



Reporte Técnico de Actividades Práctico-Experimentales Nro. 002

1. Datos de Identificación del Estudiante y la Práctica

| | |
|--|--|
| Nombre del estudiante(s) | Pilar Valentina Naranjo Quizhpe. |
| 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 programa. |
| Nombre del Docente | Lisette Geoconda López Faicán |
| Fecha | Jueves 23 de octubre del 2025 Jueves 30 de octubre del 2025 |
| Horario | 10h30 – 13h30 |
| Lugar | Aula física asignada al paralelo. |
| Tiempo planificado en el Sílabo | 6 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, Reactivos, Equipos y Herramientas

Liste los materiales, reactivos, equipos y herramientas utilizados en la práctica, confirmando los de la guía o agregando los adicionales.

4. Procedimiento / Metodología Ejecutada

- 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).

5. Resultados

- **Análisis del problema:**

Entradas:

- $c1$: Nota del primer certamen.
- $c2$: Nota del segundo certamen.
- nl : Nota de laboratorio.

Proceso:

1. Calcular el promedio de certámenes: $nc = (c1 + c2 + c3) / 3$.
2. Calcular la nota final: $nf = (nc * 0.7) + (nl * 0.3)$.
3. Despejar la nota necesaria en el tercer certamen $c3$ para lograr $nf = 6$.

Salidas:

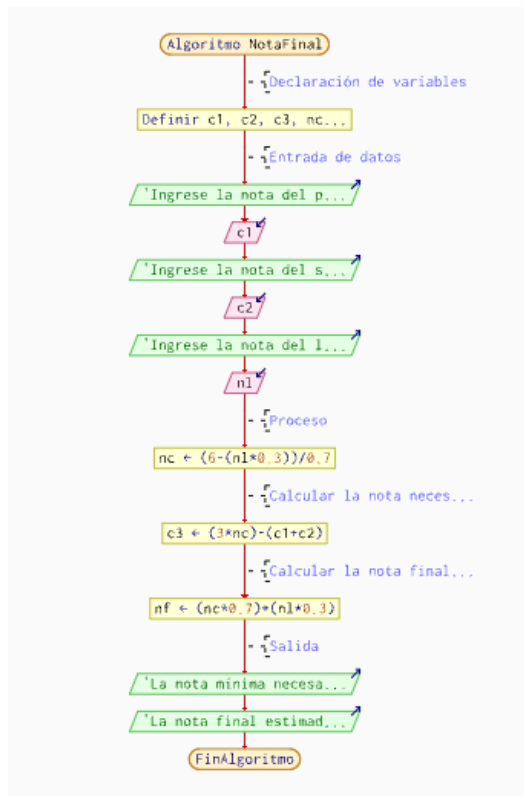
- $c3$: Nota mínima que necesita para aprobar.
- nf : Nota final calculada.

- **Diseño del algoritmo:**

Pseudocódigo en PSeInt

```
1  Algoritmo NotaFinal
2  // Declaración de variables
3  Definir c1, c2, c3, nc, nf, nl Como Real
4
5  // Entrada de datos
6  Escribir "Ingrese la nota del primer certamen (C1): "
7  Leer c1
8  Escribir "Ingrese la nota del segundo certamen (C2): "
9  Leer c2
10 Escribir "Ingrese la nota del laboratorio: "
11 Leer nl
12
13 // Proceso
14  $nc \leftarrow (6 - (nl * 0.3)) / 0.7$ 
15
16 // Calcular la nota necesaria en el tercer certamen (C3)
17  $c3 \leftarrow (3 * nc) - (c1 + c2)$ 
18
19 // Calcular la nota final estimada
20  $nf \leftarrow (nc * 0.7) + (nl * 0.3)$ 
21
22 //Salida
23 Escribir "La nota mínima necesaria en el tercer certamen es: ", c3
24 Escribir "La nota final estimada será: ", nf
25 FinAlgoritmo
```

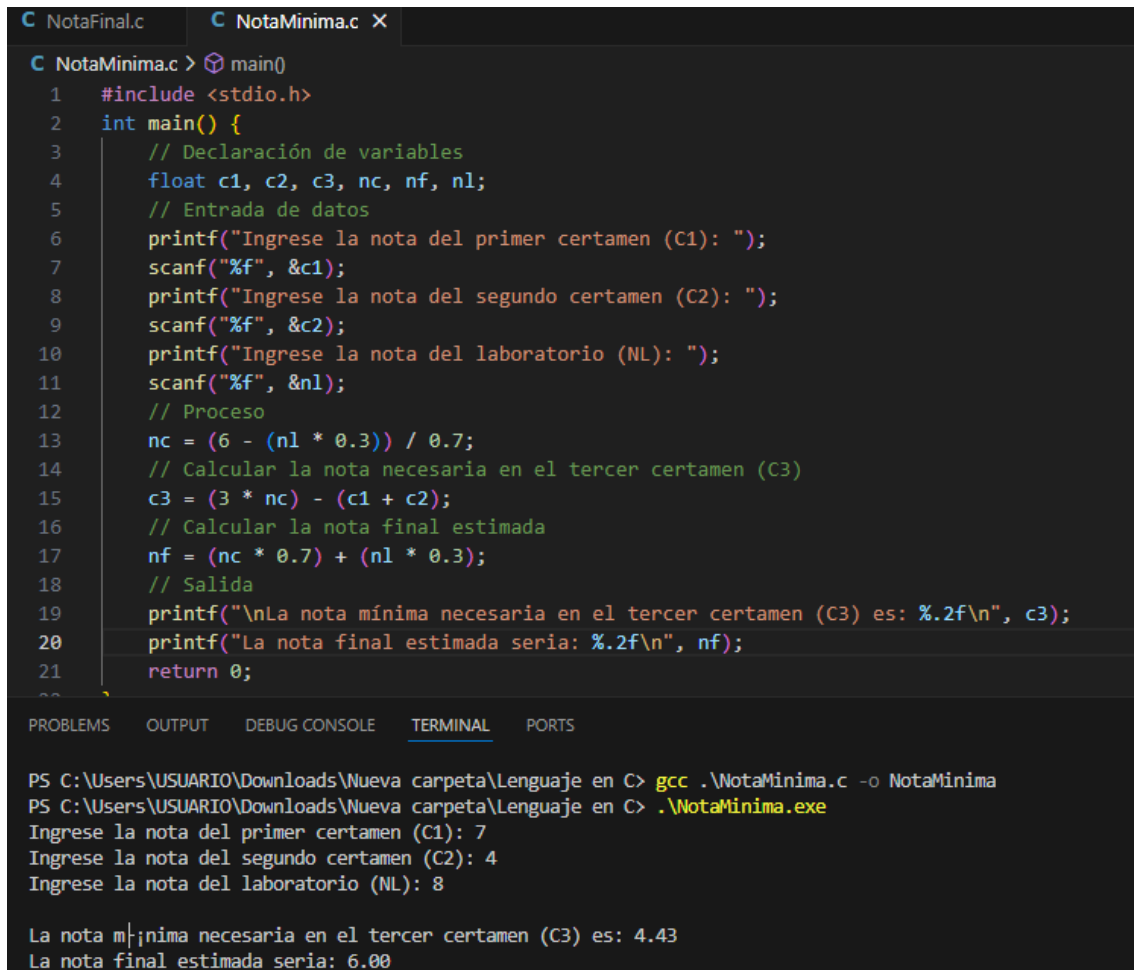
Diagrama de flujo



Pruebas de escritorio

| Datos de Entrada | Proceso | Salida |
|----------------------------|--|--------------------------------------|
| C1 = 5 C2 = 6 NL = 7 | $NC = (6 - (7 \times 0.3)) / 0.7 = 5.57$ $C3 = (3 \times 5.57) - (5 + 6) = 5.71$ $NF = (5.57 \times 0.7) + (7 \times 0.3) = 6.00$ | C3 = 5.71 NF = 6.00 |
| C1 = 4 C2 = 5 NL = 6 | $NC = (6 - (6 \times 0.3)) / 0.7 = 5.57$ $C3 = (3 \times 5.57) - (4 + 5) = 9$ $NF = (5.57 \times 0.7) + (6 \times 0.3) = 5.71$ | C3 = 9 NF = 6.00 |
| C1 = 7 C2 = 8 NL = 9 | $NC = (6 - (9 \times 0.3)) / 0.7 = 4.71$ $C3 = (3 \times 4.71) - (7 + 8) = -0.87$ $NF = (4.71 \times 0.7) + (9 \times 0.3) = 6.06$ | C3 = 0 NF = 6.06 |

- **Codificación: trasladar la solución a un lenguaje de programación C.**



```
C NotaFinal.c C NotaMinima.c X
C NotaMinima.c > main()
1  #include <stdio.h>
2  int main() {
3      // Declaración de variables
4      float c1, c2, c3, nc, nf, nl;
5      // Entrada de datos
6      printf("Ingrese la nota del primer certamen (C1): ");
7      scanf("%f", &c1);
8      printf("Ingrese la nota del segundo certamen (C2): ");
9      scanf("%f", &c2);
10     printf("Ingrese la nota del laboratorio (NL): ");
11     scanf("%f", &nl);
12     // Proceso
13     nc = (6 - (nl * 0.3)) / 0.7;
14     // Calcular la nota necesaria en el tercer certamen (C3)
15     c3 = (3 * nc) - (c1 + c2);
16     // Calcular la nota final estimada
17     nf = (nc * 0.7) + (nl * 0.3);
18     // Salida
19     printf("\nLa nota mínima necesaria en el tercer certamen (C3) es: %.2f\n", c3);
20     printf("La nota final estimada seria: %.2f\n", nf);
21     return 0;
22 }
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\USUARIO\Downloads\Nueva carpeta\Lenguaje en C> gcc .\NotaMinima.c -o NotaMinima
PS C:\Users\USUARIO\Downloads\Nueva carpeta\Lenguaje en C> .\NotaMinima.exe
Ingrese la nota del primer certamen (C1): 7
Ingrese la nota del segundo certamen (C2): 4
Ingrese la nota del laboratorio (NL): 8

La nota mínima necesaria en el tercer certamen (C3) es: 4.43
La nota final estimada seria: 6.00
```

6. Preguntas de Control

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

Los datos de entradas que es la información que el programa necesita, el proceso que son los pasos lógicos para transformar esa información y las salidas que es el resultado o solución esperada. También se deben definir restricciones, supuestos y casos límite.

- ¿Por qué es importante validar un algoritmo mediante pruebas de escritorio?

Porque permite detectar errores lógicos antes de codificar, verificando que la secuencia de pasos produzca los resultados esperados.

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

Convirtiendo cada instrucción del pseudocódigo a su equivalente en el lenguaje escogido, respetando la sintaxis, tipos de datos y operadores.



7. Conclusiones

A lo largo de la actividad, se logró entender lo crucial que es crear algoritmos bien estructurados y racionales antes de comenzar a programar. La tarea me brindó la oportunidad de afianzar el examen de datos de entrada, mecanismos y resultados, además de la verificación mediante ejemplos de prueba.

8. Recomendaciones

Siempre da prioridad a identificar adecuadamente las entradas, los procesos y los resultados del problema, ya que este es el paso más esencial para garantizar que la solución sea factible. Después de establecer la lógica en un algoritmo básico (secuencial), es imprescindible realizar pruebas de escritorio. Estas te ayudarán a economizar tiempo al comprobar la lógica antes de codificar en C, asegurando una conversión directa y sin errores de concepto.