Exercise 5 this exercise counts for the final grade

We refer to exercise 7 at the end of Chapter 7 of the book. It is an extension of the binary string transmitter of Section 7.6 that we have seen in class last Monday.

Exercise 7 should be clear. However there a few observations to make:
1) we want a more sophisticated channel that, give a list x of Int and a list ob bits y, invert the bits in y as follows: let x=[i0,…ik], then the bits i0, i0+i1, i0+i1+i2, …. of y are inverted.

2) In the encode function it is convenient to add the parity bit at the beginning of each byte. This simplifies the solution.

3) you are supposed to turn in 2 files, main.hs and transmitter.hs (containing modules Main and Transmitter)

4) The Main module must read the input which consists of 2 lines (so use getLine to read them). The first line consists of the integers i0 i1 …. separated by spaces and the second line contains the text that must be encoded, transmitted through the channel (see point (1)), and then decoded. The main should do all these operations and the print the final result (the string that arrives at the end). The module Transmitter must contain the encode and decode function and also a function channel as described in (1) and all auxiliary functions that these functions use.

5) As already seen in a previous exercise, once you have read the line containing i0 i1 …, you should use words and read :: Int in order to build a corresponding list of Int.

6) Since the channel in general introduces errors, the decode function should find them. We ask that each byte in which an error is found is transformed into a byte representing the character ‘x’.

There are 2 automatic tests for the exercise.

Be aware that moodle allows to check for similarities.