CS 7A - Spring 2015 - John Conway's Rats Sequence. Due 4/23/15

12334444, 55667777, 123334444, 556667777, 1233334444, 5566667777, . . . ,

123, 444, 888, 1677, 3489, **12333, 44556, 111, 222, 444, 888, 1677, 3489**, 12333, ...

Background Theory

A nice example of "recreational math" is given by Conway's RATS sequences, one of many base-dependent "reversal" sequences. RATS stands for Reverse, Add, Then Sort: take a number with digits in increasing order, reverse it, add to the original number, and then sort the result's digits in increasing order. Some base-10 examples are shown in the picture above.
Here is the code we wrote in class to get this started:

```
///GH The RATS class
#include "std_lib_facilities.h"

// Helper function for Rats class takes a number and returns RATS value.
unsigned ras(unsigned n) {
    vector<unsigned> digits;
    unsigned numCopy = n;
    while(numCopy!=0) {
        digits.push_back(numCopy%10);
        numCopy /= 10;
    }
    unsigned nReverse = 0, pow10 =1;
    for(int i = digits.size()-1; i >= 0; --i) {
        nReverse += digits[i]*pow10;
        pow10 *= 10;
    }
    n += nReverse;
    digits.clear();
    while(n!=0) {
        digits.push_back(n%10);
        n /= 10;
    }
    sort(digits.begin(),digits.end());
    //reverse(digits.begin(),digits.end()); //Conway doesn't reverse them
    n = 0; pow10 =1;
    for(int i = digits.size()-1; i >= 0; --i) {
        n += digits[i]*pow10;
        pow10 *= 10;
    }
    return n;
}

class Rats {  // the Rats class interface
    public:
        Rats();
        Rats(unsigned s);
        vector<unsigned> ratSeq;
        unsigned seed;
        void showRats();
```

```
};

Rats::Rats() {/*ctor*/ }

Rats::Rats(unsigned s) : seed(s) {
    ratSeq.push_back(seed);
    for(int i = 0; i < 500; ++i) { // change this to keep going until it loops
        ratSeq.push_back(ras(ratSeq[i])); //use helper function for next
    }
}

void Rats::showRats() {
    for(int i = 0; i < ratSeq.size()-1; ++i) {
        cout << ratSeq[i] << ", ";
        if((i+1)%6 == 0) cout << endl;
    }
    cout << " " << ratSeq[ratSeq.size()-1];
}
```

Here then is the main function:

```
///GH: Reverse, Add, Then Sort sequence

#include "std_lib_facilities.h"
#include "Rats.h"

int main() {
    Rats r(123);
    cout << "\nWe've created a Rats with seed " << r.seed;
    cout << "\nHere are the first 101 terms of the sequence: "
         << endl;
    r.showRats();
}
```

The output of this code is

```
We've created a Rats with seed 123
Here are the first 101 terms of the sequence:
123, 444, 888, 1677, 3489, 12333,
44556, 111, 222, 444, 888, 1677,
......(already looping)
```

1. Write code to meet these specifications:
   Precondition: A use-supplied word, like "BEST".
   Postcondition: A fully constructed base-26 Rats object from the seed "BEST", indicating the length the sequence and the length of the subsequence that loops.

   Recall that in base-26 we assign $A = 0, B = 1, C = 2, \ldots Z = 25$ so that BEST is the number $1 \cdot 26^3 + 4 \cdot 26^2 + 18 \cdot 26 + 19$ and so the next value would be In this case that would be UUWW, followed by BQRRS and so on.

   Submit the code using your initials in the usual format: say GH_Rats26.cpp