

Programming Exercise 11.6

NSA Encoder, 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 to 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 cycling?
    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 10.6 repeated the very simple sequence +1 over and over, and exercise 10.7 repeated -1 over and over. Perhaps a better encryption scheme would subtract 5 from the first letter in a line read for a file, then add 8 to the 2nd letter, then add 12 to the 3rd, 6 to the 4th, and 1 to the 5th. At the 6th letter, it would subtract 5 and continue repeating the sequence for any remaining letters in the line.

Even better, you could come up with your *own* array, with size and contents of your choosing to give you a unique encryption scheme.

Requirements. Write `nsaEncoder2.cpp` based on Exercise 10.6's `nsaEncoder1.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.

Program I/O. Same as exercise 10.6