Programming Exercise 11.7

NSA Decoder, v.2.0

Purpose. Apply arrays to do something useful and interesting.

Pretend that the National Security Agency (NSA) discovered that the encryption scheme you came up with in exercises 10.6 and 10.7 is easy to hack. All it does is bump each letter in a text file up to the next letter in the alphabet. They want something better.

Background. Pretend that you ran across this code sample on the Internet. It looks like something that can help you devise a better encryption scheme.

```
// an array of whole numbers
const int SIZE = 5;
int offset[SIZE] = {5, -8, -12, -6, -1};

int counter = 0;
while (true)
{
  // cycle through the array
  int index = counter % SIZE;
  cout << offset[index] << endl;

  // continue cycing?
  char keepGoing;
  cout << "Keep going? [Y/N]: ";
  cin >> keepGoing;
  cin.ignore(1000, 10);
  if (keepGoing == 'n' || keepGoing == 'N') break;

  // count how many loop cycles
  counter++;
}
```

This array and loop repeat the sequence +5, -8, -12, -6, -1, over and over. Your exercise 10.7 repeated the very simple sequence -1 over and over, because exercise 10.6 repeated +1 over and over. Exercise 11.3 uses a better encryption scheme that subtracts 5 from the first letter in a line read for a file, then adds 8 to the 2nd letter, then adds 12 to the 3rd, 6 to the 4th, and 1 to the 5th. At the 6th letter, it subtracts 5 and continue repeating the sequence for any remaining letters in the line.

Actually, you came come up with your *own* array, with size and contents. Only you know what that sequence is, so only you can write the decoder that reverses the process.

Requirements. Write **`nsaDecoder2.cpp`** based on Exercise 10.7's `nsaDecoder1.cpp`. But in this version, modify the "encode the line" part of the algorithm to use a repeating sequence of numbers to scramble the letters, instead of -1 for each letter. It has to match exercise 11.6's encoding scheme.

Program I/O. Same as exercise 10.7