Problem 5. (15 points) ssa.h

In this problem you are to implement a simple STL-style algorithm:

```cpp
void set_same_as(Iterator start, Iterator end, int N = 0)
```

`set_same_as` sets all elements in the range `[start, end)` to the value of the element at position `start+N`.

Example:

```cpp
list<int> ints;
for (int i = 1; i <= 10; i++)
    ints.push_back(i*i);

copy(ints.begin(), ints.end(), ostream_iterator<int>(cout, " ")); cout << endl;
set_same_as(ints.begin(), ints.end(), 3);

Output:

1 4 9 16 25 36 49 64 81 100
16 16 16 16 16 16 16 16 16 16
```

Note that the code above does output using an `ostream_iterator`, not covered in class but mentioned on slides 316-317.

To get started on this problem, take another look at this implementation of `fill_n` from slide 306,

```cpp
template<typename OutputIterator, typename Tp>
OutputIterator fill_n(OutputIterator first, int n, const Tp& value)
{
    for ( ; n > 0; --n, ++first)
        *first = value;

    return first;
}
```

and this corrected implementation of `count`, from slide 309:

```cpp
template<typename InputIterator, typename T>
int count(InputIterator first, InputIterator last, const T& value)
{
    int n = 0;
    for ( ; first != last; ++first)
        if (*first == value)
            ++n;
    return n;
}
```
Think in terms of implementing `set_same_as` using these iterator operations:

```cpp
itr++
itr != itr
*itr = *itr
```

Here's the shell of the routine:

```cpp
template<typename Iterator>
void set_same_as(Iterator start, Iterator end, int n = 0)
{
    ...
}
```

All you need to do is write the five lines or so that go between the braces!

The deliverable is a file named `ssa.h`.

There's a single test program, `$FILES/a9/ssa.cc`. Correct execution of it is worth all 15 points. `$FILES/a9/testssa` is a simple test script.

**Problem 6. (15 points) truefalse.txt**

For this problem you are to write three true/false questions about C++. The topic of a question need not be limited to the C++ material we've covered. It may involve a comparison of some sort with C or Java. You may delve into any dark and dusty corners of C++ that you can find. Questions concerning footnotes on footnotes are fine!

You are to write an "easy" question, a "medium" question, and a "hard" question, worth 2, 5, and 8 points, respectively. Here's a question I'd consider to be "easy":

*In the context "cout << i;", the operator << is known as an extractor.*

Here's a "medium":

*Given void f() { X x1; }, a call to f() could fail due to lack of space in the heap.*

Here's a "hard":

*It requires over 50% more typing to define an abstract class in C++ than it does in Java.*

A trick question is fine, like this one, which I'd call a "medium": *The output of "cout << 2 & 4 << endl;" is "0".*

I'm not interested in questions based on convoluted logic that are essentially an exercise in following the flow of control, rather than probing knowledge of C++ features.

You may work in groups of any size on this problem but each person must submit a unique set of questions, and cite the members of their group.

If you can come up with a question that I consider to be "fiendish" that'll be worth ten points of extra credit.

**NOTE:** Just for fun I'd like to post to the mailing list the full set of questions developed by the class. I won't indicate who wrote which questions. **If do not want your questions posted, include a line that says "DO NOT POST".**

**Deliverables**

Use `turnin` with the tag `397a_9` to submit your solutions for grading. The full set of deliverables for the two parts of this assignment is `Set.h`, `vowels.cc`, `inherit.txt`, `ssa.h`, `truefalse.txt`, and if you choose to submit it, `extra.txt`. 