2° curso / 2° cuatr. **Grado Ing. Inform.** 

Doble Grado Ing. Inform. y Mat.

\$lscpu

# Arquitectura de Computadores (AC)

Cuaderno de prácticas. Bloque Práctico 4. Optimización de código

Estudiante (nombre y apellidos):

Grupo de prácticas:

Fecha de entrega:

Fecha evaluación en clase:

**Denominación de marca del chip de procesamiento o procesador (se encuentra en /proc/cpuinfo):** Intel(R) Core(TM) i5-7200U CPU @ 2.50GHz

Sistema operativo utilizado: Ubuntu 18.04 LTS

[JoseJavierAlonsoRamos jjavier98@jjavier98-Lenovo-ideapad-310-15IKB:~] 2018-05-18 viernes

Versión de gcc utilizada: gcc (Ubuntu 7.3.0-16ubuntu3) 7.3.0

Volcado de pantalla que muestre lo que devuelve 1scpu en la máquina en la que ha tomado las medidas

```
Arquitectura:
                                                                             x86 64
modo(s) de operación de las CPUs:
Orden de los bytes:
                                                                            32-bit, 64-bit
Little Endian
CPU(s):
Lista de la(s) CPU(s) en línea:
Hilo(s) de procesamiento por núcleo:
Núcleo(s) por «socket»:
«Socket(s)
Modo(s) NUMA:
ID de fabricante:
                                                                             GenuineIntel
Familia de CPU:
                                                                             142
Modelo:
Nombre del modelo:
Revisión:
                                                                             Intel(R) Core(TM) i5-7200U CPU @ 2.50GHz
                                                                            602.632
CPU MHz:
                                                                             2500,0000
CPU MHz máx.:
                                                                             400,0000
CPU MHz mín.:
BogoMIPS:
                                                                             5424.00
Virtualización:
Caché L1d:
Caché L1i:
Caché L2:
                                                                             256K
Caché L3:
                                                                             3072K
CPU(s) del nodo NUMA 0:
                                                                            0-3
fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush dts acpi mmx fxsr sse sse2 s s ht tm pbe syscall nx pdpe1gb rdtscp lm constant_tsc art arch_perfmon pebs bts rep_good nopl xtopology nonstop_tsc cpuid aperfmperf tsc_known_fr eq pni pclmulqdq dtes64 monitor ds_cpl vmx est tm2 ssse3 sdbg fma cx16 xtpr pdcm pcid sse4_1 sse4_2 x2apic movbe popcnt tsc_deadline_timer aes xs ave avx f16c rdrand lahf_lm abm 3dnowprefetch cpuid_fault invpcid_single pti tpr_shadow vnmi flexpriority ept vpid fsgsbase tsc_adjust bmi1 avx2 smep bmi2 erms invpcid mpx rdseed adx smap clflushopt intel_pt xsaveopt xsavec xgetbv1 xsaves ibpb ibrs stibp dtherm arat pln pts hwp_notify
hwp_act_window hwp_epp
```

- 1. Para el núcleo que se muestra en el Figura 1, y para un programa que implemente la multiplicación de matrices (use variables globales):
  - 1.1 Modifique el código C para reducir el tiempo de ejecución del mismo. Justifique los tiempos obtenidos (use -O2) a partir de la modificación realizada. Incorpore los códigos modificados en el cuaderno.
  - 1.2 Genere los códigos en ensamblador con -O2 para el original y dos códigos modificados obtenidos en el punto anterior (incluido el que supone menor tiempo de ejecución) e incorpórelos al cuaderno de prácticas. Destaque las diferencias entre ellos en el código ensamblador.

1.3 (Ejercicio EXTRA) Intente mejorar los resultados obtenidos transformando el código ensamblador del programa para el que se han conseguido las mejores prestaciones de tiempo

**Figura 1** . Código C++ que suma dos vectores

```
struct {
    int a;
    int b;
} s[5000];

main()
{
    for (ii=0; ii<40000;ii++) {
        X1=0; X2=0;
        for(i=0; i<5000;i++) X1+=2*s[i].a+ii;
        for(i=0; i<5000;i++) X2+=3*s[i].b-ii;
        if (X1<X2) R[ii]=X1 else R[ii]=X2;
    }
}</pre>
```

# A) MULTIPLICACIÓN DE MATRICES:

CAPTURA CÓDIGO FUENTE: pmm-secuencial.c

1.1. MODIFICACIONES REALIZADAS (al menos dos modificaciones):

Modificación a) –explicación-: desenrrollado del primer bucle Modificación b) –explicación-: desenrrollado del primer y segundo bucle Modificación c) –explicación-: desenrrollado de todos los bucles

### 1.1. CÓDIGOS FUENTE MODIFICACIONES

a) Captura de pmm-secuencial-modificado\_a.c

```
 \begin{array}{l} \text{if}(\text{N\%5==0}) \\ \{ \\ \text{for}(\text{int i=0; i<N; i+=5}) \\ \text{for}(\text{int j=0; j<N; j++}) \\ \text{for}(\text{int z=0; z<N; z++}) \\ \{ \\ \text{m\_res[i][j] += m1[i][z] * m2[z][j];} \\ \text{m\_res[i+1][j] += m1[i+1][z] * m2[z][j];} \\ \text{m\_res[i+2][j] += m1[i+2][z] * m2[z][j];} \\ \text{m\_res[i+3][j] += m1[i+3][z] * m2[z][j];} \\ \text{m\_res[i+4][j] += m1[i+4][z] * m2[z][j];} \\ \} \\ \} \end{array}
```

```
else
               if(N\%4==0)
                       for(int i=0; i<N; i+=4)
                    for(int j=0; j<N; j++)
                      for(int z=0; z<N; z++)
                        m_{res[i][j]} += m1[i][z] * m2[z][j];
                        m res[i+1][i] += m1[i+1][z] * m2[z][i];
                        m res[i+2][i] += m1[i+2][z] * m2[z][i];
                        m res[i+3][i] += m1[i+3][z] * m2[z][i];
               }
               else
                       if(N\%3==0)
                              for(int i=0; i< N; i+=3)
                           for(int j=0; j<N; j++)
                              for(int z=0; z<N; z++)
                              {
                                m_{res[i][j]} += m1[i][z] * m2[z][j];
                                m_{res[i+1][j]} += m1[i+1][z] * m2[z][j];
                                m_{res[i+2][j]} += m1[i+2][z] * m2[z][j];
                       }
                       else
                              for(int i=0; i<N; ++i)
                           for(int j=0; j<N; ++j)
                              for(int z=0; z< N; ++z)
                                m_{res[i][j]} += m1[i][z] * m2[z][j];
b) ...
if(N\%5==0)
       {
               for(int i=0; i< N; i+=5)
            for(int j=0; j<N; j+=5)
              for(int z=0; z<N; z++)
                 m res[i][i] += m1[i][z] * m2[z][i];
                 m_{res[i+1][j]} += m1[i+1][z] * m2[z][j];
                 m_{res}[i+2][j] += m1[i+2][z] * m2[z][j];
                 m_{res[i+3][j]} += m1[i+3][z] * m2[z][j];
                 m_{res[i+4][j]} += m1[i+4][z] * m2[z][j];
                 m_{res[i][j+1]} += m1[i][z] * m2[z][j+1];
                 m_{res}[i+1][j+1] += m1[i+1][z] * m2[z][j+1];
                 m_{res}[i+2][j+1] += m1[i+2][z] * m2[z][j+1];
                 m_{res}[i+3][j+1] += m1[i+3][z] * m2[z][j+1];
                 m_{res}[i+4][j+1] += m1[i+4][z] * m2[z][j+1];
                 m_{res[i][j+2]} += m1[i][z] * m2[z][j+2];
                 m_{res}[i+1][j+2] += m1[i+1][z] * m2[z][j+2];
                 m_{res[i+2][j+2]} += m1[i+2][z] * m2[z][j+2];
```

 $m_{res}[i+3][j+2] += m1[i+3][z] * m2[z][j+2];$  $m_{res}[i+4][j+2] += m1[i+4][z] * m2[z][j+2];$ 

```
m_{res[i][j+3]} += m1[i][z] * m2[z][j+3];
          m_{res[i+1][j+3]} += m1[i+1][z] * m2[z][j+3];
          m_{res}[i+2][j+3] += m1[i+2][z] * m2[z][j+3];
          m_{res}[i+3][j+3] += m1[i+3][z] * m2[z][j+3];
          m_{res}[i+4][j+3] += m1[i+4][z] * m2[z][j+3];
          m_{res[i][j+4]} += m1[i][z] * m2[z][j+4];
          m \operatorname{res}[i+1][j+4] += m1[i+1][z] * m2[z][j+4];
          m \operatorname{res}[i+2][i+4] += m1[i+2][z] * m2[z][i+4];
          m res[i+3][j+4] += m1[i+3][z] * m2[z][j+4];
          m_{res[i+4][j+4]} += m1[i+4][z] * m2[z][j+4];
else
        if(N\%4==0)
                for(int i=0; i< N; i+=4)
             for(int j=0; j<N; j+=4)
                for(int z=0; z< N; z++)
                {
                  m_{res[i][j]} += m1[i][z] * m2[z][j];
                  m_{res[i+1][j]} += m1[i+1][z] * m2[z][j];
                  m res[i+2][j] += m1[i+2][z] * m2[z][j];
                  m_{res[i+3][j]} += m1[i+3][z] * m2[z][j];
                  m_{res[i][j+1]} += m1[i][z] * m2[z][j+1];
                  m_{res}[i+1][j+1] += m1[i+1][z] * m2[z][j+1];
                  m_{res[i+2][j+1]} += m1[i+2][z] * m2[z][j+1];
                  m res[i+3][j+1] += m1[i+3][z] * m2[z][j+1];
                  m_{res[i][j+2]} += m1[i][z] * m2[z][j+2];
                  m_{res}[i+1][j+2] += m1[i+1][z] * m2[z][j+2];
                  m_{res}[i+2][j+2] += m1[i+2][z] * m2[z][j+2];
                  m_{res}[i+3][j+2] += m1[i+3][z] * m2[z][j+2];
                  m_{res[i][j+3]} += m1[i][z] * m2[z][j+3];
                  m_{res[i+1][j+3]} += m1[i+1][z] * m2[z][j+3];
                  m_{res}[i+2][j+3] += m1[i+2][z] * m2[z][j+3];
                  m_{res[i+3][j+3]} += m1[i+3][z] * m2[z][j+3];
        else
                if(N\%3==0)
                {
                        for(int i=0; i< N; i+=3)
                     for(int j=0; j<N; j+=3)
                        for(int z=0; z<N; z++)
                          m_{res[i][j]} += m1[i][z] * m2[z][j];
                          m_{res}[i+1][j] += m1[i+1][z] * m2[z][j];
                          m_{res[i+2][j]} += m1[i+2][z] * m2[z][j];
                          m_{res[i][j+1]} += m1[i][z] * m2[z][j+1];
                          m_{res[i+1][j+1]} += m1[i+1][z] * m2[z][j+1];
                          m_{res}[i+2][j+1] += m1[i+2][z] * m2[z][j+1];
                          m_{res[i][j+2]} += m1[i][z] * m2[z][j+2];
                          m_{res}[i+1][j+2] += m1[i+1][z] * m2[z][j+2];
                          m_{res}[i+2][j+2] += m1[i+2][z] * m2[z][j+2];
```

```
}
                       }
                       else
                               for(int i=0; i< N; ++i)
                            for(int j=0; j<N; ++j)
                              for(int z=0; z< N; ++z)
                                m_{res[i][j]} += m1[i][z] * m2[z][j];
c)
if(N\%5==0)
        {
               for(int i=0; i< N; i+=5)
            for(int j=0; j<N; j+=5)
               for(int z=0; z<N; z+=5)
                 m_{res[i][j]} += m1[i][z] * m2[z][j];
                 m_{res[i+1][j]} += m1[i+1][z] * m2[z][j];
                 m_{res[i+2][j]} += m1[i+2][z] * m2[z][j];
                 m_{res[i+3][j]} += m1[i+3][z] * m2[z][j];
                 m res[i+4][j] += m1[i+4][z] * m2[z][j];
                 m_{res[i][j+1]} += m1[i][z] * m2[z][j+1];
                 m_{res[i+1][j+1]} += m1[i+1][z] * m2[z][j+1];
                 m_{res}[i+2][j+1] += m1[i+2][z] * m2[z][j+1];
                 m_{res[i+3][j+1]} += m1[i+3][z] * m2[z][j+1];
                 m_{res}[i+4][j+1] += m1[i+4][z] * m2[z][j+1];
                 m_{res[i][j+2]} += m1[i][z] * m2[z][j+2];
                 m_{res}[i+1][j+2] += m1[i+1][z] * m2[z][j+2];
                 m_{res}[i+2][j+2] += m1[i+2][z] * m2[z][j+2];
                 m_{res}[i+3][j+2] += m1[i+3][z] * m2[z][j+2];
                 m_{res}[i+4][j+2] += m1[i+4][z] * m2[z][j+2];
                 m_{res[i][j+3]} += m1[i][z] * m2[z][j+3];
                 m_{res[i+1][j+3]} += m1[i+1][z] * m2[z][j+3];
                 m_{res[i+2][j+3]} += m1[i+2][z] * m2[z][j+3];
                 m_{res}[i+3][j+3] += m1[i+3][z] * m2[z][j+3];
                 m_{res[i+4][j+3]} += m1[i+4][z] * m2[z][j+3];
                 m_{res[i][j+4]} += m1[i][z] * m2[z][j+4];
                 m_{res}[i+1][j+4] += m1[i+1][z] * m2[z][j+4];
                 m_{res}[i+2][j+4] += m1[i+2][z] * m2[z][j+4];
                 m_{res}[i+3][j+4] += m1[i+3][z] * m2[z][j+4];
                 m_{res}[i+4][j+4] += m1[i+4][z] * m2[z][j+4];
                 //
                 m_{res[i][j]} += m1[i][z+1] * m2[z+1][j];
                 m_{res}[i+1][j] += m1[i+1][z+1] * m2[z+1][j];
                 m_{res}[i+2][j] += m1[i+2][z+1] * m2[z+1][i];
                 m_{res[i+3][j]} += m1[i+3][z+1] * m2[z+1][j];
                 m_{res}[i+4][j] += m1[i+4][z+1] * m2[z+1][j];
                 m_{res[i][j+1]} += m1[i][z+1] * m2[z+1][j+1];
                 m_{res}[i+1][j+1] += m1[i+1][z+1] * m2[z+1][j+1];
```

```
m_{res}[i+2][j+1] += m1[i+2][z+1] * m2[z+1][j+1];
m_{res[i+3][j+1]} += m1[i+3][z+1] * m2[z+1][j+1];
m_{res[i+4][j+1]} += m1[i+4][z+1] * m2[z+1][j+1];
m_{res[i][j+2]} += m1[i][z+1] * m2[z+1][j+2];
m_{res}[i+1][j+2] += m1[i+1][z+1] * m2[z+1][j+2];
m_{res}[i+2][j+2] += m1[i+2][z+1] * m2[z+1][j+2];
m_{res}[i+3][j+2] += m1[i+3][z+1] * m2[z+1][j+2];
m \operatorname{res}[i+4][i+2] += m1[i+4][z+1] * m2[z+1][i+2];
m res[i][i+3] += m1[i][z+1] * m2[z+1][i+3];
m_{res[i+1][j+3]} += m1[i+1][z+1] * m2[z+1][i+3];
m_{res}[i+2][j+3] += m1[i+2][z+1] * m2[z+1][j+3];
m_{res[i+3][j+3]} += m1[i+3][z+1] * m2[z+1][j+3];
m_{res[i+4][j+3]} = m1[i+4][z+1] * m2[z+1][i+3];
m_{res[i][j+4]} += m1[i][z+1] * m2[z+1][j+4];
m_{res}[i+1][j+4] += m1[i+1][z+1] * m2[z+1][j+4];
m_{res}[i+2][j+4] += m1[i+2][z+1] * m2[z+1][j+4];
m_{res}[i+3][j+4] += m1[i+3][z+1] * m2[z+1][j+4];
m_{res}[i+4][j+4] += m1[i+4][z+1] * m2[z+1][j+4];
m_{res[i][j]} += m1[i][z+2] * m2[z+2][j];
m_{res[i+1][j]} += m1[i+1][z+2] * m2[z+2][i];
m_{res[i+2][j]} += m1[i+2][z+2] * m2[z+2][j];
m_{res}[i+3][j] += m1[i+3][z+2] * m2[z+2][j];
m_{res[i+4][j]} += m1[i+4][z+2] * m2[z+2][j];
m res[i][j+1] += m1[i][z+2] * m2[z+2][j+1];
m_{res}[i+1][j+1] += m1[i+1][z+2] * m2[z+2][j+1];
m_{res}[i+2][j+1] += m1[i+2][z+2] * m2[z+2][j+1];
m_{res}[i+3][j+1] += m1[i+3][z+2] * m2[z+2][j+1];
m_{res}[i+4][j+1] += m1[i+4][z+2] * m2[z+2][j+1];
m_{res[i][j+2]} += m1[i][z+2] * m2[z+2][j+2];
m_{res}[i+1][j+2] += m1[i+1][z+2] * m2[z+2][j+2];
m_{res}[i+2][j+2] += m1[i+2][z+2] * m2[z+2][j+2];
m_{res[i+3][j+2]} += m1[i+3][z+2] * m2[z+2][j+2];
m \operatorname{res}[i+4][j+2] += m1[i+4][z+2] * m2[z+2][j+2];
m \operatorname{res}[i][j+3] += m1[i][z+2] * m2[z+2][j+3];
m_{res}[i+1][j+3] += m1[i+1][z+2] * m2[z+2][j+3];
m_{res}[i+2][j+3] += m1[i+2][z+2] * m2[z+2][j+3];
m_{res}[i+3][j+3] += m1[i+3][z+2] * m2[z+2][j+3];
m_{res}[i+4][j+3] += m1[i+4][z+2] * m2[z+2][j+3];
m_{res[i][j+4]} += m1[i][z+2] * m2[z+2][j+4];
m_{res}[i+1][j+4] += m1[i+1][z+2] * m2[z+2][j+4];
m_{res[i+2][j+4]} += m1[i+2][z+2] * m2[z+2][i+4];
m_{res}[i+3][j+4] += m1[i+3][z+2] * m2[z+2][j+4];
m_{res}[i+4][j+4] += m1[i+4][z+2] * m2[z+2][j+4];
m_{res[i][j]} += m1[i][z+3] * m2[z+3][j];
m_{res}[i+1][j] += m1[i+1][z+3] * m2[z+3][j];
m_{res}[i+2][j] += m1[i+2][z+3] * m2[z+3][j];
m_{res}[i+3][j] += m1[i+3][z+3] * m2[z+3][j];
```

 $m_{res[i+4][j]} += m1[i+4][z+3] * m2[z+3][j];$ 

```
m_{res[i][j+1]} += m1[i][z+3] * m2[z+3][j+1];
                 m \operatorname{res}[i+1][j+1] += m1[i+1][z+3] * m2[z+3][j+1];
                 m_{res}[i+2][j+1] += m1[i+2][z+3] * m2[z+3][j+1];
                 m_{res}[i+3][j+1] += m1[i+3][z+3] * m2[z+3][j+1];
                 m_{res}[i+4][j+1] += m1[i+4][z+3] * m2[z+3][j+1];
                 m \operatorname{res}[i][i+2] += m1[i][z+3] * m2[z+3][i+2];
                 m \operatorname{res}[i+1][i+2] += m1[i+1][z+3] * m2[z+3][i+2];
                 m_{res}[i+2][j+2] += m1[i+2][z+3] * m2[z+3][i+2]:
                 m_{res}[i+3][j+2] += m1[i+3][z+3] * m2[z+3][j+2];
                 m \operatorname{res}[i+4][j+2] += m1[i+4][z+3] * m2[z+3][j+2];
                 m_{res}[i][j+3] += m1[i][z+3] * m2[z+3][j+3];
                 m_{res[i+1][j+3]} += m1[i+1][z+3] * m2[z+3][j+3];
                 m \operatorname{res}[i+2][j+3] += m1[i+2][z+3] * m2[z+3][j+3];
                 m_{res}[i+3][j+3] += m1[i+3][z+3] * m2[z+3][j+3];
                 m_{res}[i+4][j+3] += m1[i+4][z+3] * m2[z+3][j+3];
                 m_{res[i][j+4]} += m1[i][z+3] * m2[z+3][j+4];
                 m_{res[i+1][j+4]} += m1[i+1][z+3] * m2[z+3][j+4];
                 m_{res}[i+2][j+4] += m1[i+2][z+3] * m2[z+3][j+4];
                 m \operatorname{res}[i+3][j+4] += m1[i+3][z+3] * m2[z+3][j+4];
                 m_{res}[i+4][j+4] += m1[i+4][z+3] * m2[z+3][j+4];
                 m_{res[i][j]} += m1[i][z+4] * m2[z+4][j];
                 m \operatorname{res}[i+1][j] += m1[i+1][z+4] * m2[z+4][j];
                 m_{res[i+2][j]} += m1[i+2][z+4] * m2[z+4][j];
                 m_{res}[i+3][j] += m1[i+3][z+4] * m2[z+4][j];
                 m_{res}[i+4][j] += m1[i+4][z+4] * m2[z+4][j];
                 m_{res}[i][j+1] += m1[i][z+4] * m2[z+4][j+1];
                 m_{res}[i+1][j+1] += m1[i+1][z+4] * m2[z+4][j+1];
                 m_{res}[i+2][j+1] += m1[i+2][z+4] * m2[z+4][j+1];
                 m_{res}[i+3][j+1] += m1[i+3][z+4] * m2[z+4][j+1];
                 m_{res}[i+4][j+1] += m1[i+4][z+4] * m2[z+4][j+1];
                 m \operatorname{res}[i][i+2] += m1[i][z+4] * m2[z+4][i+2];
                 m_{res[i+1][j+2]} += m1[i+1][z+4] * m2[z+4][j+2];
                 m \operatorname{res}[i+2][j+2] += m1[i+2][z+4] * m2[z+4][j+2];
                 m_{res}[i+3][j+2] += m1[i+3][z+4] * m2[z+4][j+2];
                 m_{res}[i+4][j+2] += m1[i+4][z+4] * m2[z+4][j+2];
                 m_{res[i][j+3]} += m1[i][z+4] * m2[z+4][j+3];
                 m_{res}[i+1][j+3] += m1[i+1][z+4] * m2[z+4][j+3];
                 m_{res}[i+2][j+3] += m1[i+2][z+4] * m2[z+4][j+3];
                 m_{res}[i+3][j+3] += m1[i+3][z+4] * m2[z+4][j+3];
                 m_{res[i+4][j+3]} = m1[i+4][z+4] * m2[z+4][i+3];
                 m_{res}[i][j+4] += m1[i][z+4] * m2[z+4][j+4];
                 m_{res}[i+1][j+4] += m1[i+1][z+4] * m2[z+4][j+4];
                 m_{res[i+2][j+4]} += m1[i+2][z+4] * m2[z+4][j+4];
                 m \operatorname{res}[i+3][i+4] += m1[i+3][z+4] * m2[z+4][i+4];
                 m_{res}[i+4][j+4] += m1[i+4][z+4] * m2[z+4][j+4];
               }
       else
Depto. Arquitectura y Tecnología de Computadores
```

```
if(N\%4==0)
        for(int i=0; i< N; i+=4)
     for(int j=0; j<N; j+=4)
       for(int z=0; z<N; z+=4)
          m_{res[i][j]} += m1[i][z] * m2[z][j];
          m_{res[i+1][j]} += m1[i+1][z] * m2[z][j];
          m res[i+2][j] += m1[i+2][z] * m2[z][j];
          m res[i+3][i] += m1[i+3][z] * m2[z][i];
          m_{res[i][j+1]} += m1[i][z] * m2[z][j+1];
          m res[i+1][j+1] += m1[i+1][z] * m2[z][j+1];
          m_{res[i+2][j+1]} += m1[i+2][z] * m2[z][j+1];
          m \operatorname{res}[i+3][j+1] += m1[i+3][z] * m2[z][j+1];
          m_{res[i][j+2]} += m1[i][z] * m2[z][j+2];
          m_{res}[i+1][j+2] += m1[i+1][z] * m2[z][j+2];
          m_{res}[i+2][j+2] += m1[i+2][z] * m2[z][j+2];
          m_{res}[i+3][j+2] += m1[i+3][z] * m2[z][j+2];
          m_{res[i][j+3]} += m1[i][z] * m2[z][j+3];
          m_{res[i+1][j+3]} += m1[i+1][z] * m2[z][j+3];
          m \operatorname{res}[i+2][j+3] += m1[i+2][z] * m2[z][j+3];
          m res[i+3][i+3] += m1[i+3][z] * m2[z][i+3];
          //
          m_{res[i][j]} += m1[i][z+1] * m2[z+1][j];
          m \operatorname{res}[i+1][j] += m1[i+1][z+1] * m2[z+1][j];
          m_{res[i+2][i]} += m1[i+2][z+1] * m2[z+1][i];
          m_{res}[i+3][j] += m1[i+3][z+1] * m2[z+1][j];
          m_{res[i][j+1]} += m1[i][z+1] * m2[z+1][j+1];
          m_{res[i+1][j+1]} += m1[i+1][z+1] * m2[z+1][j+1];
          m_{res}[i+2][j+1] += m1[i+2][z+1] * m2[z+1][j+1];
          m_{res}[i+3][j+1] += m1[i+3][z+1] * m2[z+1][j+1];
          m_{res[i][j+2]} += m1[i][z+1] * m2[z+1][j+2];
          m_{res}[i+1][j+2] += m1[i+1][z+1] * m2[z+1][j+2];
          m \operatorname{res}[i+2][j+2] += m1[i+2][z+1] * m2[z+1][j+2];
          m_{res}[i+3][j+2] += m1[i+3][z+1] * m2[z+1][j+2];
          m_{res[i][j+3]} += m1[i][z+1] * m2[z+1][j+3];
          m_{res}[i+1][j+3] += m1[i+1][z+1] * m2[z+1][j+3];
          m_{res}[i+2][j+3] += m1[i+2][z+1] * m2[z+1][j+3];
          m_{res}[i+3][j+3] += m1[i+3][z+1] * m2[z+1][j+3];
          //
          m_{res[i][j]} += m1[i][z+2] * m2[z+2][j];
          m_{res}[i+1][j] += m1[i+1][z+2] * m2[z+2][j];
          m_{res}[i+2][j] += m1[i+2][z+2] * m2[z+2][j];
          m_{res}[i+3][j] += m1[i+3][z+2] * m2[z+2][i];
          m res[i][j+1] += m1[i][z+2] * m2[z+2][j+1];
          m_{res}[i+1][j+1] += m1[i+1][z+2] * m2[z+2][j+1];
          m_{res}[i+2][j+1] += m1[i+2][z+2] * m2[z+2][j+1];
          m_{res}[i+3][j+1] += m1[i+3][z+2] * m2[z+2][j+1];
```

```
m_{res[i][j+2]} += m1[i][z+2] * m2[z+2][j+2];
         m_{res}[i+1][j+2] += m1[i+1][z+2] * m2[z+2][j+2];
         m_{res}[i+2][j+2] += m1[i+2][z+2] * m2[z+2][j+2];
         m_{res}[i+3][j+2] += m1[i+3][z+2] * m2[z+2][j+2];
         m_{res[i][j+3]} += m1[i][z+2] * m2[z+2][j+3];
         m_{res}[i+1][j+3] += m1[i+1][z+2] * m2[z+2][j+3];
         m_{res}[i+2][j+3] += m1[i+2][z+2] * m2[z+2][j+3];
         m \operatorname{res}[i+3][j+3] += m1[i+3][z+2] * m2[z+2][j+3];
         m_{res[i][j]} += m1[i][z+3] * m2[z+3][j];
         m_{res[i+1][j]} += m1[i+1][z+3] * m2[z+3][j];
         m_{res}[i+2][j] += m1[i+2][z+3] * m2[z+3][j];
         m_{res[i+3][j]} += m1[i+3][z+3] * m2[z+3][j];
         m_{res[i][j+1]} += m1[i][z+3] * m2[z+3][j+1];
         m_{res}[i+1][j+1] += m1[i+1][z+3] * m2[z+3][j+1];
         m_{res}[i+2][j+1] += m1[i+2][z+3] * m2[z+3][j+1];
         m_{res}[i+3][j+1] += m1[i+3][z+3] * m2[z+3][j+1];
         m_{res[i][j+2]} += m1[i][z+3] * m2[z+3][j+2];
         m \operatorname{res}[i+1][j+2] += m1[i+1][z+3] * m2[z+3][j+2];
         m_{res}[i+2][j+2] += m1[i+2][z+3] * m2[z+3][j+2];
         m_{res}[i+3][j+2] += m1[i+3][z+3] * m2[z+3][j+2];
         m_{res[i][j+3]} += m1[i][z+3] * m2[z+3][j+3];
         m_{res[i+1][j+3]} = m1[i+1][z+3] * m2[z+3][j+3];
         m_{res}[i+2][j+3] += m1[i+2][z+3] * m2[z+3][i+3];
         m_{res}[i+3][j+3] += m1[i+3][z+3] * m2[z+3][j+3];
else
        if(N\%3==0)
                for(int i=0; i< N; i+=3)
             for(int j=0; j<N; j+=3)
               for(int z=0; z<N; z+=3)
                  m_{res[i][j]} += m1[i][z] * m2[z][j];
                  m_{res[i+1][j]} += m1[i+1][z] * m2[z][j];
                  m res[i+2][i] += m1[i+2][z] * m2[z][i];
                  m_{res[i][j+1]} += m1[i][z] * m2[z][j+1];
                  m_{res}[i+1][j+1] += m1[i+1][z] * m2[z][j+1];
                  m_{res}[i+2][j+1] += m1[i+2][z] * m2[z][j+1];
                  m_{res[i][j+2]} += m1[i][z] * m2[z][j+2];
                  m_{res}[i+1][j+2] += m1[i+1][z] * m2[z][j+2];
                  m_{res}[i+2][j+2] += m1[i+2][z] * m2[z][j+2];
                  //
                  m_{res[i][j]} += m1[i][z+1] * m2[z+1][j];
                  m res[i+1][i] += m1[i+1][z+1] * m2[z+1][i];
                  m_{res[i+2][j]} += m1[i+2][z+1] * m2[z+1][j];
                  m_{res}[i][j+1] += m1[i][z+1] * m2[z+1][j+1];
                  m_{res}[i+1][j+1] += m1[i+1][z+1] * m2[z+1][j+1];
```

```
m_{res}[i+2][j+1] += m1[i+2][z+1] * m2[z+1][j+1];
          m_{res[i][j+2]} += m1[i][z+1] * m2[z+1][j+2];
          m_{res}[i+1][j+2] += m1[i+1][z+1] * m2[z+1][j+2];
          m_{res}[i+2][j+2] += m1[i+2][z+1] * m2[z+1][j+2];
          //
          m res[i][j] += m1[i][z+2] * m2[z+2][j];
          m \operatorname{res}[i+1][i] += m1[i+1][z+2] * m2[z+2][i];
          m \operatorname{res}[i+2][j] += m1[i+2][z+2] * m2[z+2][j];
          m_{res}[i][j+1] += m1[i][z+2] * m2[z+2][j+1];
          m_{res}[i+1][j+1] += m1[i+1][z+2] * m2[z+2][j+1];
          m_{res}[i+2][j+1] += m1[i+2][z+2] * m2[z+2][j+1];
          m \operatorname{res}[i][j+2] += m1[i][z+2] * m2[z+2][j+2];
          m_{res}[i+1][j+2] += m1[i+1][z+2] * m2[z+2][j+2];
          m_{res}[i+2][j+2] += m1[i+2][z+2] * m2[z+2][j+2];
}
else
        for(int i=0; i< N; ++i)
     for(int j=0; j< N; ++j)
       for(int z=0; z< N; ++z)
          m_{res[i][j]} += m1[i][z] * m2[z][j];
```

#### 1.1. TIEMPOS: tamaño 1500

Modificación	-O2
Sin modificar	13,088
Modificación a)	7
Modificación b)	4,84
Modificación c)	4,53

```
JoseJavierAlonsoRamos jjavier98@jjavier98-Lenovo-ideapad-310-15IKB:~/G
II/2°/2° Semestre/AC/SEMINARIOS/Seminario_4/ejer1] 2018-05-21 lunes
./pmm-secuencial 1500
m[0][0]: 6000.000000
n[1499][1499]: 6000.000000
tiempo: 13.088705880
JoseJavierAlonsoRamos jjavier98@jjavier98-Lenovo-ideapad-310-15IKB:~/G
II/2°/2° Semestre/AC/SEMINARIOS/Seminario_4/ejer1] 2018-05-21 lunes
./pmm-secuencial-mod 1500
n[0][0]: 6000.000000
n[1499][1499]: 6000.000000
iempo: 7.006655758
JoseJavierAlonsoRamos jjavier98@jjavier98-Lenovo-ideapad-310-15IKB:~/G
II/2°/2° Semestre/AC/SEMINARIOS/Seminario_4/ejer1] 2018-05-21 lunes
./pmm-secuencial-mod-1 1500
1[0][0]: 6000.000000
n[1499][1499]: 6000.000000
tiempo: 4.843274749
JoseJavierAlonsoRamos jjavier98@jjavier98-Lenovo-ideapad-310-15IKB:~/G
II/2°/2° Semestre/AC/SEMINARIOS/Seminario_4/ejer1] 2018-05-21 lunes
./pmm-secuencial-mod-2 1500
n[0][0]: 6000.000000
ໆ[1499][1499]: 6000.000000
tiempo: 4.531795338
```

## 1.1. COMENTARIOS SOBRE LOS RESULTADOS:

# 1.2. CÓDIGO EN ENSAMBLADOR DEL ORIGINAL Y DE DOS MODIFICACIONES : (PONER AQUÍ SÓLO LA ZONA DEL CÓDIGO ENSAMBLADOR EVALUADA, USE COLORES PARA DESTACAR LAS DIFERENCIAS)

pmm-secuencial.s		pmm-secuenci modificado_b	ial- o.s		pmm-secuenci modificado_d	ial- C.S
call		call			call	
clock_gettime@PL xorl	T	T clock_gettime@PLT test1			clock_getti	me@PLT testl
	, %r9d		%ebp, %	éehn		%ebp, %ebp
p2a			je	осър		jne
4,,10	5		.L74			.L28
	lign 3		movq			movq
.L15:	_		96(%rsp	),		104(%rsp),
movq		%rax			%rax	
(%rsı	o),		testb			movl
%rax			\$3, %al	L		\$0,
movq			movl		100(%rsp)	
			%eax, %	ecx		movq
(%r12,%r9), %rdi			jne 			%rax,
xorl	0.4		. L75		120(%rsp)	
	, %ecx		movq			movq
movq	_	0/	64(%rsp	), <sub> </sub>	0/	88(%rsp),
(%ra)	Χ,	%rax			%rax	
%r9), %rsi	lian		movq	,		movd
. p2a.	ттап	%rdi	56(%rsp	י,(י	112/0/rcn)	%rax,
4,,10	lian 2	%i u1	mova		112(%rsp)	naalian
. pza.	lign 3		movq %rax,		4,,10	.p2align
movs	4	32(%rsp)	∕01 αx,		4,,10	.p2align 3
(%rd:		32 (701 3p)	movl		.L29:	.pzarryn 5
%rcx), %xmm1	-,		%ebx, %	eax		movq
xorl			movq	Jour		112(%rsp),
	, %eax		%rdi,		%rax	(/00 /)/
.p2a		24(%rsp)	,			movl
4,,10	J		shrl			\$0,
	lign 3		\$2, %ea	ax	32(%rsp)	•
.L7:	_		movq			movl
movq			%rax, %	6rdx		\$16, %r15d
			addq			movq
(%r15,%rax), %rd	X		<b>\$1</b> , %ra	ìΧ		(%rax),
movs			salq		%rdi	
(%rd)	Κ,		\$5, %ra	ìΧ		movq
%rcx), %xmm0	_		salq			%rdi,
mulso			\$5, %rd	xt	48(%rsp)	
(%rs:	L,		leaq			movq
%rax), %xmm0		0(	32(%rdi	L,	مراج جا خ	8(%rax),
addq	/rox	%rdx), %rdi			%rdi	mov/G
\$8, 9	∕oı dX		movq %rav			movq %rdi
cmpq	0/r10	16(%rcn)	%rax,		56(%rcn)	%rdi,
adds	, %r13	16(%rsp)	leaq		56(%rsp)	movq
auusi					uitoctura y Tocnologí	

	%×mm0,		8(,		16(%rax),
%xmm1	, o, t	%rbx,8),		%rdi	== (/5. 5.7.)
7074111112	jne	70. 57, 67,	movq	701 41	movq
	. L7		%rdi,		%rdi,
	movsd	40(%rsp)	701 GI,	64(%rsp)	701 GI,
	%xmm1,	40(%ii 3p)	movq	04(%13p)	movq
(0/rdi 0/rov)	/0XIIIII⊥,		•		•
(%rdi,%rcx)	o d d a	0(%500)	%rax,	0/ rs d i	24(%rax),
	addq	8(%rsp)	-0-1-a-	%rdi	
	\$8, %rcx	4 40	.p2align		movq
	cmpq	4,,10	0.11		32(%rax),
	%rcx, %r13		.p2align 3	%rax	
	jne	.L14:			movq
	.L10		movq		%rdi,
	addq		32(%rsp),	72(%rsp)	
	\$8, %r9	%rax			movq
	cmpq		xorl		%rax,
	%r9, %r13		%r9d, %r9d	80(%rsp)	
	jne		movq		movq
	. L15		(%rax),		120(%rsp),
	leaq	%r14		%rax	
	64(%rsp),		movq		movq
%rsi	, ,		8(%rax),		(%rax),
	xorl	%r13	<b>\</b> //	%r12	, ,,
	%edi, %edi		movq		movq
	call		16(%rax),		8(%rax),
		%r12	== (/5: 56//)	%rcx	0 (701 0017)
clock_getti	me@PLT	/01 12	movq	701 070	movq
GIOOK_gccci	incer L1		24(%rax),		16(%rax),
		%rbp	24(701 ax);	%rbp	10(/// αλ),
		701 bb	movq	701 DP	movq
			•		24(%rax),
		9/501/	24(%rsp),	%rdx	24(%) ax),
		%rax	mayra	%i ux	mayra
			movq		movq
		04	(%rax),	0/	32(%rax),
		%rsi		%rax	0.11
			movq		.p2align
			8(%rax),	4,,10	
		%rcx			.p2align 3
			movq	.L12:	
			16(%rax),		leaq
		%rdx			8(%r15),
			movq	%rdi	
			24(%rax),		leaq
		%rax			-8(%r15),
			.p2align	%r13	
		4,,10			movq
			.p2align 3		48(%rsp),
		.L17:	-	%r11	, , ,
			leaq		movq
			8(%r9),		40(%rsp),
		%rbx	. ,,	%r10	. 177
			leaq		movq
			16(%r9),		56(%rsp),
		%r11	- ( )	%r9	- ( / /
		<b>-</b>	leaq	_ <del>-</del>	leaq
					_ = = = = =

	0.4.(0/10.)		10(0/215)
	24(%r9),		-16(%r15),
%r10		%r14	
	xorl		movq
			•
	%edi, %edi		%rdi,
	.p2align	(%rsp)	
4,,10	, ,		leaq
7,710			•
	.p2align 3		16(%r15),
.L15:		%rdi	
	movq		movq
	mo v q		
			64(%rsp),
(%r15, %rdi	i), %r8	%r8	
	movsd		movq
			80(%rsp),
(%r14, %rdi	L), %xmm4	%rsi	
	movsd		movl
			\$0 <i>,</i>
0(%r13,%r0	di), %xmm3	24(%rsp)	
	movapd	' '	movq
	•		•
	%xmm4,		%rdi,
%xmm1		8(%rsp)	
	movsd	' '	movq
			72(%rsp),
(%r12, %rdi	L), %xmm2	%rdi	
	movsd		movq
			•
	(%r8,%r9),		%r13,
%×mm0		16(%rsp)	
	movapd		.p2align
	%xmm4,	4,,10	. 6 - 8 - 9
	/0×1111114 <sub>1</sub>	4,,10	
%xmm5			.p2align 3
	mulsd	.L9:	
	%×mm0,		movq
0,	70XIIIIIO <b>,</b>		•
%xmm1			(%r10),
	addsd	%rbx	
	(%rsi),		movsd
0/> cmm 4	(701 31),		
%×mm1			(%r11),
	movsd	%xmm5	
	%xmm1,		movsd
(0/roi)	, v, /		
(%rsi)	_		(%r9),
	movapd	%xmm4	
	%xmm0,		movq
%xmm1	·		•
/%XIIIII1	<u>.</u> .		16(%rsp),
	mulsd	%r13	
	%xmm3,		movapd
%xmm1	·		•
/%XIIIII1			%xmm5,
	addsd	%xmm1	
	(%rcx),		addq
%xmm1	( · · · · / /		
70XIIIII11			\$40, %r11
	movsd		movsd
	%xmm1,		(%rbx,
(%rcx)	,	%r14), %xmn	
(/01 CX)		<sup>/01</sup>	
	movapd		addq
	%xmm0,		\$40, %r10
%xmm1	· ···· <del>-</del> /		movsd
/0XIIIIIT	1		
	mulsd		(%r8),
	%xmm2,	%xmm3	
The state of the s	,	-	

%xmm1			addq
707111111111111111111111111111111111111	addsd		\$40, %r9
	(%rdx),		mulsd
%xmm1			%xmm0,
	movsd	%xmm1	
	%xmm1,		movapd
(%rdx)	,		%xmm5,
(701 07)	movsd	%×mm6	70XIIIII3,
		/0XIIIIIO	
	0(%rbp,		movsd
%rdi), %xmm			(%rdi),
	addq	%xmm2	
	\$8, %rdi		addq
	mulsd		\$40, %r8
	%xmm1,		addq
0/24mm0	/0XIIIII <b>1</b> ,		•
%×mm0			\$40, %rdi
	addsd		addq
	(%rax),		\$40, %rsi
%xmm0			addsd
	movsd		(%r12),
	%×mm0,	%xmm1	` '/'
(%rax)	, 5, 4, 1, 1, 1	, 5, 1, 1, 1	movsd
(101 ax)	moved		
	movsd		%xmm1,
		(%r12)	
(%r8,%rbx),	%xmm0		movapd
	mulsd		%xmm0,
	%×mm0,	%xmm1	,
%xmm5	, o, c,	7074111112	mulsd
707111113	addsd		
			%xmm4,
	8(%rsi),	%xmm1	
%xmm5			addsd
	movsd		(%rcx),
	%xmm5,	%xmm1	
8(%rsi)	•		movsd
(/0.02)	movapd		%xmm1,
	•	(0/rox)	/0XIIIII±,
04	%xmm3,	(%rcx)	
%xmm5	_		movapd
	mulsd		%xmm0,
	%xmm0,	%xmm1	
%xmm5	•		mulsd
	addsd		%xmm3,
	8(%rcx),	%xmm1	
0/ymmE	J(701 UA),	WHIT	added
%xmm5			addsd
	movsd		0(%rbp),
	%xmm5,	%xmm1	
8(%rcx)			movsd
	movapd		%xmm1,
	%xmm2,	0(%rbp)	,
%xmm5	· /	(/5. 57)	movapd
7071111113	mulsd		
		0/2 / 200 4	%xmm0,
	%xmm0,	%xmm1	
%xmm5			mulsd
	mulsd		%xmm2,
	%xmm1,	%xmm1	
%×mm0	,		addsd
	addsd		(%rdx),
	uuusu		(701 U.A.),

	0/0/544/	0/vmm1	
	8(%rdx),	%xmm1	
%xmm5			movsd
	movsd		%xmm1,
	%xmm5,	(%rdx)	
8(%rdx)	•	,	movsd
- ()	movapd		-40(%rsi),
	•	0/ymm1	-τ∪(/01 S⊥),
0/2/	%×mm4,	%xmm1	
%xmm5			mulsd
	addsd		%xmm1,
	8(%rax),	%×mm0	
%×mm0			addsd
	movsd		(%rax),
	%xmm0,	%×mm0	( J. ) j
Q(0/ray)	, o, (iiiii O ,	70731111110	moved
8(%rax)	moved		movsd
	movsd		%xmm0,
		(%rax)	
(%r8,%r11),			movsd
	mulsd		(%rbx,
	%xmm0,	%r13), %xmm	
%xmm5		_ , , , ,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,	mulsd
, o, ciniii <b>o</b>	addsd		
			%×mm0,
	16(%rsi),	%xmm6	
%xmm5			addsd
	movsd		8(%r12),
	%xmm5,	%×mm6	,
16(%rsi)	,		movsd
	movapd		%xmm6,
	•		, oximiio,
0/2/2007	%×mm3,	8(%r12)	move====
%xmm5			movapd
	mulsd		%xmm4,
	%xmm0,	%xmm6	
%xmm5			mulsd
	addsd		%×mm0,
	16(%rcx),	%xmm6	,
%xmm5	±5(/01 U/)/	707.1111110	addsd
CIIIIIA	moved		
	movsd	04 5	8(%rcx),
	%xmm5,	%xmm6	
16(%rcx)			movsd
	movapd		%xmm6,
	%xmm2,	8(%rcx)	
%xmm5	,		movapd
,	mulsd		•
		0/y/mm6	%xmm3,
0/	%×mmΘ,	%xmm6	
%xmm5			mulsd
	mulsd		%xmm0,
	%xmm1,	%xmm6	
%×mm0	•		addsd
	addsd		8(%rbp),
		%xmm6	- ( ~ > P ) /
0/2/225	16(%rdx),	/0AIIIIIU	moved
%xmm5	_		movsd
	movsd		%xmm6,
	%xmm5,	8(%rbp)	
16(%rdx)	•		movapd
' '	addsd		%xmm2,
	16(%rax),	%xmm6	······· <b>-</b> /
	±υ(/οι αλ ),	/0/\lillill\(\mathbf{O}\)	

%×mm0			mulsd
/UXIIIIIO	movsd		%xmm0,
		%×mm6	/υΔιιιιί <b>υ</b> ,
16(0/50)	%×mmΘ,	/oxiiiiiO	muled
16(%rax)	moved		mulsd
	movsd	04	%xmm1,
(0) -0 0) 15:	0/ 0	%×mm0	
(%r8,%r10),			addsd
	cmpq		8(%rdx),
	%rdi,	%xmm6	_
8(%rsp)			movsd
	mulsd		%xmm6,
	%xmm0,	8(%rdx)	
%xmm4			movapd
	mulsd		%xmm5,
	%xmm0,	%xmm6	
%xmm3			addsd
	mulsd		8(%rax),
	%xmm0,	%×mm0	
%xmm2	•		movsd
	mulsd		%×mm0,
	%xmm1,	8(%rax)	,
%×mm0	,	' '	movsd
	addsd		(%rbx,
	24(%rsi),	%r15), %xmm	• •
%xmm4	( /01 0 ± / /	751 20 ) / 70/111111	mulsd
	movsd		%xmm0,
	%xmm4,	%×mm6	707mm0 ;
24(%rsi)	/V/IIIII + /	/JAIIIIIO	addsd
24(701 SI)	addsd		16(%r12),
	24(%rcx),	%xmm6	TO ( /01 TZ ) ,
%xmm3	24(/01 UX),	/0AIIIIIO	movsd
ZIIIIIX	moved		
	movsd	16(0/510)	%xmm6,
24/0/2011	%xmm3,	16(%r12)	moviored
24(%rcx)	م ما ما د -ا		movapd
	addsd	04	%xmm4,
0, 2	24(%rdx),	%xmm6	
%xmm2			mulsd
	movsd		%xmm0,
	%xmm2,	%xmm6	
24(%rdx)			addsd
	addsd		16(%rcx),
	24(%rax),	%xmm6	
%×mm0			movq
	movsd		(%rsp),
	%xmm0,	%r13	
24(%rax)	-		movsd
	jne		%xmm6,
	. L15	16(%rcx)	,
	addq	, ,	movapd
	\$32, %rsi		%xmm3,
	addq	%xmm6	- /
	\$32, %r9		mulsd
	addq		%xmm0,
	\$32, %rcx	%×mm6	/0/mm/0,
	addq	/0XIIIIIO	addsd
	auuq		นนนอน

		\$32, %rdx		16(%rbp),
		addq	%xmm6	
		\$32, %rax		movsd
		cmpq		%xmm6,
		%r9,	16(%rbp)	
	16(%rsp)			movapd
	,	jne		%xmm2,
		.L17	%xmm6	,
		addq	7 07 1111111	mulsd
		\$32,		%xmm0,
	24(%rsp)	Ψ02,	%xmm6	70XIIIIIO <b>,</b>
	24(%i3p)	odda	/0XIIIIIO	mulsd
		addq		
	00 (0/ )	\$32,	0,	%xmm1,
	32(%rsp)		%×mm0	
		movq		addsd
		24(%rsp),		16(%rdx),
	%rax		%xmm6	
		cmpq		movsd
		%rax,		%xmm6,
	40(%rsp)	•	16(%rdx)	•
	/	jne	` ′	movapd
		.L14		%xmm5,
		jmp	%xmm6	, o, t
		.L11	7071111110	addsd
	.L74:			16(%rax),
	.L/4.	mova	%×mm0	10(%i ax),
		movq	70X111111U	mayad
	0/15.0.1/	56(%rsp),		movsd
	%rax		10/0/	%xmm0,
		movl	16(%rax)	
		\$0,		movsd
	76(%rsp)		_	(%rbx,
		movq	%r13), %xmm	0
		%r15,		movq
	40(%rsp)			8(%rsp),
		movq	%r13	
		%rax,		mulsd
	88(%rsp)	•		%×mm0,
	/	movq	%xmm6	,
		64(%rsp),		addsd
	%rax	(   -   -   -   -   -   -   -   -		24(%r12),
		movq	%xmm6	= : ( ) /
		%rax,	, , , , , , , , , , , , , , , , , , , ,	movsd
	80(%rsp)	701 U.A.,		%xmm6,
	00 (%i Sp)	n2olian	24(0/512)	/0XIIIIIU,
	4 10	.p2align	24(%r12)	movend
	4,,10	m0a14a.		movapd
		.p2align 3		%xmm4,
	.L30:		%xmm6	
		movq		mulsd
		80(%rsp),		%xmm0,
	%rax		%xmm6	
		movl		addsd
		\$0,		24(%rcx),
	52(%rsp)	,	%xmm6	` ''
	· - 1° /	movl	_	movsd
		\$16, %r10d		%xmm6,
l I		WILL /ULIUU	I .	/V/MIIIII//

	movq 24(%rax),	24(%rcx)	movapd
%rbx	movq (%rax),	%×mm6	%xmm3, mulsd
%r15	movq	%×mm6	%xmm0,
%r14	8(%rax),		addsd 24(%rbp),
%r13	movq 16(%rax),	%×mm6	movsd
/%I 13	movq 32(%rax),	24(%rbp)	%xmm6, movapd
%r11	movq	%×mm6	%xmm2,
%rax	88(%rsp),		mulsd %xmm0,
24(%rsp)	movq %rbx,	%×mm6	mulsd %xmm1,
2π(/01 3μ)	movq (%rax),	%×mm0	addsd
%rdi	movq	%×mm6	24(%rdx),
%rsi	8(%rax),	24(%rdx)	movsd %xmm6,
%rcx	movq 16(%rax),	24(%i ux)	addsd 24(%rax),
	movq 24(%rax),	%×mm0	movsd
%rdx	movq	24(%rax)	%×mm0,
%rax	32(%rax), .p2align	%r13), %xmm	movsd (%rbx,
4,,10	.p2align 3	701 13), 707	mulsd %xmm0,
.L12:	leaq	%xmm5	mulsd
%rbx	8(%r10),	%×mm4	%xmm0,
%r9	leaq 16(%r10),	%×mm3	mulsd %xmm0,
	leaq -16(%r10),		mulsd %xmm0,
%r12	leaq	%xmm2	addsd
%rbp	-8(%r10), xorl	%×mm5	32(%r12), mulsd
	%r8d, %r8d movq	%×mm0	%xmm1,

	0/0		may and
	%r9,		movsd
8(%rsp)			%xmm5,
	movq	32(%r12)	
	%rbx,		addsd
16(%rsp)			32(%rcx),
±0(/013P)	n2alian	%×mm4	52 (701 CA),
4 40	.p2align	/0XIIIIII4	1
4,,10			movsd
	.p2align 3		%xmm4,
.L9:		32(%rcx)	
	movq	,	addsd
	40(%rsp),		32(%rbp),
%r9	. J ( , 51 J ) /	%xmm3	(/3/ SP//
/01 3	moved	/ozimiio	moved
	movsd		movsd
			%xmm3,
(%r15,%r8),	%xmm5	32(%rbp)	
	movsd		addsd
			32(%rdx),
(%r14,%r8),	%xmm4	%xmm2	` ''
(75. = 1,751 5 )	movq		movsd
	•		
	24(%rsp),	22/2/ : ;	%xmm2,
%rbx		32(%rdx)	
	movapd		addsd
	%xmm5,		32(%rax),
%xmm1	•	%×mm0	, ,,
	movq		movsd
	(%r9,%r8),		%xmm0,
0/50	(/013,/010),	22 (0/52)	/0ΛIIIIIO,
%r9		32(%rax)	
	movsd		movsd
			-32(%r11),
0(%r13,%r8)	, %xmm3	%xmm5	
	movsd		movq
	(%rbx,		-32(%r10),
%r8), %xmm2	•	%rbx	(/0 ) /
701 U ) , 70AIIIIIZ		701 07	movsd
	movapd		
	%xmm5,		-32(%r9),
%xmm6		%xmm4	
	movsd		movapd
			%xmm5,
(%r9,%r12),	%xmm0	%xmm1	,
	mulsd		movsd
	%xmm0,		-32(%r8),
0/x/mm1	/0∧IIIIIO,	0/x/mm2	- JZ ( /OI O ) ,
%xmm1	- 44 - 1	%xmm3	
	addsd		movsd
	(%rdi),		(%rbx,
%xmm1		%r14), %xmm	0
	movsd		movq
	%xmm1,		16(%rsp),
(%rdi)	· · · · · · · · · · · · · · · · · · ·	%r13	( /oi op ) /
(701 01)	movend	/01 ±3	moved
	movapd		movsd
	%xmm0,		-32(%rdi),
%xmm1		%xmm2	
	mulsd		mulsd
	%xmm4,		%xmm0,
%xmm1	,	%xmm1	,
, 57 (IIIII) <u>+</u>	addsd	, 57 timil <u>+</u>	movapd
	uuusu		movαρα

	(%rci)		0/vmmE
0/2/mm4	(%rsi),	0/2 cmm C	%xmm5,
%xmm1		%×mm6	- 44 - 2
	movsd		addsd
	%xmm1,		(%r12),
(%rsi)		%xmm1	
	movapd		movsd
	%xmm0,		%xmm1,
%xmm1		(%r12)	
	mulsd		movapd
	%xmm3,		%×mmO,
%xmm1	,	%xmm1	•
	addsd		mulsd
	(%rcx),		%×mm4,
%xmm1	(	%xmm1	<i></i>
/V/(IIIII)	movsd		addsd
	%xmm1,		(%rcx),
(%rcx)	/0//IIIII±/	%×mm1	(701 07),
(701 67)	movand	/0×1111111	movsd
	movapd		
0/2/2021	%xmm0,	(0/50)	%xmm1,
%xmm1		(%rcx)	
	mulsd		movapd
	%xmm2,		%xmm0,
%xmm1		%xmm1	
	addsd		mulsd
	(%rdx),		%xmm3,
%xmm1		%xmm1	
	movsd		addsd
	%xmm1,		0(%rbp),
(%rdx)	,	%xmm1	. , , ,
` '	movsd		movsd
			%xmm1,
(%r11,%r8),	%xmm1	0(%rbp)	······· <b>-</b> ,
(70. 11,70. 0),	addq	(70. 50)	movapd
	\$8, %r8		%xmmΘ,
	mulsd	%×mm1	/0/IIIIIO,
		/0∧IIIII1 <b>⊥</b>	mulsd
0/2/mm0	%xmm1,		
%×mm0	addad	0/2/mm1	%xmm2,
	addsd	%×mm1	a al al c -!
	(%rax),		addsd
%×mm0	_		(%rdx),
	movsd	%xmm1	
	%×mm0,		movsd
(%rax)			%xmm1,
	movsd	(%rdx)	
			movsd
(%r9,%rbp),	%×mm0		-32(%rsi),
	mulsd	%xmm1	, ,,
	%×mm0,		mulsd
%xmm6	······································		%xmm1,
707.tmm0	addsd	%×mm0	· • · · · · · · · · · · · · · · · · · ·
	8(%rdi),	/oximilo	addsd
0/ymm6	ο(/oi u±),		
%xmm6	moved	0/2/mm20	(%rax),
	movsd	%×mm0	,
	%xmm6,		
8(%rdi)			%xmm0,
8(%rdi)	%×mm6,		movsd %xmm0,

, , , , , , , , , , , , , , , , , , , ,	addsd	%×mm6	,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,
%xmm6	%xmm0,		movapd %xmm5,
	mulsd	8(%rdx)	movand
(%r9,%r10),		0 (0) 1 3	%xmm6,
			movsd
, ,	movsd	%xmm6	` ','
8(%rax)	,		8(%rdx),
	%xmm0,	ZOZIIIIIO	addsd
%xmm0	movsd	%×mm0	%xmm1,
9/ y mm 0	8(%rax),		mulsd
	addsd	%xmm6	
%xmm6	·		%×mm0,
	%xmm5,		mulsd
8(%rdx)	movapd	%×mm6	%xmm2,
0(%rdy)	%xmm6,		movapd
	movsd	8(%rbp)	
%xmm6	` ,,		%xmm6,
	8(%rdx),	, , , , , , , , , , , , , , , , , , , ,	movsd
%xmm0	addsd	%xmm6	8(%rbp),
9/y/mm0	%xmm1,		addsd
	mulsd	%xmm6	
%xmm6	,		%xmm0,
	%xmm0,		mulsd
/0XIIIIIU	mulsd	%xmm6	/0AIIIII3,
%xmm6	%xmm2,		movapd %xmm3,
	movapd	8(%rcx)	movand
8(%rcx)		0.404	%xmm6,
	%xmm6,		movsd
	movsd	%xmm6	- ( / /
%xmm6	0(/// 0//),		8(%rcx),
	addsd 8(%rcx),	%xmm6	addsd
%xmm6	مططوط	0/24mm C	%×mm0,
	%xmm0,		mulsd
	mulsd	%xmm6	,
%xmm6	/ολιιιιιο,		%xmm4,
	movapd %xmm3,	8(%r12)	movapd
8(%rsi)	moverd	0(%,510)	%xmm6,
	%xmm6,		movsd
	movsd	%xmm6	
%xmm6	- //		8(%r12),
	8(%rsi),	, JOXIIIIIO	addsd
/oXIIIIIO	addsd	%xmm6	<b>/0</b> ΧΙΙΙΙΙΙ <b>Ο</b> ,
%xmm6	%xmm0,		mulsd %xmm0,
	mulsd	%r13), %xmm	
%xmm6			(%rbx,
	%xmm4,	,	movsd
	movapd	(%rax)	

	16(0/rdi)	1	addad
040	16(%rdi),		addsd
%xmm6			8(%rax),
	movsd	%×mm0	
	%xmm6,		movsd
16(%rdi)			%xmm0,
, ,	movapd	8(%rax)	,
	•	0 (701 ax)	movsd
0/	%xmm4,		
%xmm6			(%rbx,
	movq	%r15), %xm	
	16(%rsp),		mulsd
%rbx			%xmm0,
	mulsd	%xmm6	,
	%×mm0,	7071111110	addsd
0/2 4 ma ma C	/0XIIIIIO,		
%xmm6			16(%r12),
	addsd	%xmm6	
	16(%rsi),		movsd
%xmm6			%xmm6,
	movsd	16(%r12)	,
	%xmm6,		movapd
16(0/50=)	/0AIIIIIU,		•
16(%rsi)			%xmm4,
	movapd	%xmm6	
	%xmm3,		mulsd
%xmm6	-		%xmm0,
	mulsd	%xmm6	,
		707(1111110	addsd
0/2/mm C	%×mm0,		
%xmm6			16(%rcx),
	addsd	%xmm6	
	16(%rcx),		movq
%xmm6			(%rsp),
	movsd	%r13	
	%×mm6,		movsd
16(0/201)	707mmO <sub>1</sub>		
16(%rcx)		10/0/== }	%xmm6,
	movapd	16(%rcx)	
	%xmm2,		movapd
%xmm6			%xmm3,
	mulsd	%xmm6	,
	%×mm0,		mulsd
0/y/mmG	/0/1111110 <sub>1</sub>		
%xmm6	1	04 0	%×mm0,
	mulsd	%xmm6	
	%xmm1,		addsd
%×mm0			16(%rbp),
	addsd	%xmm6	. 177
	16(%rdx),		movsd
0/2/mm C	10(701 UA),		
%xmm6		10/6/ 1 3	%xmm6,
	movsd	16(%rbp)	
	%xmm6,		movapd
16(%rdx)			%xmm2,
, ,	movapd	%xmm6	,
	%xmm5,		mulsd
0/y/mmG	/0/IIIII/J,		
%xmm6			%xmm0,
	addsd	%xmm6	
	16(%rax),		mulsd
%×mm0	• •		%xmm1,
	movsd	%×mm0	,
	%xmm0,	, , , , , , , , , , , , , , , , , , , ,	addsd
	/0/1111110 <sub>1</sub>		auusu

	16/0/501/			16 (0/54)
	16(%rax)		0/247777	16(%rdx),
		movsd	%xmm6	
				movsd
	(%r9,%rbx),	%×mm0		%xmm6,
		movq	16(%rdx)	
		8(%rsp),	,	movapd
	%rbx	· - 1- //		%xmm5,
	DA	mulsd	%xmm6	, , , , , , , , , , , , , , , , , , ,
			/0/111111U	added
	1/2 cmama C	%xmm0,		addsd
	%xmm6			16(%rax),
		addsd	%×mm0	
		24(%rdi),		movsd
	%xmm6			%xmm0,
		movsd	16(%rax)	
		%xmm6,	, ,	movsd
	24(%rdi)	- 1		(%rbx,
	(/0. 0.1)	movapd	%r13), %xmm	•
		•	/VI 13), /0XIIIIII	
	1/2 cmama C	%xmm4,		movq
	%xmm6			8(%rsp),
		mulsd	%r13	
		%xmm0,		mulsd
	%xmm6			%xmm0,
		addsd	%xmm6	,
		24(%rsi),	_	addsd
	%xmm6	_ : (/0: 0±//		24(%r12),
	,oxiiiii O	movsd	%×mm6	(/01 ±2);
			70XIIIIIX0	mayrad
	0.4.60(1.3	%xmm6,		movsd
	24(%rsi)	_		%xmm6,
		movapd	24(%r12)	
		%xmm3,		movapd
	%xmm6			%xmm4,
		mulsd	%xmm6	,
		%xmm0,	_	mulsd
	%xmm6	,		%xmm0,
	/U/XIIIIIIU	addsd	%×mm6	/0/MIIIIO,
			/0XIIIIIX	مططمط
		24(%rcx),		addsd
	%xmm6	_		24(%rcx),
		movsd	%xmm6	
		%xmm6,		movsd
	24(%rcx)	•		%xmm6,
	, ,	movapd	24(%rcx)	,
		%xmm2,		movapd
	%×mm6	/V/IIIII/_ /		%xmm3,
	/OAIIIIIAO	mulad	0/24mm C	/0XIIIII3,
		mulsd	%xmm6	
		%xmm0,		mulsd
	%xmm6			%xmm0,
		mulsd	%xmm6	
		%xmm1,		addsd
	%×mm0	,		24(%rbp),
		addsd	%xmm6	( ( ) ~ ( ) / (
			/////////////	moved
	· · · · · · · · · · · · · · · · · · ·	24(%rdx),		movsd
	%xmm6	_		%xmm6,
		movsd	24(%rbp)	
		%xmm6,		movapd
	24(%rdx)			%xmm2,
	` /			

addsd 24(%rax), mulsd %xmm0, movsd %xmm0, movsd %xmm0, mulsd %xmm1, movsd addsd 24(%rdx), mulsd %xmm0, mulsd %xmm0, movsd (%rbx, xmm0, mulsd %xmm0, movsd (%rbx, xmm0, movsd %xmm1, xmm0, movsd %xmm0, movsd %xmm4, addsd %xmm0, movsd %xmm0, movsd %xmm0, movsd %xmm0, movsd %xmm0, movsd %xmm4, addsd %xmm0, addsd %xmm0, movsd %xmm0, movsd %xmm0, addsd %xmm4, addsd %xmm0, addsd %xmm0, movsd %xmm0, addsd %xmm0, a
%xmm0       %xmm0         movsd       %xmm6         24(%rax)       %xmm0         movsd       %xmm0         ddsd       24(%rdx),         mulsd       %xmm6         %xmm5       %xmm6         mulsd       %xmm6,         %xmm0       addsd         %xmm0       24(%rdx)         %xmm0       movsd         %xmm0       %xmm0         %xmm0       movsd         %xmm1       movsd         %xmm1       mulsd         %xmm1       mulsd         %xmm0       %xmm0         movsd       %xmm0         movsd       %xmm0         movsd       %xmm0         movsd       %xmm0         movsd       %xmm0         mulsd       %xmm0         mulsd       %xmm0         %xmm0       %xmm0         %xmm0       mulsd         %xmm0       %xmm0         %xmm0       mulsd         %xmm0       %xmm0         %xmm0       %xmm0         %xmm0       %xmm0
movsd
%xmm0,       mulsd         24(%rax)       %xmm0         movsd       %xmm0         (%r9,%rbx),       %xmm0         mulsd       %xmm6         %xmm5       %xmm6         mulsd       %xmm6,         %xmm4       24(%rdx)         mulsd       %xmm0         %xmm3       movsd         %xmm0,       movsd         %xmm0,       movsd         %xmm1,       mulsd         %xmm5       mulsd         %xmm1,       mulsd         %xmm1,       mulsd         %xmm0,       %xmm0,         movsd       %xmm0,         mulsd       %xmm0,         mulsd       %xmm0,         movsd       %xmm0,         movsd       %xmm0,         mulsd       %xmm0,         mulsd       %xmm0,         %xmm1,       %xmm0,         %xmm0,       %xmm0,         %xmm1,       %xmm1,         %xmm1,       %xmm1,         %xmm2       %xmm0,
24(%rax)
movsd       %xmm0         (%r9,%rbx), %xmm0       addsd         mulsd       %xmm6         %xmm5       movsd         mulsd       24(%rdx)         %xmm0,       addsd         %xmm4       24(%rdx)         mulsd       %xmm0         %xmm0,       movsd         %xmm1,       movsd         %xmm1,       mulsd         %xmm0,       mulsd         %xmm1,       mulsd         %xmm1,       mulsd         %xmm0,       mulsd         %xmm0, <t< td=""></t<>
(%r9,%rbx), %xmm0       addsd         mulsd       %xmm6         %xmm5       movsd         mulsd       24(%rdx)         %xmm0,       addsd         %xmm4       24(%rax),         mulsd       %xmm0         %xmm0,       movsd         %xmm1,       movsd         %xmm1,       mulsd         %xmm0,       mulsd         %xmm1,       mulsd         %xmm0,       movsd         %xmm0,       mulsd         %xmm0,
(%r9,%rbx), %xmm0 mulsd %xmm0, mulsd %xmm0,       %xmm6         %xmm5       movsd %xmm6,         %xmm6       %xmm6,         %xmm7       addsd 24(%rdx)         %xmm0, mulsd %xmm0,       movsd %xmm0,         %xmm3       movsd (%rbx, addsd 32(%rdi), mulsd %xmm0,         %xmm5       mulsd %xmm5         %xmm1, mulsd %xmm0, movsd %xmm4       %xmm0, mulsd %xmm0, addsd 32(%rsi), addsd 32(%rsi), mulsd %xmm0, movsd %xmm0, mulsd %xmm0, movsd %xmm4         %xmm4       mulsd %xmm0, mulsd %xmm0, mulsd %xmm0, movsd %xmm2
mulsd %xmm0,       %xmm6         %xmm5       movsd %xmm6,         mulsd %xmm0,       24(%rdx)         %xmm4       addsd 24(%rax),         mulsd %xmm0,       movsd %xmm0,         %xmm3       movsd %xmm0,         %xmm0,       movsd (%rbx,         addsd 32(%rdi),       mulsd %xmm0,         %xmm5       mulsd %xmm0,         %xmm0,       mulsd %xmm0,         %xmm5,       mulsd %xmm0,         32(%rdi)       %xmm4         32(%rsi),       mulsd %xmm0,         %xmm4       mulsd %xmm0,         %xmm4       mulsd %xmm0,         %xmm4       %xmm0,
%xmm5       movsd %xmm6, wxmm6, wxmm6, wxmm6, wxmm0, wxmm0, wxmm0, wxmm0, wxmm0       24(%rdx) addsd 24(%rax), wxmm0         %xmm4       mulsd wxmm0, wxmm0       movsd wxmm0, wxmm0, wxmm0, wxmm0, wxmm0, wxmm0, wxmm0, wxmm0, wxmm0, wxmm0         %xmm2       addsd yxmm1, wxmm0       wxmm0, wxm
%xmm5       mulsd %xmm0, addsd 24(%rdx)         %xmm4       24(%rdx)         mulsd %xmm0, movsd %xmm0, mulsd %xmm0, addsd 32(%rdi), mulsd %xmm0, movsd (%rbx, addsd 32(%rdi), mulsd %xmm1, mulsd %xmm1, mulsd %xmm0, addsd %xmm5, addsd 32(%rsi), addsd 32(%rsi), mulsd %xmm0, movsd %xmm4       %xmm4         %xmm4       %xmm3         %xmm4       %xmm0, movsd %xmm2
mulsd %xmm0,       24(%rdx)         mulsd %xmm0,       324(%rax),         mulsd %xmm0,       movsd %xmm0,         mulsd %xmm0,       24(%rax)         movsd (%rbx,         addsd 32(%rdi),       mulsd %xmm0,         mulsd %xmm1,       mulsd %xmm0,         movsd %xmm5,       mulsd %xmm0,         32(%rdi)       mulsd %xmm0,         addsd %xmm5,       mulsd %xmm0,         32(%rsi),       mulsd %xmm0,         movsd       %xmm3         mulsd %xmm0,       %xmm0,         movsd       %xmm0,         movsd       %xmm0,         movsd       %xmm0,
%xmm4       addsd 24(%rax), 24(%rax)
%xmm4       24(%rax),         mulsd       %xmm0         %xmm3       %xmm0,         mulsd       24(%rax)         %xmm0,       movsd         %xmm2       (%rbx,         addsd       %r13), %xmm0         mulsd       %xmm5,         %xmm1,       mulsd         %xmm1,       mulsd         %xmm0,       mulsd         %xmm5,       mulsd         32(%rdi)       %xmm0,         addsd       %xmm3         32(%rsi),       mulsd         %xmm4       %xmm0,         movsd       %xmm0,
mulsd %xmm0, %xmm0,       %xmm0 %xmm0,         mulsd %xmm0,       24(%rax)         %xmm2       movsd (%rbx, addsd 32(%rdi),         mulsd %xmm5       mulsd %xmm0,         mulsd %xmm1,       mulsd %xmm0,         movsd %xmm5,       mulsd %xmm0,         32(%rdi)       %xmm4         movsd 32(%rsi),       mulsd %xmm0,         movsd %xmm4       %xmm3         mulsd %xmm0,       %xmm0,         movsd %xmm4       %xmm0,
%xmm3       movsd %xmm0, mulsd %xmm0, %xmm0,       24(%rax) movsd (%rbx, addsd 32(%rdi),       movsd (%rbx, %r13), %xmm0 mulsd %xmm0,         %xmm5       mulsd %xmm1, mulsd %xmm1,       mulsd %xmm0, movsd %xmm4         32(%rdi)       %xmm4       mulsd %xmm0, movsd         32(%rdi)       %xmm3 32(%rsi), mulsd %xmm0,       mulsd %xmm0, movsd         %xmm4       %xmm0, movsd       mulsd %xmm0, movsd
%xmm3       %xmm0,         mulsd       24(%rax)         %xmm0,       movsd         (%rbx,       (%rbx,         addsd       %r13), %xmm0         %xmm5       mulsd         %xmm1,       mulsd         %xmm0,       %xmm0,         movsd       %xmm4         %xmm5,       mulsd         32(%rdi)       %xmm3         32(%rsi),       mulsd         %xmm4       %xmm0,         movsd       %xmm2
mulsd %xmm0,       24(%rax)         %xmm2       movsd (%rbx, addsd 32(%rdi), mulsd %xmm0, mulsd %xmm0, mulsd %xmm1, mulsd %xmm0, wxmm0, addsd 32(%rdi)         movsd %xmm4       %xmm4         32(%rdi)       mulsd %xmm0, mulsd %xmm0, mulsd %xmm0, movsd %xmm2
%xmm0,       movsd (%rbx, addsd 32(%rdi),         %xmm5       %r13), %xmm0         mulsd %xmm1,       %xmm5         %xmm1,       mulsd %xmm0,         %xmm0       %xmm4         movsd %xmm5,       mulsd %xmm0,         32(%rdi)       %xmm3         32(%rsi),       mulsd %xmm0,         %xmm4       %xmm0,         movsd       %xmm2
%xmm2       (%rbx,         addsd       %r13), %xmm0         %xmm5       mulsd         %xmm1,       mulsd         %xmm0       %xmm5         movsd       %xmm0,         %xmm5,       mulsd         32(%rdi)       %xmm3         32(%rsi),       mulsd         %xmm4       %xmm0,         movsd       %xmm2
addsd 32(%rdi), %r13), %xmm0 mulsd %xmm0, mulsd %xmm1, mulsd %xmm0, movsd %xmm5, 32(%rdi) %xmm5, mulsd %xmm0, mulsd %xmm0, mulsd %xmm0, mulsd %xmm0, mulsd %xmm0, mulsd %xmm0, movsd %xmm2
32(%rdi), mulsd %xmm0, mulsd %xmm1, mulsd %xmm0, movsd %xmm4, mulsd %xmm5, 32(%rdi)
%xmm5       %xmm0,         mulsd       %xmm5         %xmm1,       mulsd         %xmm0,       %xmm0,         movsd       %xmm4         %xmm5,       mulsd         32(%rdi)       %xmm3         32(%rsi),       mulsd         %xmm4       %xmm0,         movsd       %xmm2
mulsd %xmm1, mulsd %xmm0 %xmm0, %xmm0, movsd %xmm4, mulsd %xmm5, addsd 32(%rsi), addsd 32(%rsi), mulsd %xmm0, movsd %xmm2
%xmm1, mulsd %xmm0, movsd %xmm4 mulsd %xmm5, mulsd %xmm6, addsd 32(%rsi), mulsd %xmm0, mulsd %xmm4 %xmm0, movsd %xmm2
%xmm0
movsd %xmm4 mulsd %xmm0, addsd 32(%rsi), %xmm4 mulsd %xmm0, mulsd %xmm4 %xmm0, movsd %xmm2
%xmm5, mulsd 32(%rdi) %xmm0, addsd %xmm3 32(%rsi), mulsd %xmm4 %xmm0,
32(%rdi)
addsd %xmm3 32(%rsi), mulsd %xmm0, movsd %xmm2
32(%rsi), mulsd %xmm4 %xmm0, movsd %xmm2
%xmm4 %xmm0, movsd %xmm2
movsd %xmm2
%xmm4. addsd
757
32(%rsi) 32(%r12),
addsd %xmm5
32(%rcx), mulsd
%xmm3 %xmm1,
movsd %xmm0
%xmm3, movsd
32(%rcx) %xmm5,
addsd 32(%r12)
32(%rdx), addsd
%xmm2 32(%rcx),
movsd %xmm4
%xmm2, movsd
32(%rdx) %xmm4,
addsd 32(%rcx)
32(%rax), addsd
movsd %xmm3
%xmm0, movsd
32(%rax)
cmpq 32(%rbp)

	%rQ		added
22(%525)	%r8,		addsd
32(%rsp)			32(%rdx),
	jne	%xmm2	
	. L9		movsd
	addl		%xmm2,
	\$5,	32(%rdx)	•
52(%rsp)	,	, ,	addsd
( 56)	addq		32(%rax),
	\$40, %rdi	%×mm0	52 (701 W/) /
	•	70×1111110	moved
	addq		movsd
	\$40 <u>,</u> %rsi		%xmm0,
	movl	32(%rax)	
	52(%rsp),		movsd
%ebx			-24(%r11),
	addq	%xmm5	
	\$40, %rcx		movq
	addq		-24(%r10),
	\$40, %rdx	%rbx	= : (/0. ±0 / /
	addq	, 31 D.A	movsd
	•		
	\$40, %rax	0/2 (200 05 4	-24(%r9),
	addq	%xmm4	
	\$40, %r10		movapd
	cmpl		%xmm5,
	72(%rsp),	%xmm1	
%ebx			movsd
	jl		-24(%r8),
	.L12	%xmm3	= : (/0. 0)/
	addl	,,,,,,,,,,,,	movsd
76 (0/202)	\$5,	0/161.43	(%rbx,
76(%rsp)		%r14), %xmm	
	addq		movq
	\$40,		16(%rsp),
88(%rsp)		%r13	
	movl		movsd
	76(%rsp),		-24(%rdi),
%eax		%xmm2	. ,,
	addq		mulsd
	\$40,		%xmm0,
90(%rcn)	Ψ+υ,	%xmm1	ANTIHITO,
80(%rsp)	omp 1	∕0XIIIII1 <b>⊥</b>	mayand
	cmpl		movapd
	72(%rsp),		%xmm5,
%eax		%xmm6	
	jl		addsd
	. L30		(%r12),
	movq	%xmm1	. ,,
	40(%rsp),		movsd
%r15	. C ( /oi Op ) /		%xmm1,
		(%r12)	/U/IIIII <b>/</b>
.L11:	1000	(101 12)	mayand
	leaq		movapd
	128(%rsp),		%xmm0,
%rsi		%xmm1	
	xorl		mulsd
	%edi, %edi		%xmm4,
	xorl	%xmm1	,
	%r12d,		addsd
	, or ±2 u j		

%r12d			(%rcx),
701 1ZU	1	0/	(701 CX),
	leaq	%xmm1	_
	.LC3(%rip)		movsd
, %rbp			%xmm1,
ļ´ .	call	(%rcx)	•
	oull	(701 071)	movapd
	0.01.0		
clock_getti	.me@PLI		%xmm0,
		%xmm1	
			mulsd
			%xmm3,
		0/4	/0XIIIII3,
		%xmm1	
			addsd
			0(%rbp),
		%xmm1	( 1 ) /
		/0/(11111111111111111111111111111111111	movsd
			%xmm1,
		0(%rbp)	
		• • • • •	movapd
			%×mm0,
		0/2/2021	/0/1111110 <sub>1</sub>
		%xmm1	
			mulsd
			%xmm2,
		%xmm1	,
		/0XIIIIII	addad
			addsd
			(%rdx),
		%xmm1	
			movsd
			%xmm1,
		(0/10 ol 1)	/0XIIIII±,
		(%rdx)	
			movsd
			-24(%rsi),
		%xmm1	, ,,
		707 1	mulsd
			%xmm1,
		%×mm0	
			addsd
			(%rax),
		%×mm0	(10. 3/1)
		/0/111111U	moved
			movsd
			%xmm0,
		(%rax)	
		<b>,</b>	movsd
		0 -10	(%rbx,
		%r13), %xmm	
			mulsd
			%xmm0,
		%xmm6	- ,
		/o/tilliio	addsd
			8(%r12),
		%xmm6	
			movsd
			%×mm6,
		0 (0/54 0)	/0∧IIIIIU,
		8(%r12)	_
			movapd
			%xmm4,
1		<u> </u>	

0/2 (2000)	
%xmm6	<b>.</b>
	mulsd
	%×mm0,
%xmm6	
	addsd
	8(%rcx),
%×mm6	- ( , ,
70/4111110	movsd
0 (0)	%xmm6,
8(%rcx)	_
	movapd
	%xmm3,
%xmm6	
	mulsd
	%×mm0,
%×mm6	707tillill <b>0</b>
/0\111110	addsd
	8(%rbp),
%×mm6	
	movsd
	%xmm6,
8(%rbp)	•
(70. 54)	movapd
	%xmm2,
0/2/mm6	/0ΧΙΙΙΙΙΖ,
%xmm6	<b>.</b>
	mulsd
	%×mm0,
%xmm6	
	mulsd
	%xmm1,
%×mm0	, g, t
70/4111110	addsd
	8(%rdx),
%×mm6	
	movsd
	%xmm6,
8(%rdx)	
	movapd
	%xmm5,
%×mm6	70/1111110 j
/oXIIIIIU	addad
	addsd
	8(%rax),
%×mm0	
	movsd
	%xmm0,
8(%rax)	,
	movsd
	(%rbx,
0/-45\ 0/	
%r15), %xm	
	mulsd
	%×mm0,
%×mm6	
	addsd
	16(%r12),
%×mm6	
/0AIIIIIU	

<u> </u>		moved
		movsd
	40(0/:-40)	%xmm6,
	16(%r12)	
		movapd
		%xmm4,
	%xmm6	_
		mulsd
		%xmm0,
	%xmm6	
		addsd
		16(%rcx),
	%xmm6	
		movq
		(%rsp),
	%r13	
		movsd
		%xmm6,
	16(%rcx)	,
	,	movapd
		%xmm3,
	%xmm6	,
	-	mulsd
		%×mm0,
	%xmm6	
		addsd
		16(%rbp),
	%xmm6	(\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
	,	movsd
		%×mm6,
	16(%rbp)	707CHIHIO ;
	±0(///i bp)	movapd
		%xmm2,
	%×mm6	/U/XIIIII/L /
	70/11111U	mulsd
		%xmm0,
	%xmm6	/JAIIIIIO,
	70X111110	mulsd
		%xmm1,
	%×mm0	//////////////////////////////////////
	70X111110	addsd
		16(%rdx),
	%×mm6	±0(/01 U/),
	70/11111U	movsd
		%xmm6,
	16(%rdx)	/0∧IIIIIO,
	10 ( /01 U A )	movapd
	%xmm6	%xmm5,
	/0∧IIIIIU	addsd
	0/.vmmΩ	16(%rax),
	%×mm0	mayrad
		movsd
	10 (0/2000)	%×mmΘ,
	16(%rax)	may a d
		movsd

/0/rhy
(%rbx,
%r13), %xmm0
movq
8(%rsp),
%r13
mulsd
%×mm0,
%xmm6
addsd
24(%r12),
%xmm6
movsd
%xmm6,
24(%r12)
movapd
%xmm4,
%×mm6
mulsd
%×mm0,
%xmm6
addsd
24(%rcx),
%×mm6
movsd
%xmm6,
24(%rcx)
movapd
%xmm3,
%×mm6
mulsd
%×mm0,
%xmm6
addsd
24(%rbp),
%xmm6
movsd
%×mm6,
24(%rbp)
movapd
%xmm2,
%xmm6
mulsd
%×mm0,
%xmm6
mulsd
%xmm1,
%xmm0
addsd
24(%rdx),
%xmm6
movsd
%xmm6,
24(%rdx)
addsd
24(%rax),

%×mm0	
	movsd
	%xmm0,
24(%rax)	
	movsd
	(%rbx,
%r13), %xmm	
	mulsd
	%xmm0,
%xmm5	
	mulsd
	%xmm0,
%xmm4	
	mulsd
	%xmm0,
%xmm3	
	mulsd
	%xmm0,
%xmm2	
	addsd
	32(%r12),
%xmm5	
	mulsd
	%xmm1,
%×mm0	
	movsd
	%xmm5,
32(%r12)	
	addsd
	32(%rcx),
%×mm4	
	movsd
	%xmm4,
32(%rcx)	
	addsd
	32(%rbp),
%×mm3	
	movsd
22/2/1	%xmm3,
32(%rbp)	
	addsd
242	32(%rdx),
%xmm2	maa-l
	movsd
00/0/	%xmm2,
32(%rdx)	addad
	addsd
0/	32(%rax),
%×mm0	moved
	movsd
20/0/	%xmm0,
32(%rax)	moved
	movsd
%×mm5	-16(%r11),
/%XIIIIII)	

movd		
## wovsd		movq
## wovsd		-16(%r10),
-16(%r9), %xmm4  movapd %xmm5, %xmm1  movsd -16(%r8), %xmm3  movsd (%rbx, %r14), %xmm0  movq 16(%rsp), %r13  movsd -16(%rdi), %xmm0  %xmm0, %xmm1  movapd %xmm5, %xmm6  addsd (%r12), %xmm1  movsd %xmm1, (%r12)  movapd %xmm1, (%rcx), %xmm1  addsd (%rcx), %xmm1  movsd %xmm1, (%rcx)  movapd %xmm1, (%rcx)  movap	%rbx	
-16(%r9), %xmm4  movapd %xmm5, %xmm1  movsd -16(%r8), %xmm3  movsd (%rbx, %r14), %xmm0  movq 16(%rsp), %r13  movsd -16(%rdi), %xmm0  %xmm0, %xmm1  movapd %xmm5, %xmm6  addsd (%r12), %xmm1  movsd %xmm1, (%r12)  movapd %xmm1, (%rcx), %xmm1  addsd (%rcx), %xmm1  movsd %xmm1, (%rcx)  movapd %xmm1, (%rcx)  movap		movsd
%xmm4		
movapd %xmm5,	0/2000 4	±0(/01 3),
%xmm1  movsd	%XIIIII4	
%xmm1		
movsd		%×mm5,
movsd	%xmm1	
-16(%r8), %xmm3 movsd (%rbx, %r14), %xmm0 movq 16(%rsp), %r13 movsd -16(%rdi), %xmm2 mulsd %xmm0, %xmm1 movapd %xmm5, %xmm6 addsd (%r12), %xmm1 movsd %xmm1, (%r12) movapd %xmm0, %xmm1 mulsd %xmm4, %xmm1 addsd (%rcx), %xmm1 movsd %xmm1 mulsd %xmm4, %xmm1 addsd (%rcx), %xmm1 movsd %xmm0, %xmm1 addsd %xmm0, %xmm1 addsd %xmm1, (%rcx) movapd %xmm0, %xmm1 addsd %xmm3, %xmm1 addsd %xmm3, %xmm1 addsd %xmm3, %xmm1 addsd %xmm3, %xmm1		movsd
%xmm3		
movsd (%rbx, %r14), %xmm0 movq	0/:	-10(7010),
(%rbx, %r14), %xmme movq 16(%rsp), %r13 movsd -16(%rdi), %xmm2 mulsd %xmmo, %xmm1 movapd %xmm5, %xmm6 addsd (%r12), %xmm1 movsd %xmm1, (%r12) movapd %xmmo, %xmm1 mulsd %xmm4, %xmm1 addsd (%rcx), %xmm1 movsd %xmm1, (%rcx) movapd %xmmo, %xmm1 addsd (%rcx), %xmm1 movsd %xmm1, (%rcx) movapd %xmmo, %xmm1 addsd (%rcx), %xmm1 movsd %xmm1, (%rcx) movapd %xmm0, %xmm1 addsd o(%rbp), %xmm1 addsd o(%rbp), %xmm1	%XIIIII3	
%r14), %xmm0 movq 16(%rsp), %r13 movsd -16(%rdi), %xmm2 mulsd %xmm0, %xmm1 movapd %xmm5, %xmm6 addsd (%r12), %xmm1 movsd %xmm1, (%r12) movapd %xmm0, %xmm1 mulsd %xmm4, %xmm1 addsd (%rcx), %xmm1 movsd %xmm1, (%rcx) movapd %xmm0, %xmm1 addsd (%rcx), %xmm1 addsd (%rcx), %xmm1 mulsd %xmm0, %xmm1, (%rcx) movapd %xmm0, %xmm1 addsd (%rcx), %xmm1 addsd %xmm0, %xmm1 addsd %xmm0, %xmm1		
movq 16(%rsp), %r13  movsd -16(%rdi), %xmm2  mulsd %xmm0, %xmm1  movapd %xmm5, %xmm6  addsd (%r12), %xmm1  movsd %xmm1, (%r12)  movapd %xmm0, %xmm1  mulsd %xmm4, %xmm1  addsd (%rcx), %xmm1  movsd %xmm1, (%rcx)  movapd %xmm0, %xmm1  addsd (%rcx), %xmm1  mulsd %xmm0, %xmm1  addsd (%rcx), %xmm1  mulsd %xmm0, %xmm1  addsd %xmm3, %xmm1  addsd %xmm3, %xmm1  addsd %xmm3, %xmm1  addsd %ymm3, %xmm1		(%rbx,
movq 16(%rsp), %r13  movsd -16(%rdi), %xmm2  mulsd %xmm0, %xmm1  movapd %xmm5, %xmm6  addsd (%r12), %xmm1  movsd %xmm1, (%r12)  movapd %xmm0, %xmm1  mulsd %xmm4, %xmm1  addsd (%rcx), %xmm1  movsd %xmm1, (%rcx)  movapd %xmm0, %xmm1  addsd (%rcx), %xmm1  mulsd %xmm0, %xmm1  addsd (%rcx), %xmm1  mulsd %xmm0, %xmm1  addsd %xmm3, %xmm1  addsd %xmm3, %xmm1  addsd %xmm3, %xmm1  addsd %ymm3, %xmm1	%r14)	, %×mm0
16(%rsp),   %r13		
%r13		
movsd -16(%rdi), %xmm2  mulsd %xmm0, %xmm1  movapd %xmm5, %xmm6  addsd (%r12), %xmm1  movsd %xmm1, (%r12)  movapd %xmm0, %xmm1  mulsd %xmm4, %xmm1  addsd (%rcx), %xmm1  movsd %xmm1, (%rcx) %xmm1  mulsd %xmm0, %xmm1  addsd (%rcx), %xmm1  movsd %xmm1, (%rcx) %xmm1  movsd %xmm1, (%rcx)  movapd %xmm0, %xmm1  addsd (%rcx), %xmm1  mulsd %xmm0, %xmm1  addsd 0(%rbp), %xmm1	0/210	±υ(/01 3μ),
-16(%rdi), %xmm2  mulsd %xmm6, %xmm1  movapd %xmm5, %xmm1  movsd %xmm1, (%r12)  movapd %xmm0, %xmm1  mulsd %xmm4, %xmm1  addsd (%rcx), %xmm1  movsd %xmm4, %xmm1  mulsd %xmm4, %xmm1  addsd (%rcx), %xmm1  movapd %xmm0, %xmm1  addsd (%rcx), %xmm1  addsd (%rcx), %xmm1  movapd %xmm0, %xmm1  addsd %xmm3, %xmm1  addsd %xmm3, %xmm1	%113	
%xmm2       mulsd %xmm0,         %xmm1       movapd %xmm5,         %xmm6       addsd (%r12),         %xmm1       movsd %xmm1,         (%r12)       movapd %xmm0,         %xmm1       mulsd %xmm4,         %xmm1       movsd %xmm1,         (%rcx),       %xmm1,         (%rcx)       movapd %xmm0,         %xmm1       mulsd %xmm3,         %xmm1       addsd %(%rbp),		
%xmm2       mulsd %xmm0,         %xmm1       movapd %xmm5,         %xmm6       addsd (%r12),         %xmm1       movsd %xmm1,         (%r12)       movapd %xmm0,         %xmm1       mulsd %xmm4,         %xmm1       movsd %xmm1,         (%rcx),       %xmm1,         (%rcx)       movapd %xmm0,         %xmm1       mulsd %xmm3,         %xmm1       addsd %(%rbp),		-16(%rdi),
mulsd %xmm0, %xmm1  movapd %xmm5, %xmm6  addsd (%r12), %xmm1  movsd %xmm1, (%r12)  movapd %xmm0, %xmm1  addsd (%rcx), %xmm1  movsd %xmm1, (%rcx)  movapd %xmm1, (%rcx)  addsd (%rcx), %xmm1  movsd %xmm1, (%rcx)  addsd (%rcx), %xmm1  addsd %xmm3, %xmm1  addsd %xmm3, %xmm1  addsd %xmm3, %xmm1	%xmm2	
%xmm1  movapd %xmm5,  %xmm6  addsd (%r12),  %xmm1  movsd %xmm1, (%r12)  movapd %xmm0,  %xmm1  addsd (%rcx),  %xmm1  movsd %xmm1, (%rcx)  movapd %xmm0,  %xmm1  addsd %xmm1, (%rcx)  movapd %xmm0, %xmm1, (%rcx)  movapd %xmm0, %xmm1		mulsd
%xmm1		
movapd	0/54	/0AIIIIIO <sub>1</sub>
%xmm5, %xmm6  addsd (%r12), %xmm1  movsd %xmm1, (%r12)  movapd %xmm0, %xmm1  addsd (%rcx), %xmm1  movsd %xmm1, (%rcx)  movapd %xmm1, (%rcx)  addsd (%rcx), %xmm1  addsd (%rcx), %xmm1  addsd %xmm1, (%rcx)  movapd %xmm0, %xmm1  addsd %xmm3, %xmm1  addsd %xmm3, %xmm1	%XMM1	
%xmm6		
%xmm6		%xmm5,
addsd (%r12), %xmm1  movsd %xmm1, (%r12)  movapd %xmm0, %xmm1  addsd (%rcx), %xmm1  movsd %xmm1, (%rcx)  movapd %xmm0, %xmm1  addsd 0(%rbp), %xmm1	%xmm6	
(%r12), %xmm1  movsd %xmm1, (%r12)  movapd %xmm0, %xmm1  mulsd %xmm4, %xmm1  addsd (%rcx), %xmm1  movsd %xmm1, (%rcx)  movapd %xmm0, %xmm1  addsd %xmm3, %xmm1  mulsd %xmm3, %xmm1  addsd %xmm3, %xmm1  addsd %xmm1, %xmm1		addsd
%xmm1       movsd %xmm1,         (%r12)       movapd %xmm0,         %xmm1       mulsd %xmm4,         %xmm1       addsd (%rcx),         %xmm1       movsd %xmm1,         (%rcx)       movapd %xmm0,         %xmm1       mulsd %xmm0,         %xmm1       addsd 0(%rbp),         %xmm1       addsd 0(%rbp),		
movsd	0/s/mm1	(701 12),
(%r12)  movapd %xmm0,  %xmm1  mulsd %xmm4,  %xmm1  addsd (%rcx),  %xmm1  movsd %xmm1,  (%rcx)  movapd %xmm0,  %xmm1  addsd %xmm0,  %xmm1  mulsd %xmm3,  %xmm1  addsd 0(%rbp), %xmm1	%XIIIII1	
(%r12)		
movapd %xmm0, %xmm1  mulsd %xmm4, %xmm1  addsd (%rcx), %xmm1  movsd %xmm1, (%rcx)  movapd %xmm0, %xmm1  aulsd %xmm3, %xmm1  addsd 0(%rbp), %xmm1		%×mm1,
movapd %xmm0, %xmm1  mulsd %xmm4, %xmm1  addsd (%rcx), %xmm1  movsd %xmm1, (%rcx)  movapd %xmm0, %xmm1  aulsd %xmm3, %xmm1  addsd 0(%rbp), %xmm1	(%r12	)
%xmm0, %xmm1  mulsd %xmm4, %xmm1  addsd (%rcx), %xmm1  movsd %xmm1, (%rcx)  movapd %xmm0, %xmm1  aulsd %xmm3, %xmm1  addsd 0(%rbp), %xmm1	`	
%xmm1  mulsd %xmm4,  %xmm1  addsd (%rcx),  %xmm1  movsd %xmm1, (%rcx)  movapd %xmm0,  %xmm1  addsd %xmm3,  %xmm1		
mulsd %xmm4, %xmm1 addsd (%rcx), %xmm1 movsd %xmm1, (%rcx) movapd %xmm0, %xmm1 addsd 0(%rbp),	0/s/mm1	70×111110,
%xmm1  addsd (%rcx),  %xmm1  movsd %xmm1, (%rcx)  movapd %xmm0,  %xmm1  addsd %xmm3,  %xmm1  addsd 0(%rbp),	%XIIIII1	7 - 1
%xmm1  addsd (%rcx),  %xmm1  movsd %xmm1, (%rcx)  movapd %xmm0,  %xmm1  addsd %xmm3,  %xmm1  addsd 0(%rbp), %xmm1		
addsd (%rcx), %xmm1  movsd %xmm1, (%rcx)  movapd %xmm0, %xmm1  mulsd %xmm3, %xmm1  addsd 0(%rbp), %xmm1		%xmm4,
(%rcx), %xmm1  movsd %xmm1, (%rcx)  movapd %xmm0, %xmm1  mulsd %xmm3, %xmm1  addsd 0(%rbp),	%xmm1	
(%rcx), %xmm1  movsd %xmm1, (%rcx)  movapd %xmm0, %xmm1  mulsd %xmm3, %xmm1  addsd 0(%rbp),		addsd
<pre>%xmm1</pre>		
movsd %xmm1, (%rcx) movapd %xmm0, %xmm1 mulsd %xmm3, %xmm1	0/2/mm4	(701 67),
%xmm1, (%rcx)  movapd %xmm0, %xmm1  mulsd %xmm3, %xmm1  addsd 0(%rbp), %xmm1	%XIIIII1	ام ساید بسیم
(%rcx)  movapd %xmm0,  %xmm1  mulsd %xmm3,  %xmm1  addsd 0(%rbp), %xmm1		
movapd %xmm0, %xmm1 mulsd %xmm3, %xmm1 addsd 0(%rbp),		
movapd %xmm0, %xmm1 mulsd %xmm3, %xmm1 addsd 0(%rbp),	(%rcx	)
%xmm0, %xmm1  mulsd %xmm3, %xmm1  addsd 0(%rbp),		
%xmm1  mulsd %xmm3, %xmm1  addsd 0(%rbp),		
mulsd %xmm3, %xmm1 addsd 0(%rbp),	0/2/mm4	/0/mm0,
%xmm3, %xmm1 addsd 0(%rbp), %xmm1	%XIIIII1	7
%xmm1 addsd 0(%rbp), %xmm1		
%xmm1 addsd 0(%rbp), %xmm1		%xmm3,
addsd 0(%rbp), %xmm1	%xmm1	
0(%rbp), %xmm1		hahha
%xmm1		
	0/	o(∿inh),
movsd	%XMM1	
		movsd

	0/v/mm1
0/0/rhn)	%xmm1,
0(%rbp)	
	movapd
	%xmm0,
%xmm1	
	mulsd
	%xmm2,
%xmm1	
	addsd
	(%rdx),
%xmm1	
	movsd
	%xmm1,
(%rdx)	,
(10.011)	movsd
	-16(%rsi),
%×mm1	±0(/01 0±//
/0XIIIIII	mulsd
%×mm0	%×mm1,
%XIIIIIU	04404
	addsd
	(%rax),
%×mm0	
	movsd
	%xmm0,
(%rax)	
	movsd
	(%rbx,
%r13),	
	mulsd
	%×mm0,
%×mm6	,
	addsd
	8(%r12),
%×mm6	- ( / /
	movsd
	%×mm6,
8(%r12)	70/XIIIIIO <b>,</b>
0(70112)	movapd
	%xmm4,
%×mm6	/0AIIIII14 <sub>,</sub>
%XIIIM6	mulad
	mulsd
	%xmm0,
%xmm6	
	addsd
	8(%rcx),
%×mm6	
	movsd
	%xmm6,
8(%rcx)	
` '	movapd
	%xmm3,
%×mm6	,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,
/oximio	mulsd
	%xmm0,
	/0/\lillio <sub> </sub>

04	
%xmm6	
	addsd
	8(%rbp),
%xmm6	
	movsd
	%xmm6,
0/%rhn)	70XIIIIIO,
8(%rbp)	
	movapd
	%xmm2,
%×mm6	
	mulsd
	%xmm0,
%×mm6	,
707111110	mulsd
	%xmm1,
%×mm0	
	addsd
	8(%rdx),
%×mm6	` ''
/oximito	movsd
	%xmm6,
8(%rdx)	
	movapd
	%xmm5,
%xmm6	·
	addsd
	8(%rax),
%×mm0	ο (πιαχ),
%XIIIIIU	
	movsd
	%xmm0,
8(%rax)	
	movsd
	(%rbx,
%r15),	
701 13),	
	mulsd
	%xmm0,
%xmm6	
	addsd
	16(%r12),
%×mm6	` ','
	movsd
10/0/ 10	%xmm6,
16(%r12	
	movapd
	%xmm4,
%xmm6	
	mulsd
	%×mm0,
%×mm6	/o/mino/
%XIIIII10	المامام
	addsd
	16(%rcx),
%xmm6	
	movq
	(%rsp),
%r13	(701 37)
/oI ±3	

### ##################################			movsd
16(%rcx)			
movapd		10(0/20)	%XIIIIIO,
%xmm6  mulsd %xmm0, %xmm6  addsd 16(%rbp), %xmm6  movsd %xmm6, 16(%rbp)  movapd %xmm2, %xmm6  mulsd %xmm0, %xmm6  addsd 16(%rdx), %xmm6  movsd %xmm6, 16(%rdx)  movapd %xmm6, 16(%rdx)  movapd %xmm5, %xmm6  addsd 16(%rax), %xmm6  addsd 16(%rax), %xmm6  addsd 16(%rax), %xmm0  movapd %xmm5, %xmm6  addsd 16(%rax), %xmm0  movapd %xmm5, %xmm6  addsd 16(%rax), %xmm0  movapd %xmm0, 16(%rax)  movapd %xmm0, %xmm0  addsd addsd addsd addsd		10(%rcx)	
## ## ## ## ## ## ## ## ## ## ## ## ##			
mulsd %xmm6  addsd 16(%rbp), %xmm6  movsd %xmm6, 16(%rbp)  movapd %xmm2, %xmm6  mulsd %xmm0, %xmm6  mulsd %xmm1, %xmm0  addsd 16(%rdx), %xmm6  movsd %xmm6, 16(%rdx)  movapd %xmm5, %xmm6  addsd 16(%rax), %xmm0  movapd %xmm5, %xmm0  movapd %xmm6  addsd 16(%rax), %xmm0  movapd %xmm0, 16(%rax) movapd %xmm0, 16(%rax) %xmm0, 16(%rax) movapd %xmm0, 16(%rax) %xmm0, 16(%rax) movapd			%xmm3,
%xmm6       addsd         addsd       16(%rbp),         %xmm6       movsd         16(%rbp)       movapd         %xmm6,       mulsd         %xmm0,       %xmm0,         %xmm6       mulsd         %xmm1,       %xmm0         addsd       16(%rdx),         %xmm6       movsd         %xmm0       movsd         %xmm0,       movsd         %r13), %xmm0       movd         %(%rsp),       %r13         mulsd       %xmm0,         %xmm0,       addsd		%xmm6	
%xmm6       addsd         16(%rbp),       movsd         %xmm6,       16(%rbp)         movapd       %xmm2,         %xmm6       mulsd         %xmm0,       %xmm0,         %xmm0       addsd         ad(%rdx),       %xmm6,         16(%rdx)       movapd         %xmm6,       16(%rax),         %xmm0       movsd         %xmm0,       16(%rax),         %xmm0,       16(%rax),         %r13), %xmm0       movq         %r13       mulsd         %xmm0,       mulsd         %xmm0,       mulsd         %xmm0,       mulsd         %xmm0,       %xmm0,			
addsd 16(%rbp),  %xmm6  movsd %xmm6,  16(%rbp)  movapd %xmm2,  %xmm6  mulsd %xmm1,  %xmm0  addsd 16(%rdx),  %xmm6  movsd %xmm6,  16(%rdx)  movapd %xmm5,  %xmm6  addsd 16(%rdx),  %xmm6  movsd %xmm6,  16(%rdx)  movapd %xmm5,  %xmm6  addsd 16(%rax),  %xmm0  movsd %xmm0,  16(%rax)  movsd %xmm0,  %xmm0  addsd 16(%rax),  %xmm6  addsd 16(%rax),  %xmm0  movsd %xmm0,  %xmm0  movsd %xmm0,  %xmm0  addsd 16(%rax),  %xmm0  movsd %xmm0,  %xmm0  movsd %xmm0,  %xmm0  addsd addsd			%xmm0,
16(%rbp), %xmm6 movsd %xmm6, 16(%rbp) movapd %xmm2, %xmm6 mulsd %xmm0, %xmm1, %xmm0 addsd 16(%rdx), %xmm6 movsd %xmm6, 16(%rdx) movapd %xmm5, %xmm6 addsd 16(%rax), %xmm0 movsd %xmm0, 16(%rax) movsd %xmm0, %xmm0 addsd 16(%rax), %xmm0 movsd %xmm0, %xmm0 addsd 16(%rax), %xmm6 addsd 16(%rax), %xmm0 addsd %xmm0, %xmm6		%xmm6	
%xmm6         16(%rbp)         movapd         %xmm2,         %xmm6         mulsd         %xmm0,         %xmm6         movsd         %xmm6,         16(%rdx)         movapd         %xmm5,         %xmm6         addsd         16(%rax),         %xmm0         movsd         %xmm0,         16(%rax)         movd         (%rbx,         %r13,       mulsd         %xmm0,         %xmm6			addsd
%xmm6         16(%rbp)         movapd         %xmm2,         %xmm6         mulsd         %xmm0,         %xmm6         movsd         %xmm6,         16(%rdx)         movapd         %xmm5,         %xmm6         addsd         16(%rax),         %xmm0         movsd         %xmm0,         16(%rax)         movd         (%rbx,         %r13,       mulsd         %xmm0,         %xmm6			16(%rbp),
%xmm6,         16(%rbp)         movapd         %xmm2,         %xmm6         mulsd         %xmm1,         %xmm0         addsd         16(%rdx),         %xmm6         movapd         %xmm5,         %xmm6         addsd         16(%rax),         %xmm0         movsd         (%rbx,         %r13), %xmm0         movq         8(%rsp),         %r13         mulsd         %xmm0,         %xmm0,		%xmm6	
16(%rbp)			movsd
16(%rbp)			%xmm6,
movapd %xmm2, %xmm6  mulsd %xmm0, %xmm6  mulsd %xmm1, %xmm0  addsd 16(%rdx), %xmm6  movsd %xmm6, 16(%rdx)  movapd %xmm5, %xmm6  addsd 16(%rax), %xmm0  movsd (%rbx, %xmm0, 16(%rax)  movsd (%rbx, %r13), %xmm0  movq 8(%rsp), %r13  mulsd %xmm0, %xmm6  addsd		16(%rbp)	,
%xmm6 mulsd %xmm0, %xmm6 mulsd %xmm1, %xmm0 addsd 16(%rdx), %xmm6 movsd %xmm6, 16(%rdx)  addsd 16(%rax), %xmm6 movapd %xmm5, %xmm6 addsd 16(%rax), %xmm0 movsd %xmm0, 16(%rax) movsd %xmm0, 16(%rax) %xmm0 addsd addsd addsd addsd addsd addsd %xmm0, %xmm0 addsd addsd addsd addsd		, ,	movand
%xmm6       mulsd         %xmm0,       mulsd         %xmm1,       %xmm1,         %xmm0       addsd         d6(%rdx),       %xmm6,         16(%rdx)       movapd         %xmm5,       %xmm6         addsd       16(%rax),         %xmm0       movsd         %xmm0,       16(%rax)         %r13), %xmm0       movq         %(%rbx,       %r13         mulsd       %xmm0,         %xmm0,       %xmm0,         %xmm0,       %xmm0,			
mulsd %xmm0, %xmm6  mulsd %xmm1, %xmm0  addsd 16(%rdx), %xmm6  movsd %xmm6, 16(%rdx)  movapd %xmm5, %xmm6  addsd 16(%rax), %xmm0  movsd %xmm0, 16(%rax)  movsd %xmm0, 16(%rax)  movsd %xmm0, 16(%rax)  movsd %xmm0, 16(%rax)  movd %xmm0, 16(%rax)  movd and sddsd  %xmm0  movq and sddsd  %xmm0, %xmm6  addsd		%xmm6	· · · · · · · · · · · · · · · · · · ·
%xmm0, %xmm6 mulsd %xmm1, %xmm0 addsd 16(%rdx), %xmm6 movsd %xmm6, 16(%rdx) movapd %xmm5, %xmm6 addsd 16(%rax), %xmm0 movsd %xmm0, 16(%rax) movsd %xmm0, movq %xmm0, %xmm6		, 3,	helum
%xmm6       mulsd         %xmm1,       %xmm1,         %xmm0       addsd         movsd       %xmm6,         16(%rdx)       movapd         %xmm5,       %xmm6         addsd       16(%rax),         %xmm0       movsd         %xmm0,       16(%rax)         %r13), %xmm0       movq         %xr13), %xmm0       movq         %xmm0,       %xmm0,         %xmm0       addsd			
mulsd %xmm1, %xmm0 addsd 16(%rdx), %xmm6 movsd %xmm6, 16(%rdx) movapd %xmm5, %xmm6 addsd 16(%rax), %xmm0 movsd %xmm0, 16(%rax) movsd (%rbx, %r13), %xmm0 movq 8(%rsp), %r13 mulsd %xmm0, %xmm6 addsd		%vmm6	/0/1111110 <sub>1</sub>
%xmm0  addsd 16(%rdx), %xmm6  movsd %xmm5, %xmm6  addsd 16(%rax), %xmm0  movsd %xmm0, 16(%rax)  movsd (%rbx, %r13), %xmm0  movq ag(%rsp), %r13  mulsd %xmm0, %xmm6  addsd addsd		/0/1111110	muled
%xmm0       addsd 16(%rdx),         %xmm6       movsd %xmm6,         16(%rdx)       movapd %xmm5,         %xmm6       addsd 16(%rax),         %xmm0       movsd %xmm0,         16(%rax)       movsd (%rbx,         %r13), %xmm0       movq 8(%rsp),         %r13       mulsd %xmm0,         %xmm6       addsd			
addsd 16(%rdx), %xmm6  movsd %xmm6,  16(%rdx)  movapd %xmm5,  %xmm6  addsd 16(%rax), %xmm0  movsd %xmm0,  16(%rax)  movsd (%rbx, %r13), %xmm0  movq 8(%rsp), %r13  mulsd %xmm0, %xmm6  addsd		0/2 /	%XIIIII11,
16(%rdx), %xmm6  movsd %xmm6,  16(%rdx)  movapd %xmm5,  %xmm6  addsd 16(%rax), %xmm0  movsd %xmm0,  16(%rax)  movsd (%rbx, %r13), %xmm0  movq 8(%rsp), %r13  mulsd %xmm0, %xmm6  addsd		%xmmo	
%xmm6         movsd         %xmm6,         16(%rdx)         movapd         %xmm5,         %xmm6         addsd         16(%rax),         %xmm0,         16(%rax)         movsd         (%rbx,         %r13), %xmm0         movq         8(%rsp),         %r13         mulsd         %xmm0,         %xmm0,         %xmm0,			
movsd %xmm6, 16(%rdx) movapd %xmm5, %xmm6 addsd 16(%rax), %xmm0 movsd %xmm0, 16(%rax) movsd (%rbx, %r13), %xmm0 movq 8(%rsp), %r13 mulsd %xmm0, %xmm6 addsd			16(%rdx),
%xmm6, 16(%rdx)  movapd %xmm5,  %xmm6  addsd 16(%rax),  %xmm0  movsd %xmm0, 16(%rax)  movsd (%rbx, %r13), %xmm0  movq 8(%rsp), %r13  mulsd %xmm0, %xmm6  addsd		%xmm6	
16(%rdx)  movapd %xmm5,  %xmm6  addsd 16(%rax),  %xmm0  movsd %xmm0,  16(%rax)  movsd (%rbx, %r13), %xmm0  movq 8(%rsp),  %r13  mulsd %xmm0,  %xmm6  addsd			
movapd %xmm5, %xmm6  addsd 16(%rax), %xmm0  movsd %xmm0, 16(%rax)  movsd (%rbx, %r13), %xmm0  movq 8(%rsp), %r13  mulsd %xmm0, %xmm6  addsd			%xmm6,
%xmm5, %xmm6  addsd 16(%rax), %xmm0  movsd %xmm0, 16(%rax)  movsd (%rbx, %r13), %xmm0  movq 8(%rsp), %r13  mulsd %xmm0, %xmm6  addsd		16(%rdx)	
%xmm6			
addsd 16(%rax), %xmm0  movsd %xmm0, 16(%rax)  movsd (%rbx, %r13), %xmm0  movq 8(%rsp), %r13  mulsd %xmm0, %xmm0, %xmm6  addsd			%xmm5,
16(%rax), %xmm0  movsd %xmm0, 16(%rax)  movsd (%rbx, %r13), %xmm0  movq 8(%rsp), %r13  mulsd %xmm0, %xmm0, %xmm6		%xmm6	
%xmm0  movsd %xmm0,  16(%rax)  movsd (%rbx, %r13), %xmm0  movq 8(%rsp), %r13  mulsd %xmm0, %xmm6  addsd			addsd
%xmm0  movsd %xmm0,  16(%rax)  movsd (%rbx, %r13), %xmm0  movq 8(%rsp), %r13  mulsd %xmm0, %xmm6  addsd			16(%rax),
movsd %xmm0, 16(%rax) movsd (%rbx, %r13), %xmm0 movq 8(%rsp), %r13 mulsd %xmm0, %xmm6		%×mm0	
%xmm0, 16(%rax)  movsd (%rbx, %r13), %xmm0  movq 8(%rsp), %r13  mulsd %xmm0, %xmm6  addsd			movsd
16(%rax)  movsd (%rbx, %r13), %xmm0  movq 8(%rsp), %r13  mulsd %xmm0, %xmm6			
movsd (%rbx, %r13), %xmm0 movq 8(%rsp), %r13 mulsd %xmm0, %xmm6		16(%rax)	,
(%rbx, %r13), %xmm0 movq 8(%rsp), %r13 mulsd %xmm0, %xmm6		_ ` ′	movsd
%r13), %xmm0  movq 8(%rsp), %r13  mulsd %xmm0, %xmm6			
movq 8(%rsp), %r13 mulsd %xmm0, %xmm6		%r13), %xmr	
8(%rsp), %r13 mulsd %xmm0, %xmm6			
%r13 mulsd %xmm0, %xmm6 addsd			
mulsd %xmm0, %xmm6 addsd		%r13	5 (70. SP) /
%xmm0, %xmm6 addsd		/3. 10	mulsd
%xmm6 addsd			
addsd		%×mm6	/0/1111110 <sub>1</sub>
		/0/1111110	hahhe
		0/24mm C	24(%r12),
%xmm6		%XMM6	
movsd			
%×mm6,			%xmm6,
24(%r12)		24(%r12)	
movapd			

0/s cmama 4	
%xmm4,	
%xmm6	
mulsd	
%×mm0,	
%xmm6	
addsd	
24(%rcx	١
%xmm6	),
movsd	
%xmm6,	
24(%rcx)	
movapd	
%xmm3,	
%×mm6	
mulsd	
%×mm0,	
%xmm6	
addsd	
	,
24(%rbp	),
%xmm6	
movsd	
%xmm6,	
24(%rbp)	
movapd	
%xmm2,	
%xmm6	
mulsd	
%xmm0,	
%xmm6	
mulsd	
%xmm1,	
%×mm0	
addsd	
24(%rdx	).
%xmm6	,,
movsd	
%xmm6,	
24(%rdx)	
addsd	,
24(%rax	),
%xmm0	
movsd	
%×mm0,	
24(%rax)	
movsd	
(%rbx,	
%r13), %xmm0	
mulsd	
%×mm0,	
%xmm5	
mulsd	
%xmm0,	
%xmm4	
mulsd	
%xmm0,	

%xmm3	
	mulsd
	%xmm0,
%×mm2	
	addsd
04	32(%r12),
%xmm5	mulsd
	%xmm1,
%×mm0	/0XIIIII <b>1</b> ,
707	movsd
	%xmm5,
32(%r1	
	addsd
	32(%rcx),
%×mm4	
	movsd
22/0/5	%xmm4,
32(%rc	addsd
	32(%rbp),
%×mm3	02 (701 SP))
	movsd
	%xmm3,
32(%rb	
	addsd
	32(%rdx),
%×mm2	
	movsd %vmm2
32(%rc	%xmm2,
02(701.0	addsd
	32(%rax),
%×mm0	` ',
	movsd
	%xmm0,
32(%ra	
	movsd
%×mm5	-8(%r11),
Cililization	movq
	-8(%r10),
%rbx	
	movsd
	-8(%r9),
%xmm4	
	movapd
  %×mm1	%xmm5,
/0XIIIIII	movsd
	-8(%r8),
%×mm3	3(70.3)
	movsd
	(%rbx,
%r14),	%×mm0

	max.::
	movq
	16(%rsp),
%	r13
	movsd
	-8(%rdi),
%:	xmm2
	mulsd
	%×mm0,
0/2	xmm1
70.	movapd
	%xmm5,
%.	xmm6
	addsd
	(%r12),
%	×mm1
	movsd
	%×mm1,
	%r12)
	movapd
	%xmm0,
0/-	xmm1
/0.	mulsd
	%xmm4,
%	xmm1
	addsd
	(%rcx),
%:	xmm1
	movsd
	%xmm1,
	%rcx)
	movapd
	%xmm0,
000	
%.	xmm1
	mulsd
	%xmm3,
%:	xmm1
	addsd
	0(%rbp),
%:	xmm1
	movsd
	%×mm1,
 	(%rbp)
	movapd
	%xmm0,
0/-	xmm1
%.	
	mulsd
	%xmm2,
%	xmm1
	addsd
	(%rdx),
%	xmm1
	movsd
	%xmm1,
	%rdx)
	movsd

0/0/roi\
-8(%rsi),
%xmm1
mulsd
%xmm1,
%×mm0
addsd
(%rax),
%×mm0
movsd
%xmm0,
(%rax)
movsd
(%rbx,
%r13), %xmm0
mulsd
%xmm0,
%xmm6
addsd
8(%r12),
%×mm6
movsd
%xmm6,
8(%r12)
movapd
%xmm4,
%×mm6
mulsd
%×mm0,
%xmm6
addsd
8(%rcx),
%xmm6
movsd
%xmm6,
8(%rcx)
movapd
%xmm3,
%xmm6
mulsd
%×mm0,
%×mm6
addsd
8(%rbp),
%×mm6
movsd
%xmm6,
8(%rbp)
movapd
%xmm2,
%xmm6
mulsd
%xmm0,
%×mm6
mulsd
%xmm1,
· · · · · · · · · · · · · · · · · · ·

%×mm0	
	addsd
	8(%rdx),
%×mm6	` ''
ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ	movsd
	%xmm6,
8(%rdx)	
	movapd
	%xmm5,
%xmm6	
	addsd
	8(%rax),
%×mm0	J(/01 ux ),
/oxiiiiiu	mayad
	movsd
	%xmm0,
8(%rax)	
	movsd
	(%rbx,
%r15),	
/// 13),	mulsd
	%xmm0,
%xmm6	
	addsd
	16(%r12),
%xmm6	. , ,
"""	movsd
	%×mm6,
16/0/515	
16(%r12	
	movapd
	%xmm4,
%xmm6	
	mulsd
	%xmm0,
%×mm6	,
ZOZIMIO	addsd
	10/0/534
	16(%rcx),
%×mm6	
	movq
	(%rsp),
%r13	
	movsd
	%xmm6,
10/0/	
16(%rcx	
	movapd
	%xmm3,
%xmm6	
	mulsd
	%×mm0,
%×mm6	, , , , , , , , , , , , , , , , , , ,
/oXIIIIIO	addad
	addsd
	16(%rbp),
%xmm6	
	movsd
	%xmm6,
16(%rbp	
10(701 0)	' /

		movend
		movapd
		%xmm2,
	%×mm6	
		mulsd
		%xmm0,
	%xmm6	,
	, , , , , , , , , , , , , , , , , , , ,	mulsd
	0/ 0	%xmm1,
	%×mm0	
		addsd
		16(%rdx),
	%xmm6	
		movsd
		%xmm6,
	16(%rdx)	,
	_ (/ (/ ( ) / ( ) / ( ) / ( )	movapd
	0/vmmG	%xmm5,
	%×mm6	م ما ما م -ا
		addsd
		16(%rax),
	%×mm0	
		movsd
		%xmm0,
	16(%rax)	,
	10 (701 U.X.)	movsd
	0/1410) 0/1/1411	(%rbx,
	%r13), %xmm	
		movq
		8(%rsp),
	%r13	
		mulsd
		%xmm0,
	%xmm6	,
	707CIIIII <b>C</b>	addsd
		24(%r12),
	0/xmmG	24(/01 12),
	%xmm6	moved
		movsd
		%xmm6,
	24(%r12)	
		movapd
		%xmm4,
	%×mm6	
		mulsd
		%xmm0,
	%xmm6	, , , , , , , , , , , , , , , , , , ,
	/0/XIIIIIIU	addsd
	0/	24(%rcx),
	%xmm6	_
		movsd
		%xmm6,
	24(%rcx)	•
	` ,	movapd
		%xmm3,
	%×mm6	70XIIIIIO
	/0AIIIIIU	muled
		mulsd

	%xmm0,
%X	cmm6
	addsd
	24(%rbp),
%x	cmm6
	movsd
	%xmm6,
24	1(%rbp)
	movapd
	%xmm2,
%X	cmm6
	mulsd
	%×mm⊙,
%x	cmm6
	mulsd
	%xmm1,
0/_	cmm0
	addsd
04.	24(%rdx),
\\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\	cmm6
	movsd
	%xmm6,
24	1(%rdx)
	addsd
	24(%rax),
%x	cmm0
	movsd
	%×mm0,
24	
24	1(%rax)
	movsd
	(%rbx,
%r	13), %xmm0
	mulsd
	%×mm⊙,
%x	cmm5
	mulsd
	%xmm0,
%x	cmm4
	mulsd
	%xmm0,
0/5-	
%X	cmm3
	mulsd
	%xmm0,
%x	cmm2
	addsd
	32(%r12),
%x	cmm5
	mulsd
	%xmm1,
0/3/	∕⁄∧⊼IIIII⊥, (mm0
<sup>76X</sup>	
	movsd
	%xmm5,
32	2(%r12)
	addsd
	32(%rcx),
	\ //

	%xmm4	
		movsd
	22 (0/22)	%xmm4,
	32(%rcx)	addsd
		32(%rbp),
	%xmm3	
		movsd
	32(%rbp)	%xmm3,
	32 (%I DP)	addsd
		32(%rdx),
	%xmm2	
		movsd
	32(%rdx)	%xmm2,
	0=(//// 0///	addsd
		32(%rax),
	%×mm0	mayrad
		movsd %xmm0,
	32(%rax)	70XIIIII
	,	addl
	0.4 (0( )	\$5,
	24(%rsp)	movl
		24(%rsp),
	%ebx	
		cmpl
	%ebx	96(%rsp),
	<b>⊘</b> €IJX	jl
		.L9
		addl
	22(%ren)	\$5,
	32(%rsp)	addq
		\$40, %r12
		addq
		\$40, %rcx movl
		32(%rsp),
	%edi	(::: -
		addq
		\$40, %rbp
		addq \$40, %rdx
		addq
		\$40, %rax
		addq
		\$40, %r15 cmpl
		96(%rsp),
	%edi	
		jl

100/%ro	.L12 addl \$5,
100(%rs	addq \$40,
120(%rs	p) movl 100(%rsp),
%eax	addq \$40,
112(%rs	p) cmpl
%eax	96(%rsp), jl
.L11:	.L29 leaq
%rsi	160(%rsp), xorl
	%edi, %edi xorl
%r12d	%r12d, leaq
, %rbp	.LC3(%rip) call
clock_g	ettime@PLT

### **B) CÓDIGO FIGURA 1:**

CAPTURA CÓDIGO FUENTE: figura1-original.c

```
int X1, X2;
int R[40000];
struct timespec cgtl,
double
                  ncgt; //para tiempo de ejecución
for (int i = 0; i < 5000; ++i)
    s[i].a = i % 6;
    s[i].b = i % 12;
clock gettime(CLOCK_REALTIME,&cgt1);
for (int ii=0; ii<40000; ++ii){
    X1=0; X2=0;
     for(int i=0; i<5000;i++)
         X1+=2*s[i].a+ii;
    for(int i=0; i<5000;i++)</pre>
         X2+=3*s[i].b-ii;
    if (X1<X2)
         R[ii]=X1;
         R[ii]=X2;
clock_gettime(CLOCK_REALTIME,&cgt2);
ncgt = (double) (cgt2.tv sec-cgt1.tv sec)+
         (double) ((cgt2.tv_nsec-cgt1.tv_nsec)/(1.e+9));
printf("R[0]: %d\n", R[0]);
printf("m[39999]: %d\n", R[39999]);
printf("tiempo: %11.9f\n", ncgt);
```

1.1. MODIFICACIONES REALIZADAS (al menos dos modificaciones):

Modificación a) –explicación-: unir los dos bucles internos Modificación b) –explicación-: desenrollar el bucle interno de la modificación a

- 1.1. CÓDIGOS FUENTE MODIFICACIONES
- a) Captura figura1-modificado\_a.c

```
clock_gettime(CLOCK_REALTIME,&cgt1);

for (int ii=0; ii<40000; ++ii){
    X1=0; X2=0;

    for(int i=0; i<5000;i++)
    {
        X1+=2*s[i].a+ii;
        X2+=3*s[i].b-ii;
    }

    if (X1<X2)
        R[ii]=X1;
    else
        R[ii]=X2;
}

clock_gettime(CLOCK_REALTIME,&cgt2);</pre>
```

Capturas de pantalla (que muestren la compilación y que el resultado es correcto):

b)

```
clock_gettime(CLOCK_REALTIME,&cgt1);
for (int ii=0; ii<40000; ++ii){
   X1=0; X2=0;
    for(int i=0; i<5000;i+=5)
        X1+=2*s[i].a+ii;
        X1+=2*s[i+1].a+ii;
        X1+=2*s[i+2].a+ii;
        X1+=2*s[i+3].a+ii;
        X1+=2*s[i+4].a+ii;
        X2+=3*s[i].b-ii;
        X2+=3*s[i+1].b-ii;
        X2+=3*s[i+2].b-ii;
       X2+=3*s[i+3].b-ii;
X2+=3*s[i+4].b-ii;
        //_mm_prefetch(s[i+32], MM HINT T0);
   if (X1<X2)
       R[ii]=X1;
        R[ii]=X2;
:lock gettime(CLOCK REALTIME,&cgt2);
```

```
[JoseJavierAlonsoRamos jjavier98@jjavier98-Lenovo-ideapad-310-15IKB:~/G
II/2°/2° Semestre/AC/SEMINARIOS/Seminario_4/ejer1/B] 2018-05-28 lunes
$./f./figura1-original
R[0]: 24992
n[39999]: -199912548
tiempo: 0.307736419
[JoseJavierAlonsoRamos jjavier98@jjavier98-Lenovo-ideapad-310-15IKB:~/G
II/2°/2° Semestre/AC/SEMINARIOS/Seminario_4/ejer1/B] 2018-05-28 lunes
$./figura1-mod-1
R[0]: 24992
n[39999]: -199912548
tiempo: 0.225723487
[JoseJavierAlonsoRamos jjavier98@jjavier98-Lenovo-ideapad-310-15IKB:~/G
II/2°/2° Semestre/AC/SEMINARIOS/Seminario_4/ejer1/B] 2018-05-28 lunes
$./figura1-mod-2
R[0]: 24992
m[39999]: -199912548
tiempo: 0.177971926
```

#### **1.1. TIEMPOS:**

Modificación	-O2
Sin modificar	0,307736419
Modificación a)	0,225723487
Modificación b)	0,177971926
•••	

### 1.1. COMENTARIOS SOBRE LOS RESULTADOS:

# 1.2. CÓDIGO EN ENSAMBLADOR DEL ORIGINAL Y DE DOS MODIFICACIONES: (PONER AQUÍ SÓLO LA ZONA DEL CÓDIGO ENSAMBLADOR EVALUADA, USE COLORES PARA DESTACAR LAS DIFERENCIAS)

pmm-secuenc:	ial.s	pmm-secuenc modificado_	ial- o.s	pmm-secuencial- modificado_c.s	
call		call		call	
clock_getti	.me@PLT	clock_gettime@PLT c		clock_gettime@PLT	
	leaq		leaq		leaq
			48(%rsp),		48(%rsp),
40004+s(%ri	p), %r9	%r10		%r10	
	leaq		leaq		leaq
	48(%rsp),				
%r11		40000+s(%ri	p), %r8	3 40000+s(%rip), %r8	
	xorl		xorl		xorl
	%r10d,		%r9d, %r9d		%r9d, %r9d
%r10d			.p2align		.p2align
	leaq	4,,10		4,,10	
	-4(%r9),		.p2align 3		.p2align 3
%r8		.L3:		.L3:	
	.p2align		leaq		leaq
4,,10			s(%rip),		s(%rip),
	.p2align 3	%rax		%rax	
.L3:			movl		movl
	leaq		%r9d, %edi		%r9d, %edx
	s(%rip),		xorl		xorl

0/ray			%00y %00y		% ocy % ocy
%rax			%ecx, %ecx		%ecx, %ecx
	movl		xorl		xorl
	%r10d,		%esi, %esi		%edi, %edi
%edi			.p2align		.p2align
	xorl	4,,10		4,,10	
	%esi, %esi		.p2align 3		.p2align 3
	.p2align	.L4:		.L4:	
4,,10	, ,		movl		movl
', ', = '	.p2align 3		(%rax),		(%rax),
.L4:	ipzarryn o	%edx	(701 477)	%esi	(701 477)
	movl	/ocux	addq	70031	addq
			•		•
0/ = -1	(%rax),		\$8, %rax		\$40, %rax
%edx			leal		leal
	addq		(%rdi,		(%rdx,
	\$8, %rax	%rdx,2), %∈		%rsi,2), %	
	leal		addl		addl
	(%rdi,		%edx, %esi		%esi, %edi
%rdx,2),	%edx		movl		movl
	addl		-4(%rax),		-32(%rax),
	%edx, %esi	%edx	, , ,	%esi	, , ,
	cmpq		leal		leal
	%r8, %rax		(%rdx,		(%rdx,
	jne	%rdx,2), %e	•	%rsi,2), %	
	. L4	701 UX, Z ) , 70C	subl	701 31, 2), 7	addl
			%edi, %edx		
	leaq		addl		%esi, %edi
0/15 0 1/	4+s(%rip),				movl
%rax	vor1		%edx, %ecx		-24(%rax),
	xorl		cmpq	%esi	11
	%ecx, %ecx		%rax, %r8		leal
4 40	.p2align		jne	0/	(%rdx,
4,,10	.0.110		. L4	%rsi,2), %	
	.p2align 3		cmpl		addl
.L5:	_		%ecx, %esi		%edi, %esi
	movl		cmovl		movl
	(%rax),		%esi, %ecx		-16(%rax),
%edx			movl	%edi	
	addq		%ecx,		leal
	\$8, %rax	(%r10,%r9,4	1)		(%rdx,
	leal		addq	%rdi,2), %	Kedi
	(%rdx,		\$1, %r9		addl
%rdx,2),	%edx		cmpq		%edi, %esi
	subl		\$40000,		movl
	%edi, %edx	%r9			-8(%rax),
	addl		jne	%edi	, , , ,
	%edx, %ecx		.L3		leal
	cmpq		leaq		(%rdx,
	%r9, %rax		32(%rsp),	%rdi,2), %	
	jne	%rsi	( P / /	' ' ' '	addl
	. L5		xorl		%esi, %edi
	cmpl		%edi, %edi		movl
	%esi, %ecx		call		-36(%rax),
	cmovg		Juli	%esi	
	_	clock_getti	me@PIT	/0C3±	leal
	movl	JOHOUN_GCCCI	-mcer = 1		(%rsi,
	%ecx,			%rsi,2), %	• •
	/0 <b>℃</b> し入 ,			/OI 3エ, ∠ ) , %	IC3T

```
(%r11,%r10,4)
                                                             subl
                                                             %edx, %esi
            addq
            $1, %r10
                                                             addl
            cmpq
                                                             %esi, %ecx
            $40000,
                                                             movl
%r10
                                                             -28(%rax),
            jne
                                                %esi
            .L3
                                                             leal
            leaq
                                                             (%rsi,
            32(%rsp),
                                                %rsi,2), %esi
%rsi
                                                             subl
                                                             %edx, %esi
            xorl
            %edi, %edi
                                                             addl
            call
                                                             %esi, %ecx
                                                             movl
clock_gettime@PLT
                                                             -20(%rax),
                                                %esi
                                                             leal
                                                             (%rsi,
                                                %rsi,2), %esi
                                                             subl
                                                             %edx, %esi
                                                             addl
                                                             %ecx, %esi
                                                             movl
                                                             -12(%rax),
                                                %ecx
                                                             leal
                                                             (%rcx,
                                                %rcx,2), %ecx
                                                             subl
                                                             %edx, %ecx
                                                             addl
                                                             %ecx, %esi
                                                             movl
                                                             -4(%rax),
                                                %ecx
                                                             leal
                                                             (%rcx,
                                                %rcx,2), %ecx
                                                             subl
                                                             %edx, %ecx
                                                             addl
                                                             %esi, %ecx
                                                             cmpq
                                                             %rax, %r8
                                                             jne
                                                             .L4
                                                             cmpl
                                                             %ecx, %edi
                                                             cmovl
                                                             %edi, %ecx
                                                             movl
                                                             %ecx,
                                                 (%r10, %r9, 4)
```

addq	
\$1, %r9	
cmpq	
\$40000,	
%r9	
jne	
.L3	
leaq	
32(%rsp)	,
%rsi	
xorl	
%edi,	it
call	
clock_gettime@PLT	

2. El benchmark Linpack ha sido uno de los programas más ampliamente utilizados para evaluar las prestaciones de los computadores. De hecho, se utiliza como base en la lista de los 500 computadores más rápidos del mundo (el Top500 Report). El núcleo de este programa es una rutina denominada DAXPY (*Double precision- real Alpha X Plus Y*) que multiplica un vector por una constante y los suma a otro vector (Lección 3/Tema 1):

for 
$$(i=1; i \le N, i++)$$
  $y[i] = a*x[i] + y[i];$ 

- 2.1. Genere los programas en ensamblador para cada una de las siguientes opciones de optimización del compilador: -O0, -Os, -O2, -O3. Explique las diferencias que se observan en el código justificando al mismo tiempo las mejoras en velocidad que acarrean. Incorpore los códigos al cuaderno de prácticas y destaque las diferencias entre ellos.
- 2.2. (Ejercicio EXTRA) Para la mejor de las opciones, obtenga los tiempos de ejecución con distintos valores de N y determine para su sistema los valores de Rmax (valor máximo del número de operaciones en coma flotante por unidad de tiempo), Nmax (valor de N para el que se consigue Rmax), y N1/2 (valor de N para el que se obtiene Rmax/2). Estime el valor de la velocidad pico (Rpico) del procesador (consulte en [4] el número de ciclos por instrucción punto flotante para la familia y modelo de procesador que está utilizando) y compárela con el valor obtenido para Rmax. -Consulte la Lección 3 del Tema 1.

CAPTURA CÓDIGO FUENTE: daxpy.c

```
int main(int argc, char const *argv[])
<u>{</u>
    struct timespec cgtl,
    double
                       ncgt; //para tiempo de ejecución
    const int tamanioVector = 80000;
    const int cte = 3548;
    int vector1[tamanioVector];
    int vector2[tamanioVector];
    clock_gettime(CLOCK_REALTIME,&cgt1);
     for (int i = 0; i < tamanioVector; ++i)</pre>
         vector1[i] = i;
         vector2[i] = i;
         vector2[i] += cte*vector1[i];
    clock gettime(CLOCK REALTIME,&cgt2);
    ncgt = (double) (cgt2.tv_sec-cgt1.tv_sec)+
              (double) ((cgt2.tv_nsec-cgt1.tv_nsec)/(1.e+9));
    printf("V2[0]: %d\n", vector2[0]);
printf("V2[79999]: %d\n", vector2[79999]);
printf("tiempo: %11.9f\n", ncgt);
```

	-O0	-Os	-O2	-O3
Tiempos ejec.	0.001962	0.00123	0.000716	0.001530
F	002	7994	498	562

## CAPTURAS DE PANTALLA (que muestren la compilación y que el resultado es correcto):

```
[JoseJavierAlonsoRamos jjavier98@jjavier98-Lenovo-ideapad-310-15IKB:~/G
II/2°/2° Semestre/AC/SEMINARIOS/Seminario_4/ejer2] 2018-05-28 lunes
$./dgcc -00 daxpy.c -o daxpy-0
[JoseJavierAlonsoRamos jjavier98@jjavier98-Lenovo-ideapad-310-15IKB:~/G
II/2°/2° Semestre/AC/SEMINARIOS/Seminario_4/ejer2] 2018-05-28 lunes
Saccacc -Os daxpy.c -o daxpy-s
[JoseJavierAlonsoRamos jjavier98@jjavier98-Lenovo-ideapad-310-15IKB:~/G
II/2°/2° Semestre/AC/SEMINARIOS/Seminario_4/ejer2] 2018-05-28 lunes
$gccgcc -02 daxpy.c -o daxpy-2
[JoseJavierAlonsoRamos jjavier98@jjavier98-Lenovo-ideapad-310-15IKB:~/G
II/2°/2° Semestre/AC/SEMINARIOS/Seminario_4/ejer2] 2018-05-28 lunes
$gccgcc -03 daxpy.c -o daxpy-3
[JoseJavierAlonsoRamos jjavier98@jjavier98-Lenovo-ideapad-310-15IKB:~/G
II/2°/2° Semestre/AC/SEMINARIOS/Seminario_4/ejer2] 2018-05-28 lunes
$gcc./daxpy-0
V2[0]: 0
V2[79999]: 283916451
tiempo: 0.001962002
[JoseJavierAlonsoRamos jjavier98@jjavier98-Lenovo-ideapad-310-15IKB:~/G
II/2°/2° Semestre/AC/SEMINARIOS/Seminario_4/ejer2] 2018-05-28 lunes
$./d./daxpy-s
V2[0]: 0
V2[79999]: 283916451
tiempo: 0.001237994
[JoseJavierAlonsoRamos jjavier98@jjavier98-Lenovo-ideapad-310-15IKB:~/G
II/2°/2° Semestre/AC/SEMINARIOS/Seminario_4/ejer2] 2018-05-28 lunes
$./d./daxpy-2
V2[0]: 0
V2[79999]: 283916451
tiempo: 0.000716498
[JoseJavierAlonsoRamos jjavier98@jjavier98-Lenovo-ideapad-310-15IKB:~/G
II/2°/2° Semestre/AC/SEMINARIOS/Seminario_4/ejer2] 2018-05-28 lunes
$./d./daxpy-3
V2[0]: 0
V2[79999]: 283916451
tiempo: 0.001530562
```

### COMENTARIOS QUE EXPLIQUEN LAS DIFERENCIAS EN ENSAMBLADOR:

CÓDIGO EN ENSAMBLADOR (no es necesario introducir aquí el código como captura de pantalla, ajustar el tamaño de la letra para que una instrucción no ocupe más de un renglón): (PONER AQUÍ SÓLO LA ZONA DEL CÓDIGO ENSAMBLADOR DONDE ESTÁ EL CÓDIGO EVALUADO, USE COLORES PARA DESTACAR LAS DIFERENCIAS)

daxpy00.s	daxpy0s.s	daxpy02.s	daxpy03.s
call	call	call	call
clock_gettime@PLT	clock_gettime@PLT shrq \$2, %rbx xorl %eax,  leaq 0(, %rbx,4), %r12	<pre>clock_gettime@PLT</pre>	clock_gettime@PLT
movl 124(%rbp), %edx	imull \$3549, %eax, %edx	4,,10 .p2align .p2align 3	shrq \$2, %rbx movq %r12,

movslq %edx,	mov]		movl	%rax	leaq
%rdx movl	0(%r13,%rax,4) mov1 %edx	1 (%rbx,%rax,4	%edx,	%rbx,4), %r13	0(,
124(%rbp), %ecx mov1	(%r12,%rax,4) inco	q (%r12,%rax,4	%eax, 1)		shrq \$2, %rax negq
(%rax,%rdx,4) movq	%ra> cmpc \$800	g l	addq \$1, %rax addl		%rax andl \$3, %eax
96(%rbp), %rax movl	%rax jne .L2	%edx	\$3549, cmpq		je' .L7 cmpl
124(%rbp), %edx movslq	lead 56(%rbp), %rsi	%rax	cmpq \$80000, ine		\$1, %eax movl \$0, 0(,
%edx, %rdx movl	xorl %edi		jne .L2 leaq -64(%rbp),	%r14,4)	movl \$0, 0(,
124(%rbp), %ecx _	call		xorl	%rbx,4)	ie
movl %ecx, (%rax,%rdx,4)	clock_gettime@PLT		%edi, %edi call		.L8 cmpl \$3, %eax
movq - 96(%rbp), %rax		clock_gettin	1e@PLT	%r14,4)	movl \$1, 4(,
movl 124(%rbp), %edx				4(,%rbx,4)	movl \$3549,
movslq %edx,				(, 2, .,	jne .L9 movl
movl (%rax,				%r14,4)	\$2, 8(, movl
movq -				8(,%rbx,4)	\$7098,
112(%rbp), %rax movl				%r9d	movl \$79997,
124(%rbp), %edx movslq %edx,				88(%rbp)	movl \$3, -
%rdx movl (%rax,				88(%rbp) .L2:	movd
%rdx,4), %eax imull				88(%rbp), %xm	movl
132(%rbp), %eax addl				%r8d	\$80000, movl
%eax, movq				%ecx	%eax, subl
96(%rbp), %rax movl				%r8d	%eax,
124(%rbp), %edx movslq					salq \$2, %rcx xorl %eax,
%edx, '				%eax	pshufd
movl %ecx, (%rax,%rdx,4)				%xmm3, %xmm1	\$0, movl
addl \$1, -				%edi	%r8d, movdga
.L2: movl				), %xmm2	.LC1(%rip leaq
124(%rbp), %eax cmpl				(%r12,%rcx),	%rsi shrl
128(%rbp), %eax jl .L3				%rov	\$171 \$2, %edi addq %r13,
leaq - 64(%rbp), %rax				%rcx	xorl %edx,
movq %rax,  %rsi				%edx	paddd .LCO(%rip
movl \$0, %edi call				), %xmm1 4,,10	.p2align
Call				.,,±0	.p2align

clock_gettime@PLT		3	
CTOCK_gettIme@rEf		3 .L4:	movdqa %xmm1,
	9	%xmm0	addl \$1, %edx
			movaps %xmm1,
			pslld \$3, %xmm0 psubd %xmm1,
	9	%×mm0	pslld \$4, %xmm0 psubd %xmm1,
	9	% Y [[[[[]]]])	pslld \$3, %xmm0 psubd %xmm1,
	9		pslld \$2, %xmm0 paddd %xmm1,
	9	%XIIIIIU	movups %xmm0,
		(%rcx,%rax)	addq \$16, %rax cmp1
	9	%edi	%edx, paddd %xmm2,
	9	%xmm1	ja .L4 movl
	8	88(%rbp), %ea	-
	9	%ecx	movl %r9d,
	9	%edx	andl \$-4, %ecx subl %ecx,
		%edx	addl %ecx,
		%eax	cmpl %ecx,
			je .L5 imull \$3548,
		%eax, %ecx	movslq %eax,
	9	%rsı	cmpl \$1, %edx movl %eax,
		(%r12,%rsi,4)	leal (%rax,
		%rcx), %edi	movl
		0(%r13,%rs1,4	) leal 1(%rax),
			je .L5 movslq
	 Depto. Arquitectura y Tecnolog		

	%esi, %rdi	
	addl \$2, %eax cmpl \$2, %edx movl	
	movi %esi, (%r12,%rdi,4) leal	
	3548(%rcx,%rsi), %esi movl %esi,	
	movl %esi, 0(%r13,%rdi,4) je .L5	
	movslq %eax,	
	leal	
	7096(%rcx,%rax), %eax mov1	
	%eax, 0(%r13,%rdx,4) .L5:	
	leaq 64(%rbp), %rsi	
	xorl %edi, %edi	
	call	
	clock_gettime@PLT	