
	Universidad Nacional de San Agustín de Arequipa Facultad de Ingeniería de Producción y Servicios Escuela Profesional de Ingeniería de Sistemas	
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		
Aprobación: 2022/03/01	Código: GUIA-PRLD-001	Página: 1

## INFORME DE LABORATORIO

INFORMACION BASICA					
ASIGNATURA:	Tecnología de Objetos				
TITULO DE LA PRACTICA:	Programación con hilos: Golang, C++ y Java				
NUMERO DE LA PRACTICA:	03	AÑO LECTIVO:	2025 - B	N° SEMESTRE:	VI
FECHA DE PRESENTACION:	01 / 10 / 2025	HORA DE PRESENTACION:	--:-- PM		
INTEGRANTE (s): ■ Huayhua Hillpa, Yourdyy Yossimar ■ Villafuerte Quispe, Alexander				NOTA:	
DOCENTE (s): ■ Mg. Escobedo Quispe, Richart Smith					

## 1. Tarea

### 1.1. Objetivo

- Programar paralelamente usando threads.
- Comparar los lenguajes de programación: Java, C++ y Go.

### 1.2. Problema propuesto:

Sea un función cualquier, por ejemplo:  $f(x) = 2x$

$$2+3x +1/2$$

y los puntos  $a=2$  y  $b=20$

Hallar el área bajo la curva en el primer cuadrante, utilizando el método del trapecio. En Clase:



- Utilice Java para resolver el problema.
- Utilice C++ para gestionar la memoria dinámicamente.

La proxima clase:

- Utilice Go para resolver el problema.

## 2. Equipos, materiales y temas utilizados

- Subsistema de Windows para Linux (WSL) con Ubuntu (versión predeterminada instalada mediante Microsoft Store).
- Sistema operativo: Microsoft Windows [Versión 10.0.26100.6584]

	<p>Universidad Nacional de San Agustín de Arequipa Facultad de Ingeniería de Producción y Servicios Escuela Profesional de Ingeniería de Sistemas</p>	
<p>Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación</p>		
<p>Aprobación: 2022/03/01</p>	<p>Código: GUIA-PRLD-001</p>	<p>Página: 2</p>

- Helix 25.01.1 (e7ac2fcd)
- Visual Studio Code 1.104.0 x64
- Git version 2.41.0.windows.1
- Cuenta activa en GitHub para la gestión de repositorios remotos.
- POO.
- Lenguaje de programación Java.
- Lenguaje de programación golang
- Lenguaje de programación c++

### 3. URL de Repositorio Github

- URL del Repositorio GitHub para clonar o recuperar.
- <https://github.com/yhuayhuahi/Teo.git>
- <https://github.com/avillaq/Teo.git>
- URL para el laboratorio (03) en el Repositorio GitHub.
- <https://github.com/yhuayhuahi/Teo/tree/main/laboratorios/lab03>
- <https://github.com/avillaq/teo/tree/main/lab03>

## 4. Desarrollo de las actividades

### 4.1. Actividades

#### 4.1.1. Función Main en C++ - Primera implementación



A continuación se muestra la función principal en c++ de la primera implementación [un hilo por trapecio a generar]:

Listing 1: Función Main en cpp - Primera implementación

```

1  int main() {
2      cout << "----- CALCULADORA DE INTEGRALES CON HILOS NORMALES -----"
3          << endl;
4      cout << endl;
5
6      cout << "Ingrese la función f(x) a integrar:" << endl;
7      cout << "    Ejemplos:" << endl;
8      cout << "    - 2*x^2 + 3*x + 0.5" << endl;
9      cout << "    - sin(x)" << endl;
10     cout << "    - cos(x) + x^2" << endl;
11     cout << "    - exp(x)" << endl;
12     cout << "    - log(x)" << endl;
13     cout << "    - sqrt(x)" << endl;
14     cout << endl;
15     cout << "f(x) = ";
16
17     getline(cin, funcion_matematica);



```

	<p>Universidad Nacional de San Agustín de Arequipa Facultad de Ingeniería de Producción y Servicios Escuela Profesional de Ingeniería de Sistemas</p>	
<p>Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación</p>		
<p>Aprobación: 2022/03/01</p>	<p>Código: GUIA-PRLD-001</p>	<p>Página: 3</p>

```

18 // eliminar espacios en blanco al inicio y final
19 size_t start = funcion_matematica.find_first_not_of(" \t");
20 if (start == string::npos) {
21     cout << "Error: No se ingreso ninguna funcion" << endl;
22     return 1;
23 }
24 size_t end = funcion_matematica.find_last_not_of(" \t");
25 funcion_matematica = funcion_matematica.substr(start, end - start + 1);
26
27 cout << "Funcion ingresada: '" << funcion_matematica << "'" << endl;
28
29 // validamos la funcion
30 try {
31     double test1 = evaluarFuncion(funcion_matematica, 1.0);
32     double test2 = evaluarFuncion(funcion_matematica, 2.0);
33     if (test1 != 0.0 || test2 != 0.0) {
34         cout << "Funcion valida" << endl;
35     }
36 } catch (...) {
37     cout << "Error: La funcion no es valida" << endl;
38     exit(1);
39 }
40
41 // se piden los parametros de la integral
42 double limite_inferior, limite_superior;
43 int decimales, numero_hilos;
44
45 cout << "Limite inferior (a): ";
46 cin >> limite_inferior;
47 cout << "Limite superior (b): ";
48 cin >> limite_superior;
49 cout << "Numero de hilos (trapecios): ";
50 cin >> numero_hilos;
51 cout << "Decimales en el resultado: ";
52 cin >> decimales;
53 cout << endl;
54
55 // validaciones
56 if (limite_superior <= limite_inferior) {
57     cout << "Error: El limite superior debe ser mayor al inferior" << endl;
58     return 1;
59 }
60
61 if (numero_hilos <= 0) {
62     cout << "Error: El numero de hilos debe ser positivo" << endl;
63     return 1;
64 }
65
66 if (numero_hilos > 1000) {
67     cout << "Error: Muchos hilos pueden afectar el rendimiento" << endl;
68 }
69
70 if (decimales < 0 || decimales > 10) {
71     cout << "Error: El numero de decimales debe estar entre 0 y 10" << endl;
72     ;
73     return 1;
74 }

```

	Universidad Nacional de San Agustín de Arequipa Facultad de Ingeniería de Producción y Servicios Escuela Profesional de Ingeniería de Sistemas	
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		
Aprobación: 2022/03/01	Código: GUIA-PRLD-001	Página: 4

```

74
75     cout << "Iniciando calculo..." << endl;
76
77     // se crean y lanzan hilos
78     vector<thread> hilos;
79     hilos.reserve(numero_hilos);
80
81     for (int i = 0; i < numero_hilos; i++) {
82         hilos.emplace_back(calcularTrapecio, limite_inferior, limite_superior,
83                             i, numero_hilos);
84     }
85
86     // se espera a que terminen todos los hilos
87     for (auto& hilo : hilos) {
88         hilo.join();
89     }
90
91     cout << endl;
92     cout << "RESULTADO FINAL:" << endl;
93     cout << "    Area = " << area_total << endl;
94     cout << endl;
95     return 0;
96 }

```

#### 4.1.2. Función Main en C++ - Implementación con Pool



A continuación se muestra la función principal en c++ implementado con Pool de threads [un hilo para un grupo de trapecios a generar]:

Listing 2: Función Main en TrapecioPool.cpp

```

1  int main() {
2      cout << "----- CALCULADORA DE INTEGRALES CON POOL DE HILOS -----"
3          << endl;
4      cout << endl;
5
6      cout << "Ingrese la función f(x) a integrar:" << endl;
7      cout << "    Ejemplos:" << endl;
8      cout << "    - 2*x^2 + 3*x + 0.5" << endl;
9      cout << "    - sin(x)" << endl;
10     cout << "    - cos(x) + x^2" << endl;
11     cout << "    - exp(x)" << endl;
12     cout << "    - log(x)" << endl;
13     cout << "    - sqrt(x)" << endl;
14     cout << endl;
15     cout << "f(x) = ";
16
17     getline(cin, funcion_matematica);
18
19     // eliminar espacios en blanco al inicio y final
20     size_t start = funcion_matematica.find_first_not_of(" \t");
21     if (start == string::npos) {
22         cout << "Error: No se ingreso ninguna funcion" << endl;
23         return 1;
24     }
25     size_t end = funcion_matematica.find_last_not_of(" \t");



```

	<p>Universidad Nacional de San Agustín de Arequipa Facultad de Ingeniería de Producción y Servicios Escuela Profesional de Ingeniería de Sistemas</p>	
<p>Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación</p>		
<p>Aprobación: 2022/03/01</p>	<p>Código: GUIA-PRLD-001</p>	<p>Página: 5</p>

```

25     funcion_matematica = funcion_matematica.substr(start, end - start + 1);
26
27     cout << "Funcion ingresada: " << funcion_matematica << " " << endl;
28
29     // validamos la funcion
30     try {
31         double test1 = evaluarFuncion(funcion_matematica, 1.0);
32         double test2 = evaluarFuncion(funcion_matematica, 2.0);
33         if (test1 != 0.0 || test2 != 0.0) {
34             cout << "Funcion valida" << endl;
35         }
36     } catch (...) {
37         cout << "Error: La funcion no es valida" << endl;
38         exit(1);
39     }
40
41     // se piden los parametros de la integral
42     double limite_inferior, limite_superior;
43     int decimales, numero_hilos;
44
45     cout << "Limite inferior (a): ";
46     cin >> limite_inferior;
47     cout << "Limite superior (b): ";
48     cin >> limite_superior;
49     cout << "Numero de hilos (trapecios): ";
50     cin >> numero_hilos;
51     cout << "Decimales en el resultado: ";
52     cin >> decimales;
53     cout << endl;
54
55     // validaciones
56     if (limite_superior <= limite_inferior) {
57         cout << "Error: El limite superior debe ser mayor al inferior" << endl;
58         return 1;
59     }
60
61     if (numero_hilos <= 0) {
62         cout << "Error: El numero de hilos debe ser positivo" << endl;
63         return 1;
64     }
65
66     if (numero_hilos > 1000) {
67         cout << "Error: Muchos hilos pueden afectar el rendimiento" << endl;
68     }
69
70     if (decimales < 0 || decimales > 10) {
71         cout << "Error: El numero de decimales debe estar entre 0 y 10" << endl;
72         ;
73         return 1;
74     }
75
76     cout << "Iniciando calculo..." << endl;
77
78     ThreadPool pool(numero_hilos);
79
80     // agregar tareas al pool
81     for (int i = 0; i < numero_hilos; i++) {

```

	<p>Universidad Nacional de San Agustín de Arequipa Facultad de Ingeniería de Producción y Servicios Escuela Profesional de Ingeniería de Sistemas</p>	
<p>Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación</p>		
<p>Aprobación: 2022/03/01</p>	<p>Código: GUIA-PRLD-001</p>	<p>Página: 6</p>

```

81     pool.enqueue( [= ] {
82         calcularTrapezio( limite_inferior, limite_superior, i, numero_hilos)
83     });
84 }
85 pool.wait_for_completion();
86
87 // el destructor del pool espera a que terminen todas las tareas
88
89 cout << endl;
90 cout << "RESULTADO FINAL:" << endl;
91 cout << "    Area = " << area_total << endl;
92 cout << endl;
93
94     return 0;
95 }
```



#### 4.1.3. Función Main en Golang

A continuación se muestra la función principal en Golang de la primera implementación [un hilo por trapezio a generar]:

Listing 3: Función Main en Golang

```

1  func main() {
2      fmt.Println("----- CALCULADORA DE INTEGRALES CON HILOS NORMALES
3      -----")
4      fmt.Println()
5
6      fmt.Println("Ingrese la función f(x) a integrar:")
7      fmt.Println("    Ejemplos:")
8      fmt.Println("    - 2*x^2 + 3*x + 0.5")
9      fmt.Println("    - sin(x)")
10     fmt.Println("    - cos(x) + x^2")
11     fmt.Println("    - exp(x)")
12     fmt.Println("    - log(x)")
13     fmt.Println("    - sqrt(x)")
14     fmt.Println()
15     fmt.Print("f(x) = ")
16
17     scanner := bufio.NewScanner(os.Stdin)
18     scanner.Scan()
19     funcionMat = strings.TrimSpace(scanner.Text())
20
21     if funcionMat == "" {
22         fmt.Println("Error: No se ingreso ninguna funcion")
23         return
24     }
25
26     fmt.Printf("Funcion ingresada: '%s'\n", funcionMat)
27
28     // Validar funcion
29     test1 := evaluarFuncion(funcionMat, 1.0)
30     test2 := evaluarFuncion(funcionMat, 2.0)
31     fmt.Printf("f(1) = %f, f(2) = %f\n", test1, test2)
32     fmt.Println("Funcion valida")
33 }
```

	Universidad Nacional de San Agustín de Arequipa Facultad de Ingeniería de Producción y Servicios Escuela Profesional de Ingeniería de Sistemas	
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		
Aprobación: 2022/03/01	Código: GUIA-PRLD-001	Página: 7



```

33     var limiteInferior, limiteSuperior float64
34     var decimales, numeroHilos int
35
36     fmt.Print("Limite inferior (a): ")
37     fmt.Scanf("%f", &limiteInferior)
38     fmt.Print("Limite superior (b): ")
39     fmt.Scanf("%f", &limiteSuperior)
40     fmt.Print("Numero de hilos (trapeacios): ")
41     fmt.Scanf("%d", &numeroHilos)
42     fmt.Print("Decimales en el resultado: ")
43     fmt.Scanf("%d", &decimales)
44     fmt.Println()
45
46     // Validaciones
47     if limiteSuperior <= limiteInferior {
48         fmt.Println("Error: El limite superior debe ser mayor al inferior")
49         return
50     }
51
52     if numeroHilos <= 0 {
53         fmt.Println("Error: El numero de hilos debe ser positivo")
54         return
55     }
56
57     if numeroHilos > 1000 {
58         fmt.Println("Error: Muchos hilos pueden afectar el rendimiento")
59     }
60
61     if decimales < 0 || decimales > 10 {
62         fmt.Println("Error: El numero de decimales debe estar entre 0 y 10")
63         return
64     }
65
66     fmt.Println("Iniciando calculo...")
67
68     var wg sync.WaitGroup
69
70     for i := 0; i < numeroHilos; i++ {
71         wg.Add(1)
72         go calcularTrapezio(limiteInferior, limiteSuperior, i, numeroHilos, &wg)
73     }
74
75     wg.Wait()
76
77     fmt.Println()
78     fmt.Println("RESULTADO FINAL:")
79     fmt.Printf("    Area = %.f\n", decimales, areaTotal)
80     fmt.Println()
81 }

```

#### 4.1.4. Función Main en Go - Implementación con Pool

A continuación se muestra la función principal en Go implementado con Pool de threads [un hilo para un grupo de trapeacios a generar]:

	<p>Universidad Nacional de San Agustín de Arequipa Facultad de Ingeniería de Producción y Servicios Escuela Profesional de Ingeniería de Sistemas</p>	
<p>Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación</p>		
<p>Aprobación: 2022/03/01</p>	<p>Código: GUIA-PRLD-001</p>	<p>Página: 8</p>



Listing 4: Función Main en TrapecioPool.go

```

1  func main() {
2      fmt.Println("----- CALCULADORA DE INTEGRALES CON POOL DE HILOS
          -----")
3      fmt.Println()
4
5      fmt.Println("Ingrese la función f(x) a integrar:")
6      fmt.Println("    Ejemplos:")
7      fmt.Println("    - 2*x^2 + 3*x + 0.5")
8      fmt.Println("    - sin(x)")
9      fmt.Println("    - cos(x) + x^2")
10     fmt.Println("    - exp(x)")
11     fmt.Println("    - log(x)")
12     fmt.Println("    - sqrt(x)")
13     fmt.Println()
14     fmt.Print("f(x) = ")
15
16     scanner := bufio.NewScanner(os.Stdin)
17     scanner.Scan()
18     funcionMatPool = strings.TrimSpace(scanner.Text())
19
20     if funcionMatPool == "" {
21         fmt.Println("Error: No se ingreso ninguna funcion")
22         return
23     }
24
25     fmt.Printf("Funcion ingresada: '%s'\n", funcionMatPool)
26
27     // Validar funcion
28     test1 := evaluarFuncionPool(funcionMatPool, 1.0)
29     test2 := evaluarFuncionPool(funcionMatPool, 2.0)
30     fmt.Printf("f(1) = %f, f(2) = %f\n", test1, test2)
31     fmt.Println("Funcion valida")
32
33     var limiteInferior, limiteSuperior float64
34     var decimales, numeroHilos int
35
36     fmt.Print("Limite inferior (a): ")
37     fmt.Scanf("%f", &limiteInferior)
38     fmt.Print("Limite superior (b): ")
39     fmt.Scanf("%f", &limiteSuperior)
40     fmt.Print("Numero de hilos (trapecios): ")
41     fmt.Scanf("%d", &numeroHilos)
42     fmt.Print("Decimales en el resultado: ")
43     fmt.Scanf("%d", &decimales)
44     fmt.Println()
45
46     // Validaciones
47     if limiteSuperior <= limiteInferior {
48         fmt.Println("Error: El limite superior debe ser mayor al inferior")
49         return
50     }
51
52     if numeroHilos <= 0 {
53         fmt.Println("Error: El numero de hilos debe ser positivo")
54         return

```



	<p>Universidad Nacional de San Agustín de Arequipa Facultad de Ingeniería de Producción y Servicios Escuela Profesional de Ingeniería de Sistemas</p>	
<p>Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación</p>		
<p>Aprobación: 2022/03/01</p>	<p>Código: GUIA-PRLD-001</p>	<p>Página: 9</p>

```

55     }
56
57     if numeroHilos > 1000 {
58         fmt.Println("Error: Muchos hilos pueden afectar el rendimiento")
59     }
60
61     if decimales < 0 || decimales > 10 {
62         fmt.Println("Error: El numero de decimales debe estar entre 0 y 10")
63         return
64     }
65
66     fmt.Println("Iniciando calculo...")
67
68     // se crea el pool de hilos
69     poolSize := numeroHilos
70     if poolSize > runtime.NumCPU() {
71         poolSize = runtime.NumCPU()
72     }
73     pool := NewThreadPool(poolSize)
74
75     // se agreagn tareas al pool
76     for i := 0; i < numeroHilos; i++ {
77         i := i
78         pool.Submit(func() {
79             calcularTrapecioPool(limiteInferior, limiteSuperior, i, numeroHilos
80             )
81         })
82     }
83
84     pool.Wait()
85     pool.Close()
86
87     fmt.Println()
88     fmt.Println("RESULTADO FINAL:")
89     fmt.Printf("    Area = %.f\n", decimales, areaTotalPool)
90     fmt.Println()
91 }

```

#### 4.1.5. Función Main en Java



A continuación se muestra la función principal en Java de la primera implementación [un hilo por trapecio a generar]:

Listing 5: Función Main en Java

```

1  public static void main(String[] args) {
2      System.out.println("----- CALCULADORA DE INTEGRALES CON HILOS NORMALES -----");
3      System.out.println();
4
5      System.out.println("Ingrese la función f(x) a integrar:");
6      System.out.println(" Ejemplos:");
7      System.out.println(" - 2*x^2 + 3*x + 0.5");
8      System.out.println(" - SIN(x)");
9      System.out.println(" - COS(x) + x^2");
10     System.out.println(" - EXP(x)");
11     System.out.println(" - LOG(x)");
12     System.out.println(" - SQRT(x)");



```

	<p>Universidad Nacional de San Agustín de Arequipa Facultad de Ingeniería de Producción y Servicios Escuela Profesional de Ingeniería de Sistemas</p>	
<p>Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación</p>		
<p>Aprobación: 2022/03/01</p>	<p>Código: GUIA-PRLD-001</p>	<p>Página: 10</p>

```

13 System.out.println();
14 System.out.print("f(x) = ");
15
16 Scanner scanner = new Scanner(System.in);
17 funcionMatematica = scanner.nextLine().trim();
18
19 if (funcionMatematica.isEmpty()) {
20     System.out.println("Error: No se ingreso ninguna funcion");
21     return;
22 }
23
24 System.out.println("Funcion ingresada: '" + funcionMatematica + "'");
25
26 // validar funcion
27 try {
28     double test1 = evaluarFuncion(funcionMatematica, 1.0);
29     double test2 = evaluarFuncion(funcionMatematica, 2.0);
30     if (test1 != 0.0 || test2 != 0.0) {
31         System.out.println("Funcion valida");
32     }
33 } catch (Exception e) {
34     System.out.println("Error: La funcion no es valida");
35     System.exit(1);
36 }
37
38 double limiteInferior, limiteSuperior;
39 int decimales, numeroHilos;
40
41 System.out.print("Limite inferior (a): ");
42 limiteInferior = scanner.nextDouble();
43 System.out.print("Limite superior (b): ");
44 limiteSuperior = scanner.nextDouble();
45 System.out.print("Numero de hilos (trapezios): ");
46 numeroHilos = scanner.nextInt();
47 System.out.print("Decimales en el resultado: ");
48 decimales = scanner.nextInt();
49 System.out.println();
50
51 // validaciones
52 if (limiteSuperior <= limiteInferior) {
53     System.out.println("Error: El limite superior debe ser mayor al inferior");
54     return;
55 }
56
57 if (numeroHilos <= 0) {
58     System.out.println("Error: El numero de hilos debe ser positivo");
59     return;
60 }
61
62 if (numeroHilos > 1000) {
63     System.out.println("Error: Muchos hilos pueden afectar el rendimiento");
64 }
65
66 if (decimales < 0 || decimales > 10) {
67     System.out.println("Error: El numero de decimales debe estar entre 0 y 10");
68     return;
69 }

```

	Universidad Nacional de San Agustín de Arequipa Facultad de Ingeniería de Producción y Servicios Escuela Profesional de Ingeniería de Sistemas	
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		
Aprobación: 2022/03/01	Código: GUIA-PRLD-001	Página: 11

```

70
71     System.out.println("Iniciando calculo...");
72
73     CountdownLatch latch = new CountdownLatch(numeroHilos);
74
75     for (int i = 0; i < numeroHilos; i++) {
76         final int indice = i;
77         new Thread(() -> calcularTrapezio(limiteInferior, limiteSuperior, indice, numeroHilos,
78             latch)).start();
79     }
80
81     try {
82         latch.await();
83     } catch (InterruptedException e) {
84         Thread.currentThread().interrupt();
85     }
86
87     System.out.println();
88     System.out.println("RESULTADO FINAL:");
89     System.out.printf(" Area = %.*f%n", decimales, areaTotal);
90     System.out.println();
91
92     scanner.close();
93 }

```

#### 4.1.6. Función Main en Java - Implementación con Pool



A continuación se muestra la función principal en Java implementado con Pool de threads [un hilo para un grupo de trapezios a generar]:

Listing 6: Función Main en TrapecioPool.java

```

1     public static void main(String[] args) {
2         System.out.println("----- CALCULADORA DE INTEGRALES CON POOL DE HILOS
3             -----");
4         System.out.println();
5
6         System.out.println("Ingrese la función f(x) a integrar:");
7         System.out.println("    Ejemplos:");
8         System.out.println("    - 2*x^2 + 3*x + 0.5");
9         System.out.println("    - SIN(x)");
10        System.out.println("    - COS(x) + x^2");
11        System.out.println("    - EXP(x)");
12        System.out.println("    - LOG(x)");
13        System.out.println("    - SQRT(x)");
14        System.out.println();
15        System.out.print("f(x) = ");
16
17        Scanner scanner = new Scanner(System.in);
18        funcionMatematicaPool = scanner.nextLine().trim();
19
20        if (funcionMatematicaPool.isEmpty()) {
21            System.out.println("Error: No se ingreso ninguna funcion");
22            return;
23        }
24
25        System.out.println("Funcion ingresada: '" + funcionMatematicaPool + "'");



```

	<p>Universidad Nacional de San Agustín de Arequipa Facultad de Ingeniería de Producción y Servicios Escuela Profesional de Ingeniería de Sistemas</p>	
<p>Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación</p>		
<p>Aprobación: 2022/03/01</p>	<p>Código: GUIA-PRLD-001</p>	<p>Página: 12</p>

```

25
26 // validar funcion
27 try {
28     double test1 = evaluarFuncionPool(funcionMatematicaPool, 1.0);
29     double test2 = evaluarFuncionPool(funcionMatematicaPool, 2.0);
30     if (test1 != 0.0 || test2 != 0.0) {
31         System.out.println("Funcion valida");
32     }
33 } catch (Exception e) {
34     System.out.println("Error: La funcion no es valida");
35     System.exit(1);
36 }
37
38 double limiteInferior, limiteSuperior;
39 int decimales, numeroHilos;
40
41 System.out.print("Limite inferior (a): ");
42 limiteInferior = scanner.nextDouble();
43 System.out.print("Limite superior (b): ");
44 limiteSuperior = scanner.nextDouble();
45 System.out.print("Numero de hilos (trapecios): ");
46 numeroHilos = scanner.nextInt();
47 System.out.print("Decimales en el resultado: ");
48 decimales = scanner.nextInt();
49 System.out.println();
50
51 // validaciones
52 if (limiteSuperior <= limiteInferior) {
53     System.out.println("Error: El limite superior debe ser mayor al
54         inferior");
55     return;
56 }
57
58 if (numeroHilos <= 0) {
59     System.out.println("Error: El numero de hilos debe ser positivo");
60     return;
61 }
62
63 if (numeroHilos > 1000) {
64     System.out.println("Error: Muchos hilos pueden afectar el rendimiento");
65     ;
66 }
67
68 if (decimales < 0 || decimales > 10) {
69     System.out.println("Error: El numero de decimales debe estar entre 0 y
70         10");
71     return;
72 }
73
74 System.out.println("Iniciando calculo...");
75
76 // Crear pool de hilos
77 int poolSize = Math.min(numeroHilos, Runtime.getRuntime().
    availableProcessors());
    ExecutorService executor = Executors.newFixedThreadPool(poolSize);
78
79 // Enviar tareas al pool

```

	Universidad Nacional de San Agustín de Arequipa Facultad de Ingeniería de Producción y Servicios Escuela Profesional de Ingeniería de Sistemas	
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		
Aprobación: 2022/03/01	Código: GUIA-PRLD-001	Página: 13

```

78     for (int i = 0; i < numeroHilos; i++) {
79         final int indice = i;
80         executor.submit(() -> calcularTrapecioPool(limiteInferior,
81             limiteSuperior, indice, numeroHilos));
82     }
83     executor.shutdown();
84     try {
85         if (!executor.awaitTermination(60, TimeUnit.SECONDS)) {
86             executor.shutdownNow();
87         }
88     } catch (InterruptedException e) {
89         executor.shutdownNow();
90         Thread.currentThread().interrupt();
91     }
92
93     System.out.println();
94     System.out.println("RESULTADO FINAL:");
95     System.out.printf("    Area = %.*f%n", decimales, areaTotalPool);
96     System.out.println();
97
98     scanner.close();
99 }

```

#### 4.1.7. Pruebas de ejecución:

- Se probó el funcionamiento de golang,
- ingresando la función
- el rango (2-10) por mencionar algún ejemplo
- y la cantidad de hilos que se van a usar

**Prueba de ejecución para la primera implementación de golang**

```

alexander@DESKTOP-TV3V3T3:~/lab_teo/lab03/l-go$ ./build_run.sh
Iniciando proyecto Go...
Dependencias instaladas correctamente

----- CALCULADORA DE INTEGRALES CON HILOS NORMALES -----

Ingrese la función f(x) a integrar:
Ejemplos:
- 2*x^2 + 3*x + 0.5
- sin(x)
- cos(x) + x^2
- exp(x)
- log(x)
- sqrt(x)



f(x) = sin(x) + 5
Funcion ingresada: 'sin(x) + 5'
f(1) = 5.841471, f(2) = 5.909297
Funcion valida
Limite inferior (a): 4
Limite superior (b): 9
Numero de hilos (trapeacios): 8
Decimales en el resultado: 2

Iniciando calculo...
=> Hilo 0: [4.000000, 4.625000] -> Area = 2.577192
=> Hilo 7: [8.375000, 9.000000] -> Area = 3.524822
=> Hilo 5: [7.125000, 7.750000] -> Area = 3.668891
=> Hilo 1: [4.625000, 5.250000] -> Area = 2.545275
=> Hilo 4: [6.500000, 7.125000] -> Area = 3.425304
=> Hilo 6: [7.750000, 8.375000] -> Area = 3.706847
=> Hilo 2: [5.250000, 5.875000] -> Area = 2.732538
=> Hilo 3: [5.875000, 6.500000] -> Area = 3.068180

RESULTADO FINAL:
Area = 25.25
  
```

Figura 1: Prueba de ejecución para la implementación inicial

Prueba de ejecución para la primera implementación con Pool de threads en golang

	Universidad Nacional de San Agustín de Arequipa Facultad de Ingeniería de Producción y Servicios Escuela Profesional de Ingeniería de Sistemas	
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		
Aprobación: 2022/03/01	Código: GUIA-PRLD-001	Página: 15

```

----- CALCULADORA DE INTEGRALES CON POOL DE HILOS -----

Ingrese la función f(x) a integrar:
Ejemplos:
- 2*x^2 + 3*x + 0.5
- sin(x)
- cos(x) + x^2
- exp(x)
- log(x)
- sqrt(x)

f(x) = sin(x) + 5
Funcion ingresada: 'sin(x) + 5'
f(1) = 5.841471, f(2) = 5.909297
Funcion valida
Limite inferior (a): 4
Limite superior (b): 9
Numero de hilos (trapecios): 2
Decimales en el resultado: 2

Iniciando calculo...
=> Hilo 0: [4.000000, 6.500000] -> Area = 11.822897
=> Hilo 1: [6.500000, 9.000000] -> Area = 13.284048



RESULTADO FINAL:
    Area = 25.11

❖alexander@DESKTOP-TV3CV3T3:~/lab_teo/lab03/l-go$

```

Figura 2: Prueba de ejecución

Prueba de ejecución para la primera implementación de java

	Universidad Nacional de San Agustín de Arequipa Facultad de Ingeniería de Producción y Servicios Escuela Profesional de Ingeniería de Sistemas	
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		
Aprobación: 2022/03/01	Código: GUIA-PRLD-001	Página: 16

```

----- CALCULADORA DE INTEGRALES CON HILOS NORMALES -----

Ingresa la función f(x) a integrar:
Ejemplos:
- 2*x^2 + 3*x + 0.5
- SIN(x)
- COS(x) + x^2
- EXP(x)
- LOG(x)
- SQRT(x)

f(x) = SIN(x) + 5
Funcion ingresada: 'SIN(x) + 5'
Funcion valida
Limite inferior (a): 4
Limite superior (b): 9
Numero de hilos (trapeacios): 8
Decimales en el resultado: 2



Iniciando calculo...
=> Hilo 2: [5.250000, 5.875000] -> Area = 3.185581
=> Hilo 6: [7.750000, 8.375000] -> Area = 3.212657
=> Hilo 4: [6.500000, 7.125000] -> Area = 3.199137
=> Hilo 0: [4.000000, 4.625000] -> Area = 3.171997
=> Hilo 1: [4.625000, 5.250000] -> Area = 3.178792
=> Hilo 7: [8.375000, 9.000000] -> Area = 3.219402
=> Hilo 5: [7.125000, 7.750000] -> Area = 3.205902
=> Hilo 3: [5.875000, 6.500000] -> Area = 3.192363

```

Figura 3: Prueba de ejecución para la implementación inicial

Prueba de ejecución para la primera implementación con Pool de threads en java



	<p>Universidad Nacional de San Agustín de Arequipa Facultad de Ingeniería de Producción y Servicios Escuela Profesional de Ingeniería de Sistemas</p>	
<p>Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación</p>		
<p>Aprobación: 2022/03/01</p>	<p>Código: GUIA-PRLD-001</p>	<p>Página: 17</p>

```

————— CALCULADORA DE INTEGRALES CON POOL DE HILOS —————

Ingrese la función f(x) a integrar:
Ejemplos:
- 2*x^2 + 3*x + 0.5
- sin(x)
- cos(x) + x^2
- exp(x)
- log(x)
- sqrt(x)

f(x) = sin(x)
Funcion ingresada: 'sin(x)'
Funcion valida
Limite inferior (a): 2
Limite superior (b): 10
Numero de hilos (trapecios): 7
Decimales en el resultado: 2



Iniciando calculo ...
⇒ Hilo 6: [8.85714, 10] → Area = -0.00364662
⇒ Hilo ⇒ Hilo 03: [⇒ Hilo 2: [2, 3.14286] → Area = 0.5188764.28571,
5.42857] → Area = -0.951237
: [5.42857, 6.57143] → Area = -0.268599
⇒ Hilo 1: [3.14286, 4.28571] → Area = -0.520921
⇒ Hilo 4: [6.57143, 7.71429] → Area = 0.728301
⇒ Hilo 5: [7.71429, 8.85714] → Area = 0.873085

RESULTADO FINAL:
Area = 0.375858

```

Figura 4: Prueba de ejecución

Prueba de ejecución para la primera implementación de cpp

	Universidad Nacional de San Agustín de Arequipa Facultad de Ingeniería de Producción y Servicios Escuela Profesional de Ingeniería de Sistemas	
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		
Aprobación: 2022/03/01	Código: GUIA-PRLD-001	Página: 18

```

————— CALCULADORA DE INTEGRALES CON HILOS NORMALES —————

Ingresa la función f(x) a integrar:
Ejemplos:
- 2*x^2 + 3*x + 0.5
- sin(x)
- cos(x) + x^2
- exp(x)
- log(x)
- sqrt(x)

f(x) = sin(x)
Funcion ingresada: 'sin(x)'
Funcion valida
Limite inferior (a): 2
Limite superior (b): 9
Numero de hilos (trapeacios): 10
Decimales en el resultado: 2



Iniciando calculo ...
⇒ Hilo 1: [2.7, 3.4] → Area = 0.0601436
⇒ Hilo 0: [2, 2.7] → Area = 0.467837
⇒ Hilo 9: [8.3, 9] → Area = 0.460002
⇒ Hilo 2: [3.4, 4.1] → Area = -0.375836
⇒ Hilo 3: [4.1, 4.8] → Area = -0.635055
⇒ Hilo 7: [6.9, 7.6] → Area = 0.541226
⇒ Hilo 4: [4.8, 5.5] → Area = -0.595597
⇒ Hilo 5: [5.5, 6.2] → Area = -0.27602
⇒ Hilo 8: [7.6, 8.3] → Area = 0.654532
⇒ Hilo 6: [6.2, 6.9] → Area = 0.173373

RESULTADO FINAL:
Area = 0.474605

```

Figura 5: Prueba de ejecución para la implementación inicial

Prueba de ejecución para la primera implementación con Pool de threads en cpp

	<p>Universidad Nacional de San Agustín de Arequipa Facultad de Ingeniería de Producción y Servicios Escuela Profesional de Ingeniería de Sistemas</p>	
<p>Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación</p>		
<p>Aprobación: 2022/03/01</p>	<p>Código: GUIA-PRLD-001</p>	<p>Página: 19</p>

```

————— CALCULADORA DE INTEGRALES CON POOL DE HILOS —————

Ingrese la función f(x) a integrar:
Ejemplos:
- 2*x^2 + 3*x + 0.5
- sin(x)
- cos(x) + x^2
- exp(x)
- log(x)
- sqrt(x)

f(x) = sin(x)
Funcion ingresada: 'sin(x)'
Funcion valida
Limite inferior (a): 2
Limite superior (b): 10
Numero de hilos (trapecios): 7
Decimales en el resultado: 2

Iniciando calculo ...
⇒ Hilo ⇒ Hilo 3: [1: [3.14286, 4.28571] → Area = 5.42857-0.520921
, 6.57143] → Area = -0.268599
⇒ Hilo 2: [4.28571, 5.42857] → Area = -0.951237
⇒ Hilo 6: [8.85714, 10] → Area = -0.00364662
⇒ Hilo 0: [2, 3.14286] → Area = 0.518876
⇒ Hilo 4: [6.57143, 7.71429] → Area = 0.728301
⇒ Hilo 5: [7.71429, 8.85714] → Area = 0.873085

RESULTADO FINAL:
Area = 0.375858



```

Figura 6: Prueba de ejecución

## 4.2. Commits realizados

### 4.2.1. Primer Commit

- Este commit se hizo despues de completar en un 80 % el código de la implementación en c++
- Se crearon las clases requeridas, CmakeList para las librerias que se requieren
- Una clase main para probar el funcionamiento de las clases implementadas.

	<p>Universidad Nacional de San Agustín de Arequipa Facultad de Ingeniería de Producción y Servicios Escuela Profesional de Ingeniería de Sistemas</p>	
<p>Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación</p>		
<p>Aprobación: 2022/03/01</p>	<p>Código: GUIA-PRLD-001</p>	<p>Página: 20</p>

```
WSL at C ~ / ... /laboratorios/lab03/l-cpp C/main 15ms
12:42:30 git add .
WSL at C ~ / ... /laboratorios/lab03/l-cpp C/main 35ms
12:42:38 git commit -m "Código en cpp semicompleto"
[main 3a44365] Código en cpp semicompleto
6 files changed, 452 insertions(+), 67 deletions(-)
create mode 100644 laboratorios/lab03/l-cpp/.gitignore
create mode 100644 laboratorios/lab03/l-cpp/CMakeLists.txt
create mode 100755 laboratorios/lab03/l-cpp/build_run.sh
create mode 100644 laboratorios/lab03/l-cpp/src/trapecio.cpp
create mode 100644 laboratorios/lab03/l-cpp/src/trapecioPool.cpp
delete mode 100644 laboratorios/lab03/l-cpp/trapecio.cpp
WSL at C ~ / ... /laboratorios/lab03/l-cpp C/main 33ms
12:43:26 git push
```

Figura 7: Commit 01

#### 4.2.2. Segundo Commit



- Este commit se hizo despues terminar la implementación de código para golang, despues de probar el funcionamiento.

```
alexander@DESKTOP-TV3V3T3:~/Lab_teo/Lab03$ git add .
alexander@DESKTOP-TV3V3T3:~/Lab_teo/Lab03$ git commit -m "Calculo de la integral en Go completo"
[main 9496f59] Calculo de la integral en Go completo
7 files changed, 366 insertions(+), 2 deletions(-)
create mode 100755 lab03/l-go/build_run.sh
create mode 100644 lab03/l-go/go.mod
create mode 100644 lab03/l-go/go.sum
create mode 100644 lab03/l-go/trapecio.go
create mode 100644 lab03/l-go/trapecioPool.go
alexander@DESKTOP-TV3V3T3:~/Lab_teo/Lab03$ git push
Enumerating objects: 17, done.
Counting objects: 100% (17/17), done.
Delta compression using up to 8 threads
Compressing objects: 100% (12/12), done.
Writing objects: 100% (12/12), 3.36 KiB | 859.00 KiB/s, done.
Total 12 (delta 4), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (4/4), completed with 3 local objects.
To https://github.com/avillaq/teo.git
4cbb56c..9496f59 main -> main
alexander@DESKTOP-TV3V3T3:~/Lab_teo/Lab03$
```

Figura 8: Commit 02

#### 4.2.3. Tercer Commit

- En este commit se completo la implementación del código para java
- Se uso Maven como administrador de paquetes

	<p>Universidad Nacional de San Agustín de Arequipa Facultad de Ingeniería de Producción y Servicios Escuela Profesional de Ingeniería de Sistemas</p>	
<p>Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación</p>		
<p>Aprobación: 2022/03/01</p>	<p>Código: GUIA-PRLD-001</p>	<p>Página: 21</p>

```

~/.lab03/l-java  o Jmain  17.126s
04:24:46 git add .
~/.lab03/l-java  o Jmain  184ms
04:31:42 git commit -m "Implementación completa de Trapecio y TrapecioPool en java"
[main 7db6a42] Implementación completa de Trapecio y TrapecioPool en java
6 files changed, 425 insertions(+), 93 deletions(-)
create mode 100644 laboratorios/lab03/l-java/.gitignore
delete mode 100644 laboratorios/lab03/l-java/Trapecio.java
create mode 100644 laboratorios/lab03/l-java/pom.xml
create mode 100644 laboratorios/lab03/l-java/src/main/java/com/lab/hilos/App.java
create mode 100644 laboratorios/lab03/l-java/src/main/java/com/lab/hilos/trapecio/Trapecio.java
create mode 100644 laboratorios/lab03/l-java/src/main/java/com/lab/hilos/trapecio/TrapecioPool.java
~/.lab03/l-java  o Jmain  317ms
04:32:22 git push -u origin main

```

Figura 9: Commit 03

### 4.3. Estructura del laboratorio



A continuación se muestra la estructura de archivos y carpetas del laboratorio realizado: Claramente los archivos de compilación de Java y otros que se pudieron generar no se subieron al repositorio.

```

~/.../Teo/laboratorios/lab03  o Jmain  83ms
12:42:39 tree /f
├── l-cpp
│   ├── .gitignore
│   ├── build_run.sh
│   └── CMakeLists.txt
├── src
│   ├── trapecio.cpp
│   └── trapecioPool.cpp
├── l-go
│   ├── build_run.sh
│   ├── go.mod
│   ├── go.sum
│   ├── trapecio.go
│   └── trapecioPool.go
├── l-java
│   ├── .gitignore
│   ├── pom.xml
│   └── src
│       ├── main
│       │   └── java
│       │       ├── com
│       │       │   ├── lab
│       │       │   │   ├── hilos
│       │       │   │   │   ├── App.java
│       │       │   │   │   └── trapecio
│       │       │   │       ├── Trapecio.java
│       │       │   │       └── TrapecioPool.java

```

Figura 10: Estructura de laboratorio

	Universidad Nacional de San Agustín de Arequipa Facultad de Ingeniería de Producción y Servicios Escuela Profesional de Ingeniería de Sistemas	
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		
Aprobación: 2022/03/01	Código: GUIA-PRLD-001	Página: 22

## 5. Cuestionario

### 5.1. ¿Cuál de los LP posee ventajas para programar paralelamente?

- Java y Go poseen ventajas para programar paralelamente debido a que soportan concurrencia y paralelismo de manera nativa. En Java, se tiene la API de concurrencia y los hilos, mientras que en Go tiene goroutines y canales, lo que facilita la programación concurrente.

### 5.2. ¿Para cada lenguaje elabore una tabla cuando usa Pool de Treads?



- **Java:** Utiliza Executors para manejar pools de hilos.
- **Cpp:** Puede usar bibliotecas como ThreadPool para implementar pools de hilos.
- **Golang:** Utiliza goroutines y canales, pero no tiene un pool de hilos tradicional.

## 6. Rúbricas

### 6.1. Entregable Informe

Cuadro 1: Tipo de Informe

Informe	
Latex	El informe está en formato PDF desde Latex, con un formato limpio (buena presentación) y facil de leer.

	Universidad Nacional de San Agustín de Arequipa Facultad de Ingeniería de Producción y Servicios Escuela Profesional de Ingeniería de Sistemas	
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		
Aprobación: 2022/03/01	Código: GUIA-PRLD-001	Página: 23

## 6.2. Rúbrica para el contenido del Informe y demostración



- El alumno debe marcar o dejar en blanco en celdas de la columna **Checklist** si cumple con el ítem correspondiente.
- Si un alumno supera la fecha de entrega, su calificación será sobre la nota mínima aprobada, siempre y cuando cumpla con todos los ítems.
- El alumno debe autocalificarse en la columna **Estudiante** de acuerdo a la siguiente tabla:

Cuadro 2: Niveles de desempeño

	Nivel			
Puntos	Insatisfactorio 25 %	En Proceso 50 %	Satisfactorio 75 %	Sobresaliente 100 %
2.0	0.5	1.0	1.5	2.0
4.0	1.0	2.0	3.0	4.0

Cuadro 3: Rúbrica para contenido del Informe y demostración

Contenido y demostración		Puntos	Checklist	Estudiante	Profesor
1. GitHub	Hay enlace URL activo del directorio para el laboratorio hacia su repositorio GitHub con código fuente terminado y fácil de revisar.	2	X	2	
2. Commits	Hay capturas de pantalla de los commits más importantes con sus explicaciones detalladas. (El profesor puede preguntar para refrendar calificación).	4	X	3	
3. Código fuente	Hay porciones de código fuente importantes con numeración y explicaciones detalladas de sus funciones.	2	X	1.5	
4. Ejecución	Se incluyen ejecuciones/pruebas del código fuente explicadas gradualmente.	2	X	1.5	
5. Pregunta	Se responde con completitud a la pregunta formulada en la tarea. (El profesor puede preguntar para refrendar calificación).	2	X	2	
6. Fechas	Las fechas de modificación del código fuente están dentro de los plazos de fecha de entrega establecidos.	2	X	2	
7. Ortografía	El documento no muestra errores ortográficos.	2	X	2	
8. Madurez	El Informe muestra de manera general una evolución de la madurez del código fuente, explicaciones puntuales pero precisas y un acabado impecable. (El profesor puede preguntar para refrendar calificación).	4	X	3	
Total		20		17	

	<p>Universidad Nacional de San Agustín de Arequipa Facultad de Ingeniería de Producción y Servicios Escuela Profesional de Ingeniería de Sistemas</p>	
<p><b>Formato:</b> Guía de Práctica de Laboratorio / Talleres / Centros de Simulación</p>		
<p><b>Aprobación:</b> 2022/03/01</p>	<p><b>Código:</b> GUIA-PRLD-001</p>	<p><b>Página:</b> 24</p>

## 7. Referencias

- [1] <https://github.com/ArashPartow/exprtk>
- [2] <https://www.partow.net/programming/exprtk/index.html>
- [3] <https://www.geeksforgeeks.org/cpp/thread-pool-in-cpp/>
- [4] <https://github.com/Pramod-Devireddy/go-exprtk>
- [5] <https://germandv.me/blog/go-threadpool.html>
- [6] <https://github.com/ezylang/EvalEx>