Exercise 1

Write a command-line program that accepts the following syntax:

```
frequencies <input file> <output file>
```

The program takes a binary file as input and for each byte (interpreted as an unsigned 8-bit integer) it counts its occurrences. The output is a text file consisting of one line for each different byte found in the input file with the following format:

```
<byte><tab><occurrences><new line>
```

The byte is represented with its two-digit hexadecimal value, occurrences must be represented in base 10. The rows are sorted by byte value, from the smallest to the largest.

Finally, the program must send on standard output the entropy of the set of symbols according to the distribution just calculated and with the following format:

```
Entropy:<space><value><new line>
```

The value is expressed as a decimal number.

Exercise 2

Write a command line program with this syntax:

```
write_int32 <filein.txt> <fileout.bin>
```

The first parameter is the name of a text file that contains signed base 10 integers from -1000 to 1000 separated by whitespace. The program must create a new file, with the name passed as the second parameter, with the same numbers saved as 32-bit binary little-endian number in 2's complement.

Exercise 3

Write a command line program with this syntax:

```
read_int32 <filein.bin> <fileout.txt>
```

The first parameter is the name of a binary file containing 32-bit numbers 2’s complement, in little endian. The program must create a new file, with the name passed as the second parameter, with the same numbers saved in decimal text format separated by a new line.

Exercise 4

Write a command line program with this syntax:

```
write_int11 <filein.txt> <fileout.bin>
```

The first parameter is the name of a text file that contains base 10 integers from -1000 to 1000 separated by whitespace. The program must create a new file, with the name passed as the second parameter, with the same numbers saved as 11-bit binary in 2’s complement. The bits are inserted in the file from the most significant to the least significant. The last byte of the file, if incomplete, is filled with bits of 0.

Exercise 5

Write a command line program with this syntax:

```
read_int11 <filein.bin> <fileout.txt>
```

The first parameter is the name of a binary file that contains 11-bit numbers in 2's complement, with the bits sorted from most significant to least significant. The program must create a new file, with the name passed as the second parameter, with the same numbers saved in decimal text format separated by a new line. Ignore any excess bits in the last byte.

Exercise 6

Write a command line program with this syntax:

```
binary [c|d] <input file> <output file>
```

When the first option is "c", the first parameter is the name of a text file that contains signed base 10 integers (they can fit 32 bits) separated by whitespace.
The program should create a file with the following format:

Field	**Type and Size**	**Description**
MagicNumber	4 bytes	"BIN!"
MinValue	32 bits signed integer big endian	Lowest number in file
MaxValue	32 bits signed integer big endian	Highest number in file
NumValues	32 bits unsigned integer big endian	Number of values in file
Values	NumValues numbers with bit size depending on the range, plus 0 padding to fill last byte.	The number of bits required depends on the number of different values, that is (MaxValue – MinValue + 1). Select the minimum number of bits able to represent that range. 0 will be MinValue, 1 will be Min Value +1, …

When option "d" is specified, the program decodes the input file content generating an output text file with numbers separated by new lines.

If for example the input file contains: `3 7 -2 14`
The minimum is -2 and the maximum is 14, that is a range of 17 values, which requires 5 bits per symbol.
3 will be 00101, 7 will be 01001, -2 will be 00000, 14 will be 10000.
So the file will contain: `42 49 4E 21 FF FF FF FE 00 00 00 0E 00 00 00 04 2A 41 00`