RMIT UNIVERSITY

SCHOOL OF SCIENCE & TECHNOLOGY

EEET2482 – SOFTWARE ENGINEERING DESIGN

PROJECT – LONG DIVISION OF POLYNOMIALS

SPECIFICATIONS:

You are required to write a console program which performs long division on polynomials, where the user enters the polynomials into the console program. The result (the quotient and remainder) is to be displayed to the console. The polynomials considered for this task will be of the form

$$a_n x^n + a_{n-1}x^{n-1} + a_{n-2}x^{n-2} + \cdots + a_2 x^2 + a_1 x + a_0 = 0,$$

where a_n is the n^{th} coefficient of the n^{th} term in the polynomial. No restriction will be placed on the degree of the polynomial, i.e. n can be any positive integer. The coefficients a_n can be any positive or negative integer value (float values will not be assessed). The *divisor* is the polynomial by which another polynomial will be divided into, the *dividend* is the polynomial to be divided and the *quotient* is the result of dividing the divisor by the dividend, i.e.

$$\frac{dividend}{divisor} = quotient$$

The user must be asked by the console program to enter two polynomials, one for the dividend and the other for the divisor. The coefficients and index values are not allowed to be stored in your program code. Each polynomial must be entered in to the console program in the form

$$a_n * x\text{^}n + a_{n-1} * x\text{^}(n-1) + a_{n-2} * x\text{^}(n-2) + \cdots + a_2 * x\text{^}2 + a_1 * x + a_0$$

For example, $x^5 + 3x^2 + 1$ must be explicitly entered in to the console program as

$$x\text{^}5 + 3 * x\text{^}2 + 1$$

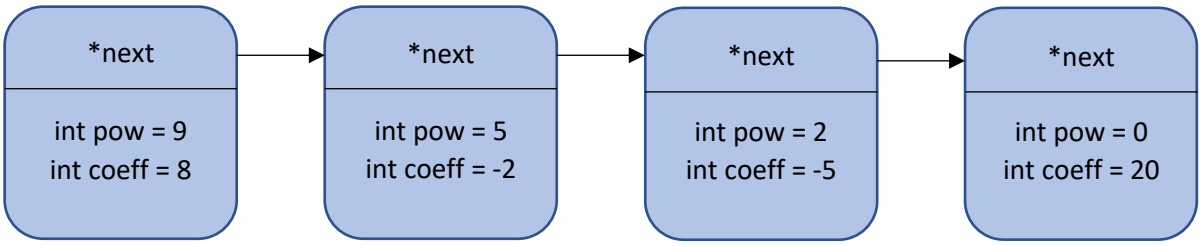
You will need to perform error checks to ensure that a valid polynomial has been entered in to your console program. A valid polynomial must meet the following criteria:

- All coefficients must be integer values. If a non-integer coefficient is detected, your console program must exit with a message stating the exact error which has been found.
- All powers must be integer values. If a non-integer power is detected, your console program must exit with a message stating the exact error which has been found.
- Any illegal characters must be detected and rejected by your console program. A message stating the exact error found must be displayed to the console, followed by program termination.
- Any term in the polynomial, where $a_n = 0$, must not be entered in to your console program.
- If the divisor is a higher degree polynomial than the dividend, your console program must terminate with an appropriate message displayed to the console.

You will be required to construct an algorithm to perform long division analytically. Numerical implementations are not allowed. You will also be required to perform long division using two different methods. Using the first method, the polynomial coefficients will be stored into a dynamic array where the index of the array is related to the degree of the polynomial term, e.g. the polynomial $10x^7 + 3x^5 - 5x^2 + 4$ is to be stored into a dynamic array as

Dynamic Array (Coefficients)	10	0	3	0	0	-5	0	4
Array Index	0	1	2	3	4	5	6	7

Using the second method, the polynomial coefficients and power terms are to be stored in a linked-list, where only the non-zero terms are stored, e.g. the polynomial $8x^9 - 2x^5 - 5x^2 + 20$ is stored as



Your algorithm for performing the long division can be applied to the method using the dynamic array and the method using the linked-list. Each node in the linked-list will be a dynamically created instance of a class, where the class definition contains two integer variables, one for the power and the other for coefficient, and a pointer which points to the next node in the linked-list.

Once your program has determined the quotient and remainder, the results must be displayed to the console as

Method 1: Dynamic Array
Quotient = $a_n x^n + a_{n-1}x^{n-1} + \cdots + a_1 x + a_0$
Remainder =$a_n x^n + a_{n-1}x^{n-1} + \cdots + a_1 x + a_0$

Method 2: Linked-List
Quotient = $a_n x^n + a_{n-1}x^{n-1} + \cdots + a_1 x + a_0$
Remainder = $a_n x^n + a_{n-1}x^{n-1} + \cdots + a_1 x + a_0$

The executable file must be named in the form *project_s1234567.exe.* The program should only take in one user argument from the command line. The input argument can only be *0* or *1*. When *project_1234567.exe 0* is executed from the command line, your program MUST print the student ID string in the form
1234567 , s1234567@student.rmit.edu.au , Full_Name
When *project_1234567.exe 1* is executed from the command line, your program must function as normal, performing the required task. If any other user input argument is used, your program must terminate with a specific error message displayed to the console. If no, or more than two, input arguments are detected, your program must also terminate with a specific error message displayed to the console.

DELIVERABLES:

1. Flowchart which shows the operation of your console program. Your flowchart should contain adequate detail to completely demonstrate your long division algorithm.
2. C++ console program which performs long division on polynomials, as specified above.
3. A zip file containing your C++ source file, flowchart (pdf) and program executable file.
4. A discussion (no more than 1 page) which addresses the following statements
    a. State the advantages and disadvantages you have observed in using a dynamic array and a linked-list to perform long division of polynomials.
    b. What assumption(s) would need to be made if a static array were used in place of a dynamic array?
    c. What other error checking methods could be implemented to make your console application more robust and how would you go about implementing these?
    d. If your program allowed float values to be entered in for the polynomial coefficients, what changes would you have to make to your console program?

EEET2482 – LONG DIVISION OF POLYNOMIALS PROJECT MARKING RUBRIC

	Weight	0 Marks - Fail NN	1 – 2 Marks (Pass) PA	3 – 5 Marks (Credit) CR	6 – 8 Marks (Distinction) DI	9 – 10 Marks (High Dist.) HD
Console Program Functionality	40%	Console program does not execute due to compilation issues. Console program executes, but does not display any results	Console program executes and displays the quotient and remainder, but are incorrect for most polynomials entered in by the user	Console program executes and displays the quotient and remainder polynomials for some of the polynomials entered in by the user	Console program executes and displays the quotient and remainder for most of the polynomials entered in by the user	Console program executes and displays the quotient and remainder for all polynomials entered in by the user.
Error Checking	20%	No error checking has been implemented	Error checking has been implemented but barely performs any functionality	Some of the error checks work and the messages displayed to the console somewhat explain the error	Most of the error checks function as expected. Some minor functionality does not work. An adequate message is displayed to the console for every detected error	All the error checks function as expected. A clear and concise message is displayed to the console for every kind of error
Discussion	15%	No discussion has been provided which address the statements given in the project guide	A discussion has been provided, but the content does not address the content specified in the laboratory guide	A brief discussion has been provided which somewhat addresses the statements given in the project guide	The discussion addresses all the statements given in the project guide, but some minor details are missing and is not concise	A clear and concise discussion has been provided which addresses each statement given in the project guide
Flowchart	25%	No flowchart has been provided or the flowchart is completely different than the algorithm implemented in the source code	A basic flowchart has been given, but is severely lacking in detail and provides no insight on how the program has been coded	The flowchart somewhat explains the structure of the console program and long division algorithm, but substantial details are missing	The flowchart is complete and depicts the structure of the console program and long division algorithm, but some minor aspects are missing	The flowchart depicts the structure of the program and details the long division algorithm in clear and concise detail

Note: Compress all your files into a zip archive and submit the zip file to Canvas by the project due date. See Canvas for the actual project due date. Late submissions will incur a penalty of 5% per day (weekend days inclusive) of your project mark.