PENN STATE UNIVERSITY
Department of Economics

Econ 597D Sec 001 Computational Economics — Gallant
Homework 6 — Fall 2015
Due Oct 6, 2015

Implement a class with the following declaration:

```
class polycoef {
private:
  INTEGER         deg;          //degree of polynomial
  REAL*           pc;           //array of len=deg+1 containing coefficients
  INTEGER         len;          //indexing is pc[i], i=0,...,deg
public:
                  polycoef();                        //default constructor
                  polycoef(INTEGER degree);          //explicit constructor
                  polycoef(const polycoef& a);       //copy constructor
                 ~polycoef();                        //destructor
  polycoef&       operator=(const polycoef& a);      //assignment operator
  REAL&           operator[](INTEGER i);             //lvalue element access
  const REAL&     operator[](INTEGER i) const;       //rvalue element access
  INTEGER         degree() const;                    //returns deg
  friend polycoef operator+(const polycoef& a, const polycoef& b);  //summation
};
```

The purpose of the class is to represent a polynomial and implement addition of polynomials.

Here is a main that uses every method in the class

```
int main()
{
  polycoef a(3);
  for (INTEGER i=0; i<=a.degree(); i++) a[i] = REAL(i);
```

```
  polycoef b(5);
  for (INTEGER i=0; i<=b.degree(); i++) b[i] = REAL(i);
  polycoef c;
  c = a + b;
  polycoef d = c;
  for (INTEGER i=0; i<=c.degree(); i++) cout << c[i] <<"  "<< d[i] << '\n';
  return 0;
}
```

This is a container class similar to `class intvec`. You can look at that code for hints on how to implement `class polycoef`. Notice that, unlike class `intvec`, the arguments of `operator+` need not have the same length; the `polycoef` that is returned will have length the larger of the two arguments.

If you are ambitious implement `operator*` for extra credit.

Turn in your code, a sample main that executes it, and the output.