



Reporte Técnico de Actividades Práctico-Experimentales Nro. 00X

1. Datos de Identificación del Estudiante y la Práctica

Nombre del estudiante(s)	Yimmy Onner Angulo Torres
Asignatura	Teoría de la programación
Ciclo	1 A
Unidad	1
Resultado de aprendizaje de la unidad	Identifica los conceptos fundamentales de la teoría de la programación, bajo los principios de solidaridad, transparencia, responsabilidad y honestidad.
Práctica Nro.	002
Tipo	Individual o Grupal
Título de la Práctica	Del diseño del algoritmo, con estructuras secuenciales, a la construcción del problema.
Nombre del Docente	Lisette Geoconda López Faicán
Fecha	28/10/2025
Horario	10h30 – 13h30
Lugar	Aula de la facultad de la Energía, las Industrias y los Recursos Naturales no Renovables
Tiempo planificado en el Sílabo	3 horas

2. Objetivo(s) de la Práctica:

- Desarrollar la capacidad de transformar un problema en una solución computacional.
- Aplicar estructuras secuenciales en el diseño del algoritmo.
- Validar la lógica del algoritmo mediante pruebas de escritorio.
- Implementar y ejecutar la solución en un lenguaje de programación.



3. Materiales y reactivos:

- Herramienta de pseudocódigo y diagramación de algoritmos: PSeInt.
- IDE de programación: Visual Studio Code u otro entorno compatible.
- Lenguaje de programación: C (según los contenidos de la unidad).

4. Equipos y herramientas

- Computador personal con sistema operativo Windows, Linux o macOS.
- Material de apoyo en el Entorno Virtual de Aprendizaje (EVA).
- Editores de texto (Word, Google Docs u otros) para la elaboración del informe técnico en formato PDF.

5. Procedimiento / Metodología

Para la resolución de este problema primero se hizo un análisis general de la solución más eficaz del problema:

A. Análisis del Problema:

Enunciado Del Ejercicio:

Un estudiante necesita saber qué calificación debe obtener en el tercer certamen (C3) para aprobar la asignatura con una nota final de 60/100 puntos.

Para calcular la nota final se utilizan las siguientes fórmulas:

1. Promedio de certámenes (NC)
$$NC = \frac{C1 + C2 + C3}{3}$$

2. Nota final del ciclo (NF):
$$NF = (NC \cdot 0.7) + (NL \cdot 0.3)$$

Donde:

- C1 y C2 son las notas de los dos primeros certámenes.
- C3 es la nota del tercer certamen (la que se debe calcular).
- NL es la nota de laboratorio.
- NF es la nota final de la asignatura.

El programa debe permitir ingresar las notas de C1, C2 y NL; calcular automáticamente la nota mínima necesaria en C3 para que el estudiante apruebe la asignatura.

Nota: Si el resultado es negativo, significa que ya aprueba con las notas actuales.

- Este enunciado nos pide calcular la tercera nota del estudiante, la misma que no esta dada desde el principio, en su lugar necesitamos aplicar los conocimientos previamente obtenidos como lo serian, el despeje de fórmulas, donde necesitamos despejar "NC" de la segunda formula, después se despejó C3 que de la primera formula para calcular la tercera nota.

- **Datos de Entrada:** not1, not2, nl;

Además, se definen 2 constantes: la nf=60; y la cantidad_notas=3;

- **Proceso:** Se despeja NC de la formula de NF quedando de esta manera:

$$nc <- (nf - nl * 0.3) / 0.7.$$

Después se despeja C3 de la otra formula que en mi caso utilice not3:

$$not3 <- (CANTIDAD_NOTAS * nc - not1 - not2);$$

- **Salida:** El valor de la not3.

B. Diseño del algoritmo:

Se diseño el algoritmo en pseint, con una estructura secuencial.

C. Elaboración del diagrama de flujo:

El diagrama de flujo fue generado dentro del mismo pseint.

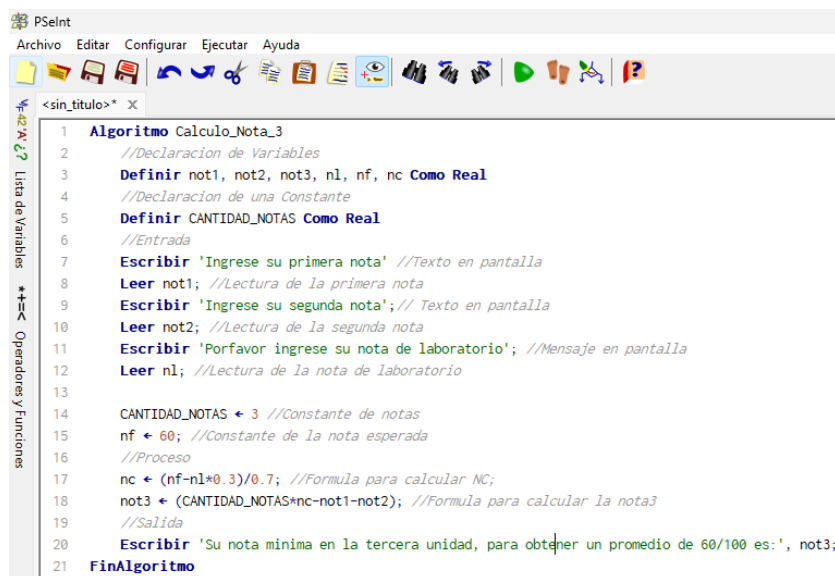
D. Trasladar el código a lenguaje C.

Se utilizó Visual Studio Code para la codificación del programa.

6. Resultados

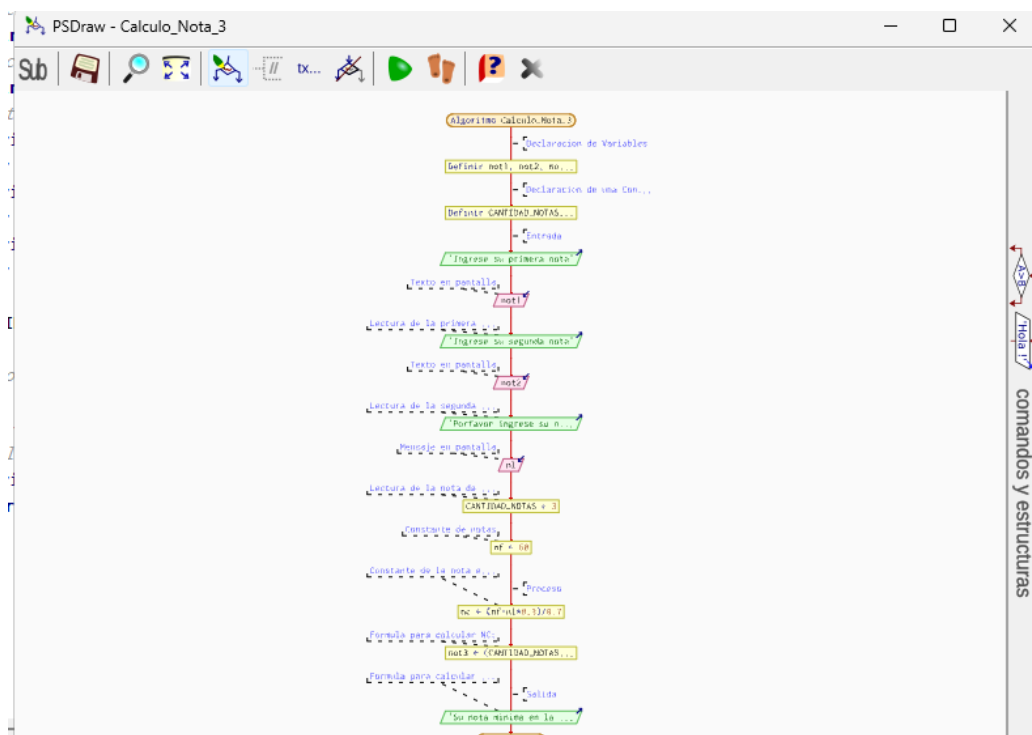
Incluya evidencias del trabajo realizado (tablas, gráficos, capturas de pantalla, registros de ejecución, modelos, programas, informes, etc.).

Pseint:

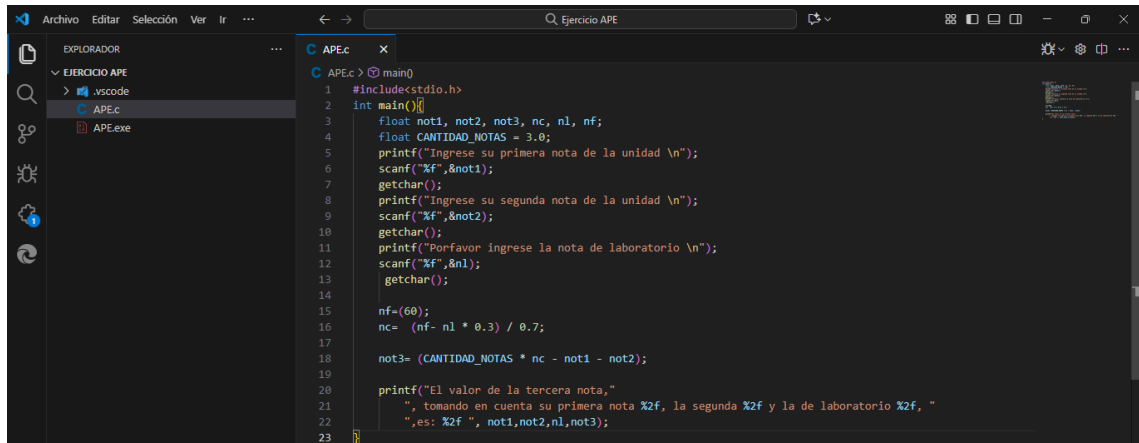


```
1 Algoritmo Calculo_Nota_3
2 //Declaracion de Variables
3 Definir not1, not2, not3, n1, nf, nc Como Real
4 //Declaracion de una Constante
5 Definir CANTIDAD_NOTAS Como Real
6 //Entrada
7 Escribir 'Ingrese su primera nota' //Texto en pantalla
8 Leer not1; //Lectura de la primera nota
9 Escribir 'Ingrese su segunda nota'; // Texto en pantalla
10 Leer not2; //Lectura de la segunda nota
11 Escribir 'Porfavor ingrese su nota de laboratorio'; //Mensaje en pantalla
12 Leer n1; //Lectura de la nota de laboratorio
13
14 CANTIDAD_NOTAS ← 3 //Constante de notas
15 nf ← 60; //Constante de la nota esperada
16 //Proceso
17 nc ← (nf-n1*0.3)/0.7; //Formula para calcular NC;
18 not3 ← (CANTIDAD_NOTAS*nc-not1-not2); //Formula para calcular la nota3
19 //Salida
20 Escribir 'Su nota minima en la tercera unidad, para obtener un promedio de 60/100 es:', not3;
21 FinAlgoritmo
```

Diagrama de Flujo:



Lenguaje C en Visual Studio Code:



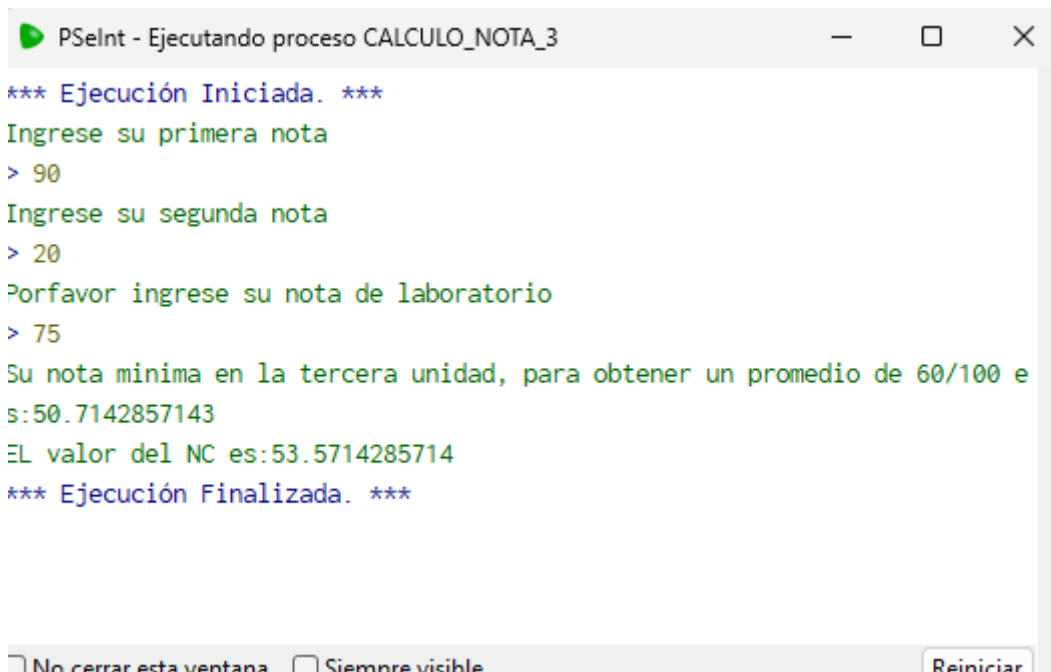
```

1 #include <stdio.h>
2 int main()
3 {
4     float not1, not2, not3, nc, n1, nf;
5     float CANTIDAD_NOTAS = 3.0;
6     printf("Ingrese su primera nota de la unidad \n");
7     scanf("%f", &not1);
8     getchar();
9     printf("Ingrese su segunda nota de la unidad \n");
10    scanf("%f", &not2);
11    getchar();
12    printf("Porfavor ingrese la nota de laboratorio \n");
13    scanf("%f", &n1);
14    getchar();
15    nf=(60);
16    nc= (nf- n1 * 0.3) / 0.7;
17
18    not3= (CANTIDAD_NOTAS * nc - not1 - not2);
19
20    printf("El valor de la tercera nota,"
21           ", tomando en cuenta su primera nota %2f, la segunda %2f y la de laboratorio %2f, "
22           ",es: %2f ", not1,not2,n1,not3);
23 }
  
```

Pruebas de escritorio en Tabla:

Not1	Not2	n1	nc<-(nf-n1*0.3)/0.7.	not3<- (CANTIDAD_NOTAS* nc-not1-not2);	Not3
80	80	90	47.14	-18.57	-18.57
60	40	50	68.28	92.85	02.85
90	20	75	53.57	50.71	50.71

Resultados:



```

PSelnt - Ejecutando proceso CALCULO_NOTA_3

*** Ejecución Iniciada. ***
Ingrese su primera nota
> 90
Ingrese su segunda nota
> 20
Porfavor ingrese su nota de laboratorio
> 75
Su nota minima en la tercera unidad, para obtener un promedio de 60/100 e
s:50.7142857143
EL valor del NC es:53.5714285714
*** Ejecución Finalizada. ***
  
```

```
PSelnt - Ejecutando proceso CALCULO_NOTA_3

*** Ejecución Iniciada. ***
Ingrese su primera nota
> 80
Ingrese su segunda nota
> 80
Porfavor ingrese su nota de laboratorio
> 90
Su nota minima en la tercera unidad, para obtener un promedio de 60/100 e
s:-18.5714285714
EL valor del NC es:47.1428571429
*** Ejecución Finalizada. ***
```

```
PSelnt - Ejecutando proceso CALCULO_NOTA_3

*** Ejecución Iniciada. ***
Ingrese su primera nota
> 60
Ingrese su segunda nota
> 40
Porfavor ingrese su nota de laboratorio
> 50
Su nota minima en la tercera unidad, para obtener un promedio de 60/100 e
s:92.8571428571
EL valor del NC es:64.2857142857
*** Ejecución Finalizada. ***
```

Terminal del Lenguaje C

```
PS C:\Users\HP\Desktop\Lenguaje C\Ejercicio APE> .\APE.exe
Ingrese su primera nota de la unidad
60
Ingrese su segunda nota de la unidad
40
Porfavor ingrese la nota de laboratorio
50
El valor de la tercera nota, tomando en cuenta su primera nota 60.000000, la segunda 40.000000 y la de laboratorio 50.000000, es: 92.857140
PS C:\Users\HP\Desktop\Lenguaje C\Ejercicio APE>
```

```
PROBLEMAS  SALIDA  CONSOLA DE DEPURACIÓN  TERMINAL  PUERTOS

● PS C:\Users\HP\Desktop\Lenguaje C\Ejercicio APE> .\APE.exe
Ingrese su primera nota de la unidad
80
Ingrese su segunda nota de la unidad
80
Porfavor ingrese la nota de laboratorio
90
El valor de la tercera nota, tomando en cuenta su primera nota 80.000000, la segunda 80.000000 y la de laboratorio 90.00000
0, es: -18.571430
❖ PS C:\Users\HP\Desktop\Lenguaje C\Ejercicio APE>

● PS C:\Users\HP\Desktop\Lenguaje C\Ejercicio APE> .\APE.exe
Ingrese su primera nota de la unidad
90
Ingrese su segunda nota de la unidad
20
Porfavor ingrese la nota de laboratorio
75
El valor de la tercera nota, tomando en cuenta su primera nota 90.000000, la segunda 20.000000 y la de laboratorio 75.00000
0, es: 50.714291
❖ PS C:\Users\HP\Desktop\Lenguaje C\Ejercicio APE>
```

7. Preguntas de Control

Responda a las preguntas del docente.

- **¿Qué elementos deben identificarse en el análisis de un problema computacional?**

Al analizar un problema computacional, es necesario reconocer los datos que se reciben, los procedimientos que se aplicarán, los resultados esperados y las condiciones bajo las cuales debe operar el sistema. En resumen, se debe entender qué información entra, cómo se procesa y qué salida se desea obtener para garantizar una solución lógica y eficiente.

- **¿Por qué es importante validar un algoritmo mediante pruebas de escritorio? FEIRNNR - Carrera de Computación**

La validación de un algoritmo con pruebas de escritorio es importante porque permite comprobar su funcionamiento paso a paso, asegurando que la lógica sea correcta antes de convertirlo en código.

Mediante esta simulación manual, se previenen errores, se mejora la comprensión del proceso y se optimiza el tiempo de desarrollo, garantizando que el algoritmo cumpla su propósito de manera precisa y eficiente

- **¿Cómo se traslada un algoritmo en pseudocódigo a un lenguaje de programación?**

Trasladar un algoritmo desde el pseudocódigo a un lenguaje de programación significa transformar la lógica escrita de forma general en instrucciones formales que la computadora pueda ejecutar. Este proceso requiere interpretar cada paso del algoritmo, usar la sintaxis del lenguaje elegido y realizar pruebas para asegurar su correcto funcionamiento, manteniendo siempre la misma lógica del diseño original.

8. Conclusiones

- **Importancia de las pruebas de escritorio:**

Las pruebas de escritorio son una herramienta esencial para validar la lógica de un algoritmo antes de llevarlo a un entorno de programación. Este proceso permite detectar errores de razonamiento, condiciones mal planteadas o cálculos incorrectos sin necesidad de ejecutar código.

Además, ayuda al programador a comprender mejor el flujo del algoritmo y anticipar posibles fallos.

De esta forma, se garantiza que el diseño previo sea sólido, coherente y eficiente, reduciendo el tiempo de depuración en etapas posteriores.

- **Traslado del pseudocódigo a un lenguaje de programación:**

Convertir un algoritmo escrito en pseudocódigo a un lenguaje de programación implica transformar las instrucciones lógicas en una sintaxis formal que la computadora pueda interpretar.

Este paso requiere comprender la estructura del pseudocódigo, seleccionar el lenguaje adecuado y aplicar correctamente sus reglas gramaticales.

Un buen traslado mantiene la esencia del algoritmo, asegurando que su lógica no se altere y que los resultados sean los esperados.

En conjunto, este proceso fortalece la calidad del desarrollo y permite construir programas funcionales y bien estructurados.

9. Recomendaciones

1. Es recomendable que antes de iniciar la práctica, el estudiante **analice detalladamente el problema** y defina claramente las entradas, procesos y salidas, con el fin de facilitar la elaboración del algoritmo.
2. Durante el diseño, se sugiere **utilizar PSeInt para simular y validar la lógica paso a paso**, realizando pruebas con distintos

valores para fortalecer la comprensión de las estructuras secuenciales.

3. En la implementación del código, se aconseja **comentar cada sección del programa** y mantener una estructura limpia y ordenada para facilitar la lectura y corrección de errores.
4. Para mejorar la práctica en contextos reales, los alumnos pueden **adaptar los algoritmos a problemas cotidianos o empresariales**, como cálculos de nómina, gestión de inventarios o control de notas académicas.
5. Se recomienda también **comparar los resultados obtenidos en PSeInt con la ejecución real en el lenguaje C**, de manera que se identifiquen diferencias entre la lógica conceptual y la implementación técnica.
6. Finalmente, se debe fomentar el **trabajo colaborativo y la retroalimentación entre compañeros**, ya que revisar y analizar distintos enfoques para resolver un mismo problema enriquece el aprendizaje y mejora la calidad de las soluciones computacionales.

10. Anexos

Se utilizó la herramienta de Visual Studio Code.

