```cpp
#include <iostream>
#include <fstream>
#include <map>
#include <set>
#include <vector>
using namespace std;

vector<char> readinfile(string filename) {
    vector<char> result;
    ifstream f(filename);
    char ch;
    while(f>>noskipws>>ch) {
        if(isprint(ch)){
            ch = toupper(ch);
            result.push_back(ch);
        }
        else {
            ch = ' '; //this basically replaces /n with a space to fix words between liens combining
            result.push_back(ch);
        }
    }
    return result;
}
void buildMap(int level, string filename, map<string, vector<char>> &markovMap) {
    vector<char> text = readinfile(filename);

    for(int i = 0; i < text.size()-level-1; i++) {
        string key;
        for(int j = 0; j < level; j++) {
            key += text[i+j];
        }
        char value = text[i+level];
        markovMap[key].push_back(value);
    }

    //print map to test
/*
    for(auto iter = markovMap.begin(); iter != markovMap.end(); iter++) {
        cout << iter->first << ": ";
        vector<char> chars = iter->second;
        for(char c: chars) {
            cout << c << " ";
        }
        cout << endl;
    }*/

}
string getText(const map<string, vector<char>> & markovMap) {
    cout << "Enter a number of characters: ";
    int textLen;
    cin >> textLen;

    string result;

    auto iter = markovMap.begin();
    advance(iter, rand() % markovMap.size());
    string key = iter->first;
    result = key;
    vector<char> chars;

    for(int i = key.length()-1; i < textLen; i++) {
        chars = markovMap.at(key);
        char value = chars.at(rand() % chars.size());
        result += value;

        key += value;
        key = key.substr(1, key.length()-1);
    }

    return result;
}

```

```cpp
int main() {
    srand(time(0));
    cout << "Hello, Markov Chain!" << endl;
    cout << "Enter level (1-10): ";
    int level;
    cin >> level;
    cout << "Enter file name: ";
    string filename;
    cin >> filename;

    map<string, vector<char>> markovMap;
    buildMap(level, filename, markovMap);

    string text = getText(markovMap);

    cout << text << endl;

    return 0;
}

```