Seminario "Computación y Algebra Lineal Densa" 2023–24	Entregable para Aula ea_la01
Nombre y apellidos del alumno:	

Práctica 01. Computación de Altas Prestaciones

1 (*Tiempo estimado: 30'*) Los polinomios son muy empleados en numerosos campos de las ciencias e ingenierías dado que su derivación e integración son triviales. Uno de sus usos más frecuentes es el ajuste de un polinomio a unos datos reales con el objetivo de calcular áreas o volúmenes de dichos datos reales.

En este ejercicio se va a analizar e intentar mejorar las prestaciones de un código que evalúa un polinomio en múltiples puntos.

No te dejes intimidar por la longitud del enunciado de este ejercicio. El código que tú debes reescribir y modificar es muy pequeño (apenas cinco o seis líneas). El texto es largo para guiarte paso a paso.

Para completar este ejercicio debes realizar lo siguiente:

1.1) Lo primero que debes hacer es **decargarte o conseguir el fichero comprimido** con el código fuente de este ejercicio. Una vez descargado dicho fichero, debes descomprimirlo.

1.2) Dentro del directorio descomprimido puedes encontrar un directorio llamado `polic_ejer`. Entra en dicho directorio y compila los ficheros fuentes. Puedes hacerlo tecleando el siguiente comando: `./c`

Dentro de este directorio, el fichero `test_polic.c` contiene el programa principal de la aplicación. Este fichero evalúa un mismo polinomio para un conjunto de valores almacenados en un vector x. Los valores del polinomio para dichos puntos se guardan en el vector y de tal forma que: $y_i = p(x_i)$, donde p es el polinomio. Es decir, para cada punto x_i se evalúa el polinomio y se guarda su valor en y_i.

1.3) Ejecuta el programa tecleando el siguiente comando: `./test_polic.x test1.in`

Este comando toma las dimensiones del vector de puntos a evaluar, el grado del polinomio y demás parámetros del fichero `test1.in`.

El fichero `test1.in` se emplea para comprobación y no para obtener tiempos, pues le indica al programa principal que evalúe un polinomio de grado pequeño en un número pequeño de puntos y muestre el resultado en pantalla.

Ten en cuenta que los dos vectores x (puntos que evaluar) e y (valores del polinomio para dichos puntos) se muestran antes de realizar el cálculo y después de realizar el cálculo.

Cuando modifiques el programa en los puntos siguientes, deberás comparar la salida del programa modificado con la salida que has obtenido al ejecutar este comando.

1.4) Ejecuta el programa tecleando el siguiente comando: `perf stat -a -d ./test_polic.x test2.in`

El fichero `test2.in` se emplea para evaluar prestaciones, pues le indica al programa principal que evalúe un polinomio de grado mayor en un número mayor de puntos y no muestre el resultado en pantalla.

El comando `perf stat -a -d` ejecuta el comando que le sigue recogiendo estadísticas detalladas e interesantes sobre el uso del procesador y el sistema de memoria.

Anota los tiempos que ha necesitado el programa.

1.5) Examina la salida atentamente. ¿Notas algún número fuera de lo normal? ¿Cuántas instrucciones por ciclo consigue este programa? ¿Cuántas instrucciones es capaz de ejecutar el procesador? Compara ambas.

1.6) ¿A qué se puede deber esta diferencia? Para determinar la causa, muy probablemente debes examinar el código del fichero `test_polic.c`. Al final de este fichero hay una función que evalúa el polinomio en un conjunto de puntos almacenado en un vector x. Dicha función se llama `polic_ruffini_1` y es muy corta.

1.7) Haz una copia del directorio `polic_ejer` con el nuevo nombre `polic_ejer_v2` con el comando siguiente: `cp -fr polic_ejer polic_ejer_v2`

1.8) El objetivo ahora es modificar el código del directorio `polic_ejer_v2` para mejorar las prestaciones. El anterior directorio `polic_ejer` se mantiene como referencia de resultados y prestaciones.

El único cambio que se debe realizar es en la función que evalúa el polinomio en múltiples puntos.

Para evitar confusiones, dentro del fichero `test_polic.c` realiza los siguientes cambios:

1. Cambia el texto de salida “Ruffini 1” por “Ruffini 2” en la llamada a la función `print_result`.
2. Cambia el nombre de la función `poli_ruffini_1` con el nuevo nombre `poli_ruffini_2`.

3. Cambia el prototipo de la función `poli_ruffini_1` con el nuevo nombre. Como es habitual, el prototipo se encuentra al principio del fichero.
4. Cambia la llamada a la función `poli_ruffini_1` con el nuevo nombre.

Ahora debes modificar la nueva función. Intenta mejorarla con la información de estadísticas que has obtenido. La modificación es sencilla si sabes cuál es el problema de la versión anterior.

Escribe el código de la nueva función.

1.9) Una vez realizada la modificación, ejecuta el comando: `./test_polic.x test1.in`
Compara los resultados con los obtenidos con la versión inicial. Si no son los mismos, tu modificación no es correcta.

1.10) Una vez comprobada que la nueva versión es correcta, ejecuta el comando siguiente: `perf stat -a - d ./test_polic.x test2.in`
Compara las prestaciones en tiempo con las obtenidas antes.

1.11) Examina las estadísticas y compáralas con las obtenidas antes.