CSC8501 Coursework – 2022
Simply Polynomial
Due Friday 21st October 2022 at 4.30pm

Set by Graham.Morgan@ncl.ac.uk

Specification (what you need to do): You will write a computer program in C++ that creates a sequence of numbers from an algebraic expression. In addition, your C++ program may also derive an algebraic expression used to create a sequence of numbers. Your program will focus on algebraic expressions of one, two, three, and four terms (Monomial, Binomial, Trinomial, Quadrinomial), commonly termed polynomials. Each algebraic expression is limited to the existence of a single variable (which can appear in multiple terms), a highest degree of 4, a highest coefficient of 9, highest constant is 1000, and restricted to the operators of **+** or **-** only between terms. Assume all values are positive. Here are valid and invalid examples:

(1) Valid: $2x^2 + x^4 - 3$ (2) Invalid: $2x^7 + y^4 - 3000$

Example (2) is invalid as a degree is higher than 4, more than one variable exists and a constant is present that is greater than 1000.

The Requirements

- Provide a command line interface that allows a user to instruct your program to create an algebraic expression of the form described in the specification
- Provide a command line interface that allows a user to instruct your program to create a number series (**output set**) using the algebraic expression provided by a user
    - This is achieved by entering two numbers: ***start number***, ***finish number***
    - All integers between, and including, **start** and **finish numbers** form the **input set**
- Allow a user to view an **output set**
- Provide a command line interface that allows **output sets** to be saved to file and read from file. The format of such a file is: each **output set** occurs on a single line (**EOL** identifies the end of the **output set**), individual numbers are separated by a comma. You should be able to share these files with your fellow students' programs
- Provide a command line interface that allows the algebraic expression that created an **output set** to be derived and then displayed to a user. This is achieved by reading an **output set** from a file and then deducing the algebraic expression used to create it **FROM THE OUTPUT SET ALONE**. You should do this for the **output sets** listed in this coursework and be able to do this for files belonging to other students where you have no prior knowledge of how the **output sets** were created.
- Provide a command line interface that reads in (**batch style**) all **output sets** from a file and saves the algebraic expressions used to create each **output set** to another file (**expression file**)

Output Set List
What are the algebraic expressions used to create the following?

A. 1 term (using 0 - 20 as **input set**)
0, 3, 24, 81, 192, 375, 648, 1029, 1536, 2187, 3000, 3993, 5184, 6591, 8232, 10125, 12288, 14739, 17496, 20577, 24000

B. 2 terms (0 - 20 as **input set**)
-9, -4, 71, 396, 1271, 3116, 6471, 11996, 20471, 32796, 49991, 73196, 103671, 142796,, 192071, 253116, 327671, 417596, 524871, 651596, 799991

C. 3 terms (0 - 20 as **input set**)
0, 14, 96, 300, 680, 1290, 2184, 3416, 5040, 7110, 9680, 12804, 16536, 20930, 26040, 31920, 38624, 46206, 54720, 64220, 74760

D. 4 terms (0 - 20 as **input set**):
7, 13, 17, 13, -5, -43, -107, -203, -337, -515, -743, -1027, -1373, -1787, -2275, -2843, -3497, -4243, -5087, -6035, -7093

E. Unknown number of terms (0 - 20 as **input set**)
114, 110, 100, 78, 38, -26, -120, -250, -422, -642, -916, -1250, -1650, -2122, -2672, -3306, -4030, -4850, -5772, -6802, -7946

F. Unknown number of terms and unknown range for **input set**
715, 625, 523, 409, 283, 145, -5, -167, -341, -527, -725, -935, -1157

Deliverables (what we want to see submitted in a single zip file):

- C++ source code authored by the student
- Executable file containing solution
- A collection of files containing **output sets** used to validate your program
- The **expression file** holding the solutions to the **output sets** in this coursework
- Demonstration (your chance to explain and show your solution):
    - You will be notified via email when the demonstration will occur
- Submit your completed solution to NESS

Learning Outcomes (what we expect you to demonstrate in a general way)

- To be able to define and discuss the main aspects of a programming language
- To be able to design and create programs
- To be able to identify appropriate techniques for analysing the efficiency of programs
- To be able to realise inappropriate usage of programming languages
- To be able to manage memory
- To be able to create and use data structures
- To be able to use condition statements, loops and functions
- To be able to utilise concurrency when appropriate
- To be able to create programs that handle run-time errors
- To be able to use appropriate techniques for debugging and analysing existing algorithms
- To be able to design programs using a well-known methodology

Marks Available (100):

- **65 Marks for satisfying the coursework requirements**
  - **5** Marks for allowing a user to create an algebraic expression
  - **5** Marks for allowing a user to specify an **input set** and display the resultant output set
  - **5** Marks for saving an output set to a file
  - **5** Marks for reading in an output set from a file
  - **20 Marks** for for deriving an algebraic expression for output sets with known number of terms (5 Marks for each output set **A** through to, and including **D**)
  - **10 Mark**s for deriving an algebraic expression for the output set **E**
  - **15** Marks for deriving an algebraic expression for the output set **F**

- **35 Marks for programming style, efficiency, and correctness**
  - **5 Marks** for having an interface that can accept, and act upon, user instruction
  - **5 Marks** for reading in a file with exception handling
  - **5 Marks** for writing out a file with exception handling
  - **5 Marks** for using object-orientation
  - **5 Marks** for concise code (limiting method/function size to 10 lines of code)
  - **10 Marks** for advanced features (2 marks for each of the following up to a maximum of 10 marks overall): threading, templates, polymorphism, lambda functions, .h files with .cpp files, auto, any, ternary operator)

Hints and tips

1. Establish what class of problem this actually is first
2. Do not rush and try to solve easiest parts first
3. The human way of doing this may not be general enough for a machine solution
4. Exploit that machines can work much more quickly and accurately than humans
5. Do not miss out on the programming style, efficiency, and correctness marks