Ejercicios de C#

Tipos valor, referencias, paso y devolución de referencias:

- Crear la clase Point(x, y) con las propiedades de las coordenadas y el método Distancia (del origen: $\sqrt{x^2 + y^2}$).
- Crear la clase Line(Point1, Point2) con:
    a. Las propiedades de los puntos.
    b. Los métodos deltaX y deltaY: valor absoluto de la diferencia de las coordenadas de los puntos.
- Para las referencias:

    A = “string”;
    B = “string”;
    C = A;

  cual es el resultado de:

    A == B
    A.Equals(B)
    A == C
    A.Equals(C)
    B == C
    B.Equals(C)

- En el main(), para las referencias:

    A = new Point(3,5);
    B = new Point(3,5);
    C = A;

  cual es el resultado de:

    A == B
    A.Equals(B)
    A == C
    A.Equals(C)
    B == C
    B.Equals(C)

- En el Main(), para el código:

    L = new Line(new Point(3, 5), new Point(3, 5));
    P = L.P1;
    Ant = L.deltaX();
    P.X = 10;
    Pos = L.deltaX();

  ¿Es Ant igual Pos?

- En la clase principal (Main()), crear el siguiente método:

    void add(Point p) { p.X = p.X + 1); }

  probar el siguiente codigo:

    L = new Line(new Point(3,5), new Point(3,5));
    P = L.P1;

```
Ant = L.deltaX();
Add(P);
Pos = L.deltaX();
```

¿Es Ant igual Posterior?

- Sobrescribir los métodos Equals y Clone de las clases Point y Line. Resolver y volver a probar los tres casos anteriores.

Aplicaciones de Consola:

Adivina el Número, generar un número entre el 0 y el 100, introducir un número e informar si es igual, mayor o menor. Hay un máximo de 10 intentos para encontrar el número que sea igual.

Decodificar las cadenas con el siguiente formato:

3+4+3,4-7*1=

en los siguientes componentes:

3 +
4 +
3,4 -
7 *
1 =

mostrando el resultado en la consola.

Crear la clase **Calculadora** que acumule y permita obtener los resultados parciales de las operaciones obtenidas en la decodificación anterior.

Crear la clase de utilidad **Valid** que permita la validación de cadenas. Todos los métodos deben ser de clase, recibir como primer parámetro la cadena a validar y devolver **true** es válido. Debe contar al menos con los siguientes métodos:

public static boolean isEmpty(String value);
public static boolean isNumeric(String value);
public static boolean isLenMax(String value, int len); //Inferior a la longitud máxima.
public static boolean isLenMin(String value, int len); //Superior a la longitud mínima.
public static boolean isPositive(String value);
public static boolean isPositive(double value);

adicionales:

public static boolean isEMail(String value);
public static boolean isURL(String value);

Crear la jerarquía de clases de figuras: triángulo, rectángulo, cuadrado, círculo, … La clase figura es abstracta con la propiedad color y el método área.

- triángulo: A = b * h / 2
- rectángulo: A = a * b
- cuadrado: A = l^2
- círculo: A= π * r^2

Crear una colección de figuras, con al menos 10, y calcular el área total de las figuras contenidas en la colección.

Crear el interfaz IGrafico con el método Pintate (imprime un literal que representa el objeto). Implementar el interfaz en las clases Figura, Linea y Punto. Crear una colección de objetos Gráficos, llenarlo con una variedad de objetos y pintar todos los elementos de la colección

Ficheros:

Leer un fichero de entrada y generar un fichero de salida, realizando los cálculos necesarios, con los siguientes formatos:

FileIN: Entrada.txt

3+4+3,4-7*1=

FileOUT: Salida.txt

3
+4
+3,4
-7
*1

3,400000

Ampliaciones:

- Fichero de entrada con múltiples líneas.
- Fichero de salida comprimido
- Descompresor del fichero de salida: zip → txt
- Compresor del fichero de entrada: txt → zip
- Procesar entrada/salida comprimida