Grupo Weboo
Universidad de La Habana

Trabajo de Control Parcial de Programación
Curso 2015-2016

Derivando Polinomios

NOTA: Si usted está leyendo este documento sin haber extraído el compactado que se le entregó, ciérrelo ahora, extraiga todos los archivos en el escritorio, y siga trabajando desde ahí. Es un error común trabajar en la solución dentro del compactado, lo cual provoca que los cambios **no se guarden**. Si usted comete este error y entrega una solución vacía, no tendrá oportunidad de reclamar.

Sea $P(x)$ un polinomio de grado n, tal que $P(x) = p_0 + p_1 \cdot x + \cdots + p_n \cdot x^n$, con coeficientes enteros. Este polinomio se puede representar computacionalmente empleando un *array* de tipo `int[]`, donde el índice i-ésimo almacena el coeficiente p_i. Por ejemplo, si $P_1(x) = 5 + 3x + 6x^2 + 2x^3$, este polinomio puede ser representado computacionalmente con el siguiente *array*:

```
int[] p1 = { 5, 3, 6, 2 };
```

Note que por conveniencia, si escribe el polinomio empezando por el término de menor grado, el coeficiente de grado k coincide con el índice k del *array*. Es decir, en el caso anterior, el coeficiente de grado 2 (que multiplica a x^2) es 7, que está almacenado en el índice `p1[2]`.

De igual forma, si el polinomio tiene coeficientes iguales a cero, el valor del *array* en el índice correspondiente será `0`. Por ejemplo, el polinomio $P(x) = x + 2x^3$ se representa en C# con el *array*:

```
int[] p = { 0, 1, 0, 2 };
```

Dada esta representación, se quiere implementar un método para encontrar la k-ésima derivada de un polinomio. Sean $P(x)$ un polinomio de grado n. La primera derivada es un polinomio $R(x)$ de grado $n - 1$ es un polinomio con la siguiente forma:

$$R(x) = p_1 + 2 \cdot p_2 x + 3p_3 \cdot x^2 + \cdots + n \cdot p_n x^{n-1}$$

Es decir, el coeficiente independiente se elimina. En cada término restante, su coeficiente se multiplica por el grado correspondiente de ese término y el grado se disminuye en 1.

Por ejemplo, la primera derivada del polinomio $P_1(x)$ del ejemplo original sería el polinomio $R_1(x) = 1 \cdot 3x^{1-1} + 2 \cdot 6x^{2-1} + 3 \cdot 2x^{3-1}$ que nos deja el resultado $R(x) = 3 + 12x + 6x^2$, que computacionalmente se representa en el *array* `{ 3, 12, 6 };`

Este proceso de derivación se puede realizar tantas veces según el orden de la derivada que se desea calcular. Por ejemplo, si lo que se quiere es obtener la segunda derivada del polinomio $P_1(x)$ se obtiene derivando nuevamente el polinomio resultante de la primera derivada $R_1(x)$, lo que daría como resultado final el $R_2(x) = 12 + 12x$ representado por el *array* `{12, 12}`.

Cuando el grado del polinomio es cero (es decir, solamente el término independiente) la derivada resulta en un polinomio de grado y valor cero, que se representa computacionalmente como un *array* de longitud `1`, y valor `0` en su primer (y único) índice.

```
int[] derivadaFinal = { 0 };
```

A partir de este punto, aplicar la derivada devuelve el mismo resultado. En ningún caso aplicando derivadas es posible obtener un *array* de longitud `0`. Por ejemplo, si continuamos derivando el polinomio anterior, obtendremos $R_3(x) = 12$, y $R_4(x) = 0$. A partir de este punto, $R_5(x)$ y todos las derivadas restantes son iguales a 0.

Usted debe implementar el siguiente método:

```
public static int[] Derivar(int[] p, int k)
{
    // Borre la siguiente línea y escriba su código aquí
    throw new NotImplementedException();
}
```

Este método devuelve el polinomio resultante de aplicar la derivada k veces, en la misma forma que el polinomio de entrada (es decir, el coeficiente de grado i almacenado en el i-ésimo índice del *array*). Por ejemplo, para el polinomio `p1` visto anteriormente el resultado sería:

```
int[] r2 = Polinomios.Derivar(p1, 2); // { 12, 12 }
```

Note que si el valor de `k` es mayor que el grado del polinomio, el resultado será el *array* `{ 0 }`.

```
int[] r6 = Polinomios.Derivar(p1, 6); // { 0 }
```

Note que si el polinomio p tiene coeficientes iguales a 0, es posible que polinomio resultante de derivar tenga también polinomios con coeficientes iguales a 0. Por ejemplo, si el polinomio $3 + 6x^2 + 2x^4 + 7x^6$ (representado por el *array* `{ 3, 0, 6, 0, 2, 0, 7 }`) se deriva 4 veces, se obtiene como resultado el polinomio $48 + 2520x^2$, que se representa en el array `{ 48, 0, 2520 }`.

Se garantiza que el coeficiente asociado al término de mayor grado nunca será 0.

Usted debe haber recibido junto a este documento una solución de *Visual Studio* con dos proyectos: una biblioteca de clases y aplicación de consola. En la biblioteca de clases encontrará la siguiente definición:

```
namespace Weboo.Examen.DerivandoPolinomios
{
    public class Polinomios
    {
        public static int[] Derivar(int[] p, int k)
        {
            // Borre la siguiente línea y escriba su código aquí
            throw new NotImplementedException();
        }
    }
}
```

Usted debe implementar este método, adicionando todos los métodos que considere necesarios para su solución.

NOTA: Todo el código de solución debe estar en este proyecto (**biblioteca de clases**), pues es el único código que será evaluado. Usted puede adicionar todo el código que considere necesario, pero no puede cambiar los nombres del *namespace*, clase o método mostrados. De lo contrario, el probador automático fallará.

La aplicación de consola se brinda para su conveniencia con algunos ejemplos de prueba, y un método utilitario que imprime en consola un *array* de tipo `int[]`, en forma de polinomio, para facilitarle la verificación.

NOTA: Los casos de prueba que aparecen en este proyecto son solamente de ejemplo. Que usted obtenga resultados correctos con estos casos no es garantía de que su solución sea correcta y de buenos resultados con otros ejemplos. De modo que usted debe probar con todos los casos que considere convenientes para comprobar la validez de su implementación.

Usted puede asumir las siguientes condiciones en la solución de su ejercicio:

- El parámetro `p` nunca será `null`.
- El *array* `p` siempre tendrá longitud mayor o igual que 1, es decir nunca será un array vacío (de longitud cero). Eso no es lo mismo que el polinomio cero, que estaría representado por el array `{ 0 }`.
- El parámetro `k` siempre será mayor o igual que `0`.
- El índice asociado al coeficiente de mayor grado (el índice final del *array* `p`) nunca tendrá valor `0`, excepto en el caso que sea exactamente el polinomio cero (`{ 0 }`).
- Cualquier otro índice, excepto el último, puede tener valor `0`.