Size	First	Middle	Last
10	21.903	20.6733	22.7345
100	22.5984	22.5406	22.5436
1000	22.5408	22.5439	22.5431
10000	22.5358	22.5425	22.4747
100000	21.3592	21.3583	21.3597

Size	Vector	List								
10	174.4447	74.4206								
100	183.843	76.2866								
1000	250.5777	74.677								
10000	1777.369	76.9513								
100000	20014.2009	90.1448								
			1. The more elements a vector has, the longer it takes to append new elements to it. The unrepeated method of appending to a vector of size n takes constant time (O(1)), while the repeated method takes linear time (O(n)).							
			2. The time it takes to append an item to a list does not change significantly as the list gets longer. This is true for both the timeAppendToListOfSize(n) and timeRepeatedAppendToListSize(n) methods, which both have Big Oh notations of O(n).							
			3. As datasets get larger, it takes longer to add an item to a vector than to a list. This means that using a list for add operations is better for large datasets.							

Size	Vector	List	List(push_front)	
10	203.5167	16.584	14.4956	
100	218.454	18.2109	14.1911	
1000	301.189	22.8378	14.3782	
10000	1798.525	22.8958	15.279	
100000	20131.644	23.5198	15.909	
				1. It takes much longer to add an item to the beginning of a vector as the vector gets larger, but it takes about the same amount of time to add an item to the beginning of a list, no matter how large the list is. This is because prepending an item to a vector requires shifting all of the existing items in the vector over to make room for the new item, while prepending an item to a list does not require any shifting.
				2.Prepending an item to a list takes about the same amount of time, regardless of how long the list is. The functions timePrependToListOfSize(n) and timeRepeatedPrependToListSize(n) both prepend one item to the list at a time, up to size n. However, the timeRepeatedPrependUntilListIsSize(n) function has a Big Oh notation of O(1). This means that the timeRepeatedPrependUntilListIsSize(n) function takes longer to execute as the size of the list grows, but the timePrependToListOfSize(n) function takes the same amount of time to execute, regardless of the size of the list.
				3. It takes the same amount of time to add an item to the beginning of a list, regardless of how large the list is. However, it takes longer to add an item to the beginning of a vector as the vector gets larger. This means that using a list for add operations is more efficient for larger datasets.
				4. Adding an item to the beginning of a linked list using push_front() is faster than using insert(). This is because push_front() simply adds a new node to the front of the list, while insert() has to shift all of the existing nodes over to make room for the new node. As a result, push_front() has a Big Oh notation of O(1), while insert() has a Big Oh notation of O(n).