MATLAB Programming NCTU Spring 2017 LAB#13 2017/6/2

For this lab, you will extend the previous lab on the BigInt class. We will focus on adding functionalities related to arrays of BigInt objects.

The following are the descriptions / tasks:

- Define a class **`BigInt`**. This class should have a single property named **`digits`**, which is a vector where every element is a digit of the big integer. You can set its default value to zero (which will be the only digit of this big integer).
- The constructor: The following forms of inputs are now accepted:
  - A single input that is a cell array of strings. Each string is converted to a **`BigInt`** object. The output **`BigInt`** array should have the same size as the input cell array. Generate an error message if any of the strings cannot be converted.
  - A single input that is an array of non-negative integer. The output **`BigInt`** array should have the same size as the input array.
- For the overloaded method **`disp`**, now print each array element in one line. Each line is proceeded by its array subscripts. For example, for an array with four elements:

```
101  202
123  345
```

  the generated output looks like

```
(1,1)   101
(2,1)   123
(1,2)   202
(2,2)   345
```

- For the overloaded operator **`plus`** and **`eq`**, they should handle the following input types:
  - Two **`BigInt`** arrays of the same size.
  - One **`BigInt`** array and one **`BigInt`** scalar, in any order.
  - One **`BigInt`** array and one scalar double, in any order.
- For the overloaded operator **`times`**, it should handle the following input types:
  - Two **`BigInt`** arrays of the same size. Otherwise, generate an error message.
- For the overloaded operator **`mtimes`**, it should handle the following input types:
  - Two **`BigInt`** arrays with sizes suitable for matrix multiplication.
  - One **`BigInt`** array and one **`BigInt`** scalar, in any order.
  - One **`BigInt`** array and one scalar double, in any order.

Try to test your class with the following code:

```
a = BigInt({'5678','123456'});  b = BigInt(99);
d = BigInt([11 22; 33 44]);  n = 808;
c = a + a;  disp(c);  c = a + n;  disp(c);
c = n + a;  disp(c);  c = a + b;  disp(c);
c = b * a;  disp(c);  c = n * c;  disp(c);
c = a .* a;  disp(c);  c = a * d;  disp(c);
disp(a + a == a * 2);
```