1. One nice example of overloading the function call operator () is to allow another form of double-array subscripting popular in some programming languages. Instead of saying

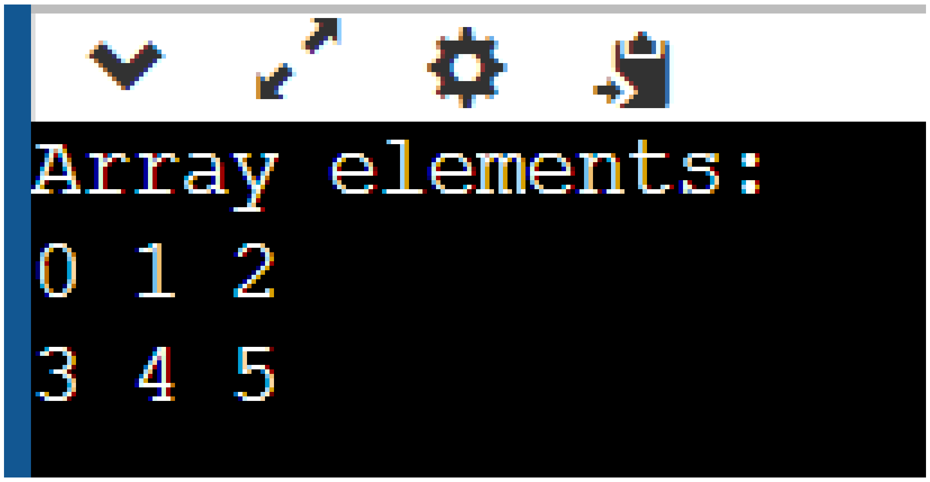
    chessBoard[row][column]

   for an array of objects, overload the function call operator to allow the alternate form

    chessBoard(row, column)

   Create a class *DoubleSubscriptedArray* that has similar features to class *Array* as following example programs. At construction time, the class should be able to create a *DoubleSubscriptedArray* of any number of rows and columns. The class should supply *operator()* to perform double-subscripting operations. For example, in a *3-by-5* *DoubleSubscriptedArray* called *chessBoard*, the user could write *chessBoard(1, 3)* to access the element at row *1* and column *3*. Remember that operator() can receive *any* number of arguments. The underlying representation of the *DoubleSubscriptedArray* could be a one-dimensional array of integers with *rows * columns* number of elements. Function *operator()* should perform the proper pointer arithmetic to access each element of the underlying array. There should be two versions of *operator()* - one that returns *int &* (so that an element of a *DoubleSubscriptedArray* can be used as an *lvalue*) and one that returns *int*. The class should also provide the following operators: ==, !=, =, << (for outputting the *DoubleSubscriptedArray* in row and column format) and >> (for inputting the entire *DoubleSubscriptedArray* contents).

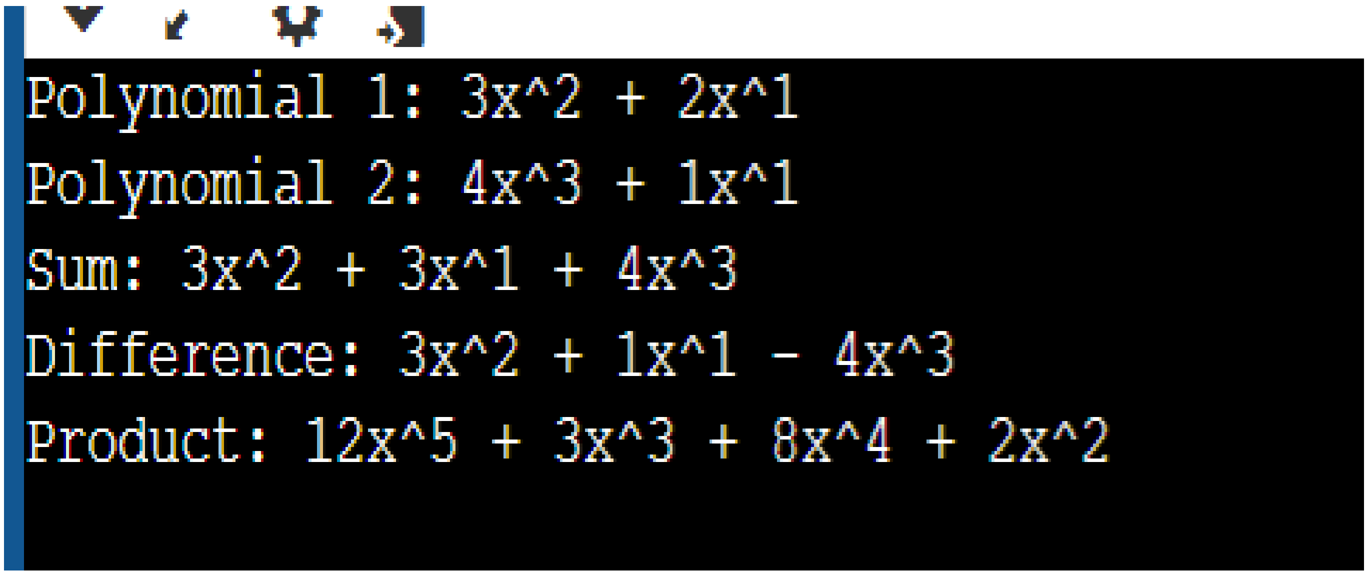
Here is the output and the main code is in the cpp file


```
Array elements:
0 1 2
3 4 5
```

2. Develop class *Polynomial*. The internal representation of a *Polynomial* is an array of terms. Each term contains a coefficient and an exponent, e.g., the term $2x^4$ has the coefficient *2* and the exponent *4*. Develop a complete class containing proper constructor and destructor functions as well as set and get functions. The class should also provide the following overloaded operator capabilities:

   a. Overload the addition operator *(+)* to add two *Polynomials.*
   b. Overload the subtraction operator *(-)* to subtract two *Polynomials*.
   c. Overload the assignment operator to assign one *Polynomial* to another.
   d. Overload the multiplication operator *(*)* to multiply two *Polynomials*.
   e. Overload the addition assignment operator *(+=)*, subtraction assignment operator *(-=)*, and multiplication assignment operator *(*=)*.

Here is the output and the main code is in the cpp file



```
Polynomial 1: 3x^2 + 2x^1
Polynomial 2: 4x^3 + 1x^1
Sum: 3x^2 + 3x^1 + 4x^3
Difference: 3x^2 + 1x^1 - 4x^3
Product: 12x^5 + 3x^3 + 8x^4 + 2x^2
```