```
alex@alex-VirtualBox:~$ nano Tarea1.c
alex@alex-VirtualBox:~$ gcc Tarea1.c -o programa
alex@alex-VirtualBox:~$ ./programa
^[[DSegmentation fault (core dumped)
alex@alex-VirtualBox:~$ ./programa 10

Dimension: 10,
Tiempo: 0.000000alex@alex-VirtualBox:~$ ./programa 100

Dimension: 100,
Tiempo: 0.000000alex@alex-VirtualBox:~$ ./programa 1000
Segmentation fault (core dumped)
alex@alex-VirtualBox:~$ ./programa 250

Dimension: 250,
Tiempo: 0.000000alex@alex-VirtualBox:~$ ./programa 500

Dimension: 500,
Tiempo: 1.000000alex@alex-VirtualBox:~$ ./programa 750
Segmentation fault (core dumped)
alex@alex-VirtualBox:~$ ./programa 550

Dimension: 550,
Tiempo: 0.000000alex@alex-VirtualBox:~$ ./programa 600
Segmentation fault (core dumped)
alex@alex-VirtualBox:~$ ./programa 575

Dimension: 575,
Tiempo: 0.000000alex@alex-VirtualBox:~$ ./programa 580
```

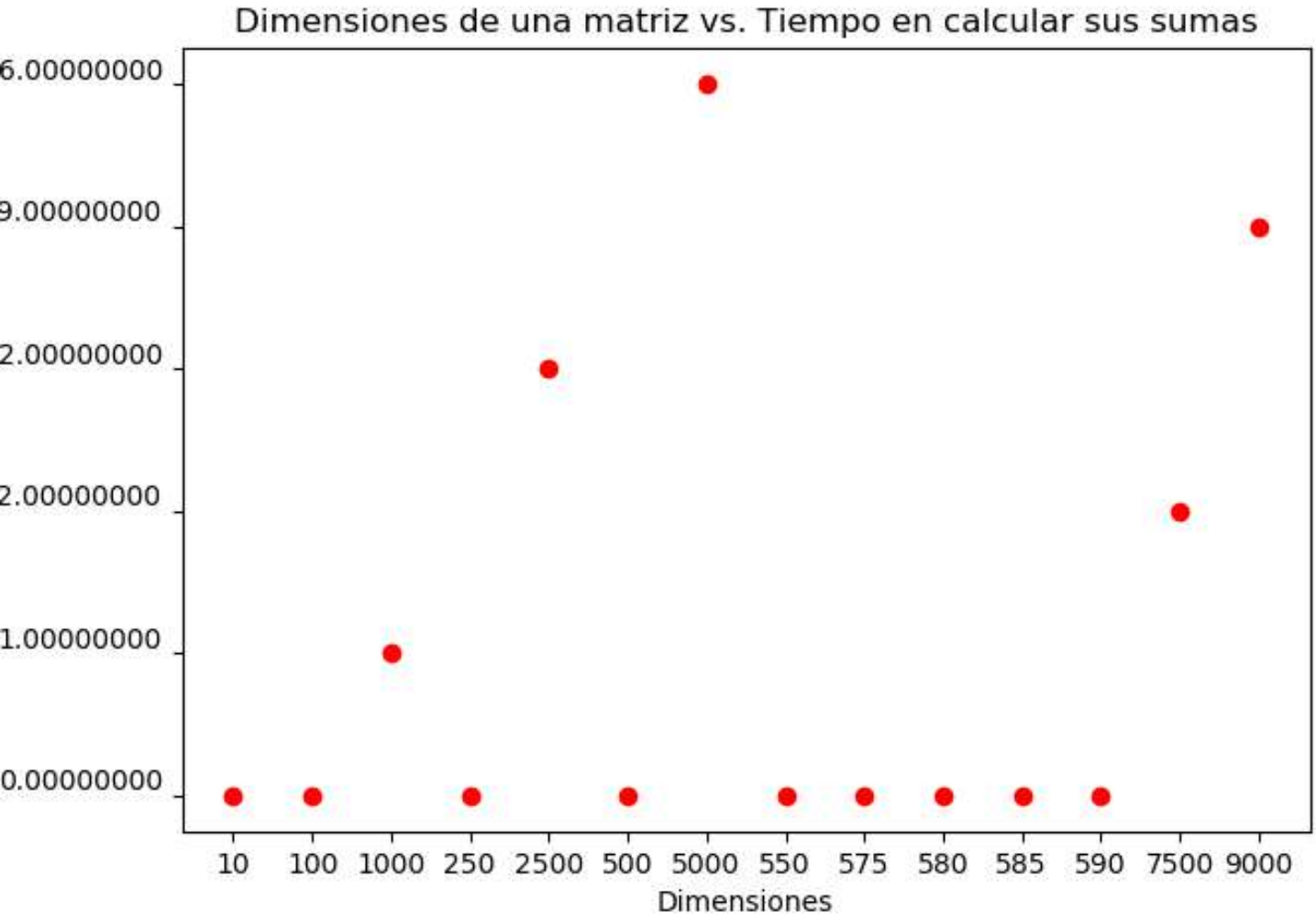
```
Dimension: 575,
Tiempo: 0.000000alex@alex-VirtualBox:~$ ./programa 580

Dimension: 580,
Tiempo: 0.000000alex@alex-VirtualBox:~$ ./programa 585

Dimension: 585,
Tiempo: 0.000000alex@alex-VirtualBox:~$ ./programa 590

Dimension: 590,
Tiempo: 0.000000alex@alex-VirtualBox:~$ ./programa 595
Segmentation fault (core dumped)
```

```
alex@alex-VirtualBox:~$ nano Tarea1.c
alex@alex-VirtualBox:~$ gcc Tarea1.c -o programa
alex@alex-VirtualBox:~$ ./programa

Tiempo: 0.000000alex@alex-VirtualBox:~$ nano Tarea1.c
alex@alex-VirtualBox:~$ gcc Tarea1.c -o programa
alex@alex-VirtualBox:~$ ./programa

Dimension: 100,
Tiempo: 0.000000alex@alex-VirtualBox:~$ nano Tarea1.c
alex@alex-VirtualBox:~$ gcc Tarea1.c -o programa
alex@alex-VirtualBox:~$ ./programa

Dimension: 1000,
Tiempo: 0.000000alex@alex-VirtualBox:~$ nano Tarea1.c
alex@alex-VirtualBox:~$ gcc Tarea1.c -o programa
alex@alex-VirtualBox:~$ ./programa
Segmentation fault
alex@alex-VirtualBox:~$ nano Tarea1.c
alex@alex-VirtualBox:~$ gcc Tarea1.c -o programa
alex@alex-VirtualBox:~$ ./programa

Dimension: 2500,
Tiempo: 2.000000alex@alex-VirtualBox:~$ nano Tarea1.c
alex@alex-VirtualBox:~$ gcc Tarea1.c -o programa
alex@alex-VirtualBox:~$ ./programa

Dimension: 5000,
Tiempo: 6.000000alex@alex-VirtualBox:~$ nano Tarea1.c
```

```
Dimension: 5000,
Tiempo: 6.000000alex@alex-VirtualBox:~$ nano Tarea1.c
alex@alex-VirtualBox:~$ gcc Tarea1.c -o programa
alex@alex-VirtualBox:~$ ./programa

Dimension: 7500,
Tiempo: 12.000000alex@alex-VirtualBox:~$ nano Tarea1.c
alex@alex-VirtualBox:~$ gcc Tarea1.c -o programa
alex@alex-VirtualBox:~$ ./programa

Dimension: 9000,
Tiempo: 59.000000alex@alex-VirtualBox:~$ cat Datos_Tarea1.txt
```

```
alex@alex-VirtualBox:~$ cat Datos_Tarea1.txt
10 0.00000000
100 0.00000000
1000 1.00000000
100 0.00000000
100 0.00000000
1000 0.00000000
2500 2.00000000
5000 6.00000000
7500 12.00000000
9000 59.00000000

10 0.00000000
100 0.00000000
250 0.00000000
500 1.00000000
550 0.00000000
575 0.00000000
580 0.00000000
585 0.00000000
590 0.00000000alex@alex-VirtualBox:~$ nano Tarea1.c
```

```
alex@alex-VirtualBox:~$ nano Graph.py
alex@alex-VirtualBox:~$ python3 Graph.py
14
['10', '0.00000000\n']
['100', '0.00000000\n']
['250', '0.00000000\n']
['500', '0.00000000\n']
['550', '0.00000000\n']
['575', '0.00000000\n']
['580', '0.00000000\n']
['585', '0.00000000\n']
['590', '0.00000000\n']
['1000', '1.00000000\n']
['2500', '2.00000000\n']
['5000', '6.00000000\n']
['7500', '12.00000000\n']
['9000', '59.00000000\n']
```


Dimensiones de una matriz vs. Tiempo en calcular sus sumas
6.00000000
9.00000000
2.00000000
2.00000000
1.00000000
0.00000000
10
100
1000
250
2500
500
5000
550
575
580
585
590
7500
9000
Dimensiones

```
  GNU nano 2.9.3                        Tarea1.c

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

int main(int argc, char * argv[])
{
        int i,j;
        /* 'dimension' es un entero que guardará el número ingresado por el
           terminal.
        */
        long int dimension = strtol(argv[1], NULL, 10);
        double tiempo = 0.0;

        // Empieza a contar tiempo.
        time_t inicio = time(NULL);

        // Matrices
        long int matrizA[dimension][dimension];
        long int matrizB[dimension][dimension];
        long int matrizRes[dimension][dimension];

        /* Definición de la semilla de 'rand'
           Para siempre generar números aleatorios.

^G Get Help    ^O Write Out   ^W Where Is    ^K Cut Text    ^J Justify
^X Exit        ^R Read File   ^\ Replace     ^U Uncut Text  ^T To Spell
```

GNU nano 2.9.3 Tarea1.c

```
	srand(time(0));

	for(i=0; i<dimension; i++)
	{
		for(j=0;j<dimension;j++)
		{
			matrizA[i][j]=rand()%1001;
			matrizB[i][j]=rand()%1001;
			matrizRes[i][j]=matrizA[i][j]+matrizB[i][j];
		}
	}

	// Termina de contar tiempo.
	time_t final = time(NULL);
	tiempo = (double)(final-inicio);

	/* Se crea la dirección de un archivo llamado "Datos_Tarea1" de tipo
	   texto.
	 * Se abre, se anotan en nuevas hileras la dimensión y el tiempo que
	   le tomó computarlo, y se cierra.
	 */
	FILE* fichero;
	fichero = fopen("Datos_Tarea1.txt","a");
```

^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify
^X Exit ^R Read File ^\ Replace ^U Uncut Text ^T To Spell

GNU nano 2.9.3 Tarea1.c

```
    // Termina de contar tiempo.
    time_t final = time(NULL);
    tiempo = (double)(final-inicio);

    /* Se crea la dirección de un archivo llamado "Datos_Tarea1" de tipo
       texto.
     * Se abre, se anotan en nuevas hileras la dimensión y el tiempo que
       le tomó computarlo, y se cierra.
     */
    FILE* fichero;
    fichero = fopen("Datos_Tarea1.txt","a");
    fprintf(fichero,"\n%ld %.8f", dimension, tiempo);
    fclose(fichero);

    return 0;
}
```

```
  GNU nano 2.9.3                    Tarea1.c

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

/* La memoria local del programa no puede soportar, en mi caso, una dimension
   > 550, así que se vuelve necesario declararla como una variable global
   (i.e. antes de main()).
 */
# define dimension 9700

long int matrizA[dimension][dimension];
long int matrizB[dimension][dimension];
long int matrizRes[dimension][dimension];

int main()
{
        int i,j;
        double tiempo = 0.0;

        // Empieza a contar tiempo.
        time_t inicio = time(NULL);
```

GNU nano 2.9.3 Tarea1.c

```
/* Definición de la semilla de 'rand'
   Para siempre generar números aleatorios.
 */
srand(time(0));

for(i=0; i<dimension; i++)
{
        for(j=0;j<dimension;j++)
        {
                matrizA[i][j]=rand()%1001;
                matrizB[i][j]=rand()%1001;
                matrizRes[i][j]=matrizA[i][j]+matrizB[i][j];
        }
}

// Termina de contar tiempo.
time_t final = time(NULL);
tiempo = (double)(final-inicio);

printf("\nDimension: %d, \nTiempo: %f",dimension, tiempo);

/* Se crea la dirección de un archivo llamado "Datos_Tarea1" de tipo
   texto.
```

```
    /* Se crea la dirección de un archivo llamado "Datos_Tarea1" de tipo
       texto.
     * Se abre, se anotan en nuevas hileras la dimensión y el tiempo que
       le tomó computarlo, y se cierra.
     */
    FILE* fichero;
    fichero = fopen("Datos_Tarea1.txt","a");
    fprintf(fichero,"\n%d %.8f", dimension, tiempo);
    fclose(fichero);

    return 0;
}
```

```
  GNU nano 2.9.3                        Graph.py

from pylab import*
import matplotlib.pyplot as plt

archivo = open('Datos_Tarea1.txt', 'r')
lineas = len(archivo.readlines())

print(lineas)

archivo.seek(0)
dimensiones = []
tiempos = []

for i in range(lineas):
        x = archivo.readline().split(' ')
        print (x)

        dimensiones.append(x[0])
        tiempos.append(x[1])

plt.plot(dimensiones, tiempos, 'ro')
plt.title("Dimensiones de una matriz vs. Tiempo en calcular sus sumas")
plt.xlabel("Dimensiones")
plt.ylabel("Tiempo (seg)")
                              [ Read 27 lines ]
```

GNU nano 2.9.3 Graph.py

```
for i in range(lineas):
        x = archivo.readline().split(' ')
        print (x)

        dimensiones.append(x[0])
        tiempos.append(x[1])

plt.plot(dimensiones, tiempos, 'ro')
plt.title("Dimensiones de una matriz vs. Tiempo en calcular sus sumas")
plt.xlabel("Dimensiones")
plt.ylabel("Tiempo (seg)")

plt.show()
archivo.close()
```