**Лабораторная работа № 5**

**ИЗБЫТОЧНОЕ КОДИРОВАНИЕ ДАННЫХ В ИНФОРМАЦИОННЫХ СИСТЕМАХ. ИТЕРАТИВНЫЕ КОДЫ**

**Цель:** приобретение практических навыков кодирования/декодирования двоичных данных при использовании итеративных кодов.

**Теоретические сведения**

**Итеративные** **коды** относятся к классу **кодов** **произведения**.

**Кодом произведения** двух исходных (базовых) помехоустойчивых кодов называется такой многомерный помехоустойчивый код, кодовыми последовательностями которого являются все двумерные таблицы со строками кода (k1) и столбцами кода (k2).

Итеративные коды могут строиться на основе использования **дву-, трехмерных матриц (таблиц)** и более высоких размерностей.

Каждая из отдельных последовательностей информационных символов кодируется определенным линейным кодом (групповым или циклическим). Получаемый таким образом итеративный код также является **линейным**.

**Простейшим** из итеративных кодов является **двумерный код** с проверкой на четность по строкам и столбцам.

**Итеративные,** иногда называемые **прямоугольными** кодами либо композиционными**,** являются одними из самых простых (с точки зрения аппаратной реализации) избыточных кодов, позволяющих исправлять ошибки в информационных словах.

Основное **достоинство** рассматриваемых кодов **– простота** как аппаратной, так и программной реализации.

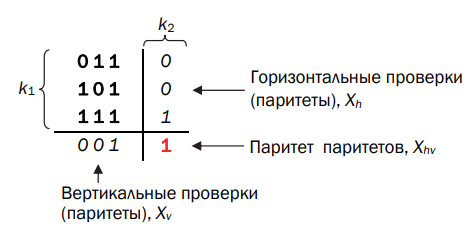
Основной **недостаток** – сравнительно **высокая избыточность**.

Кодовые слова записываются в виде таблицы.

Проверочные символы вычисляются исходя из того, что строки и столбцы должны содержать четное (нечетное) число единиц.

Хk = 011101111.

Хr = Xh, Xv, Xhv = 001.001.1.



Xn = 011101111.001.001.1.

**Избыточные символы** (называемые также **паритетами**) в приведенном кодовом слове в принятом порядке (Xh, Xv, Xhv) записываются сверху вниз, справа налево. Возможен обратный или иной порядок, только при декодировании использовать тот же порядок следования паритетов.

Двумерная матрица формируется как комбинация двух кодов с минимальным кодовым расстоянием dmin=2. Таким образом, итеративный код, состоящий из r=k1​+k2​, будет иметь минимальное кодовое расстояние, равное произведению dmin​ по строкам и столбцам, что составляет 4.

Использование символа Xhv обеспечивает минимальное кодовое расстояние такого итеративного кода dmin (r = k1 + k2 + 1) на единицу больше.

**Проверочный символ** есть сверка по модулю 2 информационных символов, записанных в соответствующие строку или столбец матрицы.

**Декодирование** начинают сразу, не ожидая поступления всего блока информации.

Проверка соответствия избыточных символов полученного слова (Yr = Yh, Yv, Yhv либо Yr = Yh, Yv) при декодировании позволяет обнаружить любое нечетное число искаженных символов, расположенных в одной строке или в одном столбце.

**Сравнение принятых избыточных символов и вновь вычисленных** для полученного слова паритетов.

Однако этим кодом *не могут быть установлены* местоположения многократных ошибок, имеющих четное число искаженных символов как по строкам, так и по столбцам.

Это происходит из-за того, что четность (паритет) по строкам и по столбцам матрицы не нарушается.

Принято считать рассматриваемый код **многомерным**, если количество измерений, по которым вычисляются и анализируются паритеты, не менее 3. Таким образом, простейшим многомерным линейным итеративным кодом является код *трехмерный*.

Дополнительно к двум кодам на основе кодов простой четности (по вертикали и горизонтали) избыточные символы вычисляются по диагонали: Xd.



**Практическое задание**

Разработать собственное приложение, которое позволяет выполнять следующие операции:

1) вписывать произвольное двоичное представление информационного слова Хk (кодируемой информации) длиной k битов в двумерную матрицу размерностью в соответствии с вариантом либо в трехмерную матрицу в соответствии с вариантом (указаны в табл. 5.2);





Рисунок 1.1 – Дву/трехмерная матрица для слова

2) вычислять проверочные биты (биты паритетов): а) по двум; б) по трем; в) по четырем направлениям (группам паритетов);



Рисунок 1.2 – Вычисление паритетов(диагональ, горизонталь, вертикаль)

3) формировать кодовое слово Xn присоединением избыточных символов к информационному слову;

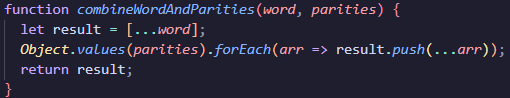


Рисунок 1.3 – Формирование Xn

4) генерировать ошибку произвольной кратности (i, i > 0), распределенную случайным образом среди символов слова Xn, в результате чего формируется кодовое слово Yn;

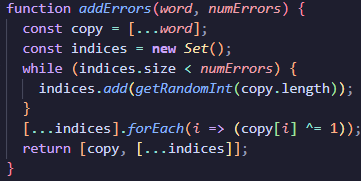


Рисунок 1.4 – Генерация ошибок

5) определять местоположение ошибочных символов итеративным кодом в слове Yn в соответствии с используемыми группами паритетов по пункту (2) и исправлять ошибочные символы (результат исправления – слово Yn’);

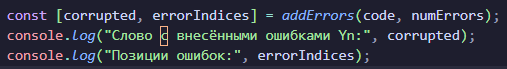


Рисунок 1.5 – Позиции ошибок

6) выполнять анализ корректирующей способности используемого кода (количественная оценка) путем сравнения соответствующих слов Xn и Yn’; результат анализа может быть представлен в виде отношения общего числа сгенерированных кодовых слов с ошибками определенной одинаковой кратности (с одной ошибкой, с двумя ошибками и т. д.) к числу кодовых слов, содержащих ошибки этой кратности, которые правильно обнаружены и которые правильно скорректированы.

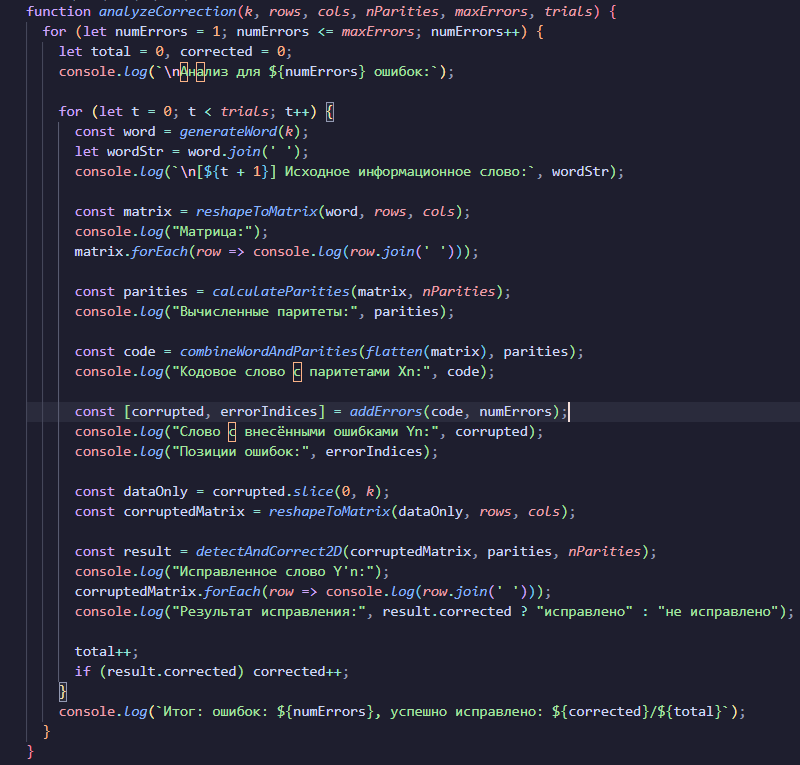


Рисунок 1.6 – Анализ корректирующей способности

Для k1 = 8, k2 = 2

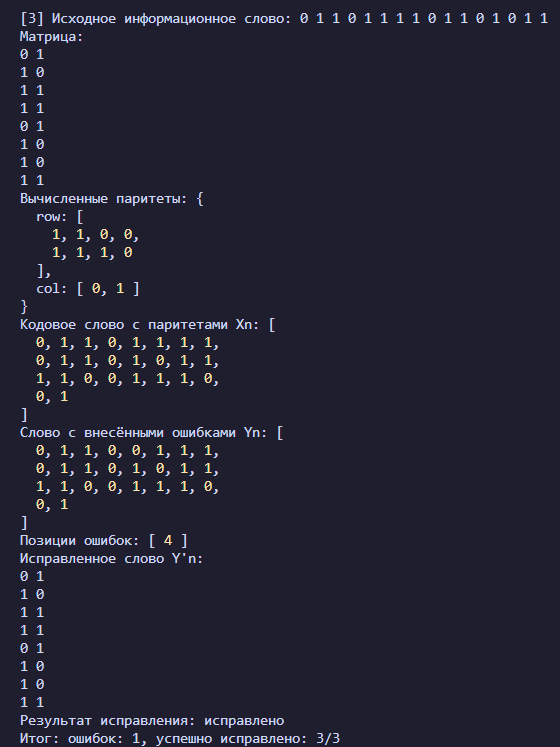


Рисунок 1.7 – Результат для 1 ошибки(остальные 2 случая из трех здесь и далее опущены)

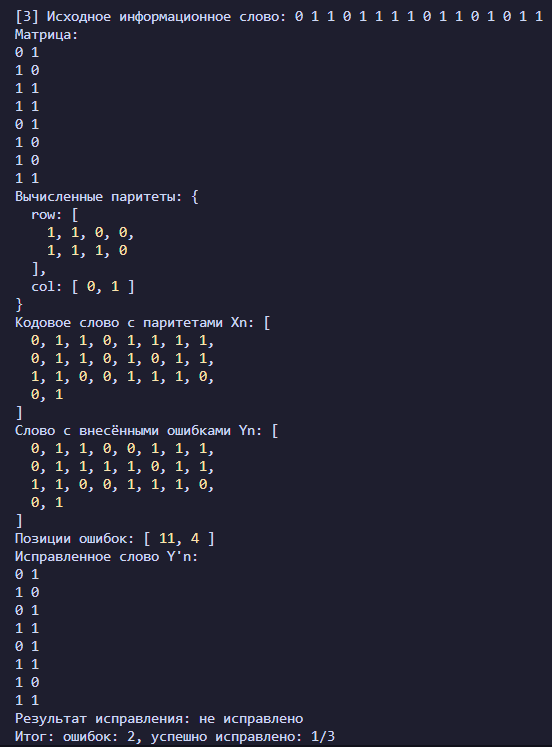


Рисунок 1.8 – Результат для 2 ошибок

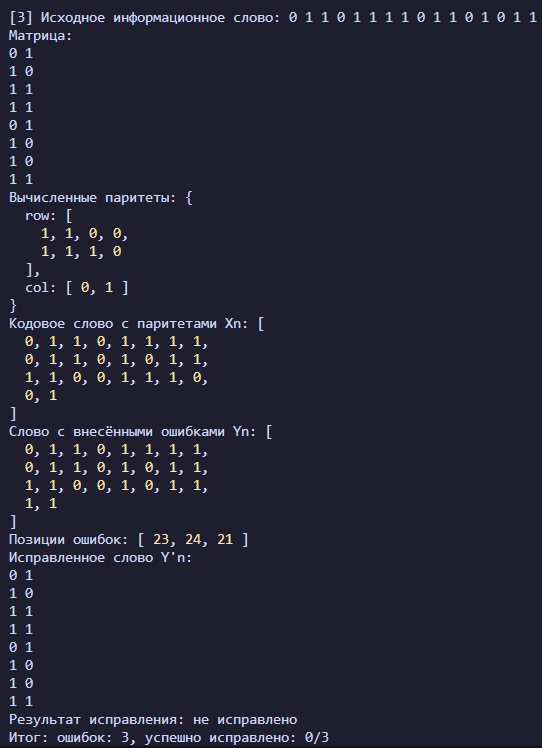


Рисунок 1.9 – Результат для 3 ошибок

Для k1 = 4, k2 = 4

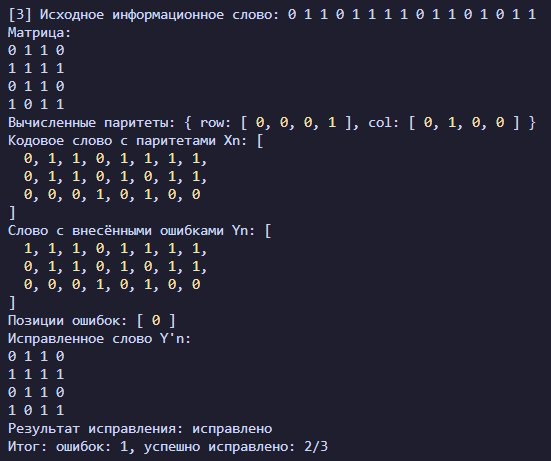


Рисунок 1.10 – Результат для 1 ошибки

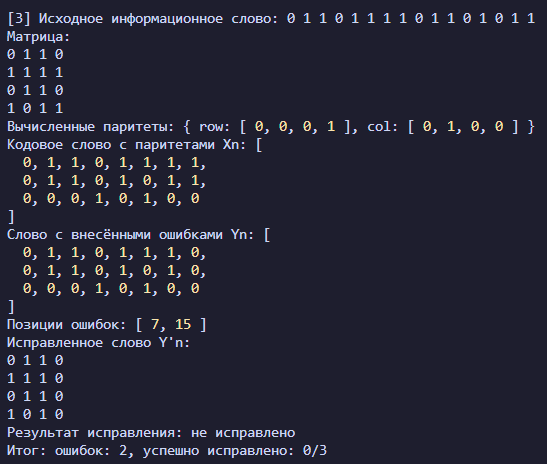


Рисунок 1.11 – Результат для 2 ошибок

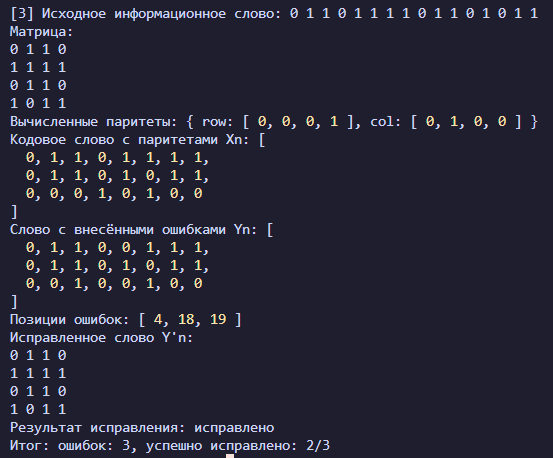


Рисунок 1.12 – Результат для 3 ошибок

Для k1 = 4, k2 = 2, z = 2.

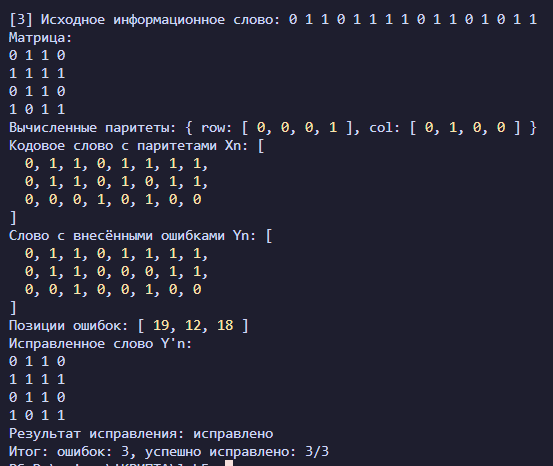


Рисунок 1.13 – Результат для 1 ошибки

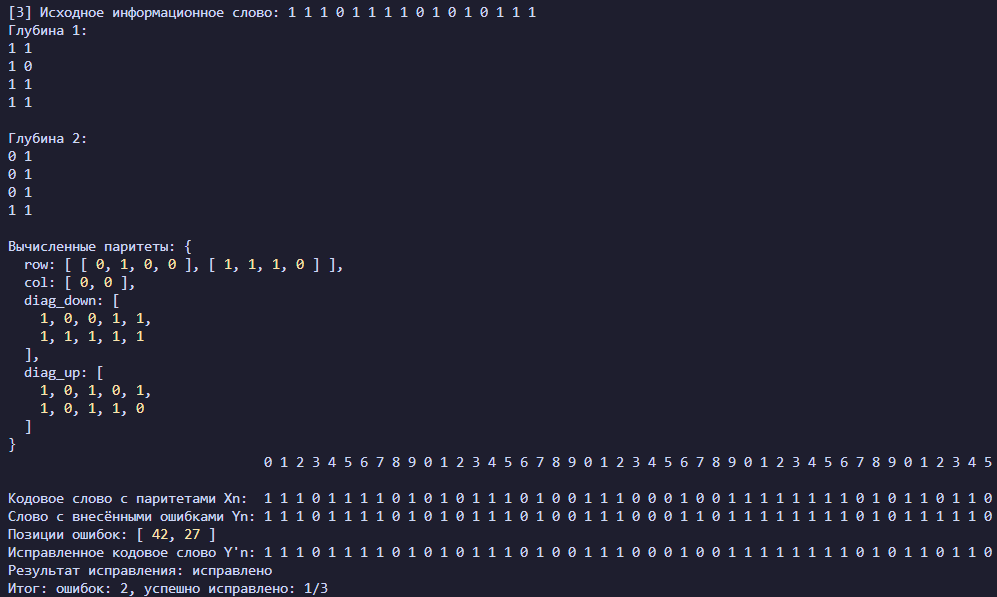


Рисунок 1.14 – Результат для 2 ошибок

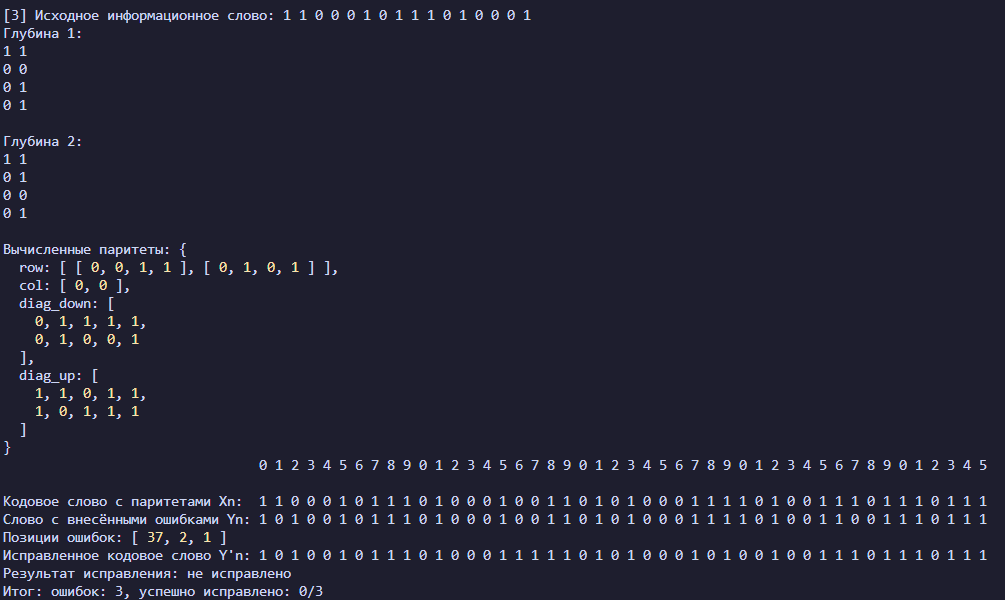


Рисунок 1.15 – Результат для 3 ошибок

Для k1 = 2, k2 = 4, z = 2.

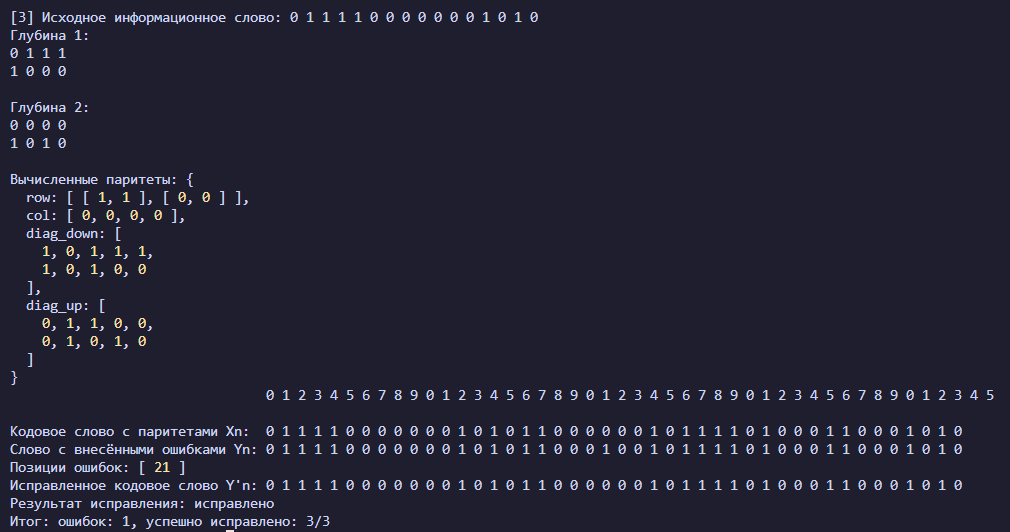


Рисунок 1.16 – Результат для 1 ошибки



Рисунок 1.17 – Результат для 2 ошибок

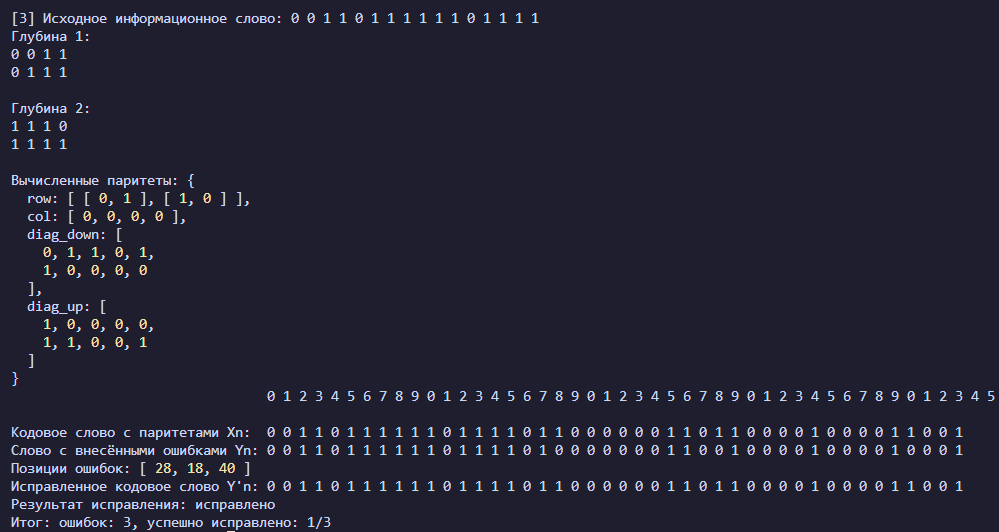


Рисунок 1.18 – Результат для 3 ошибок

**Вывод**

Проведенный анализ работы алгоритма коррекции ошибок в матрице с использованием паритетных битов демонстрирует его способность обнаруживать и исправлять *одиночные ошибки*, внесенные в кодовое слово.