National University of Computer and Emerging Sciences, Lahore Campus

	Course:	**OOP Lab**	**Code:**	**CL217**
	Program:	**BS (Computer Science)**	**Semester:**	**Summer 2019**
	Duration:	**130 minutes**	**T. Marks:**	**100 (60+40)**
	Date:	**Wednesday 10-07-2019**	**Weight**	**30%**
	Section:	**A**	**Page(s):**	**3**
	Exam:	**Lab Mid Term**	**Roll No:**	

Notes: No queries will be handled; plagiarism will be rewarded as 'F' grade.
Submission Path: \\cactus\Xeon\Summer 2019\Shakeel Zafar\Submissions\MID

Question # 1:

Complete the definition of the class given below such that the main program runs successfully. Make sure that your program doesn't consume extra memory space and it should not leak any memory.

```
class BinaryNum
{
private:
    int* binNum;          //integer array to save binary number
        int noOfBits;     //total no. of bits public:
 void Print(){   if(binNum != 0){
                        for(int i = 0 ; i< noOfBits ; i++)
                                cout<<binNum[i];
                }
                cout<<endl;
        }
};
void main()
{
        BinaryNum b1;                       //noOfBits = 0, binNum is NULL
        BinaryNum b2("101");                //noOfBits = 3, binNum is {1,0,1}
        BinaryNum b3("1001");                //noOfBits = 4, binNum is {1,0,0,1}

        cout<<"b1 = ";cout<<b1;             //Prints Nothing
        cout<<"b2 = ";cout<<b2;             //Prints 101
        cout<<"b3 = ";cout<<b3;             //Prints 1001

        b1 = b2+b3;
        cout<<"b1 = "<<b1;                  //Prints 1110
        cout<<"b1[0] = "<<b1[0]<<endl;      //Prints 1 (0th bit in b1)
        cout<<"b1[3] = "<<b1[3]<<endl;      //Prints 0 (3rd bit in b1)

        cout<<b3++;                         //Prints 1001
        cout<<"is equal= "<<b3==b2;                 //Prints 0 cout<<++b3;
                                        //Prints 1011

        b1 = b2 + "111";
        cout<<b1;                           //Prints 1100
```

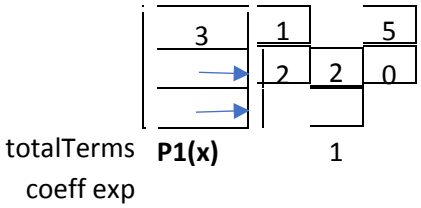
```
        cout<<b1-b2;                    //Prints 111

}
```

Question # 2:

A polynomial P1(x) = x^2+2x^1+5 has three terms: x^2, $2x^1$ and $5x^0$. Coefficients of these terms are 1, 2 and 5 respectively while exponents are 2, 1 and 0 respectively. You do not need to store exponent. Those will always be in decreasing order.

To work with Polynomials, a definition of class Polynomial is given below and memory configuration for P1 is shown as follows:

```
class Polynomial
{
private: int totalTerms;        //Total terms in a
Polynomial int* coeff;          //to save array of
coefficients };
```

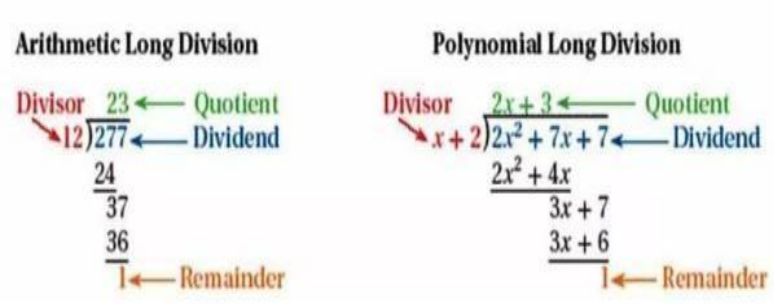


Your task is to complete the definition of Polynomial class such that the main program runs successfully. Make sure that your program doesn't consume extra memory space and it should not leak any memory.

```
void main()
{
        int coeff_P1[] = {1,2,5}, coeff_P2[] = {4,3};       //Coefficients for Polynomial P1 & P2 respectively

        Polynomial P1(3, coeff_P1);                  //Creates P1 with 3 terms (P1 = 1x^2 + 2x^1 + 5x^0 )
        Polynomial P2(2, coeff_P2);                  //Creates P2 with 2 terms (P2 = 4x^1 + 3x^0)

        cout<<"P1 = "; P1.Print();                   //Prints P1 = x^2+2x^1+5
        cout<<"P2 = "; P2.Print();                   //Prints P2 = 4x^1+3

        Polynomial P3 = P1*P2;                       //Multiplies P1 and P2 and saves result in P3. You may
                                                     //consume extra space for resultant Polynomial

        cout<<"P3 = "; P3.Print();                   //Prints P3 = 4x^3 + 11x^2 + 26x^1 + 15

        Polynomial P4 = P3/P2;                       //Prints P4 = x^2+2x^1+5
        cout<<"P4 = "; P4.Print();
}
```

P.T.O

For the above task you are given with following algorithm help.

Division Algorithm	**Multiplication Algorithm**
Step 0: Let we have A & B polynomial named for dividend and divisor respectively. We need two more polynomials named as quotient and remainder. **Step 1:** First arrange the term of dividend and the divisor in the decreasing order of their degrees. **Step 2:** To obtain the first term of quotient, divide the highest degree term of the dividend by the highest degree term of divisor. **Step 3:** To obtain the second term of the quotient, divide the highest degree term of the new dividend obtained as remainder by the highest degree term of the divisor. **Step 4:** Continue this process till the degree of remainder is less than the degree of divisor or remainder is zero. 	**Step 1:** Let we have polynomial A & B. Now, create a polynomial 'prod' which will have 'coeff' array of size m+n-1. **Step 2:** Initialize all entries in prod.coeff[] as 0. **Step 3:** Travers array A.coeff[] and do following for every element A.coeff[i] Traverse array B.coeff[] and do following for every element B.coeff[j] Prod.coeff[i+j] = prod.coeff[i+j] + A.coeff[i] * B.coeff[j]