THE BIG-WIGS

OVERVIEW

Many software programs encode data to make subsequent operations more effective. You will write a program that is able to convert data between text-files and bigwig-files.

DESCRIPTION

A text file is a sequence of characters whereas a bigwig file is a binary file composed of zero-or-more shorts. To transform a text-file T into a bigwig-file B we take the following steps.

1. Construct a sequence of symbols V. This sequence is initially given as
   a. {abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ1234567890-$#_*&%/.,?\'\";:<>()[]{}!@ \r\n\t}
2. Read a single character C from T and find it's index K in V. K will be -1 if C is not in V.
3. If K is not negative, then write K to B and move C to the beginning of V.

To transform a bigwig-file into a text-file we take the following steps.

1. Construct a sequence of symbols V. This sequence is initially given as
   a. {abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ1234567890-$#_*&%/.,?\'\";:<>()[]{}!@ \r\n\t}
2. Read a single integer K from B and find the character C in V such that V[K] = C. Output the character C and move C to the beginning of V.

This algorithm treats letters as big-wigs (important people) by moving them to the beginning of the line whenever they are handled. The algorithm is useful because it will tend to generate a file with many more low-valued integers than large-valued integers. This is often done as a pre-processing step when performing data compression.

DETAILS

Write a class named **BigWigSequence** that models the sequence of symbols V as described above. Specifically, the class must support the following methods

- A constructor that accepts no arguments. The sequence is initialized as described above.
- int charToIndex(char c) : this returns the index of 'c' in the sequence (-1 if it is not present). It also moves 'c' to the beginning of the sequence if 'c' exists.
- char intToChar(int i) : this returns the char 'c' at index 'i' of the sequence. It also moves 'c' to the beginning of the sequence if 'i' is valid. It throws an IndexOutOfBoundsException if 'i' is not a valid index.
- int length() : returns the length of the sequence.
- char charAt(int i) : returns the character at index i of the sequence.
- boolean contains(char c) : returns true if the sequence contains c and false otherwise

Write a program named **BigWig** that accepts exactly three command-line arguments.

1. Either the text "-e" or "-d".
2. The name of the input file
3. The name of the output file

If the program is executed with the "-e" option (encode), the program must take a text-file as input and transform it into a bigwig-file. If the program is executed with the "-d" option (decode), the program must take a bigwig-file and transform it into a text-file. The program must use the class **BigWigSequence** when encoding and decoding.

OUTPUT

The program must do exactly one of the following:

1. If there are not 3 arguments on the command-line print "Usage: (-e|-d) <input> <output>"
2. If the input file doesn't exist print "Error. Input file doesn't exist."
3. If there is an error reading/writing from/to files print "Error creating the file."
4. If the program successfully completes in -d mode, then print "Decoding successful"
5. If the program successfully completes in -e mode, then print "Encoding successful". After printing this message, print a histogram of the output file. That is, print the number of times each possible output value was actually written to the output file. Specifically, print N lines, each of which has two integer values. The first integer is one of the possible output values (starting at 0 and moving through N-1). The second number is the number-of-times that value was written to the output. N is the length of the BigWigSequence.