



**INSTITUTO TECNOLÓGICO SUPERIOR DE ALAMO TEMAPACHE**

**REPORTE DE PRÁCTICAS**

**CARRERA:**

INGENIERIA EN SISTEMAS COMPUTACIONALES

**ASIGNATURA:**

PROCESO PERSONAL DE DESARROLLO DE SOFTWARE

**NOMBRE DE LA PRÁCTICA:**

USO DEL PROCESO PSP2.0

**NÚMERO DE LA PRÁCTICA:**

PROGRAMA 05

**INTEGRANTES DEL EQUIPO:**

GONZALO MARTINEZ SILVERIO

**DOCENTE:**

DRA. TANIA TURRUBIATES LOPEZ

**SEMESTRE:**

7

**GRUPO:**

701A

---



### **INTRODUCCIÓN:**

El Proceso de Software Personal (PSP) es un marco de trabajo que proporciona a los ingenieros de software un conjunto de prácticas y métodos para desarrollar software de manera sistemática y mejorar continuamente sus habilidades y procesos.

PSP2 es un enfoque metodológico desarrollado para mejorar la calidad y productividad en el desarrollo de software a nivel individual. Esta metodología se centra en la gestión y mejora del proceso personal de un ingeniero de software. Además, introduce elementos clave como revisiones de diseño y código, así como una planificación de calidad detallada. A través de la estimación de defectos, el seguimiento del rendimiento y la automatización de cálculos de eficiencia.

PSP2 permite a los profesionales de desarrollo de software mejorar la calidad de su trabajo, planificar con precisión y aumentar la productividad, todo ello sin sacrificar la calidad del producto final. Con énfasis en revisiones estructuradas y medidas de calidad derivadas.

PSP proporciona un marco sistemático para el desarrollo personalizado y la gestión eficiente de proyectos de software.

Este reporte se hablará de la realización de un software sobre la integración numérica utilizando la regla de Simpson para la distribución  $t$  utilizando PSP2.0.

### **OBJETIVO:**

Capacitar a los profesionales de desarrollo de software para producir software de alta calidad de manera más eficiente, proporcionando herramientas y métodos estructurados para la evaluación, mejora y gestión de su propio proceso de desarrollo.

### **COMPETENCIA A DESARROLLAR:**

Que el alumno conozca su ritmo de trabajo y pueda hacer una evaluación del tiempo que tarda y con respecto a ello conozca su ritmo de trabajo en cada etapa.

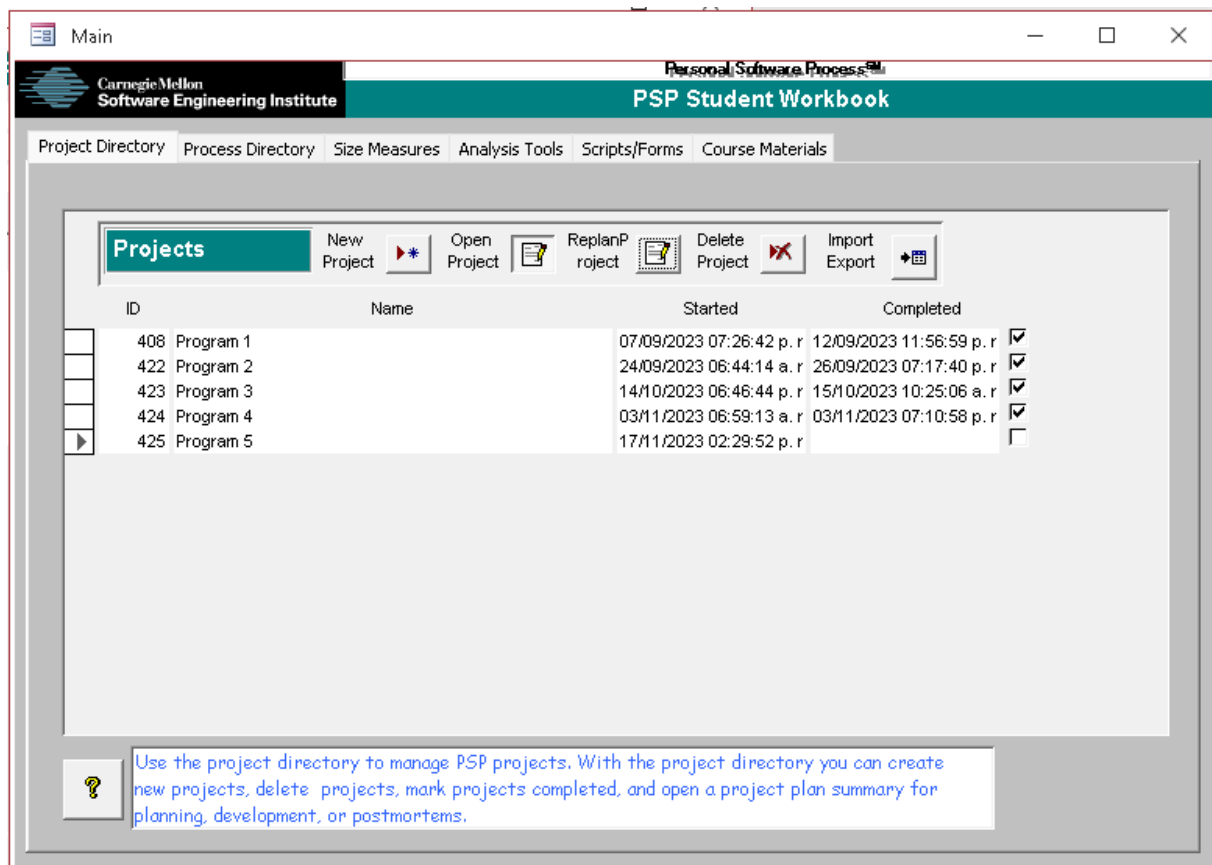


## MATERIAL Y EQUIPO (REQUERIMIENTOS):

- Computadora.
- NetBeans 8.2.
- JDK.
- Material de prácticas de PSP2 (Requerimientos, Time Log, etc.).

## DESARROLLO:

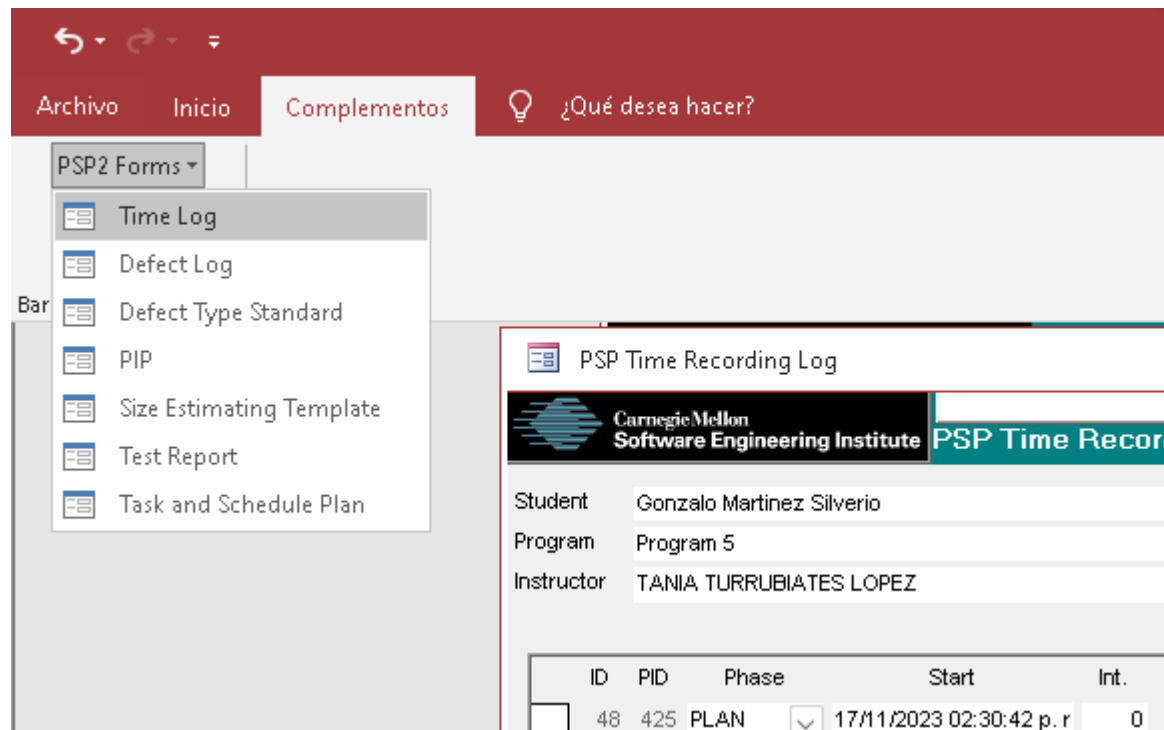
1.- Registramos el inicio de la etapa de planeación en la herramienta **PSP Student Workbook.mde**:





### ETAPA DE PLANEACION:

Comenzamos a registrar en el time log el inicio de la etapa de planeación:



#### 1.-Requerimientos del programa:

Usando PSP2, escribe un programa para integrar numéricamente una función usando la regla de Simpson. Utilice la distribución t como función.

Pruebe minuciosamente el programa. Como mínimo, calcule los valores para la integral de distribución t para los valores de la Tabla 1. Los valores esperados también se incluyen en la Tabla 1.

Prueba		Valor esperado	Valor actual
$x$	$dof$	$p$	
0 to $x= 1.1$	9	0.35006	
0 to $x= 1.1812$	10	0.36757	
0 to $x= 2.750$	30	0.49500	



# INSTITUTO TECNOLÓGICO SUPERIOR DE ALAMO TEMAPACHE

## REPORTE DE PRÁCTICAS

La regla de Simpson se puede utilizar para integrar una función de distribución estadística simétrica en un rango específico (por ejemplo, de 0 a algún valor  $x$ ).

1.  $num\_seg$  = número inicial de segmentos, un número par.
2.  $W = x/num\_seg$ , el ancho del segmento.
3.  $E$  = el error aceptable, por ejemplo: 0,00001.
4. Calcula el valor integral con la siguiente ecuación.

$$p = \frac{W}{3} \left[ F(0) + \sum_{i=1,3,5\dots}^{num\_seg-1} 4F(iW) + \sum_{i=2,4,6\dots}^{num\_seg-2} 2F(iW) + F(x) \right]$$

5. Calcule el valor integral nuevamente, pero esta vez con  $num\_seg = num\_seg * 2$ .
6. Si la diferencia entre estos dos resultados es mayor que  $E$ , duplique  $num\_seg$  y calcule el valor integral nuevamente. Continúe haciendo esto hasta que la diferencia entre los dos últimos resultados sea menor que  $E$ . El último resultado es la respuesta.

### 2.-Estimación de tiempo:

El tiempo estimado para este proyecto será de 1 hora y 30 minutos.

### 3.-Registramos la conclusión de la etapa de planeación registrándolo en el Time Log:

The screenshot shows the 'PSP Time Recording Log' window. At the top, it displays the Carnegie Mellon Software Engineering Institute logo and the title 'PSP Time Recording Log'. Below this, there are fields for 'Student' (Gonzalo Martinez Silverio), 'Program' (Program 5), and 'Instructor' (TANIA TURRUBIATES LOPEZ). To the right, there are fields for 'Start Date' (17-nov.-23), 'End Date', and 'Language' (Java). A table at the bottom lists time recording entries with columns: ID, PID, Phase, Start, Int., Stop Date and Time, Delta, and Comments. The first entry is: ID 48, PID 425, Phase PLAN, Start 17/11/2023 02:30:42 p. r, Int. 0, Stop Date and Time 17/11/2023 02:45:48 p. r, Delta 15.1, Comments Planeacion.

ID	PID	Phase	Start	Int.	Stop Date and Time	Delta	Comments
48	425	PLAN	17/11/2023 02:30:42 p. r	0	17/11/2023 02:45:48 p. r	15.1	Planeacion





# INSTITUTO TECNOLÓGICO SUPERIOR DE ALAMO TEMAPACHE

## REPORTE DE PRÁCTICAS

3.-Finalizando la etapa de Diseño registramos en el Time Log:

PSP Time Recording Log

Carnegie Mellon Software Engineering Institute

Student: Gonzalo Martinez Silverio  
Program: Program 5  
Instructor: TANIA TURRUBIATES LOPEZ

Start Date: 17-nov.-23  
End Date:  
Language: Java

ID	PID	Phase	Start	Int.	Stop Date and Time	Delta	Comments
48	425	PLAN	17/11/2023 02:30:42 p. r	0	17/11/2023 02:45:48 p. r	15.1	Planeacion
49	425	DLD	17/11/2023 02:45:50 p. r	5	17/11/2023 03:19:39 p. r	28.8	Interrupcion para ir al baño

### ETAPA DE CODIFICACION:

Registramos el inicio de la etapa de codificación en nuestro Time Log:

PSP Time Recording Log

Carnegie Mellon Software Engineering Institute

Student: Gonzalo Martinez Silverio  
Program: Program 5  
Instructor: TANIA TURRUBIATES LOPEZ

ID	PID	Phase	Start
48	425	PLAN	17/11/2023 02:30:42 p. r
49	425	DLD	17/11/2023 02:45:50 p. r
50	425	CODE	17/11/2023 03:19:42 p. r

2.- Abrimos NetBeans 8.2 nos dirigimos a File > New Project > Seleccionamos Java en Categories > Java Application en Projects > hacemos clic en Next > ingresamos el nombre el proyecto > creamos la clase llamada PROGRAMA5 > hacemos clic derecho > Create Main Class > hacemos clic en Finish y comenzamos a codificar nuestro programa conforme a nuestro diseño.

```
1 package PROGRAMA5;
2 /**
3  * PROGRAMA5_PSP
4  * Este programa realiza la integración numérica utilizando la regla de Simpson para la distribución t.
5  * @author GONZALO MARTINEZ SILVERIO
6  */
7 import java.util.Scanner;
8 import org.apache.commons.math3.distribution.TDistribution;
9
10 public class PROGRAMA5 {
```



# INSTITUTO TECNOLÓGICO SUPERIOR DE ALAMO TEMAPACHE

## REPORTE DE PRÁCTICAS

```
11 // Función para realizar la integración numérica usando la regla de Simpson para la distribución t
12 public static double integrarDistribucionT(double x, int gradosLibertad, int numSegmentos) {
13     // Calcular el ancho del segmento
14     double W = x / numSegmentos;
15     // Inicializar el valor de la integral
16     double valorIntegral = (W / 3) * (
17         // Término correspondiente a F(0)
18         distribucionT(0, gradosLibertad) +
19         // Términos con factor 4 en la sumatoria impar
20         4 * sumarDistribucionT(W, gradosLibertad, numSegmentos, 1) +
21         // Términos con factor 2 en la sumatoria par
22         2 * sumarDistribucionT(W, gradosLibertad, numSegmentos, 2) +
23         // Término correspondiente a F(x)
24         distribucionT(x, gradosLibertad)
25     );
26     return valorIntegral;
27 }
28
29 // Función para obtener el valor de la distribución t en un punto dado
30 private static double distribucionT(double x, int gradosLibertad) {
31     // Crear una distribución t con los grados de libertad dados
32     TDistribution tDistribution = new TDistribution(gradosLibertad);
33     // Calcular la densidad de probabilidad en el punto x
34     return tDistribution.density(x);
35 }
36
37 // Función para sumar los valores de la distribución t en puntos específicos
38 private static double sumarDistribucionT(double W, int gradosLibertad, int numSegmentos, int inicio) {
39     double suma = 0;
40     // Sumar los términos de la distribución t en la serie
41     for (int i = inicio; i < numSegmentos - 1; i += 2) {
42         suma += distribucionT(i * W, gradosLibertad);
43     }
44     return suma;
45 }
46
47 public static void main(String[] args) {
48     System.out.println("INSTITUTO TECNOLÓGICO SUPERIOR DE ALAMO TEMAPACHE");
49     System.out.println("=====");
50     // Scanner para leer desde el teclado
51     Scanner scanner = new Scanner(System.in);
52     double[] x = new double[3];
53     int[] gradosLibertad = new int[3];
54     double[] valoresEsperados = {0.35006, 0.36757, 0.49500};
55     // Introducir datos desde el teclado
56     for (int i = 0; i < 3; i++) {
57         System.out.print("Ingrese el valor de x[" + i + "]: ");
58         x[i] = scanner.nextDouble();
59         System.out.print("Ingrese los grados de libertad (dof) [" + i + "]: ");
60         gradosLibertad[i] = scanner.nextInt();
61         System.out.println("=====");
62     }
63     System.out.println("\n");
64     System.out.println("RESULTADOS:");
65     System.out.println("=====");
66
67     // Realizar cálculos y mostrar resultados
68     for (int i = 0; i < 3; i++) {
69         double actual = integrarDistribucionT(x[i], gradosLibertad[i], 100000);
70         System.out.println("Prueba de 0 a x = " + x[i] + ", grados de libertad (dof) = " + gradosLibertad[i]);
71         System.out.println("Valor Esperado (p): " + valoresEsperados[i]);
72         System.out.println("Valor Actual: " + actual);
73         System.out.println("=====");
74     }
75
76     // Cerrar el Scanner para liberar recursos
77     scanner.close();
78 }
```





# INSTITUTO TECNOLÓGICO SUPERIOR DE ALAMO TEMAPACHE

## REPORTE DE PRÁCTICAS

3.-Marcamos como finalizada la etapa de codificación en el Time Log:

<

### ETAPA DE COMPILACION:

Marcamos el inicio y final de la etapa de compilación:

<

Al finalizar esta etapa hicimos una prueba introduciendo los datos de la tabla 1 se encontró un error el cual procederé a solucionar en la siguiente etapa.:

### ETAPA DE TESTEO:

Registramos el inicio de nuestra etapa de Testeo:



Carnegie Mellon

Software Engineering Institute

PSP Time Recorder

Student	Gonzalo Martinez Silverio				
Program	Program 5				
Instructor	TANIA TURRUBIATES LOPEZ				

	ID	PID	Phase	Start	Int.
<input type="checkbox"/>	48	425	PLAN	<input type="checkbox"/> 17/11/2023 02:30:42 p. r	0
<input type="checkbox"/>	49	425	DLD	<input type="checkbox"/> 17/11/2023 02:45:50 p. r	5
<input type="checkbox"/>	50	425	CODE	<input type="checkbox"/> 17/11/2023 03:19:42 p. r	0
<input type="checkbox"/>	51	425	COMPILE	<input type="checkbox"/> 17/11/2023 04:01:15 p. r	0
<input type="checkbox"/>	52	425	UT	<input type="checkbox"/> 17/11/2023 04:02:21 p. r	0



# INSTITUTO TECNOLÓGICO SUPERIOR DE ALAMO TEMAPACHE

## REPORTE DE PRÁCTICAS

Ahora procedemos a ejecutar el programa nuevamente:


Introducimos los datos:

```
Output - PSP (run) x
run:
INSTITUTO TECNOLÓGICO SUPERIOR DE ALAMO TEMAPACHE
=====
Ingrese el valor de x[0]: 1.1
Ingrese los grados de libertad (dof) [0]: 9
=====
Ingrese el valor de x[1]: 1.1812
Ingrese los grados de libertad (dof) [1]: 10
=====
Ingrese el valor de x[2]: 2.750
Ingrese los grados de libertad (dof) [2]: 30
=====
```

Resultados:

```
RESULTADOS:
=====
Prueba de 0 a x = 1.1, grados de libertad (dof) = 9
Valor Esperado (p): 0.35006
Valor Actual: 0.350055590127881
=====
Prueba de 0 a x = 1.1812, grados de libertad (dof) = 10
Valor Esperado (p): 0.36757
Valor Actual: 0.3675703978913015
=====
Prueba de 0 a x = 2.75, grados de libertad (dof) = 30
Valor Esperado (p): 0.495
Valor Actual: 0.4949996078236181
=====
BUILD SUCCESSFUL (total time: 46 seconds)
```

Los resultados fueron verificados y en efecto son correctos por lo tanto marcamos como finalizada la etapa de pruebas:



Carnegie Mellon

Software Engineering Institute

Personal Software Process<sup>SM</sup>

PSP Time Recording Log

Student

Gonzalo Martinez Silverio

Program

Program 5

Instructor

TANIA TURRUBIATES LOPEZ

ID

425

Start Date

17-nov.-23

End Date

Language


Java

ID	PID	Phase	Start	Int.	Stop Date and Time	Delta	Comments	
	48	425	PLAN	17/11/2023 02:30:42 p. r	0	17/11/2023 02:45:48 p. r	15.1	Planeacion
	49	425	DLD	17/11/2023 02:45:50 p. r	5	17/11/2023 03:19:39 p. r	28.8	Interrupcion para ir al baño
	50	425	CODE	17/11/2023 03:19:42 p. r	0	17/11/2023 04:01:13 p. r	41.5	Codificacion
	51	425	COMPILE	17/11/2023 04:01:15 p. r	0	17/11/2023 04:02:20 p. r	1.1	Compilacion
	52	425	UT	17/11/2023 04:02:21 p. r	0	17/11/2023 04:06:33 p. r	4.2	Pruebas



### **ETAPA POSMORTEM:**

Registramos el inicio de esta etapa:



Carnegie Mellon  
Software Engineering Institute

PSP Time

Student	Gonzalo Martinez Silverio			
Program	Program 5			
Instructor	TANIA TURRUBIATES LOPEZ			

	ID	PID	Phase		Start
<input type="checkbox"/>	48	425	PLAN	<input type="button" value="v"/>	17/11/2023 02:30:42 p. r
<input type="checkbox"/>	49	425	DLD	<input type="button" value="v"/>	17/11/2023 02:45:50 p. r
<input type="checkbox"/>	50	425	CODE	<input type="button" value="v"/>	17/11/2023 03:19:42 p. r
<input type="checkbox"/>	51	425	COMPILE	<input type="button" value="v"/>	17/11/2023 04:01:15 p. r
<input type="checkbox"/>	52	425	UT	<input type="button" value="v"/>	17/11/2023 04:02:21 p. r
<input type="checkbox"/>	53	425	PM	<input type="button" value="v"/>	17/11/2023 04:06:34 p. r

En esta fase llevamos a cabo un exhaustivo análisis de nuestro proceso de desarrollo, centrándonos en cada etapa para evaluar nuestro desempeño y los resultados obtenidos de manera detallada.

### **ETAPA DE PLANEACION:**

La planificación de este proyecto se inició con un análisis detallado de la descripción del problema, comprendiendo los requisitos esenciales para el desarrollo del programa. A partir de estos requisitos, se realizó una estimación del tiempo necesario para completar el proyecto de manera efectiva.

### **ETAPA DE DISEÑO:**

En esta etapa en base a lo comprendido en la etapa de planeación se procedió a elaborar el diseño en base a los requerimientos, para lo cual se tuvo que diseñar un diagrama de clases detallado para visualizar las relaciones entre las entidades del sistema. La jerarquía de la clase y las interacciones se estructuraron de manera clara y coherente.

### **ETAPA DE CODIFICACION:**

Puesto que en la etapa de diseño tuvimos que codificar arreglo para probar solo se modificó a partir del diseño para poderse implementar en la clase principal, pero al no haber un diseño detallado de esta por parte del programador, el tiempo de codificación aumento considerablemente puesto que se hizo una compilación antes de terminar esta etapa la cual nos obligó a corregir los errores.

### **ETAPA DE COMPILACION:**

En esta etapa de compilación duro un tiempo mínimo casi inexistente debido a que no se encontraron errores de sintaxis.

---



### **ETAPA DE PRUEBAS:**

En esta etapa de pruebas identifiqué un error introducido durante la fase de codificación y lo corregí durante las pruebas. El error estaba relacionado con una función que no recibía un parámetro necesario.

### **EVALUACION DE RESULTADOS:**

#### **METAS ESTABLECIDAS:**

En la etapa de planeación se establecieron las metas principales en base a los requerimientos, las cuáles son:

- 1.- El tiempo estimado para la elaboración de este proyecto fue de 1hr 30 min.
- 2.- El programa debía de integrar numéricamente una función usando la regla de Simpson. Utilizar la distribución t como función
- 3.- Calcular los valores para la integral de distribución t para los valores de la tabla 1.
- 4.- El programa obtenga y muestre los resultados correctos.

.

#### **METAS NO CUMPLIDAS:**

Todas las metas fueron cumplidas por lo cual no hubo ningún inconveniente.

#### **METAS CUMPLIDAS:**

Gracias a la gran comprensión de los requerimientos y a un buen desempeño durante cada una de las etapas y un gran conocimiento se pudo cumplir todas las metas establecidas. Los datos mostrados son confiables ya que el programa respeta cada operación de las fórmulas correspondientes.

Una vez terminada esta etapa procedemos a registrarla en el time Log y procedemos a terminar el proceso.





# INSTITUTO TECNOLÓGICO SUPERIOR DE ALAMO TEMAPACHE

## REPORTE DE PRÁCTICAS

PSP Defect Recording Log

Carnegie Mellon Software Engineering Institute Personal Software Process<sup>SM</sup>

PSP Defect Recording Log

ID 425

Student Gonzalo Martinez Silverio Start Date 17-nov.-23

Program Program 5 End Date

Instructor TANIA TURRUBIATES LOPEZ Language Java

ID	PID	Date	Type	Phase Injected	Phase Removed	Fix Time	FixDefect
18	425	17/11/2023	80-Function	CODE	UT	1	1
Defect Description		Detecte un error inyectado en la etapa de codificación y lo removi en pruebas, el error consistia en una funcion que no recibia un parametro					

PSP Process Improvement Proposal

Carnegie Mellon Software Engineering Institute Personal Software Process<sup>SM</sup>

PSP Process Improvement Proposal

Student Gonzalo Martinez Silverio Start Date 17-nov.-23

Program Program 5 End Date

Instructor TANIA TURRUBIATES LOPEZ Language Java

UID #¿Nor Date 17-nov.-23

**Problem Description**  
Briefly describe the problems you encountered.

Durante la fase de codificación, identifiqué un error que se había introducido. Este error estaba relacionado con una función que no estaba recibiendo el parámetro necesario para obtener el resultado del calculo correcto.

**Proposal Description**  
Briefly describe the process improvements that you propose.

Asegúrame de que cada función tenga coherencia. Esto puede ayudar a detectar errores en las funciones individuales y asegurar que cada función esté trabajando como se espera.

**Other Notes and Comments**  
Note any other comments or observations that describe your experiences or improvement ideas.

Además de la mejora mencionada anteriormente, es importante recordar que no solo se trata de evitar errores. También se trata de hacer que el código sea de buena calidad.





# INSTITUTO TECNOLÓGICO SUPERIOR DE ALAMO TEMAPACHE

## REPORTE DE PRÁCTICAS

**PROBE Historical Data - Methods B and C**

ID	Name	End Date	Time	Plan A&M	E	Actual Time	Actual A&M	Outlier
+02	Program 4	11/03/23	240	58	58	129.37	51	<input type="checkbox"/>

**PROBE Method Parameters - Methods B and C**

	R <sup>2</sup> Size	Beta0 Size	Beta1 Size	R <sup>2</sup> Time	Beta0 Time	Beta1 Time
Method C	N/A	0	0.879310	N/A	0	2.2304598
Method B	0	0	0	0	0	0

**PROBE Historical Data - Method A**

ID	Name	End Date	Time	Plan A&M	E	Actual Time	Actual A&M	Outlier
+02	Program 4	11/03/23	240	58	58	129.37	51	<input type="checkbox"/>

**PROBE Method Parameters - Method A**

	R <sup>2</sup> Size	Beta0 Size	Beta1 Size	R <sup>2</sup> Time	Beta0 Time	Beta1 Time
Method A	0	0	0	0	0	0

Registro: 1 de 1 Sin filtro Buscar

Method A

424	58	51
-----	----	----

Registro: 1 de 1 Sin filtro Buscar

424	58	129.366666666667
-----	----	------------------

Method B

424	58	51
-----	----	----

Registro: 1 de 1 Sin filtro Buscar

424	58	129.366666666667
-----	----	------------------





# INSTITUTO TECNOLÓGICO SUPERIOR DE ALAMO TEMAPACHE

## REPORTE DE PRÁCTICAS

PSP2.0 Project Plan Summary

Carnegie Mellon  
Software Engineering Institute

Personal Software Process<sup>SM</sup>  
PSP2 Project Plan Summary

ID 425

Student

Gonzalo Martinez Silverio

Start Date

17-nov.-23

Program

Program 5

End Date

Instructor

TANIA TURRUBIATES LOPEZ

Language

Java

Summary

	Plan	Actual	To-Date
Productivity	14.9	25.7	0.0
Planned Time	130.0		980.0
Actual Time		123.9	1080.2
CPI			0.9
%Reused	8.5	3.1	3.1
%New Reusable	0.0	0.0	0.0
Test Defects/KLOC	0.0	18.9	18.9
Total Defects/KLOC	0.0	18.9	18.9
Yield%	0.0	0.0	0.0

Program Size Summary

LOC-Lines of code

	Plan Size	Actual Size	To Date
Base (B)	75.00	75.00	
Deleted (D)	0.00	1.00	
Modified (M)	0.00	1.00	
Added (A)	32.36	52.00	
Reused (R)	10.00	4.00	4.00
Added & Modified (A&M)	32.36	53.00	53.00
Total (T)	117.36	130	130.00
New Reusable (NR)	0.00	0.00	0.00
Estimated A&M (E)	32.36		

Time in Phase

Registro: 1 de 1

Filtrado

Buscar

Time in Phase

Phase	Plan	Actual	To-Date	To-Date%
PLAN	16	15	0	0.0%
DLD	30	29	0	0.0%
DLDR	2	0	0	0.0%
CODE	30	42	0	0.0%
CR	3	0	0	0.0%
COMPILE	1	1	0	0.0%
UT	3	4	0	0.0%
PM	40	33	0	0.0%
Total	125	124	0	

Defects Injected in Phase

Phase	Plan	Actual	To-Date	To-Date%
PLAN	0.0	0	0	0.0%
DLD	0.0	0	0	0.0%
DLDR	0.0	0	0	0.0%
CODE	3.0	1	0	0.0%
CR	0.0	0	0	0.0%
COMPILE	0.0	0	0	0.0%
UT	0.0	0	0	0.0%
PM	0.0	0	0	0.0%
Total	3	1	0	



## REPORTE DE PRÁCTICAS

Defects Removed in Phase				
Phase	Plan	Actual	To-Date	To-Date%
PLAN	0.0	0	0	0.0%
DLD	0.0	0	0	0.0%
DLDR	0.0	0	0	0.0%
CODE	2.0	0	0	0.0%
CR	0.0	0	0	0.0%
COMPILE	0.0	0	0	0.0%
UT	1.0	1	0	0.0%
PM	0.0	0	0	0.0%
Total	3	1	0	

Defect Removal Efficiency			
	Plan	Actual	To-Date
Defects/Hr. DLDR	0.0	0.0	0.0
Defects/Hr. CR	0.0	0.0	0.0
Defects/Hr. Compile	0.0	0.0	0.0
Defects/Hr. UT	20.0	14.3	14.3
DRL (DLDR/UT)	0.0	0.0	0.0
DRL (CR/UT)	0.0	0.0	0.0
DRL (Compile/UT)	0.0	0.0	0.0

Registro: 14 1 de 1 Filtrado Buscar

**PSP Test Report**

Personal Software Process

**PSP Test Report**

Student	Gonzalo Martinez Silverio	Start Date	17-nov.-23
Program	Program 5	End Date	
Instructor	TANIA TURRUBIATES LOPEZ	Language	Java

Test Number	1
Test Name	Prueba uno
Objective	Realizar la integración numérica usando la regla de Simpson para la distribución t, calculando el área bajo la curva de la distribución t con límites y grados de libertad especificados.
Description	El programa utiliza métodos para la integración numérica (integrarDistribucionT), el cálculo de valores de la distribución t en puntos específicos (distribucionT), y la suma de estos valores para la regla de Simpson (sumarDistribucionT). En el método principal, se solicita al usuario que ingrese el límite superior de integración (x) y los grados de libertad (gradosLibertad). Luego, realiza la integración para múltiples conjuntos de entradas y compara los resultados con valores esperados predefinidos (valoresEsperados).
Conditions	Se utiliza la biblioteca Apache Commons Math (TDistribution) para los cálculos de la distribución t. La entrada del usuario se realiza mediante la clase Scanner para x y gradosLibertad. La integración se realiza con la regla de Simpson utilizando un número específico de segmentos (numSegmentos).
Expected Results	El programa espera que el usuario ingrese valores para x y grados de libertad (gradosLibertad). Luego, realiza la integración numérica de la distribución t y compara los resultados con valores esperados predefinidos (valoresEsperados). Estos valores esperados están codificados para tres conjuntos de entradas.
Actual Results	El programa calcula la integración numérica para cada conjunto de valores de entrada y muestra los resultados reales. Luego, compara estos resultados con los valores esperados y muestra la salida para cada conjunto de entrada.



# INSTITUTO TECNOLÓGICO SUPERIOR DE ALAMO TEMAPACHE

## REPORTE DE PRÁCTICAS

PSP Task and Schedule Plan

Carnegie Mellon Software Engineering Institute

Personal Software Process

Student: Gonzalo Martinez Silverio Start Date: 28-mar.-05

Instructor: TANIA TURRUBIATES LOPEZ Default Hours/Week: 10

Task & Schedule Plan: My Task and Schedule Plan Planned Weeks: 0

ID: 1 Total Plan Hours: 0

Task and Schedule Plan Status

Task Plan Schedule Plan Cumulative Earned Value Cumulative Hours

Task ID	Order	Project	Phase	Task Name	Plan Hours	Plan Date	PV	Actual Hours	Actual Date	EV
637	1	Program 1	PLAN	Program 1 PLAN	0.00		0.00	0.2		0.00
638	2	Program 1	DLD	Program 1 DLD	0.00		0.00	1.0		0.00
639	3	Program 1	CODE	Program 1 CODE	0.00		0.00	0.8		0.00
640	4	Program 1	COMPILE	Program 1 COMPILE	0.00		0.00	0.0		0.00
641	5	Program 1	UT	Program 1 UT	0.00		0.00	0.4		0.00
642	6	Program 1	PM	Program 1 PM	0.00		0.00	1.6		0.00
733	7	Program 2	PLAN	Program 2 PLAN	0.22		0.00	0.6		0.00
734	8	Program 2	DLD	Program 2 DLD	1.01		0.00	2.3		0.00
735	9	Program 2	CODE	Program 2 CODE	0.86		0.00	1.3		0.00
736	10	Program 2	COMPILE	Program 2 COMPILE	0.01		0.00	0.0		0.00
737	11	Program 2	UT	Program 2 UT	0.42		0.00	0.5		0.00
738	12	Program 2	PM	Program 2 PM	1.65		0.00	1.7		0.00
739	13	Program 3	PLAN	Program 3 PLAN	0.21		0.00	0.4		0.00
740	14	Program 3	DLD	Program 3 DLD	0.90		0.00	0.8		0.00
741	15	Program 3	CODE	Program 3 CODE	0.57		0.00	1.0		0.00
742	16	Program 3	COMPILE	Program 3 COMPILE	0.00		0.00	0.0		0.00
743	17	Program 3	UT	Program 3 UT	0.24		0.00	0.2		0.00
744	18	Program 3	PM	Program 3 PM	0.90		0.00	0.9		0.00
745	19	Program 4	PLAN	Program 4 PLAN	0.33		0.00	0.3		0.00
746	20	Program 4	DLD	Program 4 DLD	1.20		0.00	0.5		0.00
747	21	Program 4	CODE	Program 4 CODE	0.90		0.00	0.6		0.00
748	22	Program 4	COMPILE	Program 4 COMPILE	0.01		0.00	0.0		0.00
749	23	Program 4	UT	Program 4 UT	0.31		0.00	0.2		0.00
750	24	Program 4	PM	Program 4 PM	1.24		0.00	0.5		0.00
751	25	Program 5	PLAN	Program 5 PLAN	0.00		0.00	0.3		0.00
752	26	Program 5	DLD	Program 5 DLD	0.00		0.00	0.5		0.00
753	27	Program 5	DLD	Program 5 DLD	0.00		0.00	0.0		0.00
754	28	Program 5	CODE	Program 5 CODE	0.00		0.00	0.7		0.00
755	29	Program 5	CR	Program 5 CR	0.00		0.00	0.0		0.00
756	30	Program 5	COMPILE	Program 5 COMPILE	0.00		0.00	0.0		0.00
757	31	Program 5	UT	Program 5 UT	0.00		0.00	0.1		0.00
758	32	Program 5	PM	Program 5 PM	0.00		0.00	0.6		0.00
*	avo)									

Total Task Plan Hours: 11.0

Registro: 1 de 1 Sin filtro Buscar

Concluimos el proyecto:

Main

Carnegie Mellon Software Engineering Institute

Personal Software Process

PSP Student Workbook

Project Directory Process Directory Size Measures Analysis Tools Scripts/Forms Course Materials

Projects

New Project Open Project Replan Project Delete Project Import Export

ID	Name	Started	Completed	
408	Program 1	07/09/2023 07:26:42 p. r	12/09/2023 11:56:59 p. r	✓
422	Program 2	24/09/2023 06:44:14 a. r	26/09/2023 07:17:40 p. r	✓
423	Program 3	14/10/2023 06:46:44 p. r	15/10/2023 10:25:06 a. r	✓
424	Program 4	03/11/2023 06:59:13 a. r	03/11/2023 07:10:58 p. r	✓
425	Program 5	17/11/2023 02:29:52 p. r	24/11/2023 08:13:47 p. r	✓



### **CONCLUSIONES:**

En conclusión, PSP2, se rige como un enfoque valioso y estructurado para mejorar la calidad y la productividad en el desarrollo de software a nivel individual. Al introducir revisiones sistemáticas en las fases críticas de diseño y código, así como al facilitar la planificación detallada de la calidad, PSP2 proporciona a los ingenieros de software las herramientas necesarias para identificar y corregir posibles problemas de manera temprana.

La automatización de cálculos y la introducción de medidas de calidad derivadas permiten una gestión más eficiente del proceso, brindando a los profesionales información valiosa sobre su rendimiento y permitiéndoles ajustar y mejorar continuamente sus prácticas de desarrollo.

La conclusión clave es que, al adoptar PSP2, los ingenieros de software pueden no solo elevar la calidad de su trabajo, sino también planificar con mayor precisión y aumentar la productividad sin comprometer la calidad del producto final. Con un enfoque en la mejora continua, PSP2 se presenta como un marco sólido para el desarrollo personalizado y la gestión efectiva de proyectos de software a nivel individual.

### **OBSERVACIONES PROPIAS PARA MEJORAR:**

1. Mejorar la concentración.
2. Observar más al momento de codificar.