```
template <typename T>
class Vector
{
public:
    Vector(int initSize = 0)
        : theSize{ initSize },
          theCapacity{ initSize + SPARE_CAPACITY }
    { data = new T[theCapacity]; }

    // … Cut …

    // FOCUS ON ITERATORS

    typedef T* iterator;
    typedef const T* const_iterator;

    iterator begin()
    {
        return &data[0];
    }

    iterator end()
    {
        return &data[size()];
    }

    static const int SPARE_CAPACITY = 2;

private:
    int theSize;
    int theCapacity;
    T* data;
}
```

Vector Iterators in action:

```
int main()
{
    Vector<int> myvec;
    rand_seed();
    random_vector(15, 1, 100, myvec, 0);

    Vector<int>::iterator itr;

    cout << endl << endl;
  for (itr = myvec.begin(); itr != myvec.end();
                                                    ++itr)
    {
        cout << *itr << " ";
        *itr = -*itr;

    }
    cout << endl << endl;
    for (itr = myvec.begin(); itr != myvec.end();
                                                     ++itr)
    {
        cout << *itr << " ";

    }
    cout << endl << endl;

    return 0;
}
```

Output of running the program:

```
94 39 28 12 50 70 13 13 10 89 80 24 53 10 91

-94 -39 -28 -12 -50 -70 -13 -13 -10 -89 -80 -24 -53 -10 -91
```