1. Министерство образования и науки Российской Федерации
2. Санкт-Петербургский Политехнический Университет Петра Великого
3. —
4. Институт прикладной математики и механики

**Институт кибербезопасности и защиты информации**

**ЛАБОРАТОРНАЯ РАБОТА № 9**

1. **«Методы контроля целостности»**
2. по дисциплине «Основы информационной безопасности»
3. Выполнил
4. студент гр. 4851001/20001 Козлов О. И.

<*подпись*>

1. Преподаватель
2. профессор Калинин М.О.

<*подпись*>

1. Санкт-Петербург
2. 2023

Оглавление

[1) Цель работы 3](#_Toc131460943)

[2) Постановка задачи 3](#_Toc131460944)

[3) Ход работы 4](#_Toc131460945)

[4) Контрольные вопросы 17](#_Toc131460946)

[5) Выводы 19](#_Toc131460947)

[*Приложение* 20](#_Toc131460948)

## Цель работы

*Цель работы* – приобретение навыков в разработке и использовании программных средств контроля целостности информации, анализ свойств алгоритмов расчета контрольных сумм и циклических кодов.

## Постановка задачи

1. Разработать программу, реализующую расчет восьмиразрядной контрольной суммы по формуле С = Т % (М + 1) для произвольного текстового файла.

2. Создать тестовый файл, записать в него "секретную" информацию (100 символов). С помощью созданной программы подсчитать эталонное значение контрольной суммы для данных из тестового файла.

3. Изменить один из символов входного потока. Произвести расчет контрольной суммы. Сравнить полученное значение с эталоном.

4. Добавить в конец исходной строки ее копию, то есть продублировать строку. Рассчитать контрольную сумму. Сравнить полученные результаты с предыдущими.

5. Поменять местами два символа в тестовом файле. Рассчитать контрольную сумму, сравнить ее с ранее полученными значениями.

6. Получить у преподавателя формулу полинома-делителя CRC и разработать программу вычисления CRC с помощью правил полиномиального деления. Для удобства расчетов следует использовать свойства операции XOR, выполняя полиномиальное деление в блоках текста с последующей сборкой результатов при помощи операции XOR. Программа должна выдавать значение CRC в символьном виде, а также в десятичной, двоичной и шестнадцатеричной системах счисления.

7. С помощью созданной программы подсчитать эталонное значение CRC для тестового файла.

8. Повторить выполнение п. 3 ... 5. Сравнить все полученные значения CRC с эталонным значением, полученным в п. 7.

9. Используя НЕХ-редактор (например, НVIEW, HexWorkshop) добавить в начало "секретных" данных двоичные нули. Рассчитать значение CRC, объяснить полученный результат.

10. К концу контрольного текста в режиме НЕХ дописать полученное значение CRC. Рассчитать новый код CRC, прокомментировать результат.

11. К концу модифицированного в п. 10 текста в режиме НЕХ повторно добавить СRС-код, полученный в п. 9. Рассчитать новый CRC, прокомментировать результат.

12. К концу модифицированного в п. 11 текста в режиме НЕХ дописать байт 80h. Рассчитать CRC, прокомментировать результат.

13. Выполнить расчет CRC для произвольного бинарного файла объемом более 2 Мбайт (например, файл в формате Word, Adobe). К концу исходного файла в режиме НЕХ дважды дописать CRC, вычислить значение CRC и сопоставить его с полученным в п. 11, прокомментировать результат.

14. Модифицировать программу расчета CRC, задав другой полином­делитель той же степени. Путем тестовых прогонов программы, найти два полинома, которые вычисляют значения CRC для одного и тог же рабочего файла, но при добавлении их в конец файла для проверки цикличности кодов не дают нулевой CRC (не показывают достоверность информации). Продолжить тестовые прогоны программы и найти два полинома, которые обладают свойством стандартного полинома CRC. Прокомментировать результаты исследования.

15. Модифицировать программу расчета CRC, использовав снова стандартный полином, но задействовав вместо полиномиального деления бинарные сдвиговые операции. Рассчитать CRC для исходного контрольного текста. Сравнить полученное значение CRC с эталонным CRC, вычисленным

в п. 7. Прокомментировать полученный результат.

## Ход работы

Пусть исходное число равно c = 21110 или в двоичной форме равно 110100112.

Полином LRC8 равен x8 + x7 + x6 + x4 + x2 + x0 или в двоичной форме равен 1110101012.

1. Высчитаем LRC8 с помощью полиномиального деления

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **1** | **1** | **1** | **0** | **1** | **0** | **1** | **0** | **1** |  |  |  |  |  |  |  |
|  |  | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 |  |  |  |  |  |
|  |  | **1** | **1** | **1** | **0** | **1** | **0** | **1** | **0** | **1** |  |  |  |  |  |
|  |  |  |  |  |  | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |  |
|  |  |  |  |  |  | **1** | **1** | **1** | **0** | **1** | **0** | **1** | **0** | **1** |  |
|  |  |  |  |  |  |  |  | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

*Таблица 1 - LRC8 с помощью полиномиального деления*

LRC8 = 100010102

1. Высчитаем LRC8 с помощью сдвиговых операций

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |  | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |  | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |  | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |  | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 |  | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 |  | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 |  | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 | 1 | 0 | 0 | 1 |  | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 | 0 | 0 | 1 | 1 |  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 | 0 | 1 | 1 | 1 |  | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 1 | 1 | 1 | 1 |  | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 1 | 1 | 1 | 0 |  | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 1 | 1 | 0 | 1 |  | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 1 | 0 |  | 0 | 0 | 0 |
| 1 | 1 | 1 | 1 | 0 | 1 | 0 | 0 |  | 0 | 0 |
| 1 | 1 | 1 | 0 | 1 | 0 | 0 | 1 |  | 0 |
| 1 | 1 | 0 | 1 | 0 | 0 | 1 | 1 |  |

*Таблица 2 - LRC8 с помощью сдвиговых операций*

LRC8 = 110100112

1. Высчитаем CRC8 с помощью полиномиального деления на полином

(8, 7, 1, 0)

Полином (8, 7, 1, 0): x8+x7+x1+x0 = 1100000112

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **1** | **1** | **0** | **0** | **0** | **0** | **0** | **1** | **1** |  |  |  |  |  |  |  |
|  |  |  | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 |  |  |  |  |
|  |  |  | **1** | **1** | **0** | **0** | **0** | **0** | **0** | **1** | **1** |  |  |  |  |
|  |  |  |  | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 0 |  |  |  |
|  |  |  |  | **1** | **1** | **0** | **0** | **0** | **0** | **0** | **1** | **1** |  |  |  |
|  |  |  |  |  | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 |  |  |
|  |  |  |  |  | **1** | **1** | **0** | **0** | **0** | **0** | **0** | **1** | **1** |  |  |
|  |  |  |  |  |  |  |  | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |

*Таблица 3 - CRC8 с помощью* *полиномиального деления на полином*

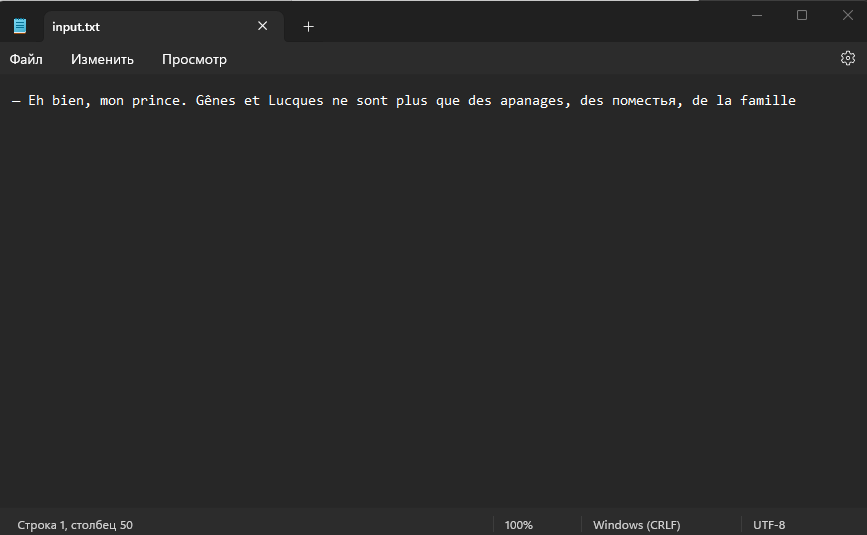
*(8, 7, 1, 0)*

*Программа CRC8*

В ходе выполнения лабораторной работы была реализована программа, которая вычисляет восьмиразрядной контрольной суммы по формуле

C = T % (M+1).

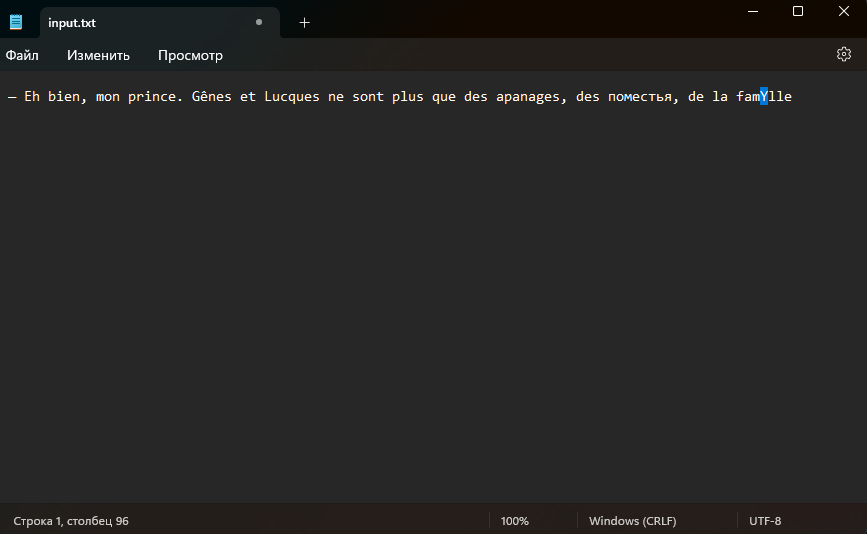
1. Исходный текст:



*Рисунок 1 – Исходный текст, состоящий из 100 символов*

**Контрольная сумма LRC8 = 49**

1. Изменим один из символов входного потока:

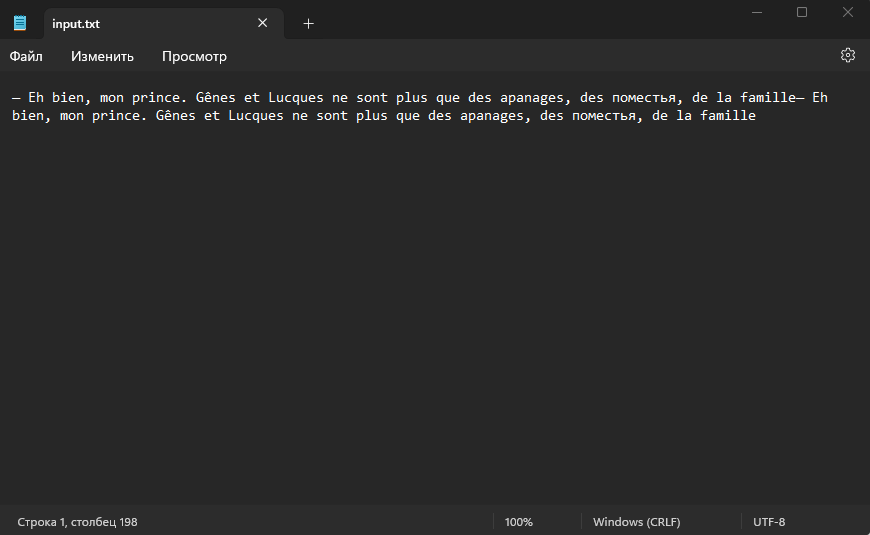


*Рисунок 2 – Исходный текст с отредактированным символом*

**Контрольная сумма LRC8 = 33**

Данная контрольная сумма не совпадает с эталонным значением для исходного текста. Из этого можно сделать вывод, что контрольная сумма меняется при редактировании текста.

1. Добавим в конец сообщения его копию:

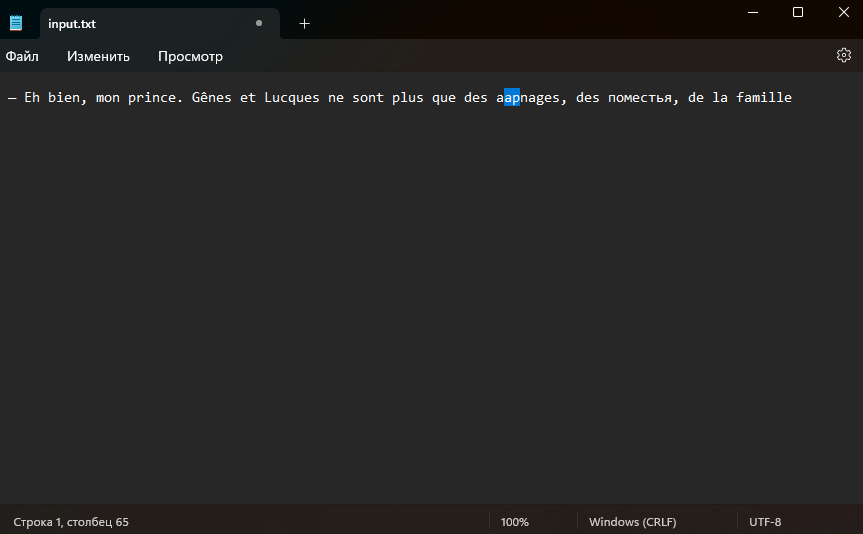


*Рисунок 3 ­­– Исходный текст (дублированный)*

**Контрольная сумма LRC8 = 97**

Данная контрольная сумма не совпадает с эталонным значением для исходного текста. Из этого можно сделать вывод, что контрольная сумма меняется при дублировании текста.

1. Поменяем местами два символа в исходном файле:



*Рисунок 4 ­­– Исходный текст c поменянными символами*

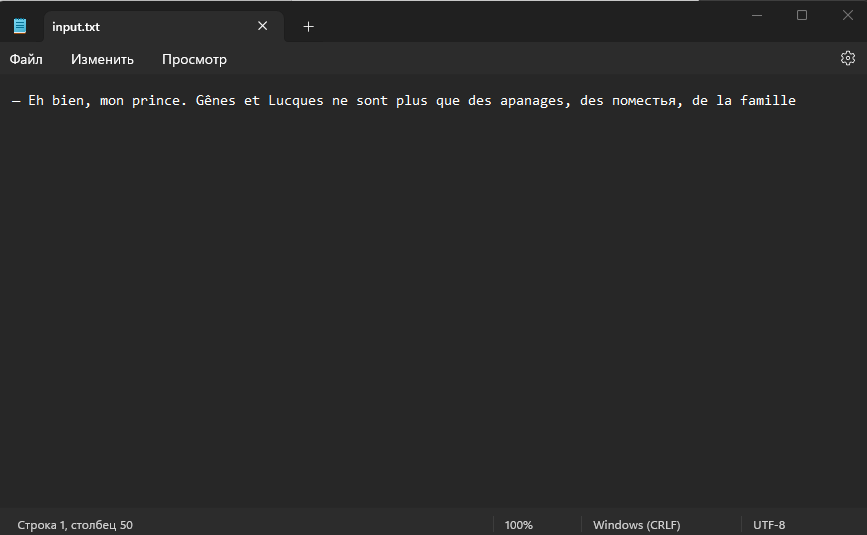
**Контрольная сумма LRC8 = 49**

Значение контрольной суммы совпадает со значением контрольной суммы исходного файла, а значит можно сделать вывод, что при перестановке символов местами значение контрольной суммы не изменяется.

*Программа CRC*

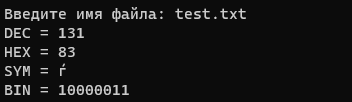
Также была реализована утилита, вычисляющая CRC, показывающая получившееся значение в символьном виде, в десятичной, двоичной и шестнадцатеричной системе счисления.

1. Возьмем стандартный полином: x8+x0 для CRC8

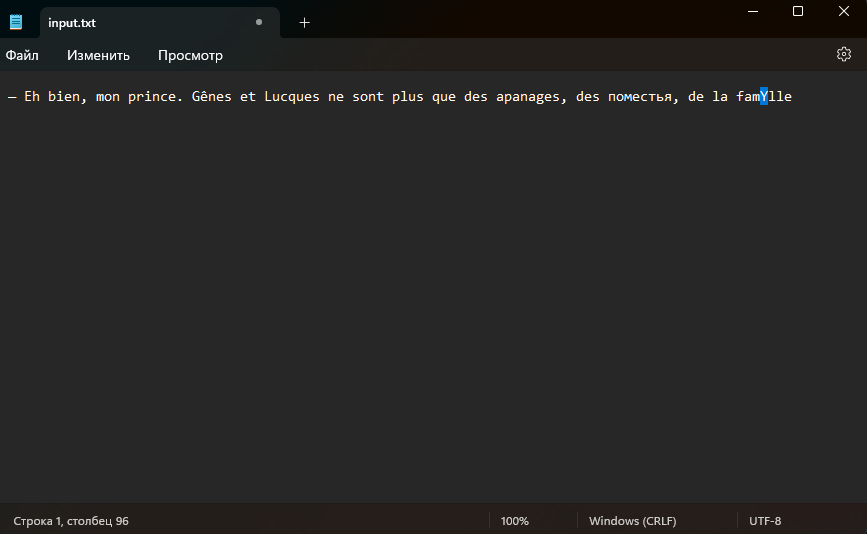


*Рисунок 5 – Исходный текст, состоящий из 100 символов*

**Контрольная сумма CRC8:**

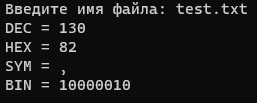
****

1. Изменим один из символов входного потока:



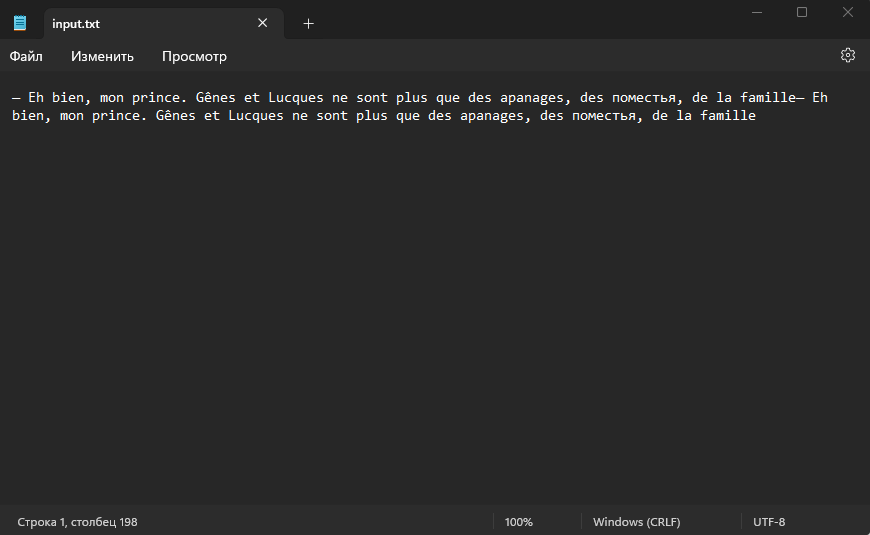
*Рисунок 6 – Исходный текст с отредактированным символом*

**Контрольная сумма CRC8:**



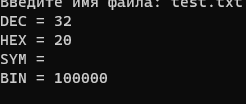
Данная контрольная сумма не совпадает с эталонным значением для исходного текста. Из этого можно сделать вывод, что контрольная сумма меняется при редактировании текста.

1. Добавим в конец сообщения его копию:



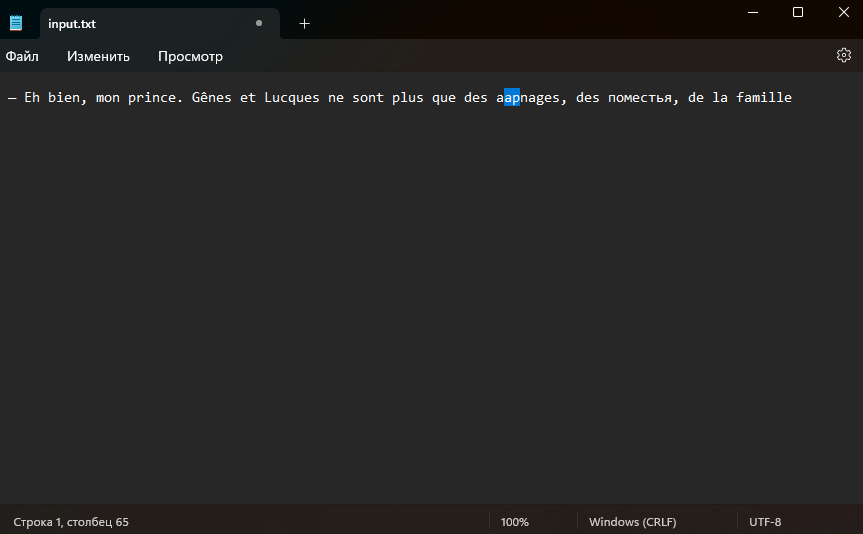
*Рисунок 7 ­­– Исходный текст (дублированный)*

**Контрольная сумма СRC8:**

****

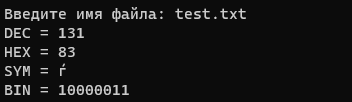
Данная контрольная сумма не совпадает с эталонным значением для исходного текста. Из этого можно сделать вывод, что контрольная сумма меняется при дублировании текста.

1. Поменяем местами два символа в исходном файле:



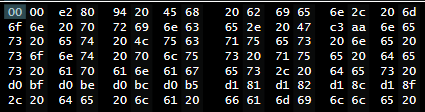
*Рисунок 8 ­­– Исходный текст c поменянными символами*

**Контрольная сумма СRC8:**

****

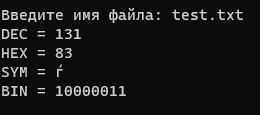
Значение контрольной суммы не отличается от значения контрольной суммы исходного файла, а значит можно сделать вывод, что при перестановке символов местами значение контрольной суммы не изменяется.

1. Добавим двоичные нули в программу:



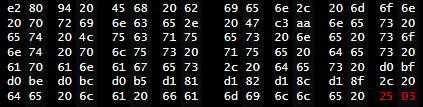
*Рисунок 9 – Добавление двоичный нулей в исходное сообщение*

**Контрольная сумма СRC8:**



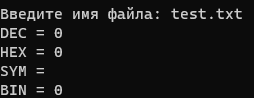
Значение контрольной суммы не отличается от значения контрольной суммы исходного файла, а значит при помощи CRC нельзя определить были ли добавлены нули в файл.

1. Добавим CRC код в конец сообщения:



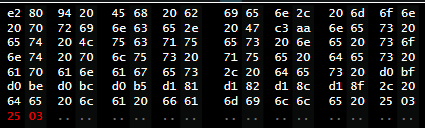
*Рисунок 10 – Добавление CRC кода в конец сообщения (HEX код)*

**Контрольная сумма СRC8:**



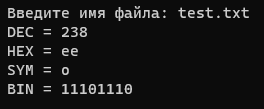
Значение CRC оказалось равно 0. Это произошло, т. к. СRC цикличен. Именно тот факт, что после добавления CRC к тексту, CRC становится равным 0, и доказывает правильность реализованной программы.

1. Добавим еще один CRC код в конец сообщения:



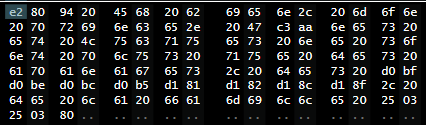
*Рисунок 11 – Двойной CRC в конце сообщения (HEX код)*

**Контрольная сумма СRC8:**

****

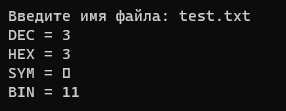
Полученное значение контрольной суммы отличается от эталонной, поскольку был произведён расчёт CRC для текста, отличного от исходного.

1. Добавим в сообщение байт 0x80 в HEX-редакторе:



*Рисунок 12 - Добавим 0x80 в сообщение в HEX редакторе*

**Контрольная сумма СRC8:**

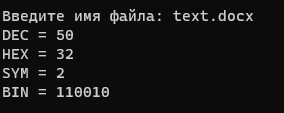
**

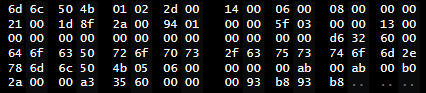
Полученное значение контрольной суммы отличается от эталонной, поскольку был произведён расчёт CRC для текста, отличного от исходного.

1. Рассчитаем CRC для Word файла:

Был создан docx документ размера 2,01 МБ

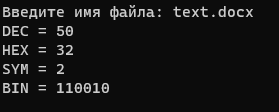
**Контрольная сумма СRC8:**





*Рисунок 13 – Двойной CRC в конце сообщения (HEX код)*

**Контрольная сумма СRC8:**



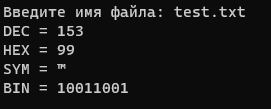
Полученное значение контрольной суммы совпадает с эталонной, поскольку чтение из бинарного файла имеет специфику. Дописанные в конец байты не влияют на результат при попытке чтения при данной реализации алгоритма. Вероятно, это вызвано не универсальностью метода поиска конца файла.

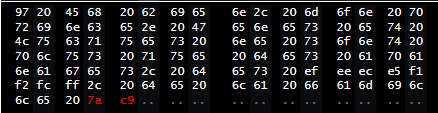
1. Выберем другие полиномиальные делители

C помощью тестовых прогонов программы c другими полиномами-делителями не были найдены полиномы, которые бы не давали нулевой CRC, то есть не показывали достоверность информации.

x8+x1+x0  = 1000000112

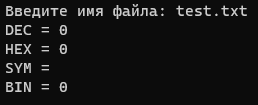
**Контрольная сумма СRC8:**





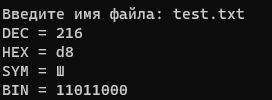
*Рисунок 14 – Добавляем CRC в конец файла*

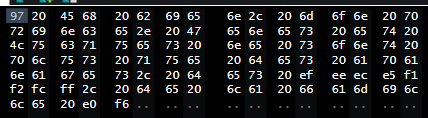
**Контрольная сумма СRC8:**



x8 + x6 + x5 + x2 = 1011001002

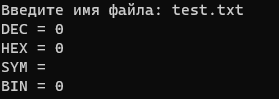
**Контрольная сумма СRC8:**





*Рисунок 15 – Добавляем CRC в конец файла*

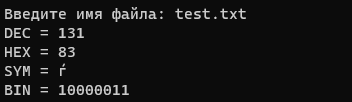
**Контрольная сумма СRC8:**



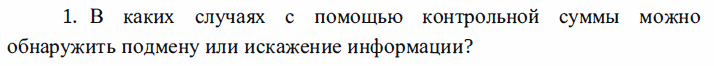
Оба полинома обладают свойствами стандартного полинома и дают нулевой CRC при проверке цикличности.

1. *Программа CRC сдвиговая:*

Для сдвиговых операций CRC, совпадает с получившимся CRC при полиномиальном делении и равен

****

## Контрольные вопросы

 С помощью контрольной суммы можно обнаружить подмену или искажение информации в том случае, если при этом конечная контрольная сумма отличается от изначальной.



При подсчёте CRC применяется полиномиальная арифметика по модулю 2, потому что он был принят из-за отсутствия переносов коэффициентов полиномов. С отменой переноса исчезает разница между сложением и вычитанием, что эквивалентно логической операции XOR, которая является симметричной.



Ошибки, которые могут возникнуть в подсчёте CRC при передаче пакета:

* Ошибка в одном бите. Е = 100...000. Такие ошибки можно отловить, если в G не менее двух битов установлено в 1. Как бы мы ни складывали сдвиги таких G, всегда получим строку, в которой, по крайней мере, два единичных бита, значит, Е не может быть кратно G;
* Ошибка в двух битах. Надо подобрать G, для которых не служат кратными числа типа 11, 101, 1001, и т.д.;
* Ошибка в нечётном количестве бит. Такие ошибки можно отловить, взяв G, в котором количество бит чётно. Это следует из того, что CRC-умножение является результатом XOR константы в аккумулятор с разными сдвигами. Если сделать XOR значения с чётным числом единиц, то в аккумуляторе инвертируется чётное количество единиц, в результате чего чётность количества единиц аккумулятора сохранится;
* Ошибка пакета. Е = 000...000111...111000...000. Ошибка локализована в непрерывном пакете внутри сообщения. Такое Е можно представить в виде произведения: E = (100…000)(111…111), где первый множитель содержит Z нулей, а второй — N единиц. Если в G младший бит — 1, то левый множитель не может быть делителем G, тогда, если G длиннее правого множителя, то ошибка будет отловлена.



Для расчёта CRC применяют определенные полиномы, потому что вычисление CRC интерпретируется как взятие остатка от деления, для этого, во-первых, выбирают делитель, также его длина должна на один бит превышать требуемую длину итогового CRC, т.е. для получения CRC16 необходим 17-битный делитель. Это связано с тем, что остаток должен быть меньше делителя, иначе можно выполнить деление ещё один раз. Не все делители дают хорошие результат.



Алгоритм CRC может указать только на сам факт искажения файла, а не на конкретную ошибку. Поэтому исправить ошибку, зная только начальное и конечное значения CRC, нельзя.

## Выводы

В результате выполнения лабораторной работы было изучено понятие о контроле целостности файлов, а также приобретены навыки в разработке и использовании программных средств контроля целостности информации.

## 

## *Приложение*

Программа LRC8

#include <stdio.h>

#include <locale.h>

int main()

{

setlocale(LC\_ALL, "ru");

FILE\* file = NULL;

unsigned long long int CRC = 0;

char Symbol = '\0';

char FILE[256] = { 0 };

printf\_s("Введите имя файла: ");

scanf\_s("%255s", FILE, 255);

fopen\_s(&file,FILE,"r");

if (file != NULL)

{

Symbol = fgetc(file);

while (Symbol != EOF)

{

CRC += (unsigned long long int)Symbol;

Symbol = fgetc(file);

}

CRC %= 256;

printf("\nCRC = %llu", CRC);

}

fclose(file);

return 0;

}

Программа CRC8

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <locale.h>

#include <time.h>

#include <wchar.h>

#include <malloc.h>

#define POLY 0x1

enum {CRC\_S = 1, CRC\_D = 2, EXIT = 3};

void CRC8\_S();

void CRC8\_D();

char buf[9] = { 0 };

int main()

{

setlocale(LC\_ALL, "1251");

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

while (1)

{

printf(" Что вы хотите сделать? ");

puts("\n");

puts(" 1) CRC - сдвигом ");

puts("");

puts(" 2) CRC - полиномиальным деление ");

puts("");

puts(" 3) Выход");

int Choose = 0;

scanf\_s("%d", &Choose);

switch (Choose)

{

case CRC\_S:

{

system("cls");

CRC8\_S();

return CRC\_S;

}

case CRC\_D:

{

system("cls");

CRC8\_D();

return CRC\_D;

}

case EXIT:

{

return EXIT;

}

default:

{

printf("Неправильный ввод! ");

Sleep(500);

system("cls");

}

}

}

}

void CRC8\_S()

{

unsigned char crc = 0xff;

int flag = 0;

wchar\_t a;

char Name\_File[256] = { 0 };

printf("Введите имя файла: ");

scanf\_s("%255s", Name\_File, 255);

FILE\* file = fopen(Name\_File, "rb");

while (1)

{

if (fread(&a, 1, 1, file) != 1)

{

break;

}

else

{

flag = 1;

}

crc ^= (int)a;

for (int j = 0; j < 8; j++) {

if (crc & 0x80) {

crc = ((crc << 1) ^ 1);

}

else {

crc <<= 1;

}

}

}

if (flag == 0) {

printf("0\n");

return;

}

crc %= 256;

crc ^= 0xff;

if (crc < 0) crc = abs(crc)+1;

char itog[255] = {0};

\_itoa\_s(crc, itog, 255, 2);

printf("Сдвигом:\nint -> %d\nbin -> %s\nhex -> %x\nsym -> %c\n", crc, itog, crc, (char)crc);

fclose(file);

}

void CRC8\_D()

{

char Name\_File[256] = { 0 };

printf("Введите имя файла: ");

scanf\_s("%255s", Name\_File, 255);

FILE\* file = fopen(Name\_File, "rb");

char Poly\_String[8];

int PolyVR = POLY;

for (int p = 0; p < 8; p++) {

Poly\_String[7 - p] = PolyVR % 10;

PolyVR /= 10;

}

fseek(file, 0, SEEK\_END);

int Size\_File = ftell(file);

rewind(file);

unsigned char\* Text\_File = (unsigned char\*)malloc((Size\_File + 1) \* sizeof(unsigned char));

Text\_File[Size\_File] = '\0';

for (int p = 0; p < Size\_File; p++)

fscanf(file, "%c", &Text\_File[p]);

unsigned char Poly\_Char = 0;

for (int p = 0; p < 8; p++)

{

if (Poly\_String[p] == 1)

Poly\_Char = Poly\_Char | (1 << (7 - p));

}

unsigned char Buffer = 0;

while(Size\_File--)

{

Buffer = Buffer ^ \*Text\_File++;

for (int p = 0; p < 8; p++)

{

if (Buffer & 0x80)

{

Buffer = (Buffer << 1) ^ Poly\_Char;

}

else Buffer <<= 1;

}

}

int CRC\_10 = Buffer;

char CRC\_2[16] = {0};

\_itoa\_s(CRC\_10, CRC\_2, 15, 2);

printf("DEC = %d\nHEX = %x\nSYM = %c\nBIN = %s\n", CRC\_10, CRC\_10, CRC\_10, CRC\_2);

fclose(file);

}