# Long Polynomial Division Discussion

1. Pros and Cons of using Dynamic Array and Linked-List for the algorithm

|  |  |  |
| --- | --- | --- |
|  | Dynamic Array | Linked-List |
| Advantages | Accessing a random term knowing the pow is really fast and efficient. O(1) | There is no need to pre-allocate any memory, term could be added at any time since it only requires to add a new node to the chain. |
| Disadvantages | The degree of the polynomial has to be defined before initialization in order to allocate the memory. | Accessing a random term is linear and slow, since you have to iterate through the whole linked-list to find the correct element. O(n) |

1. Assumption(s) need to be made if static array were used instead of dynamic array

* The performance is faster since static array is allocated in the stack.
* The stack has a very limited memory compared to the heap which is where dynamic arrays are stored. Thus if using static array, then the maximum polynomial’s degree supported by the program must be defined pre-compile time, and is very small.

1. What other error checking methods could be implemented

* Right now the application makes some assumption about the polynomial input. For instance: no spaces are allowed. It could be added by trimming all the space in the program and then process normally

1. Actually the program has already supported coeffs as double, but since the error checking doesn’t allow the users to enter floating point number as coeff; I have to reject those inputs.

If we were to allow coeffs to be double, it’s just matter of removing some code.