COMP3121 Assignment 3

Question 1

1.1 Since we already know the last two element a_j and a_k, and the sequence is beautiful, we know the last three elements in the subsequence satisfy the relationship, or $3a_i + 5a_j = a_k$. If we denote the third last element by a_i.
Hence, we can find the value of a_i by $(a_k - 5a_j)/3$. Then we may use binary search to find the position of a_i in the original sequence, which is strictly increasing by definition. This binary search cost $O(\log n)$ if total number of elements is n. Thus, total time cost of our algorithm is $O(\log n)$ since finding value of a_i costs constant time.

1.2 According to question 1.1, we know that we can find the third last element in a beautiful sequence if we already know the last two. So we may apply DP and denote $T[i, j]$ be the length of the longest beautiful sequence with the last two elements being the $i\,th$ and $j\,th$ element in the original sequence, respectively. Then we may dynamically update the entries using:
$T[i, j] = T[k, i] + 1$, if we can find an element a_k that is $(a_j - 5a_i)/3$;
Otherwise, $T[i, j] = 2$.

We update the entries in increasing order of both i and j requiring $i < j$, and initial entries are given as $T[1, j] = 2$ for all j, since all beautiful sequences should have at least 3 elements so the second last element must be indexed 2 or greater. But we set this value to be 2 since later on we may use a_i and a_j as previous two elements. Notice a sequence with length 2 is not considered beautiful but can be a prefix of a beautiful sequence. Finally, we find the maximum entry in T, which costs $O(n^2)$ to secure the length of the global longest sequence.
Since we update $O(n^2)$ elements and each time we use $O(\log n)$ time to do binary search (as we discussed before). So total time cost is $O(n^2 \log n)$.

1.3 Once we find the max length, we may trace back using the relationship between neighboring elements in a beautiful sequence. Since when we go over all elements in T to find the maximum length we also have the corresponding indices i, j at hand, we can back-trace for the 3rd last element using binary search, and the 4th last, and so on and so forth. Each search costs $O(\log n)$ and we need to search for $O(n)$ elements, less than the total number of the original sequence. Thus we can rebuild the longest beautiful sequence in $O(n \log n)$ time.