

Programming Assignment #2: Polynomial GCD

(due 23:59, Oct 28th, 2020)

Introduction:

The greatest common divisor(GCD) for two or more univariate polynomials is important and elementary in algebra. For example, polynomial GCD can be used in root-finding algorithm when polynomials have multiple roots. Further, GCD computations allow us to compute the square-free factorization, even for multivariate polynomials. In computer algebra systems, most of the modern theory of polynomial GCD has been developed to satisfy the need of efficiency.

Objective:

In this programming assignment, you are asked to write a C++ program to derive the GCD of two univariate polynomials and return the GCD.

Provided files:

- (1) **main.cpp**: it executes the function *FindGCD()* and checks the answer. It can be changed if necessary for you to debug.
- (2) **GCD.cpp & GCD.h**: these are the program files you need to implement. The function *FindGCD()* receives two arrays, each of the arrays contains all the coefficients from 1 input polynomial. *FindGCD()* will return the GCD of input polynomials in the same way(as a coefficient array).
- (3) **example**: this is an exemplary input test case, which can be used to test your program.

Implementation Details:

The polynomial $A_0x^0 + A_1x^1 + A_2x^2 + A_3x^3 + \dots + A_{998}x^{998} + A_{999}x^{999}$ will be stored in a **long integer** array with size=1000 as follows:

array[0]	array[1]	array[2]	array[998]	array[999]
A ₀	A ₁	A ₂	A ₉₉₈	A ₉₉₉

The two inputs and the answer will all be stored in this format, and the output of *FindGCD()* should also be stored in the same way, with the same array size.

The returned GCD should be a **polynomial function with irreducible integral coefficients**, and **the coefficient of the highest power of the variable should be positive**. If there is no GCD between two inputs, you should return 1. Also, if both of

the input are zero-degree polynomials (constant polynomials), you should return 1 as well.

Example:

GCD you calculate	GCD you should return
$3x^3-6$	x^3-2
$-3x^3+6$	x^3-2
6	1

The input file will look like this:

Case 1	x^3-1 x^2-2x+1 $x-1$	<- Polynomial 1 <- Polynomial 2 <- GCD of Polynomial 1 & Polynomial 2
Case 2	$x^{999}-1$ $x^{601}-1$ $x-1$	<- Polynomial 1 <- Polynomial 2 <- GCD of Polynomial 1 & Polynomial 2
Case 3	2 5 1	<- Polynomial 1 <- Polynomial 2 <- GCD of Polynomial 1 & Polynomial 2
Case 4	2 $x^{999}-1$ 1	<- Polynomial 1 <- Polynomial 2 <- GCD of Polynomial 1 & Polynomial 2
Case 5	$516x^2+1603x+1225$ $-56x-98$ $4x+7$	<- Polynomial 1 <- Polynomial 2 <- GCD of Polynomial 1 & Polynomial 2

There are 900 test patterns in the example case. The first and the second lines indicate 2 input polynomials, and the third line is the golden answer (the GCD of the input polynomials).

* Note that the input polynomials would all be nonzero polynomials, and the coefficients will always be smaller than **INT_MAX**. However, the coefficients may exceed **INT_MAX** during your calculation. (**INT_MAX**= $2^{31}-1=2147483647$)

Language:

C or C++.

Platform:

You may develop your software on UNIX/Linux.

Compile: \$ g++ main.cpp GCD.cpp

Execution: \$./a.out

Submission

Please compress the following files into a zip file and name it by your **name and student ID**. For example, “HW2_0850241_葉季佑.zip”. Then upload the compressed file to the new E3 website by the deadline (**Oct 28th, 2020**).

- (1) GCD.h
- (2) GCD.cpp

Grading policy:

- (1) Example case correctness (60%)
- (2) Hidden case correctness (10%)
- (3) Hidden case ranking (30%, ranked by run time)