт: «Информационные технологии и прикладная математика»
Кафедра: 806 «Вычислительная математика и программирование»

Лабораторная работа № 3
по курсу «Криптография»

Студент:	Обыденкова Ю. Ю.
Группа:	М8О-308Б-18
Вариант:	16
Преподаватель:	Борисов А. В.
Оценка:	
Дата:	

Москва, 2021

Постановка задачи

Сравнить 1) два осмысленных текста на естественном языке, 2) осмысленный текст и текст из случайных букв, 3) осмысленный текст и текст из случайных слов, 4) два текста из случайных букв, 5) два текста из случайных слов.

Как сравнивать: считать процент совпадения букв в сравниваемых текстах – получить дробное значение от 0 до 1 как результат деления количества совпадений на общее число букв. Расписать подробно в отчёте алгоритм сравнения и приложить сравниваемые тексты в отчёте хотя бы для одного запуска по всем пяти подпунктам. Осознать какие значения получаются в этих пяти подпунктах. Привести свои соображения о том почему так происходит.

Длина сравниваемых текстов должна совпадать. Привести соображения о том какой длины текста должно быть достаточно для корректного сравнения.

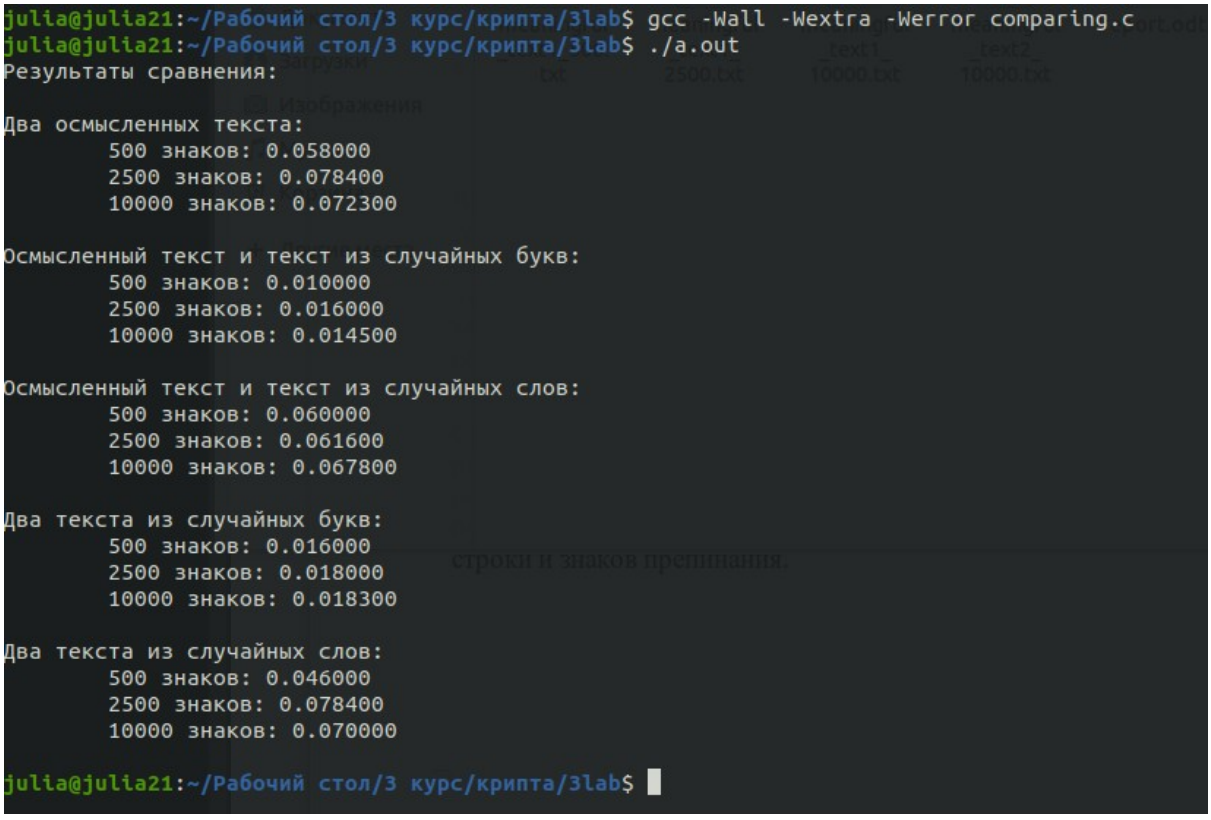
Общие сведения о программе

Открытый текст — в криптографии исходный текст, подлежащий шифрованию, либополучившийся в результате расшифровки. Может быть прочитан без дополнительной обработки.
Заменив реальный открытый текст его моделью, можно построить критерий распознавания открытого текста. При этом можно воспользоваться либо стандартными методами различения статистических гипотез, либо наличием в открытых текстах некоторых запретов, таких, например, как биграмма ЪЪ в русском тексте.
Я решила сравнивать "наивно". Т.е. я просто подсчитываю процент совпадения символов на соответсвующих местах в двух разных текстах. Алгоритм подсчета реализован в файле comparing.c:

```
while ( b uff 1 [ i ] != ’ \0 ’ && b uff 2 [ i ] != ’ \0 ’ )
{
i f ( b uff 1 [ i ] == b uff 2 [ i ] )
r e s ++;
i ++;
}
r e s = r e s / n ;
```

Механизмы генерации текстов (осмысленных, из случайных букв, из случайных слов) описаны в файлах generate.c, generate_random_words.py.
За основу для осмысленных текстов был взят роман "Гордость и предубеждение" Джейн Остин. Я "нарезала"кусочки нужно мне длины. Для текстов, состоящих из случайных букв, я просто использовала рандомное размещение букв различного регистра. Также были сгенерированы тексты различной длины. Генерация текстов из случайных слов производилась из заранее скачанного словаря, затем этот словарь был преобразован в массив и далее процесс генерации был схож с процессом генерации текста из случайных букв, только в данном случае буквами были слова из массива.

Руководство по использованию программы



Код программы

comparing.c

```
#include <stdio.h>
```

```
#include <fcntl.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>

void   calc_func(char *file1, char *file2, int n)
{
        int             i;
        int             fd1;
        int             fd2;
        char    buff1[10240];
        char    buff2[10240];
        float   res;

        printf("\t%d знаков: ", n);
        fd1 = open(file1, O_RDONLY);
        read(fd1, buff1, n);
        buff1[n] = '\0';
        close(fd1);
        fd2 = open(file2, O_RDONLY);
        read(fd2, buff2, n);
        buff2[n] = '\0';
        close(fd2);
        i = 0;
        res = 0;
        while (buff1[i] != '\0' && buff2[i] != '\0')
        {
                if (buff1[i] == buff2[i])
                        res++;
                i++;
        }
```

```
	res = res / n;
	printf("%f\n", res);
}

int		main(void)
{
	printf("Результаты сравнения:\n\n");

	printf("Два осмысленных текста:\n");
	calc_func("data/meaningful_text1_500.txt",
"data/meaningful_text2_500.txt", 500);
	calc_func("data/meaningful_text1_2500.txt",
"data/meaningful_text2_2500.txt", 2500);
	calc_func("data/meaningful_text1_10000.txt",
"data/meaningful_text2_10000.txt", 10000);
	printf("\n");

	printf("Осмысленный текст и текст из случайных букв:\n");
	calc_func("data/meaningful_text1_500.txt",
"data/random_letters1_500.txt", 500);
	calc_func("data/meaningful_text1_2500.txt",
"data/random_letters1_2500.txt", 2500);
	calc_func("data/meaningful_text1_10000.txt",
"data/random_letters1_10000.txt", 10000);
	printf("\n");

	printf("Осмысленный текст и текст из случайных слов:\n");
	calc_func("data/meaningful_text1_500.txt",
"data/random_words1_500.txt", 500);
	calc_func("data/meaningful_text1_2500.txt",
"data/random_words1_2500.txt", 2500);
	calc_func("data/meaningful_text1_10000.txt",
"data/random_words1_10000.txt", 10000);
	printf("\n");
```

```
	printf("Два текста из случайных букв:\n");
	calc_func("data/random_letters1_500.txt",
"data/random_letters2_500.txt", 500);
	calc_func("data/random_letters1_2500.txt",
"data/random_letters2_2500.txt", 2500);
	calc_func("data/random_letters1_10000.txt",
"data/random_letters2_10000.txt", 10000);
	printf("\n");


	printf("Два текста из случайных слов:\n");
	calc_func("data/random_words1_500.txt",
"data/random_words2_500.txt", 500);
	calc_func("data/random_words1_2500.txt",
"data/random_words2_2500.txt", 2500);
	calc_func("data/random_words1_10000.txt",
"data/random_words2_10000.txt", 10000);
	printf("\n");


	return (0);
}
```

generate.c

```
#include <stdio.h>
#include <fcntl.h>
#include <stdlib.h>
#include <unistd.h>

void   meaningful_text(int fd_in, char *file, int n)
{
	int		fd_out;
	char	buff[10240];
```

```
        fd_out = open(file, O_WRONLY);
        read(fd_in, buff, n);
        buff[n] = '\0';
        write(fd_out, buff, n);
        close(fd_out);
}

void   random_letters(char *file, int n)
{
        int             i;
        int             fd_out;
        char    buff[10240];

        fd_out = open(file, O_WRONLY);
        i = 0;
        while (i < n)
        {
                if ((rand() % 10) % 2 == 0)
                        buff[i] = 97 + rand() % 26;
                else
                        buff[i] = 65 + rand() % 26;
                i++;
        }
        buff[n] = '\0';
        write(fd_out, buff, n);
        close(fd_out);
}

void   random_words(int fd_in, char *file, int n)
{
```

```
        int             fd_out;
        char    buff[10240];

        fd_out = open(file, O_WRONLY);
        read(fd_in, buff, n);
        buff[n] = '\0';
        write(fd_out, buff, n);
        close(fd_out);
}

int             main(void)
{
        int             fd_in;

        fd_in = open("data/Austin Pride and Prejudice.txt", O_RDONLY);

        meaningful_text(fd_in, "data/meaningful_text1_500.txt", 500);
        meaningful_text(fd_in, "data/meaningful_text2_500.txt", 500);
        meaningful_text(fd_in, "data/meaningful_text1_2500.txt", 2500);
        meaningful_text(fd_in, "data/meaningful_text2_2500.txt", 2500);
        meaningful_text(fd_in, "data/meaningful_text1_10000.txt", 10000);
        meaningful_text(fd_in, "data/meaningful_text2_10000.txt", 10000);

        close(fd_in);

        random_letters("data/random_letters1_500.txt", 500);
        random_letters("data/random_letters2_500.txt", 500);
        random_letters("data/random_letters1_2500.txt", 2500);
        random_letters("data/random_letters2_2500.txt", 2500);
        random_letters("data/random_letters1_10000.txt", 10000);
        random_letters("data/random_letters2_10000.txt", 10000);
```

```
	fd_in = open("data/random_words.txt", O_RDONLY);

	random_words(fd_in, "data/random_words1_500.txt", 500);
	random_words(fd_in, "data/random_words2_500.txt", 500);
	random_words(fd_in, "data/random_words1_2500.txt", 2500);
	random_words(fd_in, "data/random_words2_2500.txt", 2500);
	random_words(fd_in, "data/random_words1_10000.txt", 10000);
	random_words(fd_in, "data/random_words2_10000.txt", 10000);

	return (0);
}
```

Вывод

Вывод, который можно сделать, это то, что чем больше объем текста, тем точнее будет статистика. Однако, нет смысла увеличивать объем текста до бесконечности, т.к. определенную закономерность можно выявить на относительно небольшой выборке. В текстах из случайных букв этой закономерности вообще заведомо нет, так что в этом тоже можно обойтись небольшой выборкой. Наиболее подходящим объемом выглядит текст, содержащий примерно 5000-10000 знаков.

Сравнивая два текста из случайных букв мы получаем велечину примерно равную вероятности выпадения случайной буквы из алфавита (в разных регистрах). Это объясняется тем, что в текстах, состоящих их случайных букв, сгенерированных мной, нет ни знаков пробелов, табуляции, переноса строки и знаков препинания.