All questions are worth one point unless indicated otherwise.  

**Multiple Choice:** Use the A-F choices below each sentence to answer that sentence’s question(s). For each question, place the letter of the best choice in the blank on the question’s left.

The following C++ code compiles and runs without error; use it to answer the questions on this page.

```cpp
// Data Structures Example Test Page

template<class Item>
class List {
public:
    List() { myFirst = NULL; mySize = 0; }
    unsigned getSize() const { return mySize; }
    void prepend(const Item& it) { myFirst = new Node(it, myFirst); mySize++;
    Item getFirst() const { return myFirst->myItem; }
private:
    struct Node {
        Node(const Item& it, Node* nxt) { myItem = it; myNext = nxt; }
        Item myItem;
        Node* myNext;
    };
    Node* myFirst;
    unsigned mySize;
    friend class ListTester;
};

int main() {
    List<double> aList; // 1
    aList.prepend(55); // 2
    aList.prepend(44); // 3
    aList.prepend(33); // 4
}
```

_____ 1. List is best described as a _____.
   A. method                B. friend                C. class
   D. class template        E. struct               F. None of these

_____ 2. After the line labeled 1 is performed, the expression `aList.getFirst()` will produce _____:
   A. 55                B. 44                C. 33
   D. a run-time error message  E. All of these  F. None of these

_____ 3. after the line labeled 2 is performed, the expression `aList.getFirst()` will produce _____; and
   _____ 4. after the line labeled 4 is performed, the expression `aList.getFirst()` will produce _____.
   A. 55                B. 44                C. 33
   D. a run-time error message  E. All of these  F. None of these

_____ 5. Which of these operations should also be defined by List?
   A. the assignment operator  B. an **append**() method  C. a copy constructor
   D. a destructor            E. All of these          F. None of these

_____ 6. Which of these best describes the time-complexity of the **prepend**() operation?
   A. \(O(1)\)              B. \(O(lg(N))\)             C. \(O(N)\)
   D. \(O(N^2)\)            E. \(O(2^N)\)             F. None of these
Problem Solving. Here is a test-method for the List on the preceding page:

```cpp
void ListTester::testGetLast() {
    cout << "- testing getLast()..." << flush;
    List<double> aList;
    assert( aList.getSize() == 0 );

    aList.prepend(33);
    assert( aList.getSize() == 1 );
    assert( aList.getFirst() == 33 );              // line numbers:
    assert( aList.getLast() == 33 );              // 16
    cout << " 0 " << flush;

    aList.prepend(22);
    assert( aList.getSize() == 2 );
    assert( aList.getFirst() == 22 );
    assert( aList.getLast() == 33 );              // 22
    cout << " 1 " << flush;

    aList.prepend(11);
    assert( aList.getSize() == 3 );
    assert( aList.getFirst() == 11 );
    assert( aList.getLast() == 33 );              // 28
    assert( aList.myFirst->myNext->myItem == 22 );
    cout << " 2 " << flush;

    cout << " Passed! " << endl;
}
```

When this method is compiled, the compiler produces this output:

```
ListTester.cpp: In member function ‘void ListTester::testGetLast()’:
ListTester.cpp:16: error: ‘class List<double>’ has no member named ‘getLast’
ListTester.cpp:22: error: ‘class List<double>’ has no member named ‘getLast’
ListTester.cpp:28: error: ‘class List<double>’ has no member named ‘getLast’
```

1. In the space below, define a data-independent List method that will allow this test-method to compile and pass its tests, without adding any new instance variables to the List. Neatness counts! (9 pts)

2. What is the time-complexity of your operation?