**EJERCICIO #1** (4 puntos)

*Enunciado*.- Escribir el pseudocódigo de un programa que calcule ***por iteración*** la función exponencial $e^x$ considerando suficiente precisión ( X < 1000) la suma de 11 términos de la serie

$$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots\cdots + \frac{x^{10}}{10!}$$

siendo n! = 1·2·3 ····· (n-1) · n

**Nota**.- Escribir un fichero texto (no con Eclipse)

*Planteamiento previo*.- La función que nos piden puede tomar valores de entrada enteros o reales (double) e igualmente su resultado. Esto sólo cambiará los atributos (int o double) aplicables pero, para el pseudocódigo, tal cuestión es prácticamente irrelevante. La solución por iteración (resolución por pasos sucesivos) puede implementarse con bucles en cada sumando o bien aprovechar los resultados de sumandos previos (solución más eficiente). A su vez, cada sumando puede implementar un cálculo recursivo o un bucle:

a) Sin recursividad.- for (índice = 1 hasta el número del sumando (entre 1 y 10) …)
b) Con recursividad.- Viendo que tanto el numerador como el denominador son iterables:
   1. $X^n = X * X^{(n-1)}$
   2. $N! = N * (N-1)!$

Por ejemplo, la función factorial (n!) puede codificarse (con recursividad) así:

```
private int factorial (int num) {          // en este caso, voy a calcular factoriales de números enteros (sólo)
        Si num = 0, su factorial es 1  // (por definición y como límite de la recursividad
        Si no (else)  aux = num *  factorial (num – 1)   // he aquí la recursividad derivada de la definición
        return aux                              // aux almacena el resultado de la función en cada iteración (hasta 0)
```

*Solución.-* Así que podemos calcular cada sumando con una iteración FOR desde 1 hasta 10 e ir acumulando el valor de cada sumando. El primer sumando ("1") lo agregamos al final. A su vez, podemos guardar los resultados parciales (de numerador y divisor) para aplicarlos al cálculo del siguiente sumando:

```
double exponencial (double x) {
    double pot=1, fact=1, result = 0;     // estas son variables de cálculo parcial en cada paso
    for (int i = 1; i<=10; i++)  {        // bucle de cálculo de los 11 sumandos con "i" como índice
        pot = pot * x;                    // el numerador en cada sumando
        fact = fact * i;                  // el denominador en cada sumando
        result = result + pot/fact;}      // actualizamos el valor de resultado añadiendo un sumando más
    return (result +1);}                  // el valor de la función es la suma obtenida en los 10 pasos  más "1"
```

*Discusión adicional*.- El uso de algoritmos recursivos en cada sumando para calcular los valores de n! y $x^n$ no sería más eficiente pues habría que repetirlos en cada paso (sumando). Por ejemplo,

fact (i) para calcular el numerador en cada paso (para cada "i", donde i va de 1 hasta 10 igualmente):

{float aux=1;  // defino la variable auxiliar que dará el valor del numerador

Si  i == 0, se para la recursividad poniendo aux  a "1":  aux =1;

Si i > 0, se calcula fact (i) por recursividad: aux = i * fact(i-1);

return aux;}  se halla así el valor de fact (i) en cada sumando.

Igualmente se haría para el numerador con la función potencia n-síma de X.

Es decir, en este caso la recursividad aplicada individualmente no es más eficiente que la iteración sobre valores parciales, que finalmente viene a dar una recursividad implícita pues cada valor de fact y de pot utiliza el valor calculado en el paso anterior.

**Nota**.- En cualquier caso, una solución con algoritmos recursivos en cada sumando también sería aceptada

**EJERCICIO #2** (6 puntos)

*Enunciado*.- Escribir un programa JAVA con su main, clases (ver párrafo siguiente) y métodos para ordenar un conjunto de países atendiendo a distintos atributos (podría ser un menú en el main): Ordenar por nombre, ordenar por capital, ordenar por superficie, etc. (2,5 puntos)

Declarar las clases necesarias para tratar objetos país con los siguientes atributos: nombre, continente, superficie, capital, población. Explicar el tipo de variable elegido (poner comentarios en el código). (1,0 pt.)

Codificar los métodos constructores que se estime oportuno así como los de obtención de los atributos (get) de un país y su establecimiento (set). (0,5 puntos)

Instanciar 4 (objetos de) países distintos (con valores de superficie y población inventados). (0,5 puntos)

Añadir un método que devuelva la capital de un país cuando reciba como input el nombre de tal país (introducido por teclado) si dicho país es uno del 4 que tenemos codificado: capital (nombrepais) (1,5 puntos)

*Planteamiento previo*.- Para resolver el ejercicio conforme a los conocimientos de la 1ª evaluación, se puede hacer todo dentro de una clase y, si acaso, introducir "LeerTeclado.java", que también se dio en la 1ª evaluación para seleccionar la opción de ordenación (por nombre, por continente, por superficie, etc.). Así que el esquema podría ser el siguiente:

```
public class Pais  {
        private String nombre, continente, capital;
        private double superficie, poblacion;
        métodos de la clase Pais (constructores, get, set, orden1, orden2, etc.){
        }
        construcción ("a pelo") de 4 países con sus respectivos atributos{
        }
        public static void main(String[] args) {
                ejecución tipo menú para ordenar los países que tenemos según distintos parámetros
        }
}
```

*Solución.-* Así que la declaración de la clase Pais, con sus atriburtos sería:

```
public class Pais  {
      private String nombre, continente, capital;
      private double superficie, poblacion;   // deben ser tipo double porque pueden ser grandes
                                              //  cifras o expresarse en millones (con decimales).
```

Y sus constructores, a continuación, serán:

```
      public Pais () {
      }                                       //constructor "vacío" (sólo reserva memoria para el objeto)
      public Pais (String nombre, String continente, String capital, double superficie,
      double poblacion){                      //constructor con todos los atributos de un objeto Pais.
            this.nombre=nombre;
            this.continente=continente;
            this.capital=capital;
            this.superficie=superficie;
            this.poblacion=poblacion;
      }
```

Los métodos get y set pedidos se codifican como es habitual:

```
      public String getNombre() {
            return nombre;
      }
      public String getContinente() {
            return continente;
      }
```

```
	public String getCapital() {
		return capital;
	}
	public double getSuperficie() {
		return superficie;
	}
	public double getPoblacion() {
		return poblacion;
	}
	public void setNombre(String nombre) {
		this.nombre=nombre;
	}
	public void setContinente(String continente) {
		 this.continente=continente;
	}
	public void setCapital(String capital) {
		this.capital=capital;
	}
	public void setSuperficie(double superficie) {
		this.superficie=superficie;
	}
	public void setPoblacion(double poblacion) {
		this.poblacion=poblacion;
	}
```

La instanciación de los 4 países sería algo así como:

```
static Pais Alemania=new Pais("Alemania", "Europa", "Berlín", 357168, 81.3);
static Pais USA = new Pais ("Estados Unidos", "América", "Washington", 9371200, 316.0);
static Pais Egipto = new Pais ("Egipto", "África", "El Cairo", 1000000, 90.1);
static Pais China = new Pais ("China", "Asia", "Pekín", 9596961, 1370.5);
```

La cualificación de static garantiza su tratamiento posterior, en el bloque main, para ordenarlos.
Como no podemos operar con arrays ni en la 1ª evaluación se dieron aún algoritmos de ordenación, el apartado final se hará sobre los 4 paises introducidos y simplemente implementaremos toString en diferente formato para cada tipo de ordenación. Cualquier otra implementación más general y refinada, sería valorada adicionalmente. De momento, podría bastar con algo así:

```
	public String toStringPorNombre(Pais pais) {
		return pais.nombre + " está en " + pais.continente +", su capital es "+ pais.capital + " y su población, de " + pais.poblacion + " millones de habitantes viven en " + pais.superficie + " Km2.\n";
	}
	public String toStringPorCapital(Pais pais) {
		return "La capital de " + pais.nombre +" es "+ pais.capital + ", está en " + pais.continente + " y sus " + pais.poblacion + " millones de habitantes viven en " + pais.superficie + " Km2.\n";
	}
	public String toStringPorContinente(Pais pais) {
		return "En " + pais.continente +" se sitúa "+ pais.nombre + ", con una extensión de " + pais.superficie + " y una población de " + pais.poblacion + ". Su capital es " + pais.capital + ".\n";
	}
	public String toStringPorPoblacion(Pais pais) {
		return "Los " + pais.poblacion +" habitantes de "+ pais.nombre + " viven en " + pais.superficie + ". Es un país situado en el continente " + pais.continente + " cuya capital es " + pais.capital + ".\n";
	}
	public String toStringPorSuperficie(Pais pais) {
		return pais.superficie + " Km2 es la superficie de " + pais.nombre +" en "+ pais.continente + ". Su capital es " + pais.capital + " y su población es " + pais.poblacion + " millones de habitantes.\n";
	}
```

Ya en el bloque main, podemos meter un **menú** para solicitar el criterio de ordenación y presentar a los 4 países con el correspondiente toString:

```
int opcion;

      do {
            System.out.println ("\nMENU para ordenar paises:");
            System.out.println ("1 - Ordenar por nombre");
            System.out.println ("2 - Ordenar por capital");
            System.out.println ("3 - Ordenar por superficie");
            System.out.println ("4 - Ordenar por población");
            System.out.println ("5 - Ordenar por continente");
            do {
                  System.out.println("Introduce opcion (1-6)");
                  opcion = LeerTeclado.readInteger();
            } while (opcion < 1 || opcion > 5);
```

o directamente sacar los listados ordenados secuencialmente por nombre, capital, superficie, capital y continente con un texto anunciador:

```
System.out.println ("Paises ordenados por nombre:");
System.out.println(Alemania.toStringPorNombre());
System.out.println(China.toStringPorNombre());
System.out.println(Egipto.toStringPorNombre());
System.out.println(USA.toStringPorNombre());
```

Y luego

```
System.out.println ("\n Paises ordenados por capital");
System.out.println(Alemania.toStringPorCapital());
System.out.println(Egipto.toStringPorCapital());
System.out.println(China.toStringPorCapital());
System.out.println(USA.toStringPorCapital());
```

etc.

Introducir el menú añade valor a la solución implementada al ejercicio pero no es lo más relevante del mismo.

En la opción de utilizar el menú, habría que completarlo con una sentencia case (respecto al valor de la variable opcion) tal que así:

```
switch (opcion){
                  case 1: {
                        System.out.println(Alemania.toStringPorNombre());
                        System.out.println(China.toStringPorNombre());
                        System.out.println(Egipto.toStringPorNombre());
                        System.out.println(USA.toStringPorNombre());
                  } break;
                  case 2: {
                        System.out.println(Alemania.toStringPorCapital());
                        System.out.println(Egipto.toStringPorCapital());
                        System.out.println(China.toStringPorCapital());
                        System.out.println(USA.toStringPorCapital());
                  } break;
                  case 3: {
                        System.out.println(China.toStringPorSuperficie());
                        System.out.println(USA.toStringPorSuperficie());
                        System.out.println(Egipto.toStringPorSuperficie());
                        System.out.println(Alemania.toStringPorSuperficie());
                  } break;
                  case 4: {
                        System.out.println(China.toStringPorPoblacion());
                        System.out.println(USA.toStringPorPoblacion());
                        System.out.println(Egipto.toStringPorPoblacion());
                        System.out.println(Alemania.toStringPorPoblacion());
                  } break;
```

```
            case 5: {
                  System.out.println(Egipto.toStringPorContinente());
                  System.out.println(USA.toStringPorContinente());
                  System.out.println(China.toStringPorContinente());
                  System.out.println(Alemania.toStringPorContinente());
            } break;
      }
```

En definitiva, tampoco los formateos de los textos toString (porNombre, porContinente, etc.) son lo más importante y valorado en el ejercicio pero sí la construcción de la clase Pais, con los atributos apropiados y los métodos indicados en el enunciado (no necesariamente uno de ordenación) y la instanciación de objetos de dicha clase.