Code Documentation & Analysis

- **Input:** The algorithm takes an RNA string as input. RNA strings consist of characters from the alphabet {A, U, C, G}, representing the bases Adenine, Uracil, Cytosine, and Guanine respectively.

- **Output:** Return the structure set with the start and end indices of all pairs, representing the secondary structure.

- **Algorithm:**

    1. Initialise the dynamic programming vector(opt) whenever i>= j - 4.

    2. Calculate the maximum number of base pairs in the secondary structure using a dynamic programming approach where `k` starts with 4 and `i` starts with 0 and both are incremented till `n` and `n-k`.

        2.1. Set j = i + k -1

        2.2. Compute the opt(i,j) using the recurrence relation:

        ```
        opt(i,j) = max(opt(i, j-1) , max(1 + opt(i, t-1) + opt(t+1, j-1)))
        ```

        where t ranges from i to j-5 such that it is allowed to pair with j

        2.3. The rules for forming a pair are:
        - (No sharp turns) The ends of each pair are separated by at least 4 intervening bases i.e. if (i , j) ε S, then i < j - 4.
        - The elements in each pair in S consist of either {A,U} or {C,G} (in either order).
        - S is a matching: no base appears in more than one pair.
        - (No knots) If (i,j) and (k,l) are two pairs in S, then we can't have i < k < j < l .

    3. Secondary Structure Generation:

        3.1. Iterate through all possible starting indices i and ending indices j.

        3.2.Check if the bases at indices i and j are complementary (A-U or U-A, C-G or G-C).

        3.3. If complementary, check if the length of the secondary structure formed by pairing these bases is greater than 1.

        3.4. If yes, add the pair (i, j) to the set structure, representing the start and end indices of the secondary structure.

    4. Return the secondary structure. Print the start and end indices of the secondary structure.