Conclusiones Lab 4 Proa

Ignacio Gomis Lli

Juan Pablo Uriol Balbín

Con la función Memcost1 que utiliza los constructores por defecto obtenemos los tiempos totales, a misma cantidad de operaciones:
CostMem1: 0.0336839 s

CostMem2: 0.313172 s

CostMem3: 0.333417 s

CostMem4: 0.681024 s

Como podemos observar, realizar los cambios en la memoria estática es lo mejor en tiempo, y que los unique pointer tienen un rendimiento levemente peor que los punteros simples, aunque dan una mayor protección de los datos. Finalmente, podemos observar que los shared pointers tienen una gran carga de tiempo comparado con los anteriores, debido a la implementación para llevar el conteo de objetos.

Con la funcion Memcost2 en primer lugar sobreescribimos el operador new y delete con la operación general, consiguiendo un tiempo en CostMem2 de aproximadamente 9 segundos (representa una mejora de 1/6 conrespecto a Memcost1)

A continuación sobreescribimos con el método visto en clase de emplear el pool y el allocation map:

CostMem1: 0.0316279 s //Sin efecto

CostMem2: 3.25401 s //Empeora 3 segundos

CostMem3: 3.24547 s

CostMem4: 3.61405 s

Como podemos ver este método es muy lento en comparación a la operación por defecto