МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ  
Учреждение образования  
«Белорусский государственный технологический университет»  
  
  
  
  
  
  
Отчёт по лабораторной работе №4

ИЗБЫТОЧНОЕ КОДИРОВАНИЕ ДАННЫХ В ИНФОРМАЦИОННЫХ СИСТЕМАХ. КОД ХЕММИНГА

Выполнил: студент 3 курса специальности ИСиТ Калоша И.В.  
Проверила: Ржеутская Н. В.

Минск 2020

**Избыточное кодирование данных в информационных системах. Код Хемминга**

**Цель:** приобретение практических навыков кодирования/декодирования двоичных данных при использовании кода Хемминга.

**Теоретические сведения**

Надежность системы – характеристика способности программного, аппаратного, аппаратно-программного средства выполнить при определенных условиях требуемые функции в течение конкретного периода времени.

Достоверность работы системы (устройства) – свойство, характеризующее истинность конечного (выходного) результата работы (выполнения программы), определяемое способностью средств контроля фиксировать правильность или ошибочность работы.

Ошибка устройства – неправильное значение сигнала (бита – в цифровом устройстве) на внешних выходах устройства или отдельного его узла, вызванное технической неисправностью, или воздействующими на него помехами (преднамеренными либо непреднамеренными), или иным способом.

Ошибка программы – проявляется в не соответствующем реальному (требуемому) промежуточном или конечном значении(результате) вследствие неправильно запрограммированного алгоритма или неправильно составленной программы.

Как следует из вышеприведенного определения, надежность есть внутреннее свойство объекта, заложенное в него при изготовлении и проявляющееся во время эксплуатации. Вторая особенность надежности состоит в том, что она проявляется во времени.

И третья особенность надежности выражается по-разному при различных условиях эксплуатации и различных режимах применения объекта (информационной системы в целом, отдельного ее блока, канала передачи сообщения, оперативной или внешней памяти компьютера).

Надежность является комплексным свойством, включающим в себя единичные свойства: безотказность, ремонтопригодность, сохраняемость, долговечность.

Безотказность – это свойство технического объекта непрерывно сохранять работоспособное состояние в течение некоторого времени (или наработки). Наработка, как правило, измеряется в единицах времени.

Ремонтопригодность – это свойство технического объекта, заключающееся в приспособленности к поддержанию и восстановлению работоспособного состояния путем технического обслуживания, ремонта (или с помощью дополнительных, избыточных технических средств, функционирующих параллельно с объектом).

Большинство современных цифровых систем и устройств (в том числе компьютеры и компьютерные системы, отдельные блоки и модули компьютеров – полупроводниковая, магнитная или оптическая память) содержат специальные средства, призванные автоматически восстанавливать работоспособность этих объектов при нарушении нормального функционирования.

Такие специальные средства контроля называются избыточными.

На рисунке приведена упрощенная структурная схема системы передачи данных с избыточными средствами аппаратного контроля.

Не вдаваясь в детали функционирования блоков, названия которых на схеме выделены полужирным, рассмотрим основные принципы работы представленной схемы в соответствии с задачами данной лабораторной работы. Как видим, сначала осуществляется формирование данных в виде двоичных символов. Затем кодер канала вносит в принятую информационную последовательность некоторую избыточность (данный процесс называется кодированием или помехоустойчивым кодированием), которую декодер может использовать для исправления возникающих при передаче данных по каналу связи ошибок.

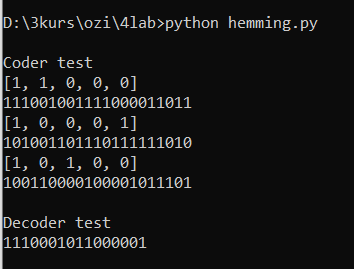


Рисунок 1 – Вывод консоли

**Листинг кода**

def is\_bool(variable):

new\_bool = [x for x in variable if x in ["0","1"]]

return len(variable) == len(new\_bool)

def counter(position, variable):

list\_bool = list(str(variable))

ecc\_value = []

for item\_on\_place in position:

y = 0

for x in item\_on\_place:

if list\_bool[x] == "1":

y += 1

if y % 2 == 0:

ecc\_value.append(0)

else:

ecc\_value.append(1)

return ecc\_value

def to\_hamming(value, position = POSITION16\_5):

bit16 = str(value)

if len(bit16) != 16:

raise ValueError("Error кооличество разрядов не равно 16")

elif is\_bool(bit16) != True:

raise ValueError("Число не двоичное")

ecc\_date = counter(ECC\_POS, bit16)

print(ecc\_date)

bit16 = list(bit16)

for x in position:

bit16.insert(int(x), str(ecc\_date.pop()))

bit21 = ''.join(bit16)

return(bit21)

**Вывод**: в данной работе был рассмотрен проверочный код Хэмминга. Было выяснено, что данный вид кодирования позволяет представлять информацию таким образом, что даже при её ошибочной передаче, удастся восстановить одну ошибку. К сожалению, две ошибки справить уже невозможно. Однако, можно разбить данные на меньшие промежутки, тогда можно будет исправить больше ошибок, но передача будет медленнее.