COMPARACIÓN DE TIEMPOS ENTRE LAS VERSIONES DE LA PRIORITY QUEUE (medido en microsegundos)

Estudiante: Rodrigo Alonso Torres Sotomayor

Priority Queue con lista enlazada

Prueba1:

```
> ./main
100 Datos -> Execution Time: 7  microsec
1000 Datos -> Execution Time: 57    microsec
10000 Datos -> Execution Time: 1175 microsec
100000 Datos -> Execution Time: 158962  microsec
1000000 Datos -> Execution Time: 14817614   microsec
>
```

Prueba2:

```
> ./main
100 Datos -> Execution Time: 6  microsec
1000 Datos -> Execution Time: 93    microsec
10000 Datos -> Execution Time: 4933 microsec
100000 Datos -> Execution Time: 162945  microsec
1000000 Datos -> Execution Time: 16415165   microsec
>
```

Priority Queue con skip list

Prueba1:

```
> ./main
100 Datos -> Execution Time: 31 microsec
1000 Datos -> Execution Time: 412   microsec
10000 Datos -> Execution Time: 10012    microsec
100000 Datos -> Execution Time: 187672  microsec
1000000 Datos -> Execution Time: 1627988    microsec
1
```

Prueba2:

```
> ./main
100 Datos -> Execution Time: 41 microsec
1000 Datos -> Execution Time: 303   microsec
10000 Datos -> Execution Time: 44611    microsec
100000 Datos -> Execution Time: 107933  microsec
1000000 Datos -> Execution Time: 1403267    microsec
1
>
```

Priority Queue Paralela (usando skiplist)

Prueba1:

```
100 Datos -> Execution Time: 8495       microsec
1000 Datos -> Execution Time: 4628      microsec
10000 Datos -> Execution Time: 2918     microsec
100000 Datos -> Execution Time: 2905    microsec
1000000 Datos -> Execution Time: 2912   microsec
```

Prueba2:

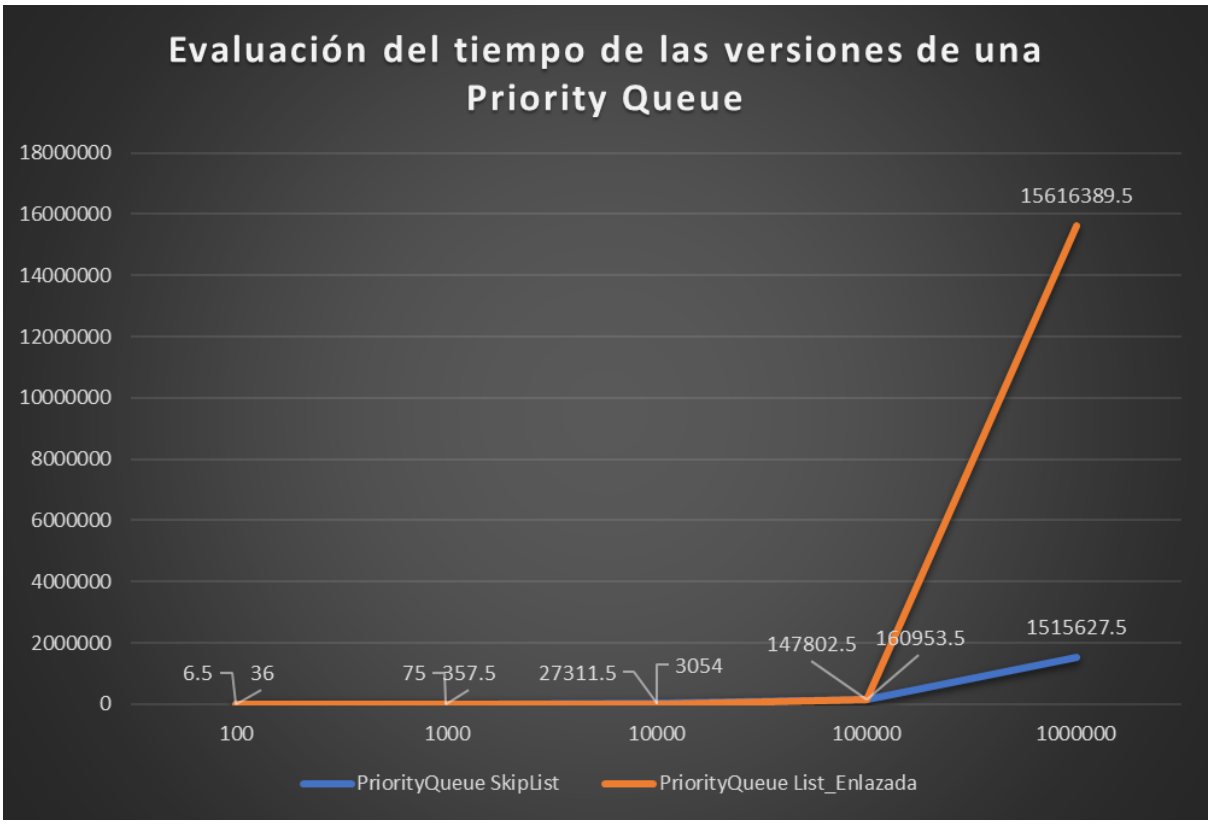
```
100 Datos -> Execution Time: 5644       microsec
1000 Datos -> Execution Time: 1960      microsec
10000 Datos -> Execution Time: 2096     microsec
100000 Datos -> Execution Time: 2234    microsec
1000000 Datos -> Execution Time: 1977   microsec
```

Tabla:

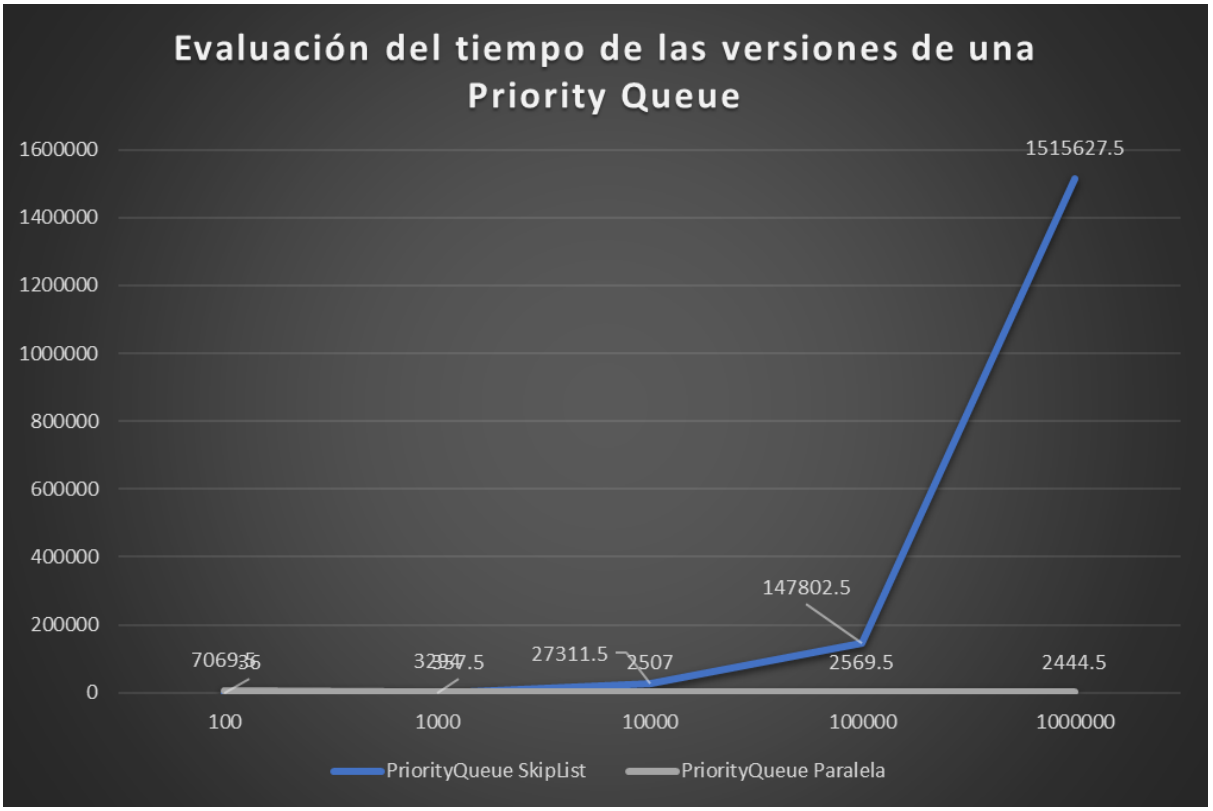
N° Datos	PriorityQueue SkipList	PriorityQueue List_Enlazada	PriorityQueue Paralela
100	36	6.5	7069.5
1000	357.5	75	3294
10000	27311.5	3054	2507
100000	147802.5	160953.5	2569.5
1000000	1515627.5	15616389.5	2444.5

Gráficas:

PriorityQueue con SkipList y con lista enlazada



PriorityQueue SkipList y Paralelizada:



Conclusión:
Como se puede observar la PriorityQueue con skiplist y usando una lista enlazada son muy rápidas y eficientes cuando trabajan con una cantidad baja de datos llegando hasta trabajar de forma eficiente con 1000 datos, pero a partir de los 10000 datos la versión paralelizada es mucho más eficiente y rápida, superando por mucho a las otras versiones. Por ello concluimos que la Priority Queue Parelizada usando SkipList es muy eficiente y rápida con el manejo de grandes cantidades de datos.