

## Introduction to Computer Programming and Data Structures

### Assignment 07

Maximum Marks: **120**

Submission Deadline: **2022-Nov-19**

Topic: Polynomial Operations

#### Assignment problem # AP0701

**Polynomial operations:** Given two polynomials  $f(x) = \sum_{i=0}^n a_i \cdot x^i$ ,  $g(x) = \sum_{i=0}^m b_i \cdot x^i$  of degree  $n$  and  $m$  respectively, find addition/ subtraction/division/multiplication of them. You should have at least the following operations.

- $poly\_A \leftarrow poly\_init(n)$ : Given a non-negative integer  $n$ , it initializes a polynomial structure  $poly\_A$ . Here it allocates memory for the coefficients and the degree in  $poly\_A$ . Consider the coefficients as `float` variables.
- $b \leftarrow poly\_display(poly\_A)$ : Given a polynomial  $poly\_A$ , it should display the polynomial. Output should be in such a way that all of your friends can understand. Finally it returns a status bit  $b$  ( $b$  = degree of the polynomial if success, else return -1, in case of failure). The coefficients must be displayed up to 2 decimal places.
- $b \leftarrow poly\_free(poly\_A)$ : Given a polynomial  $poly\_A$ , it makes the memory allocated for the coefficients free. Finally, it returns a status bit  $b$  ( $b$  = degree of the polynomial if success, else return -1, in case of failure).
- $poly\_C \leftarrow poly\_add(poly\_A, poly\_B)$ : Given two polynomials  $poly\_A$  and  $poly\_B$ , it outputs  $poly\_C = poly\_A + poly\_B$  and displays  $poly\_C$  in the terminal.
- $poly\_C \leftarrow poly\_sub(poly\_A, poly\_B)$ : Given two polynomials  $poly\_A$  and  $poly\_B$ , it outputs  $poly\_C = poly\_A - poly\_B$  and displays  $poly\_C$  in the terminal
- $PoliDivRes \leftarrow poly\_div(poly\_A, poly\_B)$ : Given two polynomials  $poly\_A$  and  $poly\_B$ , it outputs  $PoliDivRes$  which stores  $poly\_R$  (remainder) and  $poly\_Q$  (quotient) such that  $poly\_A = poly\_B * poly\_Q + poly\_R$  and displays  $poly\_R$  and  $poly\_Q$  in the terminal
- $poly\_C \leftarrow poly\_mult(poly\_A, poly\_B)$ : Given two polynomials  $poly\_A$  and  $poly\_B$ , it outputs  $poly\_C = poly\_A * poly\_B$  and displays  $poly\_C$  in the terminal.
- $poly\_C \leftarrow poly\_mult\_dnc(poly\_A, poly\_B)$ : Given two polynomials  $poly\_A$  and  $poly\_B$ , it outputs  $poly\_C = poly\_A * poly\_B$  using divide-and-conquer algorithm and displays  $poly\_C$  in the terminal.

**Input format:** A file containing  $(3k + 1)$  lines.

- Line 1 contains the number of test cases, i.e.,  $k$ .
- Each test case has three lines:
  1. line 1 contains  $n\ m\ op$ , separated by space, where  $op \in \{+, -, *, /\}$ , and  $n$  and  $m$  are degrees of the input polynomials.
  2. line 2 contains space separated coefficients of the 1st polynomial with degree  $n$  as  $a_n a_{n-1} \dots a_0$ .
  3. line 3 contains space separated coefficients of the 2nd polynomial with degree  $m$  similar to the above.

**Output format:** Any Readable format. For multiplication, output results from both `poly_mult` and `poly_mult_dnc`.

**Notes:**

- `poly_A`, `poly_B`, `poly_C`, etc., are structures that store polynomials (i.e., store the coefficient and degree).
- Free the memories occupied by the polynomials, if any, before terminating the program at any stage.

[10+10+10+10+10+25+25+30+20]