// SequenceClassImplementation.cpp : This file contains the 'main' function. Program //execution begins and ends there.

```
#include <iostream>
#include<cassert>
using namespace std;
class Sequence {
public:
    typedef double value_type;
    typedef std::size_t size_type;
    static const size_type capacity = 30;
    Sequence();
    void start();
    void advance();
    void insert(const value_type& entry);
    void attach(const value_type& entry);
    void remove_current();
    size_type size() const;
    bool is_item() const;
    value_type current() const;

private:
    value_type data[capacity];
    size_type used;
    size_type current_index;
};

Sequence::Sequence()
{
    current_index = 0;
    used = 0;
}

void Sequence::start()
{
    current_index = 0;
}
```

```cpp
Sequence::value_type Sequence::current() const
{
    return data[current_index];
}

Sequence::size_type Sequence::size() const
{
    return used;
}

bool Sequence::is_item() const
{
    if (current_index < used)
        return true;
}

void Sequence::advance()
{
    if (is_item() == true)
        current_index++;
}

void Sequence::insert(const value_type& entry)
{
    assert(size() < capacity);
    if (is_item() == false)
        current_index = 0;
    for (int i = used; i > current_index; i--)
        data[i] = data[i - 1];
    data[current_index] = entry;
    used++;
}

void Sequence::remove_current()
{
    assert(is_item() == true);
    for (int i = current_index; i < used - 1; i++) {
        data[i] = data[i + 1];
        used--;
    }
}

void Sequence::attach(const value_type& entry)
{
    assert(size() < capacity);
    if (is_item() == false)
        data[used - 1] = entry;
    for (int i = used; i > current_index; i--) {
        data[i] = data[i + 1];
    }
    data[current_index] = entry;
    used++;
}


int main()
{
    Sequence s;
    s.insert(3.4);
    s.insert(5.4);
    s.insert(8.4);
    s.insert(5.4);
    s.insert(3.6);

    s.remove_current();
    cout << s.current();
    s.remove_current();
    cout << s.current();

}
```