6.851 Advanced Data Structures (Spring'14)

Prof. Erik Demaine TAs: Timothy Kaler, Aaron Sidford

Problem 1 *Due: Tuesday, Feb. 18*

Be sure to read the instructions on the assignments section of the class web page.

Fully Retroactive Dynamic Read-only Array Describe and analyze a *fully retroactive* data structure for storing a list of elements that you can access like an array. Starting with an empty list your data structure should support `Insert`$(t, \texttt{update})$ and `Delete`$(t, \texttt{update})$ where t denotes the time of the operation and `update` is one of the following four update operations:

- `addL`(x): Add element x to the left of the list.
- `addR`(x): Add element x to the right of the list.
- `remL`$()$: Remove the leftmost element from the list.
- `remR`$()$: Remove the rightmost element from the list.

In addition your data structure should support `Query`$(t, \texttt{query})$ where t denotes the time of the operation and `query` is one of following two query operations:

- `size`$()$: Return the number of elements in the list.
- `get`(i): Counting from the left, return the ith element in the list.

Your data structure should support retroactive updates, `Insert`$(t, \texttt{update})$ and `Delete`$(t, \texttt{update})$ where $\texttt{update} \in \{\texttt{addL}(x), \texttt{addR}(x), \texttt{remL}(), \texttt{remR}()\}$, and retroactive queries, `Query`$(t, \texttt{query})$ where $\texttt{query} \in \{\texttt{size}(), \texttt{get}(i)\}$, in $O(\log m)$ time per operation, where m denotes the number of update operations (i.e. `addL`, `addR`, `remL`, and `remR`) in the current timeline of the data structure. The space usage should be $O(m)$, though partial credit will be given for a solution using $O(m \log m)$ space.