```
1  //Johanny Mateo
2  //December 5, 2016
3  //CISC 1110
4  //Assignment 7
5
6  #include <iostream>
7  #include <fstream>
8  #include <string>
9  #include <iomanip>      //setw() function is called to organize output
10 using namespace std;
11
12 bool isvaliddna (string);               //is it valid?
13 void print_occurs (string, string);     //how many times does the motif occur
14
15 int main () {
16     //line holds the current line while reading in
17     string dna, motif, filename, line;
18
19     ifstream inputFile;
20     cout << "Please enter the input filename: ";
21     //shortinput.txt    (valid DNA, no N)
22     //shortinput2.txt   (valid DNA, has N)
23     //shortinput3.txt   (invalid DNA)
24     getline (cin, filename);
25     inputFile.open(filename.c_str());
26     cout << endl;
27
28     if (!inputFile){
29         cout << "Input file could not be opened.\n\n";
30         exit(1);
31     }
32
33     while (getline(inputFile, line)) {  //run this loop to get each line,
34         dna += line;                    //and concatenate it to the object dna
35     }
36
37     cout << "Original DNA Sequence:\n" << dna << endl;
38
39     cout    << "\nEnter a motif. A valid motif contains the letters "
40             << "A, C, G, T, N. If you enter a valid motif, I will tell you "
41             << "How many times it occurs in the DNA sequence, and at which "
42             << "positions: ";
43     cin >> motif;
44     cout << endl;
45
46     if (isvaliddna(dna) && isvaliddna(motif)) {
47         print_occurs(dna, motif);
48     }
49
50     else
51         cout << "Either the DNA sequence or the motif is invalid.\n";
52
53     cout << endl;
54     return 0;
55 }
56
57 bool isvaliddna (string test) {
58     //test.length() is the size of the entire string array
59     for (int i = 0; i < test.length(); i++) {
60
61         if (test[i] != 'A' && test[i] != 'C' && test[i] != 'G' &&
62             test[i] != 'T' && test[i] != 'N'){
63             return false;
64         }
65     }
66     return true;
67 }
68
69 void print_occurs (string original, string pattern) {
```

```
70
71      int numocc = 0;     //counter for number of occurences
72
73      //create an array to hold the position of the pattern. array size has to be
74      //the length of DNA.
75      unsigned long position[original.length()];
76
77      //index is the index for the locations of the postion array
78      int index = 0;
79
80      size_t pos = original.find(pattern);        //find position of 1st pattern
81
82      while (pos != string::npos) {   //if atleast 1 pattern is present
83          numocc++;                   //increase the number of occurrences
84          position[index] = pos;      //add its position to the postition array
85          index++;                    //go to the next index in the position array
86          pos = original.find(pattern, pos+1);    //continue searching
87      }
88
89      if (numocc != 0) {      //if the pattern appears at least once
90          cout    << "The number of times this motif occurs "
91                  << "in the DNA sequence is: "
92                  << numocc << ". It occurs at positions:" << endl;
93
94          //iterate through the position array, and print its contents
95          //(the positon in the DNA sequence in which the motif appears
96          //numocc is the number of elements in the array
97          for (int i = 0; i < numocc; i++){
98
99              //there are about 5000-6000 possible positions, there's 4 characters
100             //in the max possible location, setw to 5 to have at least one space
101             //in between two intergers and a neat output
102             cout <<setw(5) << position[i];
103         }
104         cout << endl;
105     }
106
107     else {      //if numocc == 0, the pattern is not in the original sequence
108         cout << "This motif does not occur in this DNA sequence.\n";
109     }
110
111     return;
112 }
```