Exercise 1

Write a command-line C++ 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 in base ten. The rows are sorted by byte value, from the smallest to the largest.

Exercise 2

Write a command line program in C++ 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 numbers in 2's complement.

Exercise 3

Write a command line program in C++ 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 in C++ 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 equal to 0. Since the incomplete byte will have at most 7 padding bits, there’s no risk of interpreting padding as another value.

Exercise 5

Write a command line program in C ++ 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.