Ejercicio 4: Mejor y peor caso

Retome el ejercicio de ordenación mediante el algoritmo de la burbuja. Debe modificar el código que genera los datos de entrada para situarnos en dos escenarios diferentes:
• El mejor caso posible. Para este algoritmo, si la entrada es un vector que ya está ordenado el tiempo de cómputo es menor ya que no tiene que intercambiar ningún elemento.
• El peor caso posible. Si la entrada es un vector ordenado en orden inverso estaremos en la peor situación posible ya que en cada iteración del bucle interno hay que hacer un intercambio. Calcule la eficiencia empírica en ambos escenarios y compárela con el resultado del ejercicio 1.

1) Creamos los ficheros 'mejorcasoposible.cpp' y 'peorcasoposible.cpp' en el que incluímos distintas formas en la representación del vector.
   -En el mejor caso posible introducimos los elementos ordenados:

   ```
   for (int i=0; i<tam; i++)
     v[i]=i;
   ```

   -En el peor caso posible introducimos los elementos ordenados pero en orden inverso

   ```
   for(int i=0; i<tam; i++)
     v[i]=tam-i;
   ```

   Por lo que suponemos que en el mejor caso posible tardará menos ya que los elementos están ordenados, no como ocurre en el peor caso

2) Generamos los ejecutables mediante:

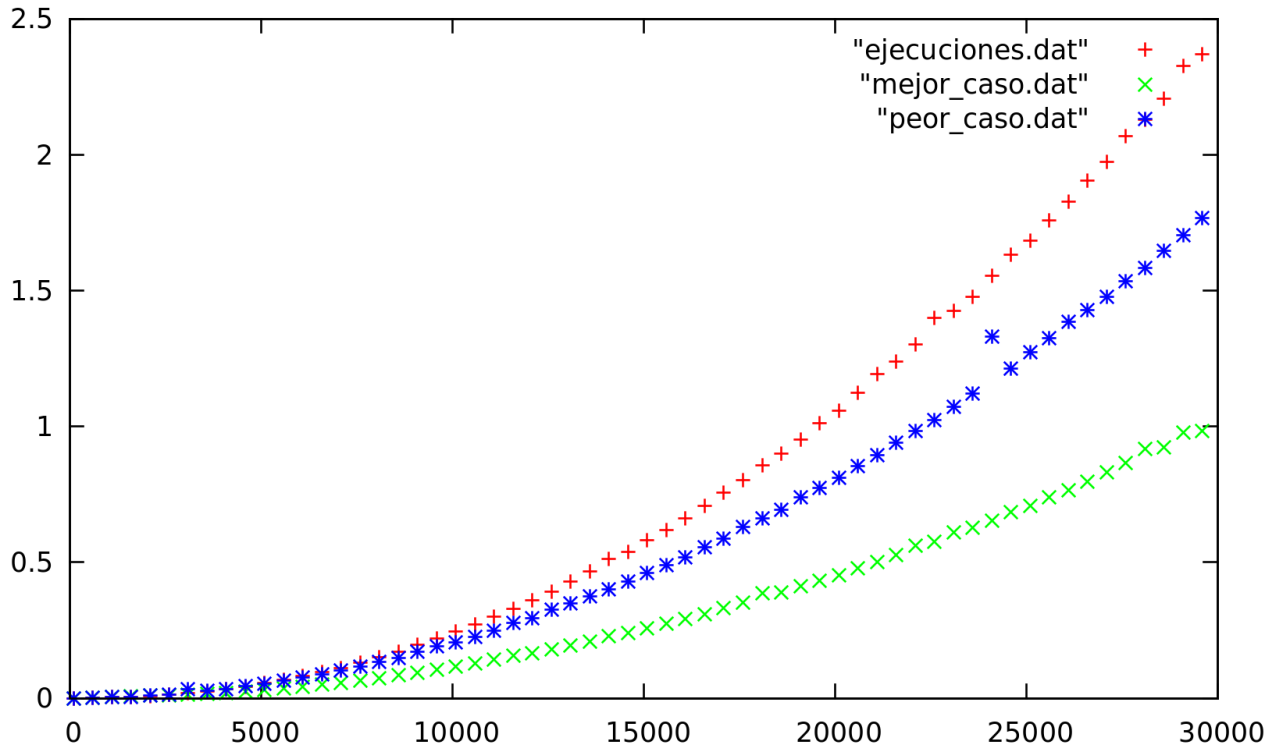
g++ mejorcasoposible.cpp –o mejor_caso
g++ peorcasoposible.cpp -o peor_caso

3) Creamos los scripts mejor_caso.csh y peor_caso.csh que generarán los tiempos de ejecución en archivos .dat dierentes

   ./mejor_caso.csh
   ./peor_caso.csh

4) Con la ayuda de gnuplot, representamos ambas gráficas y las comparamos con la del ejercicio1:

>gnuplot
>plot “mejor,caso.dat” , “peor_caso.dat” , “ejecuciones.dat”



Como podemos observar, la eficiencia del mejor caso posible es mayor que la del mayor. Sin embargo, la eficiencia empírica del ejercicio1, 'ejecuciones.dat' es menos eficiente que 'peor_caso.dat', es decir, que se tarda más en ordenar un vector con elementos aleatorios que un vector en el que todos los elementos están ordenados en orden inverso, ¿Por qué ocurre esto?
Esto es porque en el peor caso, el procesador puede es capaz de predecir que siempre va a tener que intercambiar todos los valores por lo que irá más rápido que en el vector aleatorio.

Características del ordenador:

Fabricante	Lenovo
Procesador	Intel Core i7 4710HQ
Sistema Operativo	Ubuntu
Versión SO	14.04
RAM	8192MB
CPU	64 bits