1.

A two dimensional dynamic array of `double`

Based on code from Display 10.9, write a `class TwoD` that implements a two-dimensional dynamic array of double.

Student should supply:

- Default constructor for which you choose a default maximum rows and maximum columns
- Parameterized constructor, which sets maximum rows and maximum columns
- A `void` member function that allows setting a particular row-column entry
- A `double` member function that allows retrieving a particular row-column entry
- Remark: Suggest accessor and mutator member functions for these accesses.
- Overload + as friend to add to matrices of the same size
- Overload =
- Copy constructor
- Destructor
- Use `const` for members that do not change

Notes: The function call operator is overloaded to serve as index. That way, matrix(i, j) returns the (i, j) element, while matrix(i) returns the ith row.

2.

Using dynamic arrays, implement a polynomial class with infix operators +, -, *.

Discussion:

In a polynomial, a variable is placeholder for coefficient. If term is missing, the coefficient is 0. For example, the cubic polynomial

    2x^3 -3x + 4

or, written out as C++ code,

```
2*x*x*x -3*x + 4
```

has the coefficients for terms as listed:

```
degree 3 term has coefficient 2,
degree  2 term has coefficient 0,
degree  1 term has coefficient -3, and
degree  0 term has coefficient 4.
```

Note that the term with degree 2 is missing, but we will use a coefficient of 0 to indicate that. Observe that the size of the coefficient array is `4`,one more than the degree.

Use of sparse matrix techniques not recommended. We will assume that there are few missing terms.

The student is to provide these member functions:

- default constructor,
- copy constructor,
- operator=
- destructor
- parameterized constructor to create an arbitrary polynomial
- operator+
- operator-
- operator*
- assign and inspect function (or functions) for
- coefficients, indexed by exponent
- function to evaluate polynomial as a value of type double

The student is to decide on whether these are to be member, friend, or neither (standalone).

3.

Write a program that accepts a C-string input from the user and verifies whether the input string forms a palindrome (for example, “madam” is a palindrome, wherein the reverse of the string is the same as the string) or not. Your program should work by using two pointers. The “head” pointer should be set to the address of the first character in the string, and the “tail” pointer should be set to the address of the last character in the string (i.e., the character before the terminating null). The program should compare the characters referenced by these pointers, increment “head” to point to the next character, decrement “tail” to point to the second-to-last character, and so on, until all characters have been compared.

4.
Extend HW5-1, the TwoD array class.
Place the class definition and implementation in a namespace, then provide access to the names in the namespace. Test. Access to namespace may be by qualification, using definition, or using directive. Global namespace directives are not allowed.

5.
Write a program that takes input from a file of number of type double. The program outputs to the screen the average and standard deviation of the numbers in the file. (Suggestion: Modify Project 2. The data file is exactly like the data file of #2).

Average = $\frac{\sum x_i}{\text{count}}$

Standard Deviation = $\frac{\sqrt{(x_i - a)^2}}{\text{count}}$

Instructor supplies data file and name.
Read entire file, compute average, then close and reopen the file to compute the Standard Deviation.

File:test

6
File Input to the program is the body of the letter. Keyboard Input to the program is the name and address of the recipients. The file contains all the letter except the names, which are denoted by #N#. This occurs *exactly* once in the letter. The program should copy the file to the output file, until the #N# is encountered. At this point, the program should ask the user for a name, accept the name, place it in the file. Then the program should finish copying the file to the output file. The main program has compiled-in file names for the input file and output file. It opens the files with error checking, and calls the function to do the work. A function should be defined to do the work that accepts one input and one output file stream. Obtain the file names from your instructor. The student should create files to test your program.

File:letterBody