Lab Assignment #1 Grading Rubric

40 marks in Total

All parts completed according to the given specifications. Subtract marks for each component that is missing or that is not implemented according to specifications. **Provide feedback for subtracted marks.**

Detailed Mark Distribution:

3 marks for `Polynomial(int A[], int size);`

- Demonstrated that class constructor takes as input two parameters: an array of integers, `int A[]`, and the size of the array, `int size`. The inputted array contains all the polynomial coefficients. The constructor creates the matching Polynomial and populates an internal data with inputted values.

3 marks for `Polynomial();`

- Demonstrated that class constructor with no inputs generates a polynomial of random size from the `[0 to 1000]` range with coefficients from the `[-1000, 1000]` range. The constructor creates the matching Polynomial and populates an internal array with inputted values. Check that consecutive calls to this constructor generate different polynomials.

4 marks for `Polynomial(string fileName);`

- Demonstrated that class constructor takes as input one parameter: name of the file in which polynomial coefficient are stored, `string fileName`. Inside the file, the first line contains the polynomial size while other lines contain polynomial coefficients, with one coefficient stored per line. The constructor creates the matching Polynomial and populates an internal array with inputted values [2 marks]. Coefficients for negative exponents should not be stored when reading from a file [1 mark]. Missing coefficients should be set to zero when reading from a file [1 mark].

3 marks for `void print();`

- Demonstrated that the method prints the current polynomial in the required format. The method was used in at least three different situations [2 marks]. The coefficient for the highest exponent is not zero (e.g., handled when printing out the polynomial) [1 mark].
- The test cases for `print()` and the constructors can rely on visual inspection.

4 marks for `bool operator==(const Polynomial& target);`

- Demonstrated that the method compares the current and target polynomials [2 marks]. Provided three different test cases that clearly demonstrate required functionality [1 mark].
- Can handle an empty polynomial (e.g., treat an empty polynomial as a zero polynomial) [1 mark].

6 marks for `void Polynomial operator+(const Polynomial& target);`

- Demonstrated that the method adds the current and target polynomials [5 marks]. Provided three different test cases that clearly demonstrate required functionality [1 mark].

6 marks for `Polynomial operator-(const Polynomial& target);`

- Demonstrated that the method subtracts the current and target polynomials [5 marks]. Provided three different test cases that clearly demonstrate required functionality [1 mark].

6 marks for `Polynomial operator*(const Polynomial& target);`

- Demonstrated that the method multiplies the current and target polynomials [5 marks]. Provided three different test cases that clearly demonstrate required functionality [1 mark].

5 marks for `Polynomial derivative();`

- Demonstrated that the method computes the derivative of the current polynomial [4 marks]. Provided three different test cases that clearly demonstrate required functionality [1 mark].

If the code does not compile under Dev-C++, try it under another IDE, such as CLion for Windows or XCode for Mac OS. If the compilation issues are minor (e.g., one or two mistakes), correct those. If it still does not compile, make a note of this and award partial grades based on code walkthrough.