

Arquitectura de Computadores (AC)

Cuaderno de prácticas.

Bloque Práctico 0. Entorno de programación

Estudiante (nombre y apellidos):

Grupo de prácticas y profesor de prácticas:

Fecha de entrega:

Fecha evaluación en clase:

Antes de comenzar a realizar el trabajo de este cuaderno consultar el fichero con los normas de prácticas que se encuentra en SWAD

Parte I. Ejercicios basados en los ejemplos del seminario práctico

Crear el directorio con nombre bp0 en atcgrid y en el PC local.

NOTA: En las prácticas se usa slurm como gestor de colas. Consideraciones a tener en cuenta:

- Slurm está configurado para asignar recursos a los procesos (llamados *tasks* en slurm) a nivel de core físico. Esto significa que por defecto slurm asigna un core a un proceso, para asignar más de uno se debe usar con sbatch/srun la opción `--cpus-per-task`.
- En slurm, por defecto, `cpu` se refiere a cores lógicos (ej. en la opción `--cpus-per-task`), si no se quieren usar cores lógicos hay que añadir la opción `--hint=nomultithread` a sbatch/srun.
- Para asegurar que solo se crea un proceso hay que incluir `-n1` en sbatch/srun.
- Para que no se ejecute más de un proceso en un nodo de atcgrid hay que usar `--exclusive` con sbatch/srun (se recomienda no utilizarlo en los srun dentro de un script).
- Los srun dentro de un script heredan las opciones fijadas en el sbatch que se usa para enviar el script a la cola slurm.

1. Ejecutar `lscpu` en el PC y en un nodo de cómputo de atcgrid. (Crear directorio ejer1)

(a) Mostrar con capturas de pantalla el resultado de estas ejecuciones.

RESPUESTA:

```
Actividades Terminal
vie 18:41
c3estudiante18@atcgrid:~/bp0/ejer1

Archivo Editar Ver Buscar Terminal Ayuda

JavierRamirezPulido>vie feb 28 lscpu
Architecture:          x86_64
CPU op-mode(s):        32-bit, 64-bit
Byte Order:             Little Endian
CPU(s):                 8
On-line CPU(s) list:    0-7
Thread(s) per core:     2
Core(s) per socket:     4
Socket(s):              1
NUMA node(s):           1
Vendor ID:              GenuineIntel
CPU family:             6
Model:                  158
Model name:             Intel(R) Xeon(R) CPU E3-1230 v6 @ 3.50GHz
Stepping:               9
CPU MHz:                1599.822
CPU max MHz:            3900.0000
CPU min MHz:            800.0000
BogoMIPS:               7008.00
Virtualization:         VT-x
L1d cache:              32K
L1i cache:              32K
L2 cache:               256K
L3 cache:               8192K
NUMA node0 CPU(s):      0-7
Flags:                  fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush dts acpi mmx fxsr sse sse2 ss ht tm pbe
                        syscall nx pdpe1gb rdtscp lm constant_tsc art arch_perfmon pebs bts rep_good nopl xtopology nonstop_tsc aperfmperf eagerfpu pni pclmulqdq dte
                        s64 monitor ds_cpl vmx smx est tm2 sse3 sdbg fma cx16 xtpr pdcm pcid sse4_1 sse4_2 x2apic movbe popcnt tsc_deadline_timer aes xsave avx f16c
                        rdrand lahf_lm abm 3dnowprefetch epb intel_pt ssbd ibrs ibpb stibp tpr_shadow vmmi flexpriority ept vpid fsgsbase tsc_adjust bmi1 hle avx2 sme
                        p bmi2 erms invpcid rtm mpx rdseed adx smap clflushopt xsaveopt xsavec xgetbv1 dtherm ida arat pln pts hwp hwp_notify hwp_act_window hwp_epp m
                        d_clear spec_ctrl intel_stibp flush_l1d
JavierRamirezPulido>vie feb 28
```

En atcgrid

```

JavierRamirezPulido>vie feb 28 lscpu
Arquitectura:                x86_64
modo(s) de operación de las CPUs: 32-bit, 64-bit
Orden de los bytes:          Little Endian
CPU(s):                       2
Lista de la(s) CPU(s) en línea: 0,1
Hilo(s) de procesamiento por núcleo: 1
Núcleo(s) por «socket»:      2
«Socket(s)»:                  1
Modo(s) NUMA:                 1
ID de fabricante:             GenuineIntel
Familia de CPU:                6
Modelo:                        55
Nombre del modelo:             Intel(R) Celeron(R) CPU N2815 @ 1.86GHz
Revisión:                      3
CPU MHz:                       1822.724
CPU MHz máx.:                  2132,8000
CPU MHz mín.:                   533,2000
BogoMIPS:                       3732.40
Virtualización:                 VT-x
Caché L1d:                      24K
Caché L1i:                       32K
Caché L2:                       1024K
CPU(s) del nodo NUMA 0:         0,1
Indicadores:                    fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush dts acpi mmx fxsr sse sse2
                                ss ht tm pbe syscall nx rdtscp lm constant_tsc arch_perfmon pebs bts rep_good nopl xtopology tsc_reliable nonstop_tsc cpuid aperfmperf tsc_k
                                nownfreq pni pclmulqdq dtes64 monitor ds_cpl vmx est tm2 ssse3 cx16 xtpr pdcm sse4_1 sse4_2 movbe popcnt tsc_deadline_timer rdrand lahf_lm 3d
                                nowprefetch ept tpr_shadow vnmi flexpriority ept vpid tsc_adjust smep erms dtherm ida arat
JavierRamirezPulido>vie feb 28

```

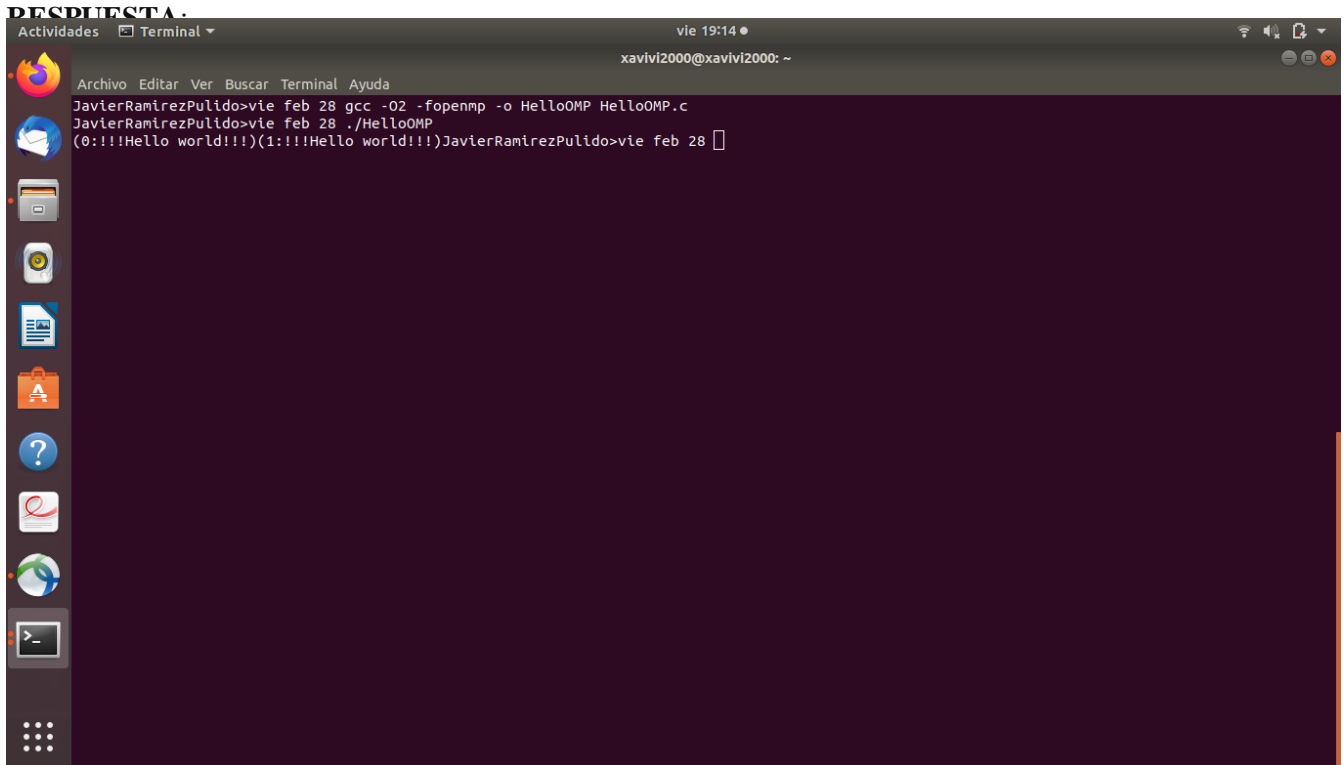
En mi PC

(b) ¿Cuántos cores físicos y cuántos cores lógicos tienen los nodos de cómputo de atcgrid y el PC? Razonar las respuestas

RESPUESTA: En PC tengo un procesador (<<Socket(s)>>) con 2 núcleos físicos (Núcleo(s) por <<socket>>) y una hebra de procesamiento por núcleo (Hilo(s) de procesamiento por núcleo) por lo que tengo 2 CPUs (son los cores lógicos) y tengo 2 cores físicos. En atcgrid tengo un solo socket(s), que es lo mismo que tener un solo procesador, y además este tiene 4 cores físicos (Core(s) per socket) y en cada núcleo 2 líneas de procesamiento (Thread(s) per core). Por tanto, tengo 8 cores lógicos (CPU(s)).

2. Compilar y ejecutar en el PC el código HelloOMP.c del seminario (recordar que se debe usar un directorio independiente para cada ejercicio dentro de bp0 que contenga todo lo utilizado, implementado o generado durante el desarrollo del mismo, para el presente ejercicio el directorio sería **ejer2**, como se indica en las normas de prácticas).

(a) Adjuntar capturas de pantalla que muestren la compilación y ejecución en el PC.



```
DESDIESTA
Actividades Terminal vie 19:14
xavivi2000@xavivi2000: ~
Archivo Editar Ver Buscar Terminal Ayuda
JavierRamirezPulido>vie feb 28 gcc -O2 -fopenmp -o HelloOMP HelloOMP.c
JavierRamirezPulido>vie feb 28 ./HelloOMP
(0:!!!Hello world!!!)(1:!!!Hello world!!!)JavierRamirezPulido>vie feb 28
```

Compilación y ejecución en el PC.

(b) Justificar el número de “Hello world” que se imprimen en pantalla teniendo en cuenta la salida que devuelve `lscpu`.

RESPUESTA: Se imprime un “Hello World” por cada CPU que tenga (número de cores lógicos del sistema)

3. Copiar el ejecutable de `HelloOMP.c` que ha generado anteriormente y que se encuentra en el directorio `ej2` del PC al directorio `ej2` de su home en el *front-end* de `atcgrid`. Ejecutar este código en un nodo de cómputo de `atcgrid` a través de `cola ac` del gestor de colas (no use ningún *script*) utilizando directamente en línea de comandos:

(a) `srun -p ac -n1 --cpus-per-task=12 --hint=nomultithread HelloOMP`

Adjuntar capturas de pantalla que muestren el envío a la cola de la ejecución y el resultado de esta ejecución tal y como la devuelve el gestor de colas.

[illegible]

Actividades Terminal

vie 19:39 ●

c3estudiante18@atcgriid:~/bp0/ejer2

Archivo Editar Ver Buscar Terminal Ayuda

JavierRamirezPulido>vie feb 28 pwd
/home/c3estudiante18/bp0/ejer2
JavierRamirezPulido>vie feb 28 ls
HelloOMP
JavierRamirezPulido>vie feb 28 srun -p ac -n1 --cpus-per-task=12 --hint=nomultithread HelloOMP
(0:!!!Hello world!!!)(4:!!!Hello world!!!)(6:!!!Hello world!!!)(8:!!!Hello world!!!)(2:!!!Hello world!!!)(1:!!!Hello world!!!)(11:!!!Hello world!!!)(7:!!!Hello world!!!)(9:!!!Hello world!!!)(3:!!!Hello world!!!)(5:!!!Hello world!!!)(10:!!!Hello world!!!)JavierRamirezPulido>vie feb 28

```
(b) srun -p ac -n1 --cpus-per-task=24 HelloOMP
```

Adjuntar capturas de pantalla que muestren el envío a la cola de la ejecución y el resultado de esta ejecución tal y como la devuelve el gestor de colas.

```

c3estudiante18@atcgrid:~/bp0/ejer2
JavierRamirezPulido>vie feb 28 srun -p ac -n1 --cpus-per-task=24 HelloOMP
(4:!!!Hello world!!!)(0:!!!Hello world!!!)(3:!!!Hello world!!!)(23:!!!Hello world!!!)(1:!!!Hello world!!!)(9:!!!Hello world!!!)(10:!!!Hello wo
rld!!!)(6:!!!Hello world!!!)(8:!!!Hello world!!!)(21:!!!Hello world!!!)(18:!!!Hello world!!!)(22:!!!Hello world!!!)(12:!!!Hello world!!!)(20:!!
!!!Hello world!!!)(11:!!!Hello world!!!)(15:!!!Hello world!!!)(17:!!!Hello world!!!)(7:!!!Hello world!!!)(13:!!!Hello world!!!)(16:!!!Hello wor
ld!!!)(5:!!!Hello world!!!)(2:!!!Hello world!!!)(14:!!!Hello world!!!)(19:!!!Hello world!!!)JavierRamirezPulido>vie feb 28

```

(c) `srun -p ac -n1 HelloOMP`

Adjuntar capturas de pantalla que muestren el envío a la cola de la ejecución y el resultado de esta ejecución tal y como la devuelve el gestor de colas.

RESPUESTA:

```

c3estudiante18@atcgrid:~/bp0/ejer2
JavierRamirezPulido>vie feb 28 srun -p ac -n1 HelloOMP
(0:!!!Hello world!!!)(1:!!!Hello world!!!)JavierRamirezPulido>vie feb 28

```

(d) ¿Qué orden `srun` usaría para que `HelloOMP` utilice los 12 cores físicos de un nodo de cómputo de `atcgrid` (se debe imprimir un único mensaje desde cada uno de ellos, en total, 12)?

De las órdenes que he utilizado, la que imprime un mensaje por cada uno de los 12 cores físicos es la primera (`srun -p ac -n1 --cpus-per-task=12 --hint=nomultithread HelloOMP`)

4. Modificar en su PC `HelloOMP.c` para que se imprima “world” en un `printf` distinto al usado para “Hello”, en ambos `printf` se debe imprimir el identificador del thread que escribe en pantalla. Nombrar al código resultante `HelloOMP2.c`. Compilar este nuevo código en el PC y ejecutarlo. Copiar el fichero ejecutable resultante al front-end de atcgrid (directorio `ejer4`). Ejecutar el código en un nodo de cómputo de atcgrid usando el script `script_helloomp.sh` del seminario (el nombre del ejecutable en el script debe ser `HelloOMP2`).

(a) Utilizar: `sbatch -p ac -n1 --cpus-per-task=12 --hint=nomultithread script_helloomp.sh`. Adjuntar capturas de pantalla que muestren el nuevo código, la compilación, el envío a la cola de la ejecución y el resultado de esta ejecución tal y como la devuelve el gestor de colas.

RESPUESTA:

The top screenshot shows a text editor window titled "HelloOMP2.c" with the following code:

```
/* Compilar con:
gcc-O2 -fopenmp-o HelloOMPHelloOMP.c
*/
#include<stdio.h>
#include<omp.h>

int main(void){

#pragma omp parallel
printf("(d:!!!Hello)",omp_get_thread_num());
#pragma omp parallel
printf("(d: world!!!)",omp_get_thread_num());

return(0);
}
```

The bottom screenshot shows a terminal window with the following commands and output:

```
JavierRamirezPulido>vie feb 28 gcc -O2 -fopenmp -o HelloOMP2 HelloOMP2.c
JavierRamirezPulido>vie feb 28 ./HelloOMP2
(0:!!!Hello)(1:!!!Hello)(1: world!!!)(0: world!!!)JavierRamirezPulido>vie feb 28
```

[illegible]

The image shows two screenshots of a Linux terminal window. The top screenshot shows the command `sbatch -p ac -n1 --cpus-per-task=12 --hint=nomultithread script_helloomp.sh` being executed, resulting in a submitted batch job 14019. The bottom screenshot shows the output of `cat slurm-14017.out`, which contains details about the job execution, including the user, job ID, script name, directory, and the number of nodes and CPUs assigned. The output also shows the execution of the `helloOMP` script with 12 threads, followed by a list of threads printing 'Hello' and 'world'.

```

Actividades Terminal vie 20:13 ● c3estudiante18@atcgrid:~/bp0/ejer4
Archivo Editar Ver Buscar Terminal Ayuda
JavierRamirezPulido>vie feb 28 sbatch -p ac -n1 --cpus-per-task=12 --hint=nomultithread script_helloomp.sh
Submitted batch job 14019
JavierRamirezPulido>vie feb 28

Actividades Terminal vie 20:18 ● c3estudiante18@atcgrid:~/bp0/ejer4
Archivo Editar Ver Buscar Terminal Ayuda
JavierRamirezPulido>vie feb 28 cat slurm-14017.out
Id. usuario del trabajo: c3estudiante18
Id. del trabajo: 14017
Nombre del trabajo especificado por usuario: helloOMP
Directorio de trabajo (en el que se ejecuta el script): /home/c3estudiante18/bp0/ejer4
Cola: ac
Nodo que ejecuta este trabajo:atcgrid
Nº de nodos asignados al trabajo: 1
Nodos asignados al trabajo: atcgrid1
CPUs por nodo: 24

1. Ejecución helloOMP una vez sin cambiar nº de threads (valor por defecto):
(0:!!!Hello)(10:!!!Hello)(5:!!!Hello)(7:!!!Hello)(4:!!!Hello)(11:!!!Hello)(2:!!!Hello)(3:!!!Hello)(6:!!!Hello)(8:!!!Hello)(0:!!!Hello)(9:!!!Hello)(9: world!!!)(2: world!!!)(11: world!!!)(8: world!!!)(7: world!!!)(4: world!!!)(1: world!!!)(6: world!!!)(3: world!!!)(10: world!!!)(5: world!!!)(0: world!!!)
2. Ejecución helloOMP varias veces con distinto nº de threads:
- Para 12 threads:
(0:!!!Hello)(2:!!!Hello)(6:!!!Hello)(7:!!!Hello)(10:!!!Hello)(1:!!!Hello)(4:!!!Hello)(11:!!!Hello)(3:!!!Hello)(9:!!!Hello)(5:!!!Hello)(8:!!!Hello)(8: world!!!)(0: world!!!)(11: world!!!)(7: world!!!)(5: world!!!)(2: world!!!)(9: world!!!)(3: world!!!)(10: world!!!)(6: world!!!)(4: world!!!)(1: world!!!)
- Para 6 threads:
(0:!!!Hello)(1:!!!Hello)(4:!!!Hello)(5:!!!Hello)(3:!!!Hello)(2:!!!Hello)(2: world!!!)(3: world!!!)(4: world!!!)(0: world!!!)(5: world!!!)(1: world!!!)
- Para 3 threads:
(1:!!!Hello)(0:!!!Hello)(2:!!!Hello)(1: world!!!)(2: world!!!)(0: world!!!)
- Para 1 threads:
(0:!!!Hello)(0: world!!!)JavierRamirezPulido>vie feb 28

```

(b) ¿Qué nodo de cómputo de atcgrid ha ejecutado el script? Explicar cómo ha obtenido esta información.

RESPUESTA: El nodo de cómputo de atcgrid que ha ejecutado el script es atcgrid1, en el apartado que dice los nodos asignados al trabajo (\$SLURM_JOB_NODELIST)

NOTA: Utilizar siempre con `sbatch` las opciones `-n1` y `--cpus-per-task`, `--exclusive` y, para usar cores físicos y no lógicos, no olvide incluir `--hint=nomultithread`. Utilizar siempre con `srun`, si lo usa fuera de un script, las opciones `-n1` y `--cpus-per-task` y, para usar cores físicos y no lógicos, no olvide incluir `--hint=nomultithread`. Recordar que los `srun` dentro de un script heredan las opciones utilizadas en el `sbatch` que se usa para enviar el script a la cola slurm. Se recomienda usar `sbatch` en lugar de `srun` para enviar trabajos a ejecutar a través slurm porque éste último deja bloqueada la ventana hasta que termina la ejecución, mientras que usando `sbatch` la ejecución se realiza en segundo plano.

Parte II. Resto de ejercicios

5. Generar en el PC el ejecutable del código fuente C del Listado 1 para vectores locales (para ello antes de compilar debe descomentar la definición de VECTOR_LOCAL y comentar las definiciones de VECTOR_GLOBAL y VECTOR_DYNAMIC). El comentario inicial del código muestra la orden para compilar (siempre hay que usar -O2 al compilar como se indica en las normas de prácticas). Incorporar volcados de pantalla que demuestren la compilación y la ejecución correcta del código en el PC (leer lo indicado al respecto en las normas de prácticas).

RESPUESTA:

```

Actividades Terminal
vie 20:48
xavivi2000@xavivi2000: ~

Archivo Editar Ver Buscar Terminal Ayuda
JavierRamirezPulido>vie feb 28 gcc -O2 SumaVectores.c -o SumaVectores -lrt
SumaVectores.c: In function 'main':
SumaVectores.c:45:32: warning: format '%u' expects argument of type 'unsigned int', but argument 3 has type 'long unsigned int' [-Wformat=]
printf("Tamaño Vectores:%u (%u B)\n",N, sizeof(unsigned int));
                           ~^
                           %lu
JavierRamirezPulido>vie feb 28 ./SumaVectores 5
Tamaño Vectores:5 (4 B)
Tiempo:0.000000405 / Tamaño Vectores:5
/ V1[0]+V2[0]=V3[0](0.500000+0.500000=1.000000) /
/ V1[1]+V2[1]=V3[1](0.600000+0.400000=1.000000) /
/ V1[2]+V2[2]=V3[2](0.700000+0.300000=1.000000) /
/ V1[3]+V2[3]=V3[3](0.800000+0.200000=1.000000) /
/ V1[4]+V2[4]=V3[4](0.900000+0.100000=1.000000) /
JavierRamirezPulido>vie feb 28

```

Ejecucion en el PC.

```

Actividades Terminal
sáb 00:21
xavivi2000@xavivi2000: ~

Archivo Editar Ver Buscar Terminal Ayuda
JavierRamirezPulido>sáb feb 29 ./sumavectores_script.sh
Tamaño Vectores:65536 (4 B)
Tiempo:0.000876608 / Tamaño Vectores:65536
5535](13107.100000+0.100000=13107.200000) /
Tamaño Vectores:131072 (4 B)
Tiempo:0.002003626 / Tamaño Vectores:131072
V3[131071](26214.300000+0.100000=26214.400000) /
Tamaño Vectores:262144 (4 B)
Tiempo:0.003490605 / Tamaño Vectores:262144
V3[262143](52428.700000+0.100000=52428.800000) /
Tamaño Vectores:524288 (4 B)
./sumavectores_script.sh: línea 6: 15945 Violación de segmento ('core' generado) ./SumaVectores $P
Tamaño Vectores:1048576 (4 B)
./sumavectores_script.sh: línea 6: 15947 Violación de segmento ('core' generado) ./SumaVectores $P
Tamaño Vectores:2097152 (4 B)
./sumavectores_script.sh: línea 6: 15949 Violación de segmento ('core' generado) ./SumaVectores $P
Tamaño Vectores:4194304 (4 B)
./sumavectores_script.sh: línea 6: 15951 Violación de segmento ('core' generado) ./SumaVectores $P
Tamaño Vectores:8388608 (4 B)
./sumavectores_script.sh: línea 6: 15953 Violación de segmento ('core' generado) ./SumaVectores $P
Tamaño Vectores:16777216 (4 B)
./sumavectores_script.sh: línea 6: 15955 Violación de segmento ('core' generado) ./SumaVectores $P
Tamaño Vectores:33554432 (4 B)
./sumavectores_script.sh: línea 6: 15957 Violación de segmento ('core' generado) ./SumaVectores $P
Tamaño Vectores:67108864 (4 B)
./sumavectores_script.sh: línea 6: 15959 Violación de segmento ('core' generado) ./SumaVectores $P
JavierRamirezPulido>sáb feb 29

```

Ejecucion para vectores locales.

6. En el código del Listado 1 se utiliza la función `clock_gettime()` para obtener el tiempo de ejecución del trozo de código que calcula la suma de vectores. El código se imprime la variable `ncgt`,

(a) ¿qué contiene esta variable?

RESPUESTA: Es una variable de tipo `double` que contiene la diferencia de tiempos entre el final y el inicio de la ejecución, siendo esta la combinación de los segundos y nanosegundos que tarda en calcularse la suma de los vectores (tiempo de ejecución).

(b) ¿en qué estructura de datos devuelve `clock_gettime()` la información de tiempo (indicar el tipo de estructura de datos, describir la estructura de datos, e indicar los tipos de datos que usa)?

RESPUESTA: Lo devuelve en:

```
Struct timespec{
    time_t tv_sec; ->Tiempo transcurrido en segundos
    long tv_nsec; ->Tiempo transcurrido en nanosegundos
}
```

(c) ¿qué información devuelve exactamente la función `clock_gettime()` en la estructura de datos descrita en el apartado (b)? ¿qué representan los valores numéricos que devuelve?

RESPUESTA: Almacena en el segundo argumento que se le pasa el instante de tiempo y devuelve 1 o 0. 1 si no se ha ejecutado la función correctamente y 0 si lo ha hecho todo como debería.

7. Rellenar una tabla como la Tabla 1 en una hoja de cálculo con los tiempos de ejecución del código del Listado 1 para vectores locales, globales y dinámicos. Obtener estos resultados usando scripts (partir del script que hay en el seminario). Debe haber una tabla para `atcgrid` y otra para su PC en la hoja de cálculo. En la columna “Bytes de un vector” hay que poner el total de bytes reservado para un vector. (NOTA: Se recomienda usar en la hoja de cálculo el mismo separador para decimales que usan los códigos al imprimir. Este separador se puede modificar en la hoja de cálculo.)

RESPUESTA:

```
Actividades Terminal
sáb 00:21
xavivi2000@xavivi2000: ~

JavierRamirezPulido@sáb feb 29 ./sumavectores_script.sh
Tamaño Vectores:65536 (4 B)
Tiempo:0.000876608 / Tamaño Vectores:65536 / V1[0]+V2[0]=V3[0](6553.600000+6553.600000=13107.200000) / / V1[65535]+V2[65535]=V3[65535](13107.100000+0.100000=13107.200000) /
Tamaño Vectores:131072 (4 B)
Tiempo:0.002003626 / Tamaño Vectores:131072 / V1[0]+V2[0]=V3[0](13107.200000+13107.200000=26214.400000) / / V1[131071]+V2[131071]=V3[131071](26214.300000+0.100000=26214.400000) /
Tamaño Vectores:262144 (4 B)
Tiempo:0.003490605 / Tamaño Vectores:262144 / V1[0]+V2[0]=V3[0](26214.400000+26214.400000=52428.800000) / / V1[262143]+V2[262143]=V3[262143](52428.700000+0.100000=52428.800000) /
Tamaño Vectores:524288 (4 B)
./sumavectores_script.sh: línea 6: 15945 Violación de segmento ('core' generado) ./SumaVectores $P
Tamaño Vectores:1048576 (4 B)
./sumavectores_script.sh: línea 6: 15947 Violación de segmento ('core' generado) ./SumaVectores $P
Tamaño Vectores:2097152 (4 B)
./sumavectores_script.sh: línea 6: 15949 Violación de segmento ('core' generado) ./SumaVectores $P
Tamaño Vectores:4194304 (4 B)
./sumavectores_script.sh: línea 6: 15951 Violación de segmento ('core' generado) ./SumaVectores $P
Tamaño Vectores:8388608 (4 B)
./sumavectores_script.sh: línea 6: 15953 Violación de segmento ('core' generado) ./SumaVectores $P
Tamaño Vectores:16777216 (4 B)
./sumavectores_script.sh: línea 6: 15955 Violación de segmento ('core' generado) ./SumaVectores $P
Tamaño Vectores:33554432 (4 B)
./sumavectores_script.sh: línea 6: 15957 Violación de segmento ('core' generado) ./SumaVectores $P
Tamaño Vectores:67108864 (4 B)
./sumavectores_script.sh: línea 6: 15959 Violación de segmento ('core' generado) ./SumaVectores $P
JavierRamirezPulido@sáb feb 29
```

Ejecucion en PC de vector local.

```

Actividades Terminal
sáb 00:31
xavivi2000@xavivi2000: ~

JavierRamirezPulido>sáb feb 29 ./sunavectores_script.sh
Tamaño Vectores:65536 (4 B)
Tiempo:0.001627689 / Tamaño Vectores:65536 / V1[0]+V2[0]=V3[0](6553.600000+6553.600000=13107.200000) / / V1[65535]+V2[65535]=V3[65535](13107.100000+0.100000=13107.200000) /
Tamaño Vectores:131072 (4 B)
Tiempo:0.001762465 / Tamaño Vectores:131072 / V1[0]+V2[0]=V3[0](13107.200000+13107.200000=26214.400000) / / V1[131071]+V2[131071]=V3[131071](26214.300000+0.100000=26214.400000) /
Tamaño Vectores:262144 (4 B)
Tiempo:0.003166931 / Tamaño Vectores:262144 / V1[0]+V2[0]=V3[0](26214.400000+26214.400000=52428.800000) / / V1[262143]+V2[262143]=V3[262143](52428.700000+0.100000=52428.800000) /
Tamaño Vectores:524288 (4 B)
Tiempo:0.007977144 / Tamaño Vectores:524288 / V1[0]+V2[0]=V3[0](52428.800000+52428.800000=104857.600000) / / V1[524287]+V2[524287]=V3[524287](104857.500000+0.100000=104857.600000) /
Tamaño Vectores:1048576 (4 B)
Tiempo:0.013338833 / Tamaño Vectores:1048576 / V1[0]+V2[0]=V3[0](104857.600000+104857.600000=209715.200000) / / V1[1048575]+V2[1048575]=V3[1048575](209715.100000+0.100000=209715.200000) /
Tamaño Vectores:2097152 (4 B)
Tiempo:0.022510472 / Tamaño Vectores:2097152 / V1[0]+V2[0]=V3[0](209715.200000+209715.200000=419430.400000) / / V1[2097151]+V2[2097151]=V3[2097151](419430.300000+0.100000=419430.400000) /
Tamaño Vectores:4194304 (4 B)
Tiempo:0.055899085 / Tamaño Vectores:4194304 / V1[0]+V2[0]=V3[0](419430.400000+419430.400000=838860.800000) / / V1[4194303]+V2[4194303]=V3[4194303](838860.700000+0.100000=838860.800000) /
Tamaño Vectores:8388608 (4 B)
Tiempo:0.098794843 / Tamaño Vectores:8388608 / V1[0]+V2[0]=V3[0](838860.800000+838860.800000=1677721.600000) / / V1[8388607]+V2[8388607]=V3[8388607](1677721.500000+0.100000=1677721.600000) /
Tamaño Vectores:16777216 (4 B)
Tiempo:0.235459141 / Tamaño Vectores:16777216 / V1[0]+V2[0]=V3[0](1677721.600000+1677721.600000=3355443.200000) / / V1[16777215]+V2[16777215]=V3[16777215](3355443.100000+0.100000=3355443.200000) /
Tamaño Vectores:33554432 (4 B)
Tiempo:0.411876206 / Tamaño Vectores:33554432 / V1[0]+V2[0]=V3[0](3355443.200000+3355443.200000=6710886.400000) / / V1[33554431]+V2[33554431]=V3[33554431](6710886.300000+0.100000=6710886.400000) /
Tamaño Vectores:67108864 (4 B)
Tiempo:0.358521370 / Tamaño Vectores:33554432 / V1[0]+V2[0]=V3[0](3355443.200000+3355443.200000=6710886.400000) / / V1[33554431]+V2[33554431]=V3[33554431](6710886.300000+0.100000=6710886.400000) /
JavierRamirezPulido>sáb feb 29 gcc -O2 SumaVectoresC.c -o SumaVectores -lrt
SumaVectoresC.c: In function 'main':
SumaVectoresC.c:45:32: warning: format '%u' expects argument of type 'unsigned int', but argument 3 has type 'long unsigned int' [-Wformat=]

```

Ejecucion en PC con vectores globales.

```

Actividades Terminal
sáb 00:31
xavivi2000@xavivi2000: ~

JavierRamirezPulido>sáb feb 29 ./sunavectores_script.sh
Tamaño Vectores:65536 (4 B)
Tiempo:0.001283573 / Tamaño Vectores:65536 / V1[0]+V2[0]=V3[0](6553.600000+6553.600000=13107.200000) / / V1[65535]+V2[65535]=V3[65535](13107.100000+0.100000=13107.200000) /
Tamaño Vectores:131072 (4 B)
Tiempo:0.001757881 / Tamaño Vectores:131072 / V1[0]+V2[0]=V3[0](13107.200000+13107.200000=26214.400000) / / V1[131071]+V2[131071]=V3[131071](26214.300000+0.100000=26214.400000) /
Tamaño Vectores:262144 (4 B)
Tiempo:0.003027025 / Tamaño Vectores:262144 / V1[0]+V2[0]=V3[0](26214.400000+26214.400000=52428.800000) / / V1[262143]+V2[262143]=V3[262143](52428.700000+0.100000=52428.800000) /
Tamaño Vectores:524288 (4 B)
Tiempo:0.008578521 / Tamaño Vectores:524288 / V1[0]+V2[0]=V3[0](52428.800000+52428.800000=104857.600000) / / V1[524287]+V2[524287]=V3[524287](104857.500000+0.100000=104857.600000) /
Tamaño Vectores:1048576 (4 B)
Tiempo:0.013035048 / Tamaño Vectores:1048576 / V1[0]+V2[0]=V3[0](104857.600000+104857.600000=209715.200000) / / V1[1048575]+V2[1048575]=V3[1048575](209715.100000+0.100000=209715.200000) /
Tamaño Vectores:2097152 (4 B)
Tiempo:0.022134136 / Tamaño Vectores:2097152 / V1[0]+V2[0]=V3[0](209715.200000+209715.200000=419430.400000) / / V1[2097151]+V2[2097151]=V3[2097151](419430.300000+0.100000=419430.400000) /
Tamaño Vectores:4194304 (4 B)
Tiempo:0.043759359 / Tamaño Vectores:4194304 / V1[0]+V2[0]=V3[0](419430.400000+419430.400000=838860.800000) / / V1[4194303]+V2[4194303]=V3[4194303](838860.700000+0.100000=838860.800000) /
Tamaño Vectores:8388608 (4 B)
Tiempo:0.086914224 / Tamaño Vectores:8388608 / V1[0]+V2[0]=V3[0](838860.800000+838860.800000=1677721.600000) / / V1[8388607]+V2[8388607]=V3[8388607](1677721.500000+0.100000=1677721.600000) /
Tamaño Vectores:16777216 (4 B)
Tiempo:0.179956898 / Tamaño Vectores:16777216 / V1[0]+V2[0]=V3[0](1677721.600000+1677721.600000=3355443.200000) / / V1[16777215]+V2[16777215]=V3[16777215](3355443.100000+0.100000=3355443.200000) /
Tamaño Vectores:33554432 (4 B)
Tiempo:0.354177358 / Tamaño Vectores:33554432 / V1[0]+V2[0]=V3[0](3355443.200000+3355443.200000=6710886.400000) / / V1[33554431]+V2[33554431]=V3[33554431](6710886.300000+0.100000=6710886.400000) /
Tamaño Vectores:67108864 (4 B)
Tiempo:0.875013248 / Tamaño Vectores:67108864 / V1[0]+V2[0]=V3[0](6710886.400000+6710886.400000=13421772.800000) / / V1[67108863]+V2[67108863]=V3[67108863](13421772.700000+0.100000=13421772.800000) /
JavierRamirezPulido>sáb feb 29

```

Ejecución en PC de vector dinámico.

Nº de Componentes	Bytes de un vector	Tiempo para vect. locales	Tiempo para vect. globales	Tiempo para vect. dinámicos
65536	524288	0.000876608	0.001627689	0.001283573
131072	1048576	0.002003626	0.001762465	0.001757881
262144	2097152	0.003490605	0.003166931	0.003027025
524288	4194304	0	0.007977144	0.008578521
1048576	8388608	0	0.013338833	0.013035048

2097152	16777216	0	0.022510472	0.022134136
4194304	33554432	0	0.055899085	0.043759359
8388608	67108864	0	0.098794843	0.086914224
16777216	134217728	0	0.235459141	0.179956898
33554432	268435456	0	0.411876206	0.354177358
67108864	536870912	0	0.358521370	0.875013248

TIEMPO EN PC.

```

Actividades Terminal
c3estudiante18@atcgrid:~/bp0/ejer7

xavivi2000@xavivi2000:~$ ssh -X c3estudiante18@atcgrid.ugr.es
^[[A^C
xavivi2000@xavivi2000:~$ ssh -X c3estudiante18@atcgrid.ugr.es
c3estudiante18@atcgrid.ugr.es's password:
Last login: Sat Feb 29 22:41:27 2020 from vpn-s241147.ugr.es
[c3estudiante18@atcgrid ~]$ export PS1="JavierRamirezPulido>"
JavierRamirezPulido>sáb feb 29 clear
JavierRamirezPulido>sáb feb 29 ls
bp0 C2 sess0 slurm-6147.out slurm-6162.out slurm-6246.out
JavierRamirezPulido>sáb feb 29 cd bp0/ejer7/
JavierRamirezPulido>sáb feb 29 ls
slurm-14075.out SumaVectoresLocal
slurm-14077.out sumavectores_script_dinamico.sh
slurm-14080.out sumavectores_script_global.sh
SumaVectoresdinamico sumavectores_script_local.sh
SumaVectoresglobal
JavierRamirezPulido>sáb feb 29 sbatch -p ac -n1 sumavectores_script_local.sh
Submitted batch job 14661
JavierRamirezPulido>sáb feb 29 cat slurm-14661.out
Tamaño Vectores:65536 (4 B)
Tiempo:0.000466043 / Tamaño Vectores:65536 / V1[0]+V2[0]=V3[0](6553.600000+6553.600000=13107.200000) / V1[65535]+V2[65535]=V3[65535](13107.100000+0.100000=13107.200000) /
Tamaño Vectores:131072 (4 B)
Tiempo:0.000938710 / Tamaño Vectores:131072 / V1[0]+V2[0]=V3[0](13107.200000+13107.200000=26214.400000) / V1[131071]+V2[131071]=V3[131071](26214.300000+0.100000=26214.400000) /
Tamaño Vectores:262144 (4 B)
Tiempo:0.001913339 / Tamaño Vectores:262144 / V1[0]+V2[0]=V3[0](26214.400000+26214.400000=52428.800000) / V1[262143]+V2[262143]=V3[262143](52428.700000+0.100000=52428.800000) /
/var/spool/slurmd/job14661/slurm_script: línea 6: 32674 Violación de segmento ('core' generado) ./SumaVectoresLocal $P
/var/spool/slurmd/job14661/slurm_script: línea 6: 32676 Violación de segmento ('core' generado) ./SumaVectoresLocal $P
/var/spool/slurmd/job14661/slurm_script: línea 6: 32680 Violación de segmento ('core' generado) ./SumaVectoresLocal $P
/var/spool/slurmd/job14661/slurm_script: línea 6: 32683 Violación de segmento ('core' generado) ./SumaVectoresLocal $P
/var/spool/slurmd/job14661/slurm_script: línea 6: 32685 Violación de segmento ('core' generado) ./SumaVectoresLocal $P
/var/spool/slurmd/job14661/slurm_script: línea 6: 32687 Violación de segmento ('core' generado) ./SumaVectoresLocal $P
/var/spool/slurmd/job14661/slurm_script: línea 6: 32689 Violación de segmento ('core' generado) ./SumaVectoresLocal $P
/var/spool/slurmd/job14661/slurm_script: línea 6: 32691 Violación de segmento ('core' generado) ./SumaVectoresLocal $P
JavierRamirezPulido>sáb feb 29

```

Ejecución en atcgrid de vector local.

```

Actividades Terminal
c3estudiante18@atcgrid:~/bp0/ejer7

JavierRamirezPulido>sáb feb 29 ls
slurm-14075.out slurm-14080.out SumaVectoresglobal sumavectores_script_dinamico.sh sumavectores_script_local.sh
slurm-14077.out SumaVectoresdinamico SumaVectoresLocal sumavectores_script_global.sh
JavierRamirezPulido>sáb feb 29 cat slurm-14075.out
Tamaño Vectores:65536 (4 B)
Tiempo:0.000390089 / Tamaño Vectores:65536 / V1[0]+V2[0]=V3[0](6553.600000+6553.600000=13107.200000) / V1[65535]+V2[65535]=V3[65535](13107.100000+0.100000=13107.200000) /
Tamaño Vectores:131072 (4 B)
Tiempo:0.000936638 / Tamaño Vectores:131072 / V1[0]+V2[0]=V3[0](13107.200000+13107.200000=26214.400000) / V1[131071]+V2[131071]=V3[131071](26214.300000+0.100000=26214.400000) /
Tamaño Vectores:262144 (4 B)
Tiempo:0.001741031 / Tamaño Vectores:262144 / V1[0]+V2[0]=V3[0](26214.400000+26214.400000=52428.800000) / V1[262143]+V2[262143]=V3[262143](52428.700000+0.100000=52428.800000) /
Tamaño Vectores:524288 (4 B)
Tiempo:0.002882777 / Tamaño Vectores:524288 / V1[0]+V2[0]=V3[0](52428.800000+52428.800000=104857.600000) / V1[524287]+V2[524287]=V3[524287](104857.500000+0.100000=104857.600000) /
Tamaño Vectores:1048576 (4 B)
Tiempo:0.005537820 / Tamaño Vectores:1048576 / V1[0]+V2[0]=V3[0](104857.600000+104857.600000=209715.200000) / V1[1048575]+V2[1048575]=V3[1048575](209715.100000+0.100000=209715.200000) /
Tamaño Vectores:2097152 (4 B)
Tiempo:0.010104788 / Tamaño Vectores:2097152 / V1[0]+V2[0]=V3[0](209715.200000+209715.200000=419430.400000) / V1[2097151]+V2[2097151]=V3[2097151](419430.300000+0.100000=419430.400000) /
Tamaño Vectores:4194304 (4 B)
Tiempo:0.018697845 / Tamaño Vectores:4194304 / V1[0]+V2[0]=V3[0](419430.400000+419430.400000=838860.800000) / V1[4194303]+V2[4194303]=V3[4194303](838860.700000+0.100000=838860.800000) /
Tamaño Vectores:8388608 (4 B)
Tiempo:0.034042194 / Tamaño Vectores:8388608 / V1[0]+V2[0]=V3[0](838860.800000+838860.800000=1677721.600000) / V1[8388607]+V2[8388607]=V3[8388607](1677721.500000+0.100000=1677721.600000) /
Tamaño Vectores:16777216 (4 B)
Tiempo:0.065334338 / Tamaño Vectores:16777216 / V1[0]+V2[0]=V3[0](1677721.600000+1677721.600000=3355443.200000) / V1[16777215]+V2[16777215]=V3[16777215](3355443.100000+0.100000=3355443.200000) /
Tamaño Vectores:33554432 (4 B)
Tiempo:0.127241262 / Tamaño Vectores:33554432 / V1[0]+V2[0]=V3[0](3355443.200000+3355443.200000=6710886.400000) / V1[33554431]+V2[33554431]=V3[33554431](6710886.300000+0.100000=6710886.400000) /
Tamaño Vectores:67108864 (4 B)
Tiempo:0.251606304 / Tamaño Vectores:67108864 / V1[0]+V2[0]=V3[0](6710886.400000+6710886.400000=13421772.800000) / V1[67108863]+V2[67108863]=V3[67108863](13421772.700000+0.100000=13421772.800000) /
JavierRamirezPulido>sáb feb 29

```

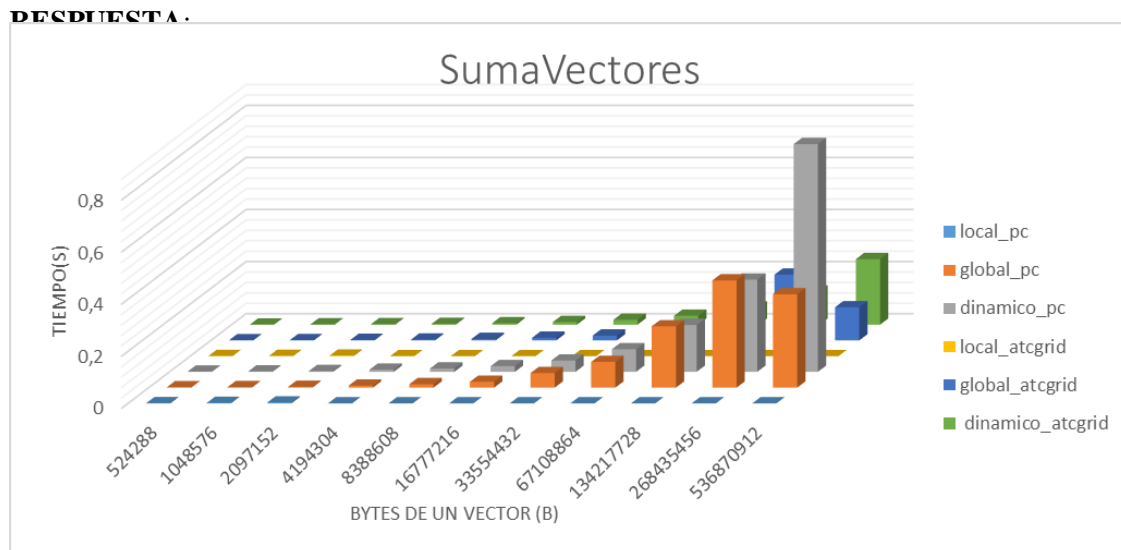
Ejecución en atcgrid de vector dinamico.

Ejecución en atcgrid de vector global.

Nº de Componentes	Bytes de un vector	Tiempo para vect. locales	Tiempo para vect. globales	Tiempo para vect. dinámicos
65536	524288	0.000465898	0.000502096	0.000390089
131072	1048576	0.000940067	0.000929968	0.000936638
262144	2097152	0.001909479	0.001844774	0.001741031
524288	4194304	0	0.002965241	0.002882777
1048576	8388608	0	0.005555266	0.005537820
2097152	16777216	0	0.010195113	0.010104788
4194304	33554432	0	0.018552039	0.018697845
8388608	67108864	0	0.035116335	0.034042194
16777216	134217728	0	0.064015600	0.065334338
33554432	268435456	0	0.127085610	0.127241262
67108864	536870912	0	0.252533318	0.251606304

EN ATCGRID

8. Con ayuda de la hoja de cálculo representar **en una misma gráfica** los tiempos de ejecución obtenidos en atcgrid y en su PC para vectores locales, globales y dinámicos (eje y) en función del tamaño en bytes de un vector (por tanto, los valores de la segunda columna de la tabla, que están en escala logarítmica, deben estar en el eje x). Utilizar escala logarítmica en el eje de ordenadas (eje y). ¿Hay diferencias en los tiempos de ejecución?



Si la hay, por lo general los tiempos en atcgrid son más pequeños que los tiempos de ejecución en mi pc, por lo que son más rápidos.

9. (a) Cuando se usan vectores locales, ¿se obtiene error para alguno de los tamaños?, ¿a qué cree que es debido lo que ocurre? (Incorporar volcados de pantalla como se indica en las normas de prácticas)

RESPUESTA: Sí, en el atcgrid se obtiene un error a partir del tamaño 524288 (este es el primero en dar error) y en el PC se obtiene un error a partir del mismo debido a que se accede a una posición de memoria no permitida por superar el tamaño de pila. Como vemos, la pila tiene un tamaño menor que $524288 * 8B$ (vector de double).

- (b) Cuando se usan vectores globales, ¿se obtiene error para alguno de los tamaños?, ¿a qué cree que es debido lo que ocurre? (Incorporar volcados de pantalla como se indica en las normas de prácticas)

RESPUESTA: En este caso, ni en el PC ni en atcgrid se obtiene error, calcula para todos los tamaños sin problema. Esto ocurre porque el tamaño de la pila no pone límites a la longitud del vector, además, existe una condición según la cual si N es más grande que el valor más alto (33554432), le asigna este valor y nunca lo supera.

- (c) Cuando se usan vectores dinámicos, ¿se obtiene error para alguno de los tamaños?, ¿a qué cree que es debido lo que ocurre? (Incorporar volcados de pantalla como se indica en las normas de prácticas)

RESPUESTA: Tampoco existe error para ningún tamaño ni en PC ni en atcgrid. Al usar memoria dinámica (la cual se puede volver a usar), la función malloc reserva un tamaño N para el vector, por lo que no existe problema con la pila.

10. (a) ¿Cuál es el máximo valor que se puede almacenar en la variable N teniendo en cuenta su tipo? Razonar respuesta.

RESPUESTA: Al ser de tipo unsigned int, su tamaño máximo es de 4B (32 bits).
 $2^{31}-1 = 4294967295 \rightarrow$ Tamaño.

- (b) Modificar el código fuente C (en el PC) para que el límite de los vectores cuando se declaran como variables globales sea igual al máximo número que se puede almacenar en la variable N y generar el ejecutable. ¿Qué ocurre? ¿A qué es debido? (Incorporar volcados de pantalla que muestren lo que ocurre)

RESPUESTA: Ocurre un error de compilación que no permite crear el ejecutable directamente. Se debe a que al declarar los vectores, el tamaño máximo de N (anteriormente 33554432) se multiplica por 8B, sobrepasando el tamaño de memoria.

```

Actividades Terminal
sáb 23:52
xavivi2000@xavivi2000: ~

Archivo Editar Ver Buscar Terminal Ayuda
JavierRamirezPulido>sáb feb 29 gcc -O2 SumaVectores.c -o SumaVectores -lrt
SumaVectores.c: In function 'main':
SumaVectores.c:45:32: warning: format '%u' expects argument of type 'unsigned int', but argument 3 has type 'long unsigned int' [-Wformat=]
printf("Tamaño Vectores:%u (%u B)\n",N, sizeof(unsigned int));
                           ^~
                           %lu
/tmp/ccx29nYJ.o: En la función 'main':
SumaVectores.c:(.text.startup+0x76): reubicación truncada para ajustar: R_X86_64_PC32 contra el símbolo 'v2' definido en la sección COMMON en
/tmp/ccx29nYJ.o
SumaVectores.c:(.text.startup+0xc9): reubicación truncada para ajustar: R_X86_64_PC32 contra el símbolo 'v3' definido en la sección COMMON en
/tmp/ccx29nYJ.o
collect2: error: ld returned 1 exit status
JavierRamirezPulido>sáb feb 29

```

Entrega del trabajo

Leer lo indicado en las normas de prácticas sobre la entrega del trabajo del bloque práctico en SWAD.

Listado 1 . Código C que suma dos vectores

```

/* SumaVectores.c
Suma de dos vectores: v3 = v1 + v2

Para compilar usar (-lrt: real time library, no todas las versiones de gcc necesitan que se incluya -lrt):
gcc -O2 SumaVectores.c -o SumaVectores -lrt
gcc -O2 -S SumaVectores.c -lrt //para generar el código ensamblador

Para ejecutar use: SumaVectoresC longitud
*/

#include <stdlib.h> // biblioteca con funciones atoi(), malloc() y free()
#include <stdio.h> // biblioteca donde se encuentra la función printf()
#include <time.h> // biblioteca donde se encuentra la función clock_gettime()

//Sólo puede estar definida una de las tres constantes VECTOR_ (sólo uno de los ...
//tres defines siguientes puede estar descomentado):
//#define VECTOR_LOCAL // descomentar para que los vectores sean variables ...
// locales (si se supera el tamaño de la pila se ...
// generará el error "Violación de Segmento")
//#define VECTOR_GLOBAL// descomentar para que los vectores sean variables ...
// globales (su longitud no estará limitada por el ...
// tamaño de la pila del programa)
#define VECTOR_DYNAMIC // descomentar para que los vectores sean variables ...

```

```

// dinámicas (memoria reutilizable durante la ejecución)
#ifdef VECTOR_GLOBAL
#define MAX 33554432 // = 2^25
double v1[MAX], v2[MAX], v3[MAX];
#endif

int main(int argc, char** argv) {

    int i;
    struct timespec cgt1, cgt2; double ncgt; // para tiempo de ejecución

    // Leer argumento de entrada (nº de componentes del vector)
    if (argc < 2) {
        printf("Faltan nº componentes del vector\n");
        exit(-1);
    }

    unsigned int N = atoi(argv[1]); // Máximo N = 2^32-1 = 4294967295 (sizeof(unsigned int) = 4 B)
#ifdef VECTOR_LOCAL
    double v1[N], v2[N], v3[N]; // Tamaño variable local en tiempo de ejecución ...
                                // disponible en C a partir de actualización C99
#endif
#ifdef VECTOR_GLOBAL
    if (N > MAX) N = MAX;
#endif
#ifdef VECTOR_DYNAMIC
    double *v1, *v2, *v3;
    v1 = (double*) malloc(N * sizeof(double)); // malloc necesita el tamaño en bytes
    v2 = (double*) malloc(N * sizeof(double)); // si no hay espacio suficiente malloc devuelve NULL
    v3 = (double*) malloc(N * sizeof(double));
    if ( (v1 == NULL) || (v2 == NULL) || (v3 == NULL) ) {
        printf("Error en la reserva de espacio para los vectores\n");
        exit(-2);
    }
#endif

    // Inicializar vectores
    for (i = 0; i < N; i++) {
        v1[i] = N * 0.1 + i * 0.1; v2[i] = N * 0.1 - i * 0.1; // los valores dependen de N
    }

    clock_gettime(CLOCK_REALTIME, &cgt1);
    // Calcular suma de vectores
    for (i = 0; i < N; i++)
        v3[i] = v1[i] + v2[i];

    clock_gettime(CLOCK_REALTIME, &cgt2);
    ncgt = (double) (cgt2.tv_sec - cgt1.tv_sec) +
           (double) ((cgt2.tv_nsec - cgt1.tv_nsec) / (1.0e+9));

    // Imprimir resultado de la suma y el tiempo de ejecución
    if (N < 10) {

```



```

printf("Tiempo(seg.):%11.9f\t / Tamaño Vectores:%lu\n", ncgt, N);
for(i=0; i<N; i++)
    printf("/ V1[%d]+V2[%d]=V3[%d] (%8.6f+%8.6f=%8.6f) /\n",
           i, i, i, v1[i], v2[i], v3[i]);
}
else
    printf("Tiempo(seg.):%11.9f\t / Tamaño Vectores:%u\t/ V1[0]+V2[0]=V3[0] (%8.6f+%8.6f=%8.6f) /\n",
           V1[0]+V2[0]=V3[0] (%8.6f+%8.6f=%8.6f) /\n",
           ncgt, N, v1[0], v2[0], v3[0], N-1, N-1, N-1, v1[N-1], v2[N-1], v3[N-1]);

#ifdef VECTOR_DYNAMIC
free(v1); // libera el espacio reservado para v1
free(v2); // libera el espacio reservado para v2
free(v3); // libera el espacio reservado para v3
#endif
return 0;
}

```