```
    1: // Copyright 2023 Thomas O'Connor
    2: #ifndef RANDWRITER_HPP
    3: #define RANDWRITER_HPP
    4:
    5: #include <algorithm>
    6: #include <fstream>
    7: #include <iostream>
    8: #include <list>
    9: #include <random>
   10: #include <string>
   11: #include <unordered_map>
   12: #include <utility>
   13: #include <SFML/System.hpp>
   14:
   15: #define ORDERZERO "refrence_text"
   16:
   17: using std::cout;
   18: using std::endl;
   19: using std::list;
   20: using std::pair;
   21: using std::string;
   22: using std::unordered_map;
   23:
   24: class RandWriter {
   25:  public:
   26:   // Create a Markov model of order k from given text
   27:   // Assume that text has length at least k.
   28:   RandWriter(string text, int k);
   29:
   30:   // Order k of Markov model
   31:   int orderK() const { return _K; }
   32:
   33:   // Number of occurences of kgram in text
   34:   // Throw an exception if kgram is not length k
   35:   int freq(string kgram) const;
   36:
   37:   // Number of times that character c follows kgram
   38:   // if order=0, return num of times that char c appears
   39:   // (throw an exception if kgram is not of length k)
   40:   int freq(string kgram, char c) const;
   41:
   42:   // Random character following given kgram
   43:   // (throw an exception if kgram is not of length k)
   44:   // (throw an exception if no such kgram)
   45:   char kRand(string kgram);
   46:
   47:   // Generate a string of length L characters by simulating a trajectory
   48:   // through the corresponding Markov chain. The first k characters of
   49:   // the newly generated string should be the argument kgram.
   50:   // Throw an excpetion if kgram is not of length k.
   51:   // Assume that L is at least k
   52:   string generate(string kgram, int L);
   53:
   54:   // Overload the stream insertion operator << and display the internal s
tate
   55:   // of the Markov model. Print out the order, alphabet, and the frequenc
ies
   56:   // of the k-grams and k+1-grams
   57:   friend std::ostream& operator<<(std::ostream& out, RandWriter& obj);
   58:
   59:  private:
   60:   int _K;
   61:   // key is kgram, data is frequency of that kgram
   62:   // and probability alphabet/dictionary of kgram
   63:   unordered_map<string, pair<int, string>> MarkovModel;
```

```
64: };
65:
66: // helper function
67: void cycleString(string& str, char item);
68:
69: #endif
```