	Carátula para entrega de prácticas
Facultad de Ingeniería	Laboratorio de docencia

Laboratorios de computación salas A y B

Profesor:	Jesus Cruz Navarro
Asignatura:	Estructura de Datos y Algoritmos II
Grupo:	Grupo 1
No de Práctica(s):	Práctica 1 – Algoritmos de ordenamiento. Parte I
Integrante(s):	Edwin Jaret Santiago Díaz
No. de Equipo de cómputo empleado:	22
No. de Lista o Brigada:	22
Semestre:	2022 - 2
Fecha de entrega:	14 febrero 2022
Observaciones:	

CALIFICACIÓN: _____

Algoritmos de ordenamiento. Parte I

Objetivos

1.- Implementar los algoritmos BubbleSort, BubbleSort Optimizado y MergeSort para ordenar colecciones de datos.
2.- Comparar los tiempos de ejecución de los algoritmos antes mencionados.

Desarrollo

1) Se implementaron los algoritmos **BubbleSort**, **BubbleSortOptimizado** y **MergeSort**, cada algoritmo tiene su propia función. Los tres algoritmos reciben como parámetro un arreglo de números enteros, sin embargo, el tercer algoritmo recibe otros dos parámetros los cuales son desde donde inicia el arreglo (la posición) y la longitud del arreglo.

2) Se ejecutó el programa para utilizar los tres algoritmos, cada algoritmo recibe el mismo arreglo de números. Lo que se quiere comprobar son los tiempos que toman cada algoritmo al ordenar de menor a mayor el mismo arreglo de números. Estos fueron los resultados.

```
arreglo1 = [3, 5, 0, 9, 7, 4, 1, 6, 8, 2]
arreglo2 = [3, 5, 0, 9, 7, 4, 1, 6, 8, 2]
arreglo3 = [3, 5, 0, 9, 7, 4, 1, 6, 8, 2]
```

```
        ¡Practica 1!
        Ejercicio 2

La lista de numeros a ordenar es: [3, 5, 0, 9, 7, 4, 1, 6, 8, 2]

Arreglo ordenado:  [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
El tiempo que le tomó al algoritmo BubbleSort es de: 0.000016900

Arreglo ordenado:  [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
El tiempo que le tomó al algoritmo BubbleSortOptimizado es de: 0.000015600

Arreglo ordenado:  [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
El tiempo que le tomó al algoritmo MergeSort es de: 0.000081600
```

3) Ahora, se ponen a prueba los tres algoritmos pasándole tres distintos tipos de arreglos para poder observar el tiempo que le toman a cada algoritmo cuando tiene que ordenar una lista ordenada ascendente (mejor de los casos), una lista ordenada descendente (peor de los casos) y una lista aleatoria (caso promedio), el tamaño de la lista puede variar entre 500, 1000, 5000, 10000 y 20000 datos con valores entre -100 y 100. Además, cada ejecución se repitió 3 veces porque el tiempo de ejecución no siempre es el mismo y así poder obtener un promedio del tiempo ejecutado.

   Estos fueron los resultados:

500	Tiempo promedio.
¡Caso Promedio! Tiempo BubbleSort : 0.018547500003 Tiempo BubbleSortOptimizado : 0.017708600004 Tiempo MergeSort : 0.001453899997 ¡Caso Promedio! Tiempo BubbleSort : 0.033832700006 Tiempo BubbleSortOptimizado : 0.033500900012 Tiempo MergeSort : 0.003078099995	**Caso promedio:** BubbleSort: 0.02883213 [s] BubbleSortOptimizado: 0.028747833 [s] MergeSort: 0.0024906 [s]

¡Caso Promedio! Tiempo BubbleSort : 0.034116299998 Tiempo BubbleSortOptimizado : 0.035034000000 Tiempo MergeSort : 0.002923500011 ¡Mejor de los casos! Tiempo BubbleSort : 0.013687600003 Tiempo BubbleSortOptimizado : 0.000043299995 Tiempo MergeSort : 0.001277000003 ¡Mejor de los casos! Tiempo BubbleSort : 0.024377900001 Tiempo BubbleSortOptimizado : 0.000092100003 Tiempo MergeSort : 0.003129999997 ¡Mejor de los casos! Tiempo BubbleSort : 0.030602700004 Tiempo BubbleSortOptimizado : 0.000106699998 Tiempo MergeSort : 0.004398799996 ¡Peor de los casos! Tiempo BubbleSort : 0.026001599996 Tiempo BubbleSortOptimizado : 0.024141199989 Tiempo MergeSort : 0.001143300004 ¡Peor de los casos! Tiempo BubbleSort : 0.058624099998 Tiempo BubbleSortOptimizado : 0.034190700011 Tiempo MergeSort : 0.001560200006 ¡Peor de los casos! Tiempo BubbleSort : 0.065498199998 Tiempo BubbleSortOptimizado : 0.022239300000 Tiempo MergeSort : 0.000992000001	**Mejor de los casos:** BubbleSort: 0.02288940 [s] BubbleSortOptimizado: 0.00008069 [s] MergeSort: 0.00293526 [s] **Peor de los casos:** BubbleSort: 0.05004129 [s] BubbleSortOptimizado: 0.02685706 [s] MergeSort: 0.00123183 [s]
1000	
¡Caso Promedio! Tiempo BubbleSort : 0.066472899998 Tiempo BubbleSortOptimizado : 0.066271899996 Tiempo MergeSort : 0.002801700000 ¡Caso Promedio! Tiempo BubbleSort : 0.084965599992 Tiempo BubbleSortOptimizado : 0.075077200003 Tiempo MergeSort : 0.003096400003 ¡Caso Promedio! Tiempo BubbleSort : 0.076341000007 Tiempo BubbleSortOptimizado : 0.077323200007 Tiempo MergeSort : 0.003005500010 ¡Mejor de los casos! Tiempo BubbleSort : 0.043762899993 Tiempo BubbleSortOptimizado : 0.000099299999 Tiempo MergeSort : 0.003060799994	**Caso promedio:** BubbleSort: 0.07592649 [s] BubbleSortOptimizado: 0.07289067 [s] MergeSort: [0.00296786 [s]

<table>
<tr>
<td>

```
¡Mejor de los casos!
Tiempo BubbleSort : 0.045834399993
Tiempo BubbleSortOptimizado : 0.000086000000
Tiempo MergeSort : 0.002796400004
¡Mejor de los casos!
Tiempo BubbleSort : 0.044978499995
Tiempo BubbleSortOptimizado : 0.000086399989
Tiempo MergeSort : 0.003699600013
```

```
¡Peor de los casos!
Tiempo BubbleSort : 0.102599799997
Tiempo BubbleSortOptimizado : 0.103660900000
Tiempo MergeSort : 0.003001900011
¡Peor de los casos!
Tiempo BubbleSort : 0.103523099999
Tiempo BubbleSortOptimizado : 0.104931999987
Tiempo MergeSort : 0.002552000005
¡Peor de los casos!
Tiempo BubbleSort : 0.102922200007
Tiempo BubbleSortOptimizado : 0.101385799993
Tiempo MergeSort : 0.002856299994
```

</td>
<td>

Mejor de los casos:

BubbleSort: 3.918755549 [s]
BubbleSortOptimizado: 0.00089566 [s]
MergeSort: 0.03164349 [s]

Peor de los casos:

BubbleSort: 0.10301502 [s]
BubbleSortOptimizado: 0.10332622 [s]
MergeSort: 0.00280302 [s]

</td>
</tr>
<tr>
<td>

5000

</td>
<td></td>
</tr>
<tr>
<td>

```
¡Caso Promedio!
Tiempo BubbleSort : 1.982600800009
Tiempo BubbleSortOptimizado : 1.899931900000
Tiempo MergeSort : 0.019199599992
¡Caso Promedio!
Tiempo BubbleSort : 1.412711199999
Tiempo BubbleSortOptimizado : 1.400387999995
Tiempo MergeSort : 0.013083099999
¡Caso Promedio!
Tiempo BubbleSort : 1.371252400000
Tiempo BubbleSortOptimizado : 1.738433299994
Tiempo MergeSort : 0.013073800001
```

```
¡Mejor de los casos!
Tiempo BubbleSort : 0.823455399994
Tiempo BubbleSortOptimizado : 0.000287499992
Tiempo MergeSort : 0.011099399999
¡Mejor de los casos!
Tiempo BubbleSort : 1.177789699999
Tiempo BubbleSortOptimizado : 0.000395199997
Tiempo MergeSort : 0.015374800001
¡Mejor de los casos!
Tiempo BubbleSort : 0.967722899994
Tiempo BubbleSortOptimizado : 0.000741399999
Tiempo MergeSort : 0.019058000005
```

</td>
<td>

Caso promedio:

BubbleSort: 1.49885477 [s]
BubbleSortOptimizado: 0.07289076 [s]
MergeSort: 0.01511882 [s]

Mejor de los casos:

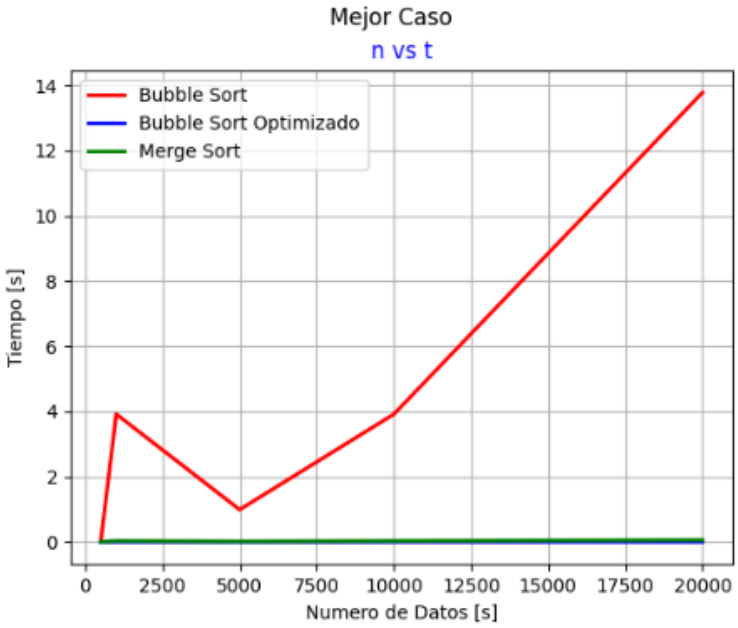
BubbleSort: 0.9865599 [s]
BubbleSortOptimizado: 0.00047469 [s]
MergeSort: 0.01517739 [s]

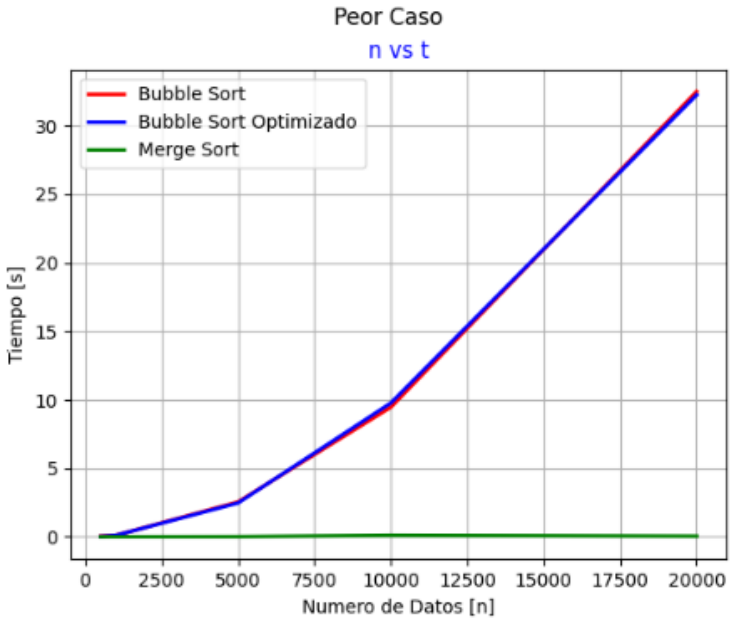
</td>
</tr>
</table>

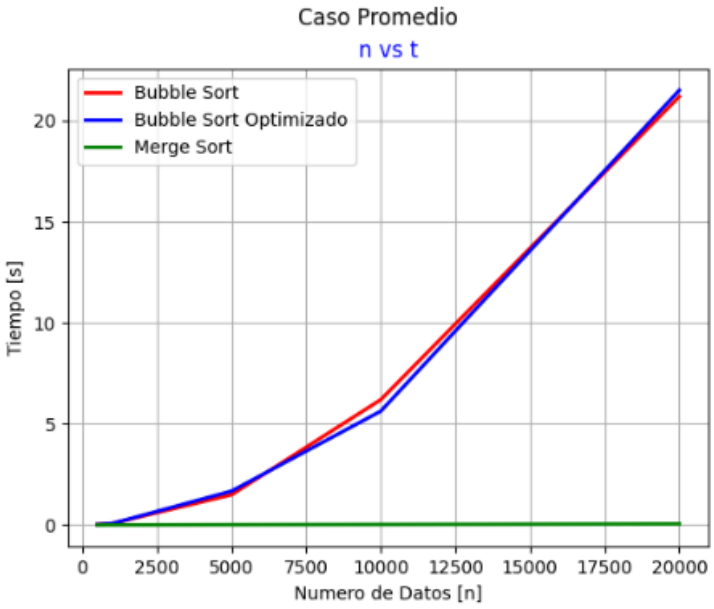
¡Peor de los casos! Tiempo BubbleSort : 2.740119299997 Tiempo BubbleSortOptimizado : 2.595650799994 Tiempo MergeSort : 0.014620299989 ¡Peor de los casos! Tiempo BubbleSort : 2.719179199994 Tiempo BubbleSortOptimizado : 2.731345299995 Tiempo MergeSort : 0.013459399997 ¡Peor de los casos! Tiempo BubbleSort : 2.177311599997 Tiempo BubbleSortOptimizado : 2.085347400003 Tiempo MergeSort : 0.010981299987	Peor de los casos: BubbleSort: 2.54553669 [s] BubbleSortOptimizado: 2.47078116 [s] MergeSort: 0.01302032 [s]
10,000	
¡Caso Promedio! Tiempo BubbleSort : 6.040382300009 Tiempo BubbleSortOptimizado : 5.864024700000 Tiempo MergeSort : 0.031309800004 ¡Caso Promedio! Tiempo BubbleSort : 6.912402499991 Tiempo BubbleSortOptimizado : 5.679760100000 Tiempo MergeSort : 0.026533199998 ¡Caso Promedio! Tiempo BubbleSort : 5.653707999998 Tiempo BubbleSortOptimizado : 5.358989999993 Tiempo MergeSort : 0.025586799995	**Caso promedio:** BubbleSort: 6.20216426 [s] BubbleSortOptimizado: 5.6342836 [s] MergeSort: 0.02780992 [s]
¡Mejor de los casos! Tiempo BubbleSort : 4.093035099999 Tiempo BubbleSortOptimizado : 0.001405999996 Tiempo MergeSort : 0.032458899994 ¡Mejor de los casos! Tiempo BubbleSort : 3.434989900008 Tiempo BubbleSortOptimizado : 0.000612600008 Tiempo MergeSort : 0.028677700000 ¡Mejor de los casos! Tiempo BubbleSort : 4.228241500008 Tiempo BubbleSortOptimizado : 0.000668400011 Tiempo MergeSort : 0.033793900002	**Mejor de los casos:** BubbleSort: 3.91875549 [s] BubbleSortOptimizado: 0.00089566 [s] MergeSort: 0.0316439 [s]
¡Peor de los casos! Tiempo BubbleSort : 8.767702800003 Tiempo BubbleSortOptimizado : 8.496002100001 Tiempo MergeSort : 0.029080099994 ¡Peor de los casos! Tiempo BubbleSort : 9.977699000010 Tiempo BubbleSortOptimizado : 10.375059200000 Tiempo MergeSort : 0.040629400013	**Peor de los casos:** BubbleSort: 9.46475733 [s] BubbleSortOptimizado: 9.77235423 [s] MergeSort: 0.105338272 [s]

¡Peor de los casos! Tiempo BubbleSort : 9.648870199992 Tiempo BubbleSortOptimizado : 10.446001399992 Tiempo MergeSort : 0.246438800008	
20,000	
¡Caso Promedio! Tiempo BubbleSort : 21.267218800000 Tiempo BubbleSortOptimizado : 21.443891899995 Tiempo MergeSort : 0.055362300001 ¡Caso Promedio! Tiempo BubbleSort : 21.232189100003 Tiempo BubbleSortOptimizado : 21.566688499996 Tiempo MergeSort : 0.054928199999 ¡Caso Promedio! Tiempo BubbleSort : 21.081704600001 Tiempo BubbleSortOptimizado : 21.509504300004 Tiempo MergeSort : 0.055468499995	**Caso promedio:** BubbleSort: 21.19370417 [s] BubbleSortOptimizado: 21.50669489 [s] MergeSort: 0.05525299 [s]
¡Mejor de los casos! Tiempo BubbleSort : 14.042525299999 Tiempo BubbleSortOptimizado : 0.001298300005 Tiempo MergeSort : 0.048941499990 ¡Mejor de los casos! Tiempo BubbleSort : 13.741682499996 Tiempo BubbleSortOptimizado : 0.001225500004 Tiempo MergeSort : 0.062931900000 ¡Mejor de los casos! Tiempo BubbleSort : 13.560694500004 Tiempo BubbleSortOptimizado : 0.001287300009 Tiempo MergeSort : 0.063042499998	**Mejor de los casos:** BubbleSort: 13.78163409 [s] BubbleSortOptimizado: 0.00127036 [s] MergeSort: 0.05830529 [s]
¡Peor de los casos! Tiempo BubbleSort : 33.202719599998 Tiempo BubbleSortOptimizado : 31.084818200005 Tiempo MergeSort : 0.046084700007 ¡Peor de los casos! Tiempo BubbleSort : 31.385569299993 Tiempo BubbleSortOptimizado : 34.208612200004 Tiempo MergeSort : 0.045891199989 ¡Peor de los casos! Tiempo BubbleSort : 32.921547200007 Tiempo BubbleSortOptimizado : 31.471598800010 Tiempo MergeSort : 0.046412900003	**Peor de los casos:** BubbleSort: 32.50327869 [s] BubbleSortOptimizado: 32.2550973 [s] MergeSort: 0.04612959 [s]

4) Con los datos obtenidos de los 3 casos previamente implementados, se realizaron 3 gráficas. Cada gráfica es de cada caso y en ellas se representa a través de líneas el tiempo que toma el algoritmo BubbleSort (línea roja), BubbleSortOptimizado (línea azul) y MergeSort (línea verde).


Mejor Caso
n vs t
Bubble Sort
Bubble Sort Optimizado
Merge Sort
Tiempo [s]
Numero de Datos [s]


Peor Caso
n vs t
Bubble Sort
Bubble Sort Optimizado
Merge Sort
Tiempo [s]
Numero de Datos [n]


Caso Promedio
n vs t
Bubble Sort
Bubble Sort Optimizado
Merge Sort
Tiempo [s]
Numero de Datos [n]

5) Por último, se puso a prueba el algoritmo MergeSort al pasarle 2 millones de números entre -100 y 100, se calculó el tiempo y esto es lo obtenido:

    a. Mejor de los casos: 7.438630 segundos
    b. Caso promedio: 8.247891 segundos
    c. Peor de los casos: 7.498036 segundos

    Como BubleSort es un algoritmo que toma mucho tiempo, tomamos los datos obtenidos previamente y se calcula cuánto tardaría el algoritmo al ordenar una lista de 20 millones de números entre -100 y 100, esto es lo obtenido:

    d. En el mejor de los casos: 13781.63409 (3.82 hrs) segundos aproximadamente
    e. En el caso promedio: 21193.70417 (5.88 hrs) segundos aproximadamente
    f. En el peor de los casos: 32503.27869 segundos (9.02 hrs) aproximadamente

Conclusiones

El flujo del programa para ejecutar los algoritmos fue:

1) Los casos promedio
2) El mejor caso
3) El peor caso

Esto se implementó para que se pudiera generar una lista de números como solicitó el profesor, después ser ordenados aplicando el caso promedio y una vez ordenados, se puede aplicar el mejor caso, para el peor caso se utilizó la función **.reverse()** a cada arreglo para que los números estén ordenados decrecientemente.

El algoritmo MergeSort es mucho más eficaz que el algoritmo BubbleSort y BubbleSortOptimizado debido a que toma menos tiempo y ahorra recursos. No imaginé que este algoritmo le tomaría 8 segundos aproximadamente para ordenar una lista de 2 millones de números.

Se cumplieron con todos los ejercicios propuestos por el profesor, así como sus gráficas realizadas con la librería matplotlib. Esta librería no viene incluida en la carpeta de la práctica.