University of Freiburg
Dept. of Computer Science
Prof. Dr. F. Kuhn
S. Faour, M. Fuchs, A. Malyusz

Algorithm Theory
Exercise Sheet 2

Due: Monday, 4th of November 2024, 10:00 am

Assumption: You may assume that calculations with real numbers can be performed with arbitrary precision in constant time.

Exercise 1: Faster Polynomial Multiplication *(5 Points)*

Let $p(x) := 4x^3 - 2x + 1$ and $q(x) := -3x^2 + x + 5$. The goal is to compute $p(x) \cdot q(x)$ with the help of the FFT algorithm. Please, make use of the following sketch:

1. Illustrate the **divide** procedure of the algorithm (for both functions p and q). More precisely, for the i-th divide step (with focus on $p(x)$), write down all the polynomials p_{ij} for $j \in \{0, ..., 2^i - 1\}$ that you obtain from further dividing the polynomials from the previous divide step $i-1$ (we define $p_{00} := p$, and the first split is into p_{10} and p_{11} and so on...).
2. Illustrate the **combine** procedure of the algorithm (for both functions p and q). That is, starting with the polynomials of the smallest degree as base cases, compute the DFT of p_{ij} (respectively q_{ij}) bottom up with the recursive formula given in the lecture.
3. **Multiply** the polynomials. More specific, give the point value representation of $p(x) \cdot q(x)$.
4. Use the **inverse** DFT procedure from the lecture to get the final coefficients for $p(x) \cdot q(x)$.

Write down all intermediate results to get partial points in the case of a typo.

Exercise 2: Space Trader's Dilemma *(7 Points)*

A space trader has just landed on a distant planet, known for its rare minerals. As usual, they brought their interstellar cargo pod, which can hold exactly k minerals. There are n types of minerals available on the planet, and each type is available in an unlimited supply. The value of one mineral of type i is a_i, with the values sorted in increasing order.
The trader must carefully select exactly k minerals (potentially taking multiple minerals of the same type) to maximize their profit. Your task is to find all possible distinct total values that the space trader can achieve by selecting exactly k minerals.

Runtime Complexity

The expected runtime complexity for solving this problem should be $O(W \log W + n \log k)$, where $W = k \cdot a_n$ is the maximum achievable total value using k minerals.

Hint: One approach to solving this problem is to define a polynomial whose coefficients represent possible mineral values, then apply the Discrete Fourier Transform (DFT). After manipulating the resulting point representation, use the inverse transform to obtain the final answer.

Exercise 3: Counting k-Inversions in a String *(8 Points)*

You are given a string S consisting of only the characters 'A' and 'B'. For each integer k between 1 and $n-1$ (where n is the length of the string), we define a k-inversion as a pair of indices (i, j) such that:

$$1 \leq i < j \leq n,$$

$$j - i = k,$$

$$S[i] = \text{'B'}, \quad S[j] = \text{'A'}.$$

In other words, a k-inversion is a pair of indices (i, j) such that the character at position i is 'B', the character at position $i + k$ is 'A'. For each $k \in \{1, 2, \ldots, n-1\}$, your task is to compute the number of k-inversions in the string S.

Input

The input consists of a single string S, where the string consists only of the characters 'A' and 'B'. The length of the string is denoted by n.

Output

Output $n-1$ integers. The k-th integer (for $k \in \{1, 2, \ldots, n-1\}$) should represent the number of k-inversions in the string.

Example

Input:

```
BABA
```

Output:

```
2
0
1
```

Explanation

Consider the string $S = \text{'BABA'}$, which has length $n = 4$.
- For $k = 1$, the valid pairs are $(1, 2)$ and $(3, 4)$, so there are 2 k-inversions. - For $k = 2$, there are no valid pairs, so the number of k-inversions is 0. - For $k = 3$, the valid pair is $(1, 4)$, so there is 1 k-inversion.

Runtime Complexity

A naive solution would involve iterating through all pairs of indices, leading to a time complexity of $O(n^2)$. We can improve the time complexity to efficiently solve the problem in $O(n \log n)$.

Hint: Try defining two polynomials. The product of these polynomials will reveal coefficients that correspond to counts of k-inversions for specific k values.