```
Running tests for sequence class with a dynamic array

START OF TEST 1:
Testing insert, attach, and the constant member functions (4 points).
Starting with an empty sequence.
Testing that size() returns 0 ... Passed.
Testing that is_item() returns false ... Passed.
I'll call start() and look at the items one more time...
All tests passed for this sequence.

I am now using attach to put 10 into an empty sequence.
Testing that size() returns 1 ... Passed.
Testing that is_item() returns true ... Passed.
The cursor should be at item [0] of the sequence
(counting the first item as [0]). I will advance the cursor
to the end of the sequence, checking that each item is correct...Passed.
I'll call start() and look at the items one more time...
The cursor should be at item [0] of the sequence
(counting the first item as [0]). I will advance the cursor
to the end of the sequence, checking that each item is correct...Passed.
All tests passed for this sequence.

I am now using insert to put 10 into an empty sequence.
Testing that size() returns 1 ... Passed.
Testing that is_item() returns true ... Passed.
The cursor should be at item [0] of the sequence
(counting the first item as [0]). I will advance the cursor
to the end of the sequence, checking that each item is correct...Passed.
I'll call start() and look at the items one more time...
The cursor should be at item [0] of the sequence
(counting the first item as [0]). I will advance the cursor
to the end of the sequence, checking that each item is correct...Passed.
All tests passed for this sequence.

I am now using attach to put 10,20,30 in an empty sequence.
Then I move the cursor to the start and insert 5.
Testing that size() returns 4 ... Passed.
Testing that is_item() returns true ... Passed.
The cursor should be at item [0] of the sequence
(counting the first item as [0]). I will advance the cursor
to the end of the sequence, checking that each item is correct...Passed.
I'll call start() and look at the items one more time...
The cursor should be at item [0] of the sequence
(counting the first item as [0]). I will advance the cursor
to the end of the sequence, checking that each item is correct...Passed.
All tests passed for this sequence.

I am now using attach to put 10,20,30 in an empty sequence.
Then I move the cursor to the start, advance once, and insert 15.
Testing that size() returns 4 ... Passed.
Testing that is_item() returns true ... Passed.
The cursor should be at item [1] of the sequence
(counting the first item as [0]). I will advance the cursor
to the end of the sequence, checking that each item is correct...Passed.
I'll call start() and look at the items one more time...
The cursor should be at item [0] of the sequence
(counting the first item as [0]). I will advance the cursor
to the end of the sequence, checking that each item is correct...Passed.
All tests passed for this sequence.

I am now using attach to put 10,20,30 in an empty sequence.
Then I move the cursor to the start and attach 15 after the 10.
Testing that size() returns 4 ... Passed.
Testing that is_item() returns true ... Passed.
The cursor should be at item [1] of the sequence
(counting the first item as [0]). I will advance the cursor
to the end of the sequence, checking that each item is correct...Passed.
I'll call start() and look at the items one more time...
The cursor should be at item [0] of the sequence
```

```
(counting the first item as [0]). I will advance the cursor
to the end of the sequence, checking that each item is correct...Passed.
All tests passed for this sequence.

All tests of this first function have been passed.
Test 1 got 4 points out of a possible 4.
END OF TEST 1.


START OF TEST 2:
Testing situations where the cursor goes off the sequence (4 points).
Using attach to put 20 and 30 in the sequence, and then calling
advance, so that is_item should return false ... passed.
Inserting 10, which should go at the sequence's front.
Then calling advance three times to run cursor off the sequence ... passed.
Calling attach to put the numbers 40, 50, 60 ...300 at the sequence's end.
Now I will test that the sequence has 10, 20, 30, ...300.
All tests of this second function have been passed.
Test 2 got 4 points out of a possible 4.
END OF TEST 2.


START OF TEST 3:
Testing remove_current (4 points).
Using attach to build a sequence with 10,30.
Insert a 20 before the 30, so entire sequence is 10,20,30.
Testing that size() returns 3 ... Passed.
Testing that is_item() returns true ... Passed.
The cursor should be at item [1] of the sequence
(counting the first item as [0]). I will advance the cursor
to the end of the sequence, checking that each item is correct...Passed.
I'll call start() and look at the items one more time...
The cursor should be at item [0] of the sequence
(counting the first item as [0]). I will advance the cursor
to the end of the sequence, checking that each item is correct...Passed.
All tests passed for this sequence.

Remove the 20, so entire sequence is now 10,30.
Testing that size() returns 2 ... Passed.
Testing that is_item() returns true ... Passed.
The cursor should be at item [1] of the sequence
(counting the first item as [0]). I will advance the cursor
to the end of the sequence, checking that each item is correct...Passed.
I'll call start() and look at the items one more time...
The cursor should be at item [0] of the sequence
(counting the first item as [0]). I will advance the cursor
to the end of the sequence, checking that each item is correct...Passed.
All tests passed for this sequence.

Remove the 30, so entire sequence is now just 10 with no cursor.
Testing that size() returns 1 ... Passed.
Testing that is_item() returns false ... Passed.
I'll call start() and look at the items one more time...
All tests passed for this sequence.

Set the cursor to the start and remove the 10.
Testing that size() returns 0 ... Passed.
Testing that is_item() returns false ... Passed.
I'll call start() and look at the items one more time...
All tests passed for this sequence.

Using attach to build another sequence with 10,30.
Insert a 20 before the 30, so entire sequence is 10,20,30.
Testing that size() returns 3 ... Passed.
Testing that is_item() returns true ... Passed.
The cursor should be at item [1] of the sequence
(counting the first item as [0]). I will advance the cursor
to the end of the sequence, checking that each item is correct...Passed.
I'll call start() and look at the items one more time...
```

```
The cursor should be at item [0] of the sequence
(counting the first item as [0]). I will advance the cursor
to the end of the sequence, checking that each item is correct...Passed.
All tests passed for this sequence.

Remove the 20, so entire sequence is now 10,30.
Testing that size() returns 2 ... Passed.
Testing that is_item() returns true ... Passed.
The cursor should be at item [1] of the sequence
(counting the first item as [0]). I will advance the cursor
to the end of the sequence, checking that each item is correct...Passed.
I'll call start() and look at the items one more time...
The cursor should be at item [0] of the sequence
(counting the first item as [0]). I will advance the cursor
to the end of the sequence, checking that each item is correct...Passed.
All tests passed for this sequence.

Set the cursor to the start and remove the 10,
so the sequence should now contain just 30.
Testing that size() returns 1 ... Passed.
Testing that is_item() returns true ... Passed.
The cursor should be at item [0] of the sequence
(counting the first item as [0]). I will advance the cursor
to the end of the sequence, checking that each item is correct...Passed.
I'll call start() and look at the items one more time...
The cursor should be at item [0] of the sequence
(counting the first item as [0]). I will advance the cursor
to the end of the sequence, checking that each item is correct...Passed.
All tests passed for this sequence.

Remove the 30 from the sequence, resulting in an empty sequence.
Testing that size() returns 0 ... Passed.
Testing that is_item() returns false ... Passed.
I'll call start() and look at the items one more time...
All tests passed for this sequence.

Build a new sequence by inserting 30, 10, 20 (so the sequence
is 20, then 10, then 30). Then remove the 20.
Testing that size() returns 2 ... Passed.
Testing that is_item() returns true ... Passed.
The cursor should be at item [0] of the sequence
(counting the first item as [0]). I will advance the cursor
to the end of the sequence, checking that each item is correct...Passed.
I'll call start() and look at the items one more time...
The cursor should be at item [0] of the sequence
(counting the first item as [0]). I will advance the cursor
to the end of the sequence, checking that each item is correct...Passed.
All tests passed for this sequence.

Just for fun, I'll empty the sequence then fill it up, then
empty it again. During this process, I'll try to determine
whether any of the sequence's member functions access the
array outside of its legal indexes.
All tests of this third function have been passed.
Test 3 got 4 points out of a possible 4.
END OF TEST 3.


START OF TEST 4:
Testing the resize member function (2 points).
I will now resize a sequence to a larger capacity, and then
attach that many items. The sequence should NOT need to
resize itself under this situation.
    Test passed.
Now I will call resize(1) for the sequence, but the actual
sequence should not change because the sequence already has
60 items.
    Test passed.
All tests of this fourth function have been passed.
```

```
Test 4 got 2 points out of a possible 2.
END OF TEST 4.


START OF TEST 5:
Testing the copy constructor (2 points).
Copy constructor test: for an empty sequence.
Testing that size() returns 0 ... Passed.
Testing that is_item() returns false ... Passed.
I'll call start() and look at the items one more time...
All tests passed for this sequence.

Copy constructor test: for a sequence with cursor at tail.
Testing that size() returns 60 ... Passed.
Testing that is_item() returns true ... Passed.
The cursor should be at item [59] of the sequence
(counting the first item as [0]). I will advance the cursor
to the end of the sequence, checking that each item is correct...Passed.
I'll call start() and look at the items one more time...
The cursor should be at item [0] of the sequence
(counting the first item as [0]). I will advance the cursor
to the end of the sequence, checking that each item is correct...Passed.
All tests passed for this sequence.

Copy constructor test: for a sequence with cursor near middle.
Testing that size() returns 60 ... Passed.
Testing that is_item() returns true ... Passed.
The cursor should be at item [30] of the sequence
(counting the first item as [0]). I will advance the cursor
to the end of the sequence, checking that each item is correct...Passed.
I'll call start() and look at the items one more time...
The cursor should be at item [0] of the sequence
(counting the first item as [0]). I will advance the cursor
to the end of the sequence, checking that each item is correct...Passed.
All tests passed for this sequence.

Copy constructor test: for a sequence with cursor near middle.
Testing that size() returns 60 ... Passed.
Testing that is_item() returns true ... Passed.
The cursor should be at item [0] of the sequence
(counting the first item as [0]). I will advance the cursor
to the end of the sequence, checking that each item is correct...Passed.
I'll call start() and look at the items one more time...
The cursor should be at item [0] of the sequence
(counting the first item as [0]). I will advance the cursor
to the end of the sequence, checking that each item is correct...Passed.
All tests passed for this sequence.

Copy constructor test: for a sequence with no current item.
Testing that size() returns 60 ... Passed.
Testing that is_item() returns false ... Passed.
I'll call start() and look at the items one more time...
All tests passed for this sequence.

All tests of this fifth function have been passed.
Test 5 got 2 points out of a possible 2.
END OF TEST 5.


START OF TEST 6:
Testing the assignment operator (2 points).
Assignment operator test: for an empty sequence.
Testing that size() returns 0 ... Passed.
Testing that is_item() returns false ... Passed.
I'll call start() and look at the items one more time...
All tests passed for this sequence.

Assignment operator test: for a sequence with cursor at tail.
Testing that size() returns 60 ... Passed.
Testing that is_item() returns true ... Passed.
The cursor should be at item [59] of the sequence
(counting the first item as [0]). I will advance the cursor
to the end of the sequence, checking that each item is correct...Passed.
I'll call start() and look at the items one more time...
The cursor should be at item [0] of the sequence
(counting the first item as [0]). I will advance the cursor
to the end of the sequence, checking that each item is correct...Passed.
All tests passed for this sequence.

Assignment operator test: for a sequence with cursor near middle.
Testing that size() returns 60 ... Passed.
Testing that is_item() returns true ... Passed.
The cursor should be at item [30] of the sequence
(counting the first item as [0]). I will advance the cursor
to the end of the sequence, checking that each item is correct...Passed.
I'll call start() and look at the items one more time...
The cursor should be at item [0] of the sequence
(counting the first item as [0]). I will advance the cursor
to the end of the sequence, checking that each item is correct...Passed.
All tests passed for this sequence.

Assignment operator test: for a sequence with cursor near middle.
Testing that size() returns 60 ... Passed.
Testing that is_item() returns true ... Passed.
The cursor should be at item [0] of the sequence
(counting the first item as [0]). I will advance the cursor
to the end of the sequence, checking that each item is correct...Passed.
I'll call start() and look at the items one more time...
The cursor should be at item [0] of the sequence
(counting the first item as [0]). I will advance the cursor
to the end of the sequence, checking that each item is correct...Passed.
All tests passed for this sequence.

Assignment operator test: for a sequence with no current item.
Testing that size() returns 60 ... Passed.
Testing that is_item() returns false ... Passed.
I'll call start() and look at the items one more time...
All tests passed for this sequence.

Checking correctness of a self-assignment x = x;
Testing that size() returns 60 ... Passed.
Testing that is_item() returns true ... Passed.
The cursor should be at item [1] of the sequence
(counting the first item as [0]). I will advance the cursor
to the end of the sequence, checking that each item is correct...Passed.
I'll call start() and look at the items one more time...
The cursor should be at item [0] of the sequence
(counting the first item as [0]). I will advance the cursor
to the end of the sequence, checking that each item is correct...Passed.
All tests passed for this sequence.

All tests of this sixth function have been passed.
Test 6 got 2 points out of a possible 2.
END OF TEST 6.


START OF TEST 7:
Testing insert/attach when current DEFAULT_CAPACITY exceeded (3 points).
Testing to see that attach works correctly when the
current capacity is exceeded.
Testing that size() returns 60 ... Passed.
Testing that is_item() returns true ... Passed.
The cursor should be at item [59] of the sequence
(counting the first item as [0]). I will advance the cursor
to the end of the sequence, checking that each item is correct...Passed.
I'll call start() and look at the items one more time...
The cursor should be at item [0] of the sequence
(counting the first item as [0]). I will advance the cursor
```

```
to the end of the sequence, checking that each item is correct...Passed.
All tests passed for this sequence.

Testing to see that insert works correctly when the
current capacity is exceeded.
Testing that size() returns 60 ... Passed.
Testing that is_item() returns true ... Passed.
The cursor should be at item [0] of the sequence
(counting the first item as [0]). I will advance the cursor
to the end of the sequence, checking that each item is correct...Passed.
I'll call start() and look at the items one more time...
The cursor should be at item [0] of the sequence
(counting the first item as [0]). I will advance the cursor
to the end of the sequence, checking that each item is correct...Passed.
All tests passed for this sequence.

All tests of this seventh function have been passed.
Test 7 got 3 points out of a possible 3.
END OF TEST 7.

Your sequence implementation has scored
21 points out of the 21 points based on this test program.

```