Tarea 1: Calibración

Análisis y Diseño de Algoritmos

Prof. Miguel Pizaña

Objetivo: Determinar cuántas operaciones por segundo hace su computadora.

1. Compilar y ejecutar el programa anexo en Linux o en algún sistema tipo Unix.
2. Entregar reporte de este primer experimento.

El reporte debe contener:

1. Título de la tarea, nombre y matrícula del estudiante (de manera individual).
2. Breve descripción de qué es lo que hace el programa y qué resultados se obtuvieron.
3. Descripción del tipo de sistema en el que se corrió: S.O. y versión, número de procesadores, número de núcleos de cada procesador, descripción completa del(los) procesador(es), frecuencia de operación y cantidad de memoria RAM.
4. Una captura de pantalla de la ejecución del programa en su computadora.
5. Conclusiones: Reportar el tiempo que tardó el programa en segundos, reportar el número de operaciones por segundo que puede hacer su computadora (en un único núcleo, un único hilo), indicar si tuvo algunas dificultades en realizar la tarea y qué fué lo que aprendió de ella.

```c
/*************************
Determinar cuanto tardan en realizarse
10^9 operaciones en su computadora.
*************************/

/*******************************************
Compilar en Linux con:

   gcc -lrt -o calibracion calibracion.c

Compilar en macOS con:

   gcc -o calibracion calibracion.c

En Linux, necesitas tener instalada la biblioteca librt
que es la biblioteca de tiempo real del compilador gcc.
*******************************************/

#include <stdio.h>
#include <stdlib.h>
#include <time.h>

//Esta subrutina simplemente convierte la
//estructura de tiempo a un long long int.
//Solo funciona para tiempos relativamente pequenios
//(hasta 37 anios en la mayoria de los sistemas)
long long int time2int(struct timespec t){
      long long int T;
     //convierte tiempo a entero
     T=t.tv_sec;
     T*=(10000000001);
     T+=t.tv_nsec;
     return T;
}

```

```
37 int main(){
38   struct  timespec t1, t2;
39   long long int T;
40   long int i;
41
42     clock_gettime(CLOCK_PROCESS_CPUTIME_ID, &t1);
43
44       //Codigo a medir. No agregar nada aqui en medio.
45       for(i=0;i<10000000001;i++);
46
47     clock_gettime(CLOCK_PROCESS_CPUTIME_ID, &t2);
48     T=time2int(t2)-time2int(t1);
49     //Imprimimos el tiempo que se tardo.
50     printf("%lld\n",T);
51   return 0;
52 }
```