Worth: 2% **Due:** By 8:59pm on Tuesday 3 February

Remember to write your *full name* and *student number* prominently on your submission.

---

*Please read and understand the policy on Collaboration given on the Course Information Sheet. Then, to protect yourself, list on the front of your submission **every** source of information you used to complete this homework (other than your own lecture and tutorial notes). For example, indicate clearly the **name** of every student with whom you had discussions, the **title and sections** of every textbook you consulted (including the course textbook), the **source** of every web document you used (including documents from the course webpage), etc.*

*For each question, please write up detailed answers carefully. Make sure that you use notation and terminology correctly, and that you explain and justify what you are doing. Marks **will** be deducted for incorrect or ambiguous use of notation and terminology, and for making incorrect, unjustified, ambiguous, or vague claims in your solutions.*

In bioinformatics, we often work with long strings (representing DNA sequences) that must be broken up into substrings. For this problem set, we consider a simplification of one particular type of problem that is encountered in practice.

Suppose we have a string $s = s_1 s_2 \cdots s_m$, where each individual symbol s_i comes from a fixed alphabet (for example, $s = GTCGGTAA$) and a number of *break positions* $0 < p_1 < \cdots < p_k < m$ (for example $p_1 = 2$, $p_2 = 3$, $p_3 = 6$). The goal is to break up s into substrings $s_1 \cdots s_{p_1}$, $s_{p_1+1} \cdots s_{p_2}$, …, $s_{p_k+1} \cdots s_m$ (in our example, s splits up into GT, C, GGT and AA).

Breaking up an arbitrary string t into substrings $t_1 t_2$ takes time proportional to the length of t, denoted $|t|$. This is independent of where in t the break occurs, because every character of t must be copied no matter what. But when performing multiple breaks, the order in which the breaks are performed can have an impact on the total time required. For example, imagine some input with length $|s| = 20$ and breakpoints $p_1 = 2$, $p_2 = 8$, $p_3 = 10$.

- If the breaks are performed in the order $[2, 8, 10]$, then the total time is $20 + 18 + 12 = 50$:
  - the first break ($s_1 s_2 / s_3 \cdots s_{20}$) takes time 20,
  - the second break ($s_3 \cdots s_8 / s_9 \cdots s_{20}$) takes time 18,
  - the third break ($s_9 s_{10} / s_{11} \cdots s_{20}$) takes time 12.
- If the breaks are performed in the order $[8, 2, 10]$, then the total time is $20 + 8 + 12 = 40$:
  - the first break ($s_1 \cdots s_8 / s_9 \cdots s_{20}$) takes time 20,
  - the second break ($s_1 s_2 / s_3 \cdots s_8$) takes time 8,
  - the third break ($s_9 s_{10} / s_{11} \cdots s_{20}$) takes time 12.
- Other permutations of $[p_1, p_2, p_3]$ may yield even smaller total times.

Follow the dynamic programming framework from class to give a detailed solution to this problem. Make sure to keep a clear distinction between each of the steps and to explain what you are doing and why at each step—you will **not** get full marks if your answer contains only equations or pseudocode with no explanation or justification.

To be precise, the input is as described above (a string $s = s_1 s_2 \cdots s_m$ and break positions $0 < p_1 < \cdots < p_k < m$) and the output is a permutation of $[p_1, \ldots, p_k]$ giving an order in which to perform the breaks that minimizes the total time required.