

Examen Parcial Part Parcial

Tema: Secuencial V.S. Threads

Nota

Estudiante	Escuela	Asignatura
Sebastian Gabriel David Pizango sdavidp@ulasalle.edu.pe	Carrera Profesional de Ingeniería de Software	Lenguaje de Programación III Semestre: I Código: 20231001

Examen	Tema	Duración
Parcial	Secuencial V.S. Threads	06 horas

Semestre académico	Fecha de inicio	Fecha de entrega
2025 - A	Del 30 Abril 2025	Al 7 Mayo 2025

1. Examen

- Suma de elementos de una matriz cuadrada de forma secuencial y forma con threads comparando los resultados
- Utilizar Git para evidenciar su trabajo.
- Enviar trabajo al profesor en un repositorio GitHub Privado, dándole permisos como colaborador.

2. Equipos, materiales y temas utilizados

- Sistema Operativo Windows 11 64 bits.
- NVIM 0.10.3
- OpenJDK 64-Bits 17.0.7.
- Git 2.39.2.
- Cuenta en GitHub con el correo institucional.
- Programación Orientada a Objetos.
- Matrices, threads.

3. URL de Repositorio Github

- URL del Repositorio GitHub para clonar o recuperar.
- `https://github.com/sebastian-david-pizango/ProyectoLp3/tree/master`
- URL para el parcial 01 en el Repositorio GitHub.
- `https://github.com/sebastian-david-pizango/ProyectoLp3/tree/master/Parcial`

4. Actividades con el repositorio GitHub

4.1. Creando e inicializando repositorio GitHub

- Como es el primer laboratorio se creo el repositorio GitHub.
- Se realizaron los siguientes comandos en la computadora:

Listing 1: Creando directorio de trabajo y repositorio de GitHub

```
$ mkdir ProyectoLp3
```

Listing 2: Dirigiéndonos al directorio de trabajo y repositorio

```
$ cd ProyectoLp3
```

Listing 3: Inicializando directorio para repositorio GitHub

```
$ nvim README.md
$ git init
$ git config --global user.name "Sebastian Gabriel David Pizango"
$ git config --global user.email sdavidp@ulasalle.edu.pe
$ git add .
$ git commit -m "Primer Commit"
$ git remote add origin https://github.com/ProyectoLp3.git
$ git push -u origin master
```

4.2. Commits

Listing 4: Primer Commit Creando carpeta/archivo para laboratorio 01

```
$ mkdir Solitario
$ touch lab01/raiz_digital.cpp
$ git add .
$ git commit -m "Creando carpeta y archivo para laboratorio 01 de raz digital"
$ git push
```

- Se creo el archivo `.gitignore` para no considerar los archivos `*.class` que son innecesarios hacer seguimiento.

Listing 5: Creando .gitignore

```
$ vim lab01/.gitignore
```

Listing 6: lab01/.gitignore

```
*.exe
```

Listing 7: Commit: Creando .gitignore para archivos *.class

```
$ git add .  
$ git commit -m "Creando .gitignore para archivos *.exe"  
$ git push
```

- Para el siguiente commit se implemento el algoritmo de ordenamiento por Inserción, se imprime el arreglo caso definido en el mismo código.
- El pseudocódigo utilizado es el siguiente:

INSERTION-SORT(A)

```
1  for  $j = 2$  to  $A.length$   
2       $key = A[j]$   
3      // Insert  $A[j]$  into the sorted sequence  $A[1..j-1]$ .  
4       $i = j - 1$   
5      while  $i > 0$  and  $A[i] > key$   
6           $A[i + 1] = A[i]$   
7           $i = i - 1$   
8       $A[i + 1] = key$ 
```

- Utilizado en los siguientes laboratorios:

Listing 8: Creando .gitignore

```
$ nvim Solitario/raiz_digital.cpp
```

Listing 9: Raiz Digital.cpp

```
1
2
3 #include <iostream>
4 using namespace std;
5
6 // Funcin para calcular la raz digital de un nmero
7 int raizDigital(int arr[], int size) {
8     int suma = 0;
9     for (int i = 0; i < size; i++) {
10         suma += arr[i];
11     }
12
13     while (suma >= 10) { // Repetir hasta que el nmero sea de un solo dgito
14         int temp = 0;
15         while (suma > 0) {
16             temp += suma % 10; // Extraer el ltimo dgito y sumarlo
17             suma /= 10; // Eliminar el ltimo dgito
18         }
19         suma = temp; // Actualizar suma con la nueva suma de dgitos
20     }
21     return suma;
22 }
23
24 int main() {
25     int numero;
26     while (true) {
27         cin >> numero;
28         if (numero == 0) break; // Condicin de salida
29
30         // Convertir nmero a array de dgitos
31         int digitos[10], size = 0, temp = numero;
32         while (temp > 0) {
33             digitos[size++] = temp % 10;
34             temp /= 10;
35         }
36
37         if (size == 1) {
38             cout << numero << endl;
39         } else {
40             cout << raizDigital(digitos, size) << endl;
41         }
42     }
43     return 0;
44 }
```

Listing 10: Compilando y probando código

```
$ cd Solitario
$ gcc raiz_digital.cpp
$ g++ a.exe
```

```
9
22
4
123
6
```

Listing 11: Commit: Probando algoritmo de Inserción con arreglo

```
$ git add .
$ git commit -m "Codigo del primer laboratorio de raiz digital"
$ git push
```

Listing 12: MatrizSecuencial.cpp

```
1 #include <iostream>
2 #include <vector>
3 #include <chrono>
4 #include <fstream>
5 #include <cstdlib> // para rand()
6 #include <ctime> // <--- Agrega esta la para usar time()
7 using namespace std;
8
9 class Matriz {
10 private:
11     vector<vector<int>> datos;
12     int filas, columnas;
13
14 public:
15     Matriz(int n, int m) : filas(n), columnas(m) {
16         datos.resize(n, vector<int>(m)); // ajusta el tama del vector principal (las filas) y
17         // adem puede inicializar su contenido.
18     }
19
20     void llenar() {
21         for (int i = 0; i < filas; ++i)
22             for (int j = 0; j < columnas; ++j)
23                 datos[i][j] = rand() % 10;
24     }
25
26     void mostrar() const {
27         for (int i = 0; i < filas; ++i) {
28             for (int j = 0; j < columnas; ++j) {
29                 cout << datos[i][j] << " ";
30             }
31             cout << endl;
32         }
33     }
34
35     int sumar() const {
36         int suma = 0;
37         for (int i = 0; i < filas; ++i)
38             for (int j = 0; j < columnas; ++j)
39                 suma += datos[i][j];
40         return suma;
41     }
42 }
```

```
41 };
42 int main() {
43     srand(time(nullptr)); // <-- Sembrar el generador de nmeros aleatorios
44     int size = 0;
45     cout << "Ingrese la matriz maxima" << endl;
46     cin >> size;
47
48     ofstream archivo("resultados.txt");
49     archivo << "Tama de la matriz,Tiempo promedio (nanosegundos)\n";
50
51     for (int i = 1; i <= size; ++i) {
52         Matriz matriz(i, i);
53         matriz.llenar();
54
55         auto start = chrono::high_resolution_clock::now();
56
57         cout << "Matriz de: " << i << " x " << i << endl;
58         //matriz.mostrar();
59
60         int suma = matriz.sumar();
61
62         auto end = chrono::high_resolution_clock::now();
63         long long duracion = chrono::duration_cast<chrono::nanoseconds>(end - start).count();
64         long long tiempoPromedio = duracion / size;
65
66         cout << "\nLa suma es: " << suma << endl;
67         cout << "Tiempo: " << tiempoPromedio << endl;
68
69         archivo << i << "," << tiempoPromedio << "\n";
70     }
71
72     archivo.close();
73     return 0;
74 }
```

Listing 13: Compilando y probando el código

```
$ cd Grupo
$ g++ MatrizSecuencial.cpp -o MatrizSecuencial.exe
$ g++ MatrizSecuencial.exe
Ingresa la matriz maxima
9
Matriz de: 1 x 1

La suma es: 3
Tiempo: 1504
Matriz de: 2 x 2

La suma es: 22
Tiempo: 179
Matriz de: 3 x 3

La suma es: 55
Tiempo: 580
Matriz de: 4 x 4

La suma es: 69
Tiempo: 945
Matriz de: 5 x 5

La suma es: 124
Tiempo: 190
Matriz de: 6 x 6

La suma es: 169
Tiempo: 210
Matriz de: 7 x 7

La suma es: 225
Tiempo: 586
Matriz de: 8 x 8

La suma es: 288
Tiempo: 578
Matriz de: 9 x 9

La suma es: 370
Tiempo: 560
```

Listing 14: Comandos de gnu plot

```
set title 'Tiempo Secuencial de la Matriz';  
set xlabel 'Tamaño de la Matriz';  
set ylabel 'Tiempo Promedio (en nanosegundos)';  
set grid;  
set style data linespoints;  
set datafile separator ',';  
plot 'resultados.txt' using 1:2 with linespoints
```

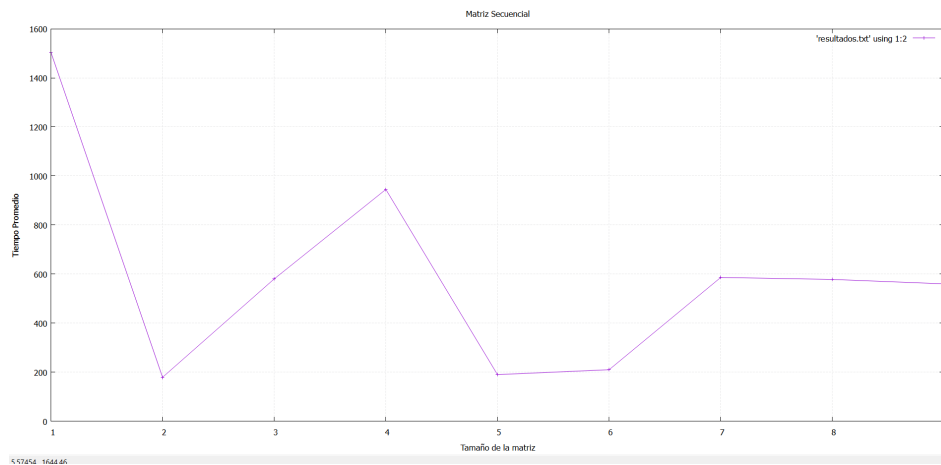


Figura 1: Resultado de los comandos gnuplot

Listing 15: Commit al repositorio

```
$ git add .  
$ git commit -m "Codigo Matriz "  
$ git push
```

Listing 16: MatrizConclases.cpp

```
1 #include <iostream>  
2 #include <chrono>  
3 #include <fstream>  
4 #include <pthread.h>  
5 #include <vector>  
6 #include <cstdlib>  
7 using namespace std;  
8  
9 class MatrizParalela {  
10 private:  
11     vector<vector<int>> datos;  
12     int filas, columnas;  
13     int sumatotal;  
14     pthread_mutex_t mutex;  
15  
16     struct HiloParams {  
17         int inicio;  
18         int fin;
```



```
19     MatrizParalela* matrizPtr;
20 };
21
22 static void* sumarRango(void* arg) {
23     HiloParams* params = (HiloParams*)arg;
24     int sumaLocal = 0;
25
26     for (int i = params->inicio; i < params->fin; ++i)
27         for (int j = 0; j < params->matrizPtr->columnas; ++j)
28             sumaLocal += params->matrizPtr->datos[i][j];
29
30     pthread_mutex_lock(&params->matrizPtr->mutex);
31     params->matrizPtr->sumatotal += sumaLocal;
32     pthread_mutex_unlock(&params->matrizPtr->mutex);
33
34     return NULL;
35 }
36
37 public:
38     MatrizParalela(int n, int m) : filas(n), columnas(m), sumatotal(0) {
39         datos.resize(n, vector<int>(m));
40         pthread_mutex_init(&mutex, NULL);
41     }
42
43     ~MatrizParalela() {
44         pthread_mutex_destroy(&mutex);
45     }
46
47     void llenar(int limite = 100) {
48         for (int i = 0; i < filas; ++i)
49             for (int j = 0; j < columnas; ++j)
50                 datos[i][j] = rand() % limite;
51     }
52
53     int sumarConHilos(int numHilos = 4) {
54         sumatotal = 0;
55         pthread_t hilos[numHilos];
56         HiloParams params[numHilos];
57
58         int paso = filas / numHilos;
59
60         for (int i = 0; i < numHilos; ++i) {
61             int inicio = i * paso;
62             int fin = (i == numHilos - 1) ? filas : inicio + paso;
63
64             params[i] = {inicio, fin, this};
65             pthread_create(&hilos[i], NULL, sumarRango, (void*)&params[i]);
66         }
67
68         for (int i = 0; i < numHilos; ++i) {
69             pthread_join(hilos[i], NULL);
70         }
71
72         return sumatotal;
73     }
74 };
```

```
75
76 int main() {
77     srand(time(nullptr));
78
79     int size;
80     cout << "Ingrese la matriz maxima (tama de la matriz cuadrada): ";
81     cin >> size;
82
83     ofstream archivo("resultados.txt");
84     archivo << "Tama de la matriz, Tiempo (nanosegundos)\n";
85
86     for (int i = 1; i <= size; ++i) {
87         MatrizParalela matriz(i, i);
88         matriz.llenar();
89
90         auto start = chrono::high_resolution_clock::now();
91         int suma = matriz.sumarConHilos();
92         auto end = chrono::high_resolution_clock::now();
93
94         long long tiempo = chrono::duration_cast<chrono::nanoseconds>(end - start).count();
95
96         cout << "Matriz de: " << i << "x" << i << endl;
97         cout << "Suma total: " << suma << endl;
98         cout << "Tiempo: " << tiempo << " nanosegundos\n" << endl;
99
100        archivo << i << "," << tiempo << "\n";
101    }
102
103    archivo.close();
104    return 0;
105 }
```

Listing 17: Compilando y probando el código

```
$ cd Grupo
$ g++ MatrizClases.cpp -o MatrizClases.exe
$ g++ MatrizClases.exe
```

Listing 18: Comandos de gnu plot 2

```
set title 'Tiempo con Clases de la Matriz';  
set xlabel 'Tamaño de la Matriz';  
set ylabel 'Tiempo Promedio (en nanosegundos)';  
set grid;  
set style data linespoints;  
set datafile separator ',';  
plot 'resultados.txt' using 1:2 with linespoints;
```

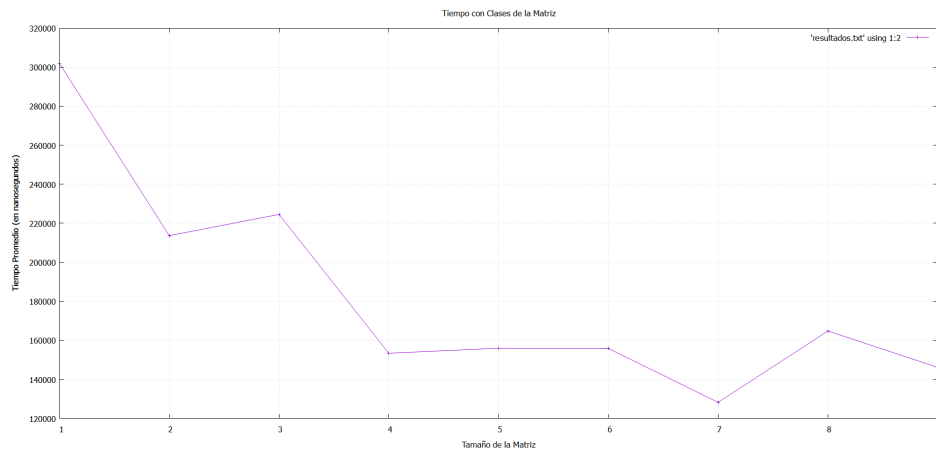


Figura 2: Resultado de los comandos gnuplot 2

Listing 19: Comandos de gnu plot Combinado

```
set title 'Comparacion de Graficos';  
set xlabel 'Tamao de la Matriz';  
set ylabel 'Tiempo Promedio (en nanosegundos)';  
set grid;  
set style data linespoints;  
set datafile separator ',';  
set datafile separator ','; set style data linespoints; set grid; set xlabel 'Tamao del  
Matriz'; set ylabel 'Tiempo Promedio (en nanosegundos)'; set title 'Comparaci  
on de Graficos'; plot 'ConThreads.txt' using 1:2 title 'Con Threads', 'SinThreads.txt' using  
1:2 title 'Sin Threads'
```

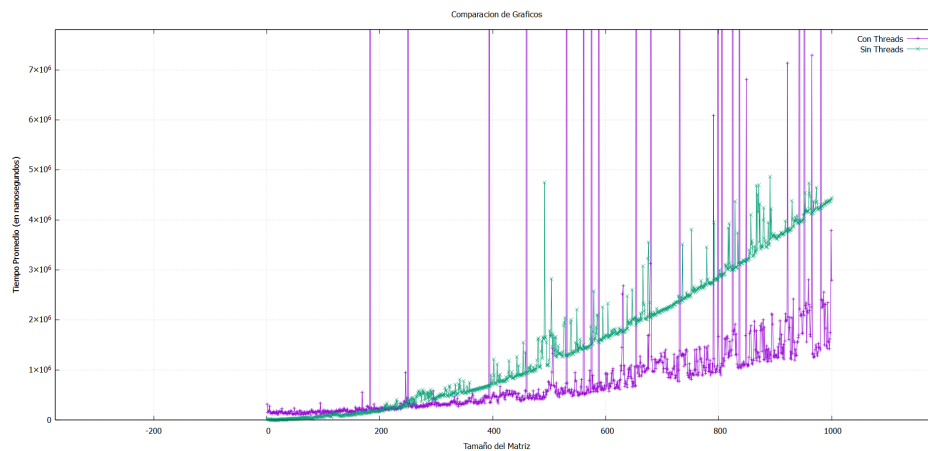


Figura 3: Resultado de los comandos gnu plot Combinado(Con 1k de pruebas)

4.3. Estructura del repositorio

- El contenido que se entrega en este laboratorio es el siguiente:

```
|--- ProyectoLp3 [REPOSITORY]  
|   |--- Grupo [DIRECTORY]  
|       |--- lab02 [DIRECTORY]  
|           |--- .gitignore  
|           |--- Hello.java  
|           |--- Program.java  
|           |--- README.md  
|           |--- TestProgram01.java  
|           |--- TestProgram02.java  
|           |--- TestProgram03.java  
|           |--- TestProgram04.java  
|       |--- lab03 [Directory]  
|           |--- Peor [DIRECTORY]  
|               |--- a.exe  
|               |--- worst.cpp  
|           |--- RangoPeor [DIRECTORY]  
|               |--- RangoPeor2.cpp  
|           |--- README.md
```

```
|      |--- comandos.txt
|      |--- simulacion.jpeg
|      |--- lab04 [Directory]
|      |--- Codigomatriz.cpp
|      |--- Grafico Matriz.PNG
|      |--- README.md
|      |--- a.exe
|      |--- codigomatrizp.cpp
|      |--- con threads.png
|      |--- sin threads.png
|      |--- Parcial [DIRECTORY]
|      |--- Comparacin.jpeg
|      |--- Con threads.jpeg
|      |--- Examen_Parcial_01.pdf
|      |--- MatrizSecuencial.cpp
|      |--- MatrizThread.cpp
|      |--- README.md
|      |--- sin threads.jpeg
|      |--- Solitario [DIRECTORY]
|      |--- .gitignore
|      |--- raiz digital.cpp
|      |---README.md
8 directories, 30 files
```

5. Calificación

Tabla 1: Rúbrica de calificación para la parte 1: Tiempo máximo de exposición 10 min

	Contenido y demostración	Puntos	Checklist	Estudiante	Profesor
1. Guía en LA-TEX del framework	El entregable cumple con todos los pasos de la guía base entregada y es reproducible para obtener el producto.	5	X	5	
2. Puntualidad y responsabilidad	Asistencia en punto el día de la presentación y con toda la predisposición a presentar el trabajo.	3	X	3	
3. Manejo del tiempo	Utiliza el tiempo de manera eficiente.	3	X	3	
4. Obtención del producto final	Muestra el producto final y explica cada uno de sus componentes utilizando la guía paso a paso.	5	X	4	
5. Manejo del escenario palabra y seguridad	Expone manejando el escenario preocupándose para que todos entiendan con una voz adecuada para ello.	4	X	3	
Total		20	-	18	

6. Referencias

- <https://www.w3schools.com/java/default.asp>
- <https://www.geeksforgeeks.org/insertion-sort/>