Estructura de Datos y Algoritmos
Unit 4. Algoritmos II. Recursión
Conjunto de problemas.

Problema 1 - Escribir un método recursivo que devuelve la suma de los N primeros enteros.

Problema 2 – Escribir un método recursivo que tome dos enteros, x y n (podemos suponer que n> = 0) y devuelve x power n.

Problema 3 – Escribir un método recursivo que tome un número entero y devuelva el número de sus dígitos (por ejemplo, 2356 tiene 4 dígitos). **Pista**: 2356/10 = 235 , 235/10=23 y 23/10= 2 y no puedo seguir dividiendo entre 10

Problema 4 – Escribir un método recursivo que toma un array de enteros y devuelve la suma de sus elementos.

Problema 5 – Escribir un método recursivo que tome un array de enteros y compruebe si el array está ordenado (orden ascendente).

Problema 6 – Escribir un método recursivo que toma un array de enteros y devuelve el elemento más pequeño.

Problema 7 – Escribir un método recursivo que toma un array de enteros y un entero y devuelve el índice de este entero en el array. Si el entero no existe en el array, el método devuelve -1.

Problema 8 – Escribir un método recursivo que toma dos enteros y devuelve su división entera sin el operador / **Pista:** puede usar el operador - en la llamada recursiva.

Problema 9 – Escribir un método recursivo que tome dos enteros x y y, tal que devuelva x veces y utilizando el método ruso. Este método ruso consiste en:

1) Haz dos columnas. Escribir el número más grande en la primera columna y el más pequeño en la segunda.
2) En la primera columna, dividir el número por 2 repetidamente hasta llegar a 1. En la segunda columna, multiplicar el número por 2 hasta que tenga las mismas filas que en la primera columna.
3) Tachar las filas cuyo valor en la primera columna sea un número par (x% 2 == 0)

146	37
73	74
~~36~~	~~148~~
~~18~~	~~296~~
9	592
~~4~~	~~1184~~
~~2~~	~~2368~~
1	4736

4) Finalmente, sumar los números no tachados en la segunda columna.
   Resultado = 74+592+4736 = 5402

Problema 10: Escriba un método recursivo que tome un entero positivo n y devuelva la suma de enteros positivos de 2 a n.

Problema 10:
Escribir un método recursivo que tome dos parámetros n y k, y devuelva el valor del *Coeficiente binomial C (n, k).* El coeficiente binomial es el número de k-combinaciones de un conjunto de n elementos. En otras palabras, el número total de subconjuntos de k elementos escogidos de un conjunto con n elementos.

C(n,k) = n! / (k!*(n-k)!) where n!= n *(n-1)*(n-2)*…*2*1

Problema 11 – Escribir un método recursivo que ordene enteros en orden ascendente. **Pista:** almacena el elemento más pequeño en la primera posición del array, y luego ordena el resto del array mediante una llamada recursiva.

Problema 12 – Escribir un método recursivo que toma un array e invierte sus elementos.

Problema 13 – Escribir un método recursivo que toma un número entero y devuelve una cadena que contiene la representación binaria del número.

Problema 14
¿Qué devuelve este método para diferentes valores de x?

```
static int f(int x){
        if (x >100) return x- 10;
        else  return(f(f(x+11)));
}
```

Problema 15 – Escribir un método recursivo que tome una cadena y verifica si es una palabra capicúa (por ejemplo, *level*, *did*, *rotator*).

Problema 16 – Implementar el algoritmo de búsqueda binaria usando recursión. Escriba un método recursivo que tome un array ordenado y un entero y devuelva la posición de este número en el array.

Problema 17 – Escribir un método recursivo que toma una cadena y devuelve su longitud (sin utilizar la propiedad de la longitud). Se puede usar la subcadena del método.

Problema 18 – Escribir un método recursivo que toma una cadena y la imprime. **Pista:** debes imprimir cada carácter.

Problema 19- ¿Qué significa este método?

```
public static int mystery (int n) {
```

```
        if (n == 0)  return 0;
        else   return n + mystery(n-1);
}
```

Problem 20. Una permutación de un conjunto de objetos es un orden particular de esos objetos. Cuando hablamos de "todas las permutaciones" de un conjunto, nos referimos a todas las formas posibles de ordenar esos objetos. Ejemplos:

- Dado el conjunto vacío {}, la única permutación posible es {}
- Dado el conjunto {A}, la única permutación posible es {A}
- Dado el conjunto {A, B}, las posibles permutaciones son {AB, BA}
- Dado el conjunto {A, B, C}, las posibles permutaciones son
- {ABC, ACB, BAC, BCA, CAB, CBA}

Escribir un método que tome un array de caracteres y encuentre todas sus permutaciones.
Pista: en el caso general, encontrar todas las permutaciones de n-1 de esos objetos. Luego, inserte el objeto restante en todas las posiciones posibles de cada permutación de n-1 objetos

Problema 21 – Escribir un método que implemente la función de Ackerman. La función de Ackermann se puede definir de la siguiente manera:

- A(0, y) = y + 1
- A(x, 0) = A(x - 1, 1)
- A(x, y) = A(x - 1, A(x, y - 1))

Esta función tarda mucho tiempo en computarse (¿por qué?)

Problema 22. Escribir un método recursivo que toma una cadena y devuelve una nueva cadena donde todos los caracteres x en minúscula han sido reemplazados por caracteres y. No puede usar los métodos replaceAll o replace.

Problema 23. Escribe un método recursivo que toma un número entero y devuelve la suma de sus dígitos. Si usa x% 10, puede obtener su dígito más a la derecha (126% 10 es 6). x / 10 elimina este dígito (126/10 es 12).

Problema 24 – Escribir un método recursivo que toma una cadena y devuelve su inversa.

Problema 25 – Escribir un método recursivo que toma una cadena y elimina sus duplicados consecutivos. Por ejemplo, "aabccba" se transforma en "abcba".

Problema 26 – Escribir un método recursivo que tome dos enteros, x e y, y calcule x mod sin usar el operador %.

Problema 27. Qué es f(0)? f(0) = 997

```
public static int f(int x) {
        if (x >1000)            return x-4;
```

```
        else            return f(f(x+5));
}
```

Problema 28 – Escribir un método recursivo que toma dos cadenas ordenadas alfabéticamente y las concatena en una nueva cadena. El resultado también debe estar ordenado alfabéticamente.

Problema 29. Explicar la funcionalidad de las siguientes funciones:

a)
```
int fun1(int x, int y)
{
  if(x == 0)
    return y;
  else
    return fun1(x - 1,  x + y);
}
```

b)
```
void fun2(int arr[], int start_index, int end_index)
{
  if(start_index >= end_index)
     return;
  int min_index;
  int temp;

  /* Assume that minIndex() returns index of minimum value in
    array arr[start_index...end_index] */
  min_index = minIndex(arr, start_index, end_index);

  temp = arr[start_index];
  arr[start_index] = arr[min_index];
  arr[min_index] = temp;

  fun2(arr, start_index + 1, end_index);
}
```

Problema 30. Escribir un método recursivo que compruebe si un número n es primo (hay que comprobar si n es divisible por cualquier número por debajo n).

Problema 31. Escribir un método recursivo que imprima un número entero con comas en los lugares correctos. Por ejemplo, 12345 se imprimirá como 12,345.

Problema 32. Escribir un método recursivo para invertir las palabras en una cadena, es decir, "cat is running" se convierte en "running is cat".

Problema 33. Una palabra se considera elfish si contiene las letras: e, l, y f en ella, en cualquier orden. Por ejemplo, diríamos que las siguientes palabras son elfish: whiteleaf, tasteful y waffles, porque cada uno contiene esas letras. Escribir un método recursivo llamado elfish? que, dada una palabra, se indique si esa palabra es elfish o no. Escribir una función de predicado más generalizada llamada x-ish? que, dadas dos palabras, devuelva verdadero si todas las letras de la primera palabra están contenidas en el segundo.

Problema 34. Escribir un método recursivo de Java que cuente el número de ocurrencias del carácter 'a' en una cadena. **Sugerencia**: una método signature que funciona es public static int countA (String s). Considerar usar los métodos charAt o startsWith en String.