1. Цель работы: реализовать программу, которая будет по заданным параметрам n и d строить код Рида-Соломона. Программа должна уметь выписывать порождающий многочлен, кодировать заданное с клавиатуры слово, декодировать введенное с клавиатуры слово при помощи декодера Питерсона-Горнстейна-Цирлера.

2. Построение кода Рида-Соломона

Коды Рида-Соломона (RS) являются q-ичными кодами, т. е. они строятся над полем GF (q) и его расширениями. В данном исследовании расширения поля рассматриваться не будут.

Код задается параметрами n — длина кодового слова и d — минимальное расстояние кода. Код может исправить $t = \lfloor\frac{d-1}{2}\rfloor$ошибок. Длина информационной последовательности равна $k = n - d + 1$.

Для начала необходимо построить поле GF (q), где $q = n + 1$, причем q должно быть простым числом, т. к. расширения полей мы не рассматриваем.

Рассматривать построение данного кода будем на примере RS (10, 6, 5), т. е. $n = 10, d = 5, t = \frac{5-1}{2} = 2, k = 10 - 5 + 1 = 6$. Далее выпишем все элементы поля GF (11) и найдем их порядок.

Элемент поля	Порядок
0	-
1	1
2	10
3	5
4	5
5	5
6	10
7	10
8	10
9	5
10	2

Далее найдем элемент поля, имеющий порядок ord = q — 1, т. е. ord = 10. Такой элемент называется примитивных и обозначается как αТаких элементов несколько, поэтому выберем первый в списке ($\alpha = 2$). Теперь с помощью степеней данного элемента можно перебрать все элементы поля. Покажем это.

Степени α	Элемент поля

0	1
1	2
2	4
3	8
4	5
5	10
6	9
7	7
8	3
9	6
10	1

Следующим этапом является получение порождающего многочлена *g(x)*. Для этого необходимо взять d — 1 минимальных многочленов и перемножить

$$g(x) = \prod_{i=1}^{d-1} (x - \alpha^i)$$

$$deg(g(x)) = d - 1$$

Для нашего примера будет получен следующий многочлен

$$g(x) = (x-2)(x-2^2)(x-2^3)(x-2^4) = x^4 + 3x^3 + 5x^2 + 8x + 1$$

Наконец, чтобы получить кодовое слово, необходимо информационную последовательность представить в виде многочлена и умножить его на порождающий многочлен. Вычисления производятся в поле GF (11). Например, закодируем последовательность$m = (4,1,7,0,9,3)$.

$$m \to m(x) = 3x^5 + 9x^4 + 7x^2 + x + 4$$

$$a(x) = m(x) \cdot g(x) = (3x^5 + 9x^4 + 7x^2 + x + 4) \cdot (x^4 + 3x^3 + 5x^2 + 8x + 1) =$$

$$= 3x^9 + 7x^8 + 9x^7 + 10x^6 + 9x^5 + 7x^4 + 7x^3 + 2x^2 + 4$$

Результатом будет являться вектор: $a = (4,0,2,7,7,9,10,9,7,3)$.

3. Декодер Питерсона-Горнстейна-Цирлера

Необходимо знать, сколько ошибок может исправлять код. Это было выяснено ранее, поэтому $t = \lfloor \frac{d-1}{2} \rfloor = \frac{5-1}{2} = 2$.

Алгоритм:

Изначально $v = t$. Вычислить синдром s, подставляя корни порождающего многочлена в полученный из канала многочлен (полученное кодовое слово представляется в виде многочлена).

1) Построить матрицу M

$$M = \begin{bmatrix} s_1 & s_2 & \cdots & s_v \\ s_2 & s_3 & \cdots & s_{v+1} \\ \vdots & \vdots & \ddots & \vdots \\ s_v & s_{v+1} & \cdots & s_{2v-1} \end{bmatrix}$$

2) Вычислить определитель $det(M)$.

3) Если $det(M) = 0$, $v = v - 1$, вернуться к шагу 1. Если $det(M) \neq 0$, перейти к шагу 4.

4) Поиск многочлена локаторов $\Lambda(x)$, $\Lambda(\alpha^i) = 0$, где i — позиция ошибки.

$$\begin{bmatrix} \Lambda_v \\ \Lambda_{v-1} \\ \Lambda_1 \end{bmatrix} = M^{-1} \cdot \begin{bmatrix} -s_{v+1} \\ -s_{v+2} \\ -s_{2v} \end{bmatrix}$$

$$\Lambda(x) = \Lambda_v x^v + \Lambda_{v-1} x^{v-1} + \cdots + \Lambda_1 x + 1$$

5) Поиск корней $\Lambda(x)$— процедура Ченя.

Перебор всех элементов поля, подстановка их в многочлен локаторов ошибок, проверка на равенство нулю.

$$\alpha_0 \to \Lambda(\alpha_0) \stackrel{?}{=} 0$$

$$\alpha_1 \to \Lambda(\alpha_1) \stackrel{?}{=} 0$$

$$\cdots$$

$$\alpha_{n-1} \to \Lambda(\alpha_{n-1}) \stackrel{?}{=} 0$$

Если $\Lambda(\alpha_i) = 0$, где $\alpha_i = \alpha_{primitive}^j$, то позицией ошибки будет являться $x = -jmod(q-1)$.

6) В случае, если корни не были найдены, то алгоритм завершает работу, т. к. произошло более t ошибок, найти их нельзя.

7) Поиск значений ошибок (В кодах БЧХ данной процедуры не требуется, т. к. коэффициенты равны 0 или 1, производится инверсия коэффициента на месте ошибки).

Строится матрица

$$\lambda = \begin{bmatrix} \alpha^{-i_1} & \alpha^{-i_2} & \cdots & \alpha^{-i_v} \\ (\alpha^{-i_1})^2 & (\alpha^{-i_2})^2 & \cdots & (\alpha^{-i_v})^2 \\ \vdots & \vdots & \ddots & \vdots \\ (\alpha^{-i_1})^v & (\alpha^{-i_2})^v & \cdots & (\alpha^{-i_v})^v \end{bmatrix}$$

Значения ошибок вычисляются как

$$[x_1 \dots x_v] = \lambda^{-1} \cdot \begin{bmatrix} s_1 \\ s_2 \\ \vdots \\ s_v \end{bmatrix}$$

Рассмотрим пример декодирования на примере. Порождающий многочлен возьмем из п. 2.

Пусть из канала пришло следующее слово: $b = (0,7,7,6,0,0,8,4,5,4)$.

$$b \to b(x) = 4x^9 + 5x^8 + 4x^7 + 8x^6 + 6x^3 + 7x^2 + 7x$$

Подставляя в *b(x)* числа $2^1 = 2, 2^2 = 4, 2^3 = 8, 2^4 = 5$, получим синдром $s = (9,6,9,1)$. Изначально$v = t = 2$.

1) $M = \begin{bmatrix} 9 & 6 \\ 6 & 9 \end{bmatrix}$

2) $det(M) = 1$

3) $M^{-1} = \begin{bmatrix} 9 & 5 \\ 5 & 9 \end{bmatrix}$

4) $\Lambda(x) = \Lambda_2 x^2 + \Lambda_1 x + 1$, $\Lambda_0 = 1(\text{всегда})$.

$M^{-1} \cdot \begin{bmatrix} -s_{v+1} \\ -s_{v+2} \\ -s_{2v} \end{bmatrix} = \begin{bmatrix} 9 & 5 \\ 5 & 9 \end{bmatrix} \cdot \begin{bmatrix} -9 \\ -1 \end{bmatrix} = \begin{bmatrix} 2 \\ 1 \end{bmatrix}$

$\Lambda(x) = 2x^2 + x + 1$

5) Процедура Ченя

$\Lambda(0) = 1(\text{всегда, т. к} \Lambda_0 = 1)$

$\Lambda(2^0) = 4$

$\Lambda(2^1) = 0 \Rightarrow x_1 = 2^1$

$\Lambda(2^2) = 4$

$\Lambda(2^3) = 5$

$\Lambda(2^4) = 1$

$\Lambda(2^5) = 2$

$\Lambda(2^6) = 7$

$\Lambda(2^7) = 7$

$\Lambda(2^8) = 0 \Rightarrow x_2 = 2^8$

$\Lambda(2^9) = 2$

$x_1^{-1} = 2^{-1} = 2^9$
$x_2^{-1} = 2^{-8} = 2^2$, следовательно, позиции ошибок — 2, 9.

6) Корни найдены, поэтому находим значения ошибок.

7) $\lambda = \begin{bmatrix} 2^2 & 2^9 \\ (2^2)^2 & (2^9)^2 \end{bmatrix} = \begin{bmatrix} 4 & 6 \\ 5 & 3 \end{bmatrix}$

$det(\lambda) = 4$

$\lambda^{-1} = \begin{bmatrix} 9 & 4 \\ 7 & 1 \end{bmatrix}$

$\lambda^{-1} \cdot \begin{bmatrix} s_1 = 9 \\ s_2 = 6 \end{bmatrix} = \begin{bmatrix} 9 & 4 \\ 7 & 1 \end{bmatrix} \cdot \begin{bmatrix} 9 \\ 6 \end{bmatrix} = \begin{bmatrix} 6 \\ 3 \end{bmatrix}$ — получили значения ошибок.

Следовательно, вектор ошибок равен $e = (0,0,6,0,0,0,0,0,0,3)$. Вычтем его из принятого вектора и получим кодовое слово

$b = b - e = (0,7,7,6,0,0,8,4,5,4) - (0,0,6,0,0,0,0,0,0,3) = (0,7,1,6,0,0,8,4,5,1)$

Если представить полученный вектор в виде многочлена, то получим синдром, состоящий из нулей.

4. Примеры работы программы

```
ilya@acer:~/Desktop/6 семестр/Теория кодиро
GF (11)
ord (alpha) = 10
+---------------+---------------+
| Element       | Ord           |
+---------------+---------------+
| 0             | -             |
| 1             | 1             |
| 2             | 10            |
| 3             | 5             |
| 4             | 5             |
| 5             | 5             |
| 6             | 10            |
| 7             | 10            |
| 8             | 10            |
| 9             | 5             |
| 10            | 2             |
+-------------------------------+
Primitive element: 2
+---------------+---------------+
| Alpha degrees | Element       |
+---------------+---------------+
| 0             | 1             |
| 1             | 2             |
| 2             | 4             |
| 3             | 8             |
| 4             | 5             |
| 5             | 10            |
| 6             | 9             |
| 7             | 7             |
| 8             | 3             |
| 9             | 6             |
| 10            | 1             |
+-------------------------------+

(n, k, d, t) = (10, 6, 5, 2)
GF (11)
Polynom: x^4 + 3x^3 + 5x^2 + 8x + 1
Roots of polynom: [2, 4, 8, 5]
msg: [4, 7, 1, 3, 1, 2]
Code word: [4, 6, 0, 3, 0, 2, 9, 5, 7, 2]
c: [0, 7, 1, 6, 0, 0, 8, 4, 5, 1]
e: [0, 0, 6, 0, 0, 0, 0, 0, 0, 3]
```

```
e: [0, 0, 6, 0, 0, 0, 0, 0, 0, 3]
b: [0, 7, 7, 6, 0, 0, 8, 4, 5, 4]
s: [9, 6, 9, 1]
M:
  9.0  6.0
  6.0  9.0

Det = 45
Inverse M:
  9.0 -6.0
 -6.0  9.0

Inverse det = 1
Inverse M:
  9.0  5.0
  5.0  9.0

s:
 -9.0
 -1.0

Lambdas:
  2.0
  1.0

Locator polynom: [1, 1, 2]
Error positions: [2, 9]
lambda:
  4.0  6.0
  5.0  3.0

Inverse det = 3
Inverse lambda:
  9.0  4.0
  7.0  1.0

s:
  9.0
  6.0

Error size:
  6.0
  3.0
```

```
Founded error vector: [0, 0, 6, 0, 0, 0, 0, 0, 0, 3]
Result: [0, 7, 1, 6, 0, 0, 8, 4, 5, 1]
Result syndrom: [0, 0, 0, 0]
```