Actividad 4.

Se quieren implementar nuevas funcionalidades:

a) `HashMap<String, Double> pageRank()`
`// Post: el resultado es el valor del algoritmo PageRank para cada web de la lista de webs`

El algoritmo Page Rank (https://en.wikipedia.org/wiki/PageRank) corresponde a una familia de algoritmos utilizados para asignar de forma numérica la relevancia de los documentos (o páginas web) de un grafo. El sistema PageRank es utilizado por el popular motor de búsqueda Google para determinar la importancia o relevancia de una página web. El algoritmo PageRank calcula la probabilidad de que una persona, accediendo al grafo y siguiendo enlaces, llegue a un nodo en particular. El cálculo de PageRank requiere varios pases, llamados iteraciones, para ajustar el valor aproximado. Inicialmente, todos los nodos reciben la misma probabilidad:

$\forall A \in nodos, PR(A) = \frac{1}{N}$ donde **N** es el número de nodos del grafo.

Después, a cada iteración, se recalcula el valor asociado a cada nodo mediante la fórmula:

$PR(A) = \frac{1-d}{N} + d * \sum_{i=1}^{n} \frac{PR(i)}{C(i)}$ donde:

- **PR(A)** es el PageRank de la página A.
- **d** (damping factor) es un factor de amortiguación que tiene un valor entre 0 y 1 (un valor típico es 0.85)
- **PR(i)** son los valores de PageRank que tienen cada una de las páginas **i** que enlazan a A.
- **C(i)** es el número total de enlaces salientes de la página i (sean o no hacia A).

El proceso continúa hasta que la suma de las diferencias en valor absoluto entre los valores de una iteración y la siguiente es menor a un umbral predeterminado (p. ej. 0.0001).

Como ejemplo, supongamos un grafo formado por cuatro nodos A, B, C y D. B tiene enlaces hacia C y A, C tiene un enlace hacia A, y el nodo D tiene enlaces de salida hacia A, B y C.

Inicialmente cada nodo tendrá un valor de 0.25. En la primera iteración, B transferirá la mitad de su valor, 0.125, a A y la otra mitad a C. C transferirá todo su valor, 0.25, a A. Como D tiene tres arcos salientes, transferirá un tercio de su valor (aproximadamente 0.083) a A, B y C. Después de esta primera iteración, A tendrá un PageRank aproximado de 0.427.

```
b) public class Par {

      String web;

      Double pageRank;

      }

   ArrayList<Par> buscar(String palabraClave)

   // Post: El resultado es una lista de pares (página web, pagerank) en
   las que aparece la palabra clave, ordenadas de mayor a menor por su
   pagerank, de manera que en las primeras posiciones aparecerán las
   páginas web com pagerank más alto

c) (opcional) ArrayList<Par> buscar(String palabraClave1,

                                       String palabraClave2)

   // Post: El resultado es una lista de las páginas web en las que
   aparecen las palabras clave, ordenadas de mayor a menor por su pagerank,
   de manera que en las primeras posiciones aparecerán las páginas web com
   pagerank más alto
```

Se deberá entregar (18-XII-2020, fecha límite entrega actividad 4):

- Programas que implementen lo pedido (ejecutados correctamente). **Se deberá demostrar que el programa funciona realmente con conjuntos de datos no triviales (es decir, procesando miles de líneas del fichero original).** La documentación deberá incluir:

- Ejemplos de ejecución de los métodos pedidos, indicando datos de prueba y resultados, junto con el número de datos usados y de pruebas realizadas. En caso de que sea relevante, también se indicará el tiempo de ejecución de cada prueba.

- Documentación describiendo el problema planteado, las alternativas examinadas, implementaciones, y eficiencia

NOTA: debido a que algunos de los resultados pedidos requerirán un alto tiempo de computación, esto exigirá que los algoritmos funcionen correctamente un tiempo antes de la fecha de entrega, ya que si no no se podrán obtener los resultados.

Además debéis rellenar y entregar el Checklist para verificar que habéis realizado todo lo que se os pide.

AYUDA: diferencia en cada iteración (tiempo de cada iteración aproximadamente 8 segundos)**.**

iteracion:	0	diff:	0.782973717764194
iteracion:	1	diff:	0.344276535594372
iteracion:	2	diff:	0.163840596038833
iteracion:	3	diff:	0.087185768116801
iteracion:	4	diff:	0.050037776206004
iteracion:	5	diff:	0.031404833276485
iteracion:	6	diff:	0.020827658897577
iteracion:	7	diff:	0.014681098383163
iteracion:	8	diff:	0.010699860422163
iteracion:	9	diff:	0.008041772386592
iteracion:	10	diff:	0.006132009166671
iteracion:	11	diff:	0.004749528325397
iteracion:	12	diff:	0.00370432722091
iteracion:	13	diff:	0.002914058287575
iteracion:	14	diff:	0.002300827614045
iteracion:	15	diff:	0.001826896845737
iteracion:	16	diff:	0.001454288501982
iteracion:	17	diff:	0.001163004906129
iteracion:	18	diff:	9.32E-04
iteracion:	19	diff:	7.50E-04
iteracion:	20	diff:	6.04E-04
iteracion:	21	diff:	4.88E-04
iteracion:	22	diff:	3.96E-04
iteracion:	23	diff:	3.21E-04
iteracion:	24	diff:	2.62E-04
iteracion:	25	diff:	2.14E-04
iteracion:	26	diff:	1.75E-04
iteracion:	27	diff:	1.43E-04
iteracion:	28	diff:	1.17E-04
iteracion:	29	diff:	9.67E-05

numeros de iteraciones totales: 30