Reporte Técnico de Actividades Práctico-Experimentales Nro. 00X

1. Datos de Identificación del Estudiante y la Práctica

Nombre del estudiante(s)	Emerson Sebastian Chamba Galarza
Asignatura	Teoría de la programación
Ciclo	Primer Ciclo
Unidad	2
Resultado de aprendizaje de la unidad	Aplica las estructuras de programación en la resolución de problemas básicos, bajo los principios de solidaridad, transparencia, responsabilidad y honestidad
Práctica Nro.	001
Tipo	Individual o Grupal
Título de la Práctica	Aplicación de estructuras condicionales en la resolución de problemas.
Nombre del Docente	Lissette Geoconda López Faicán
Fecha	Jueves 20 de noviembre del 2025
Horario	10h30 – 13-h30
Lugar	Aula física asignada al paralelo
Tiempo planificado en el Sílabo	6 horas

2. Objetivo(s) de la Práctica

- Comprender y aplicar las estructuras condicionales simples, dobles y múltiples en la resolución de problemas.
- Diseñar y codificar un algoritmo que utilice sentencias de decisión para analizar y clasificar información.
- Validar el funcionamiento del programa mediante la ejecución práctica

3. Materiales, Reactivos, Equipos y Herramientas

- Herramientas de modelado de diagrama de flujo (Psient, Draw.io, Lucidchart, otros)
- IDE de programación: Visual Studio Code u otro entorno compatible.
- Lenguaje de programación: C (según los contenidos de la unidad).

4. Procedimiento / Metodología Ejecutada

La práctica se centró en aplicar las estructuras de programación condicionales para clasificar el nivel de desempeño de la nota final de la Unidad 1, cumpliendo el objetivo de comprender y aplicar dichas estructuras en la resolución de problemas2. La metodología seguida fue el aprendizaje basado en problemas 3, que comenzó con la contextualización del ejercicio del cálculo de la nota final 4y las condiciones específicas de clasificación:

- Nota ≥ 9 → "Excelente"
- Nota ≥ 7 y < 9 → "Bueno"
- Nota ≥ 5 y < 7 → "Regular"
- Nota < 5 → "Deficiente"

Luego, en la fase de desarrollo, se analizó el problema (entradas, proceso, salidas) 6, se diseñó el algoritmo para el cálculo y la clasificación de la nota 7, y se realizó la codificación en lenguaje C, implementando una estructura condicional múltiple (if-else if-else) para clasificar el nivel de desempeño8. Finalmente, se realizaron las pruebas de escritorio y la ejecución para validar los resultados.

5. Resultados

Datos de Entrada:

- Notas del ACD, APE Y AA
- Nota del ES1 Y ES2

Proceso:

- promedio_acd = (acd1 + acd2)/2 * 0.2;
- promedio_ape = (ape1 + ape2)/2 * 0.25;
- promedio_aa = (aa1 + aa2)/2 * 0.2;

- nota_es_sin_pond =((es1 * 0.4) + (es2 * 0.6)) * 0.35;

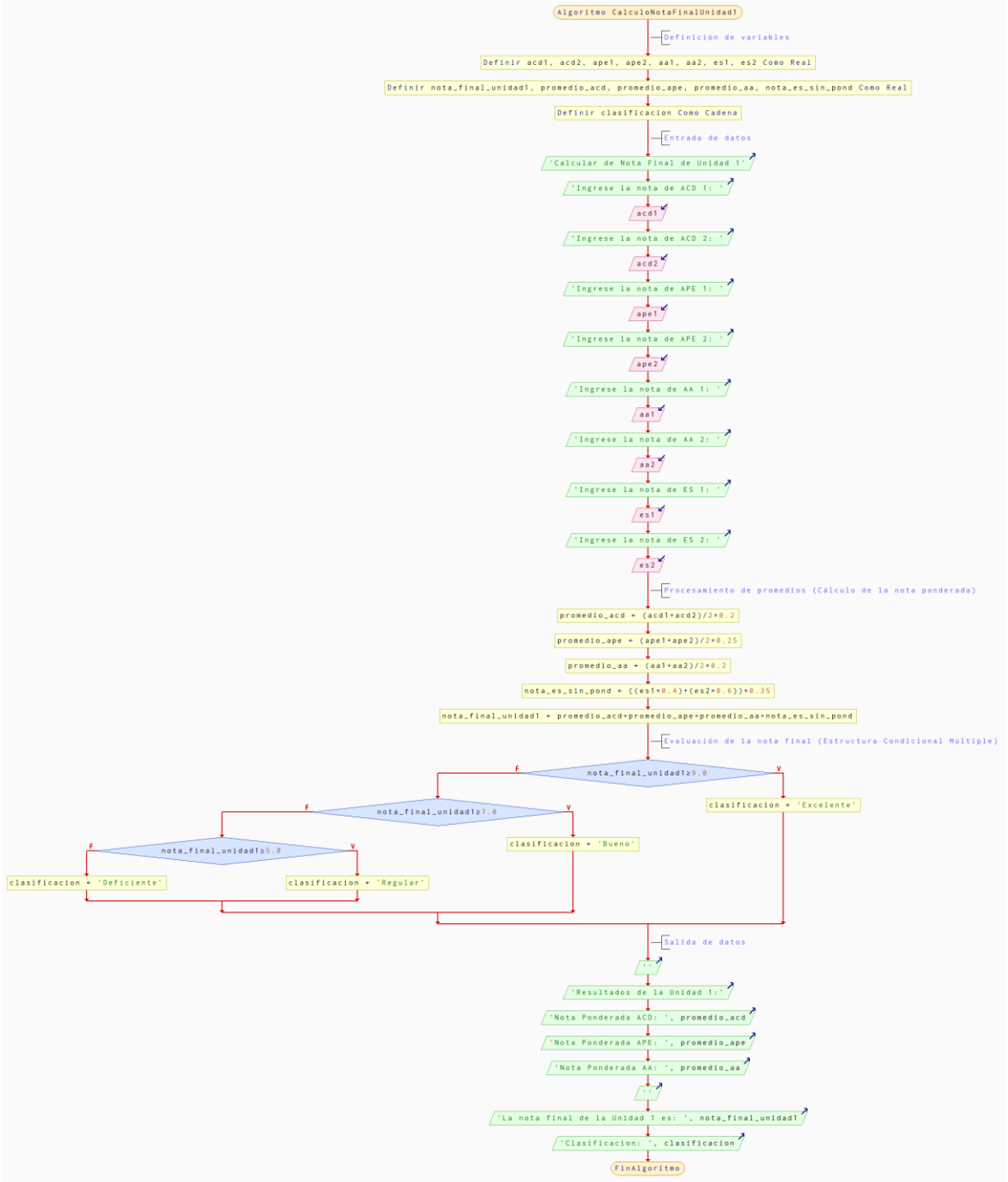
- nota_final_unidad1 = promedio_acd + promedio_ape + promedio_aa + nota_es_sin_pond;

- if (nota_final_unidad1 >= 9.0) {
  clasificacion = "Excelente";
- } else if (nota_final_unidad1 >= 7.0 && nota_final_unidad1 < 9.0)
  clasificacion = "Bueno";
- } else if (nota_final_unidad1 >= 5.0 && nota_final_unidad1 < 7.0) {
  clasificacion = "Regular";
- } else {
  clasificacion = "Deficiente";
  }

Salida:

- Nota ponderada de ACD
- Nota ponderada de APE
- Nota ponderada de AA
- Nota final de Unidad 1
- Calificación de la nota final

Diagrama de Flujo



Codificación en C

```c
#include <stdio.h>

int main() {
    // Datos de entrada

    float acd1, acd2, ape1, ape2, aa1, aa2, es1, es2;

    float nota_final_unidad1;

    float promedio_acd, promedio_ape, promedio_aa;

    float nota_es_sin_pond;
```

```
    char *clasificacion;

    printf("Calcular de Nota Final de Unidad 1\n");

    printf("Ingrese la nota de ACD 1: ");
    scanf("%f", &acd1);
    printf("Ingrese la nota de ACD 2: ");
    scanf("%f", &acd2);


    printf("Ingrese la nota de APE 1: ");
    scanf("%f", &ape1);
    printf("Ingrese la nota de APE 2: ");
    scanf("%f", &ape2);


    printf("Ingrese la nota de AA 1: ");
    scanf("%f", &aa1);
    printf("Ingrese la nota de AA 2: ");
    scanf("%f", &aa2);


    printf("Ingrese la nota de ES 1: ");
    scanf("%f", &es1);
    printf("Ingrese la nota de ES 2: ");
    scanf("%f", &es2);

    // Procesamiento de promedios

    promedio_acd = (acd1 + acd2)/2 * 0.2;
    promedio_ape = (ape1 + ape2)/2 * 0.25;
    promedio_aa = (aa1 + aa2)/2 * 0.2;


    nota_es_sin_pond =( (es1 * 0.4) + (es2 * 0.6) ) * 0.35;

    nota_final_unidad1 = promedio_acd + promedio_ape + promedio_aa + nota_es_sin_pond;


    // Evaluación de la nota final

   if (nota_final_unidad1 >= 9.0) {
        clasificacion = "Excelente";
    } else if (nota_final_unidad1 >= 7.0 && nota_final_unidad1 < 9.0) {
        clasificacion = "Bueno";
    } else if (nota_final_unidad1 >= 5.0 && nota_final_unidad1 < 7.0) {
        clasificacion = "Regular";
    } else {
        clasificacion = "Deficiente";
    }

    // Datos de salida
```

```c
        printf("\nResultados de la Unidad 1:\n");

        printf("Nota Ponderada ACD: %.2f\n", promedio_acd);
        printf("Nota Ponderada APE: %.2f\n", promedio_ape);
        printf("Nota Ponderada AA: %.2f\n", promedio_aa);



        printf("\nLa nota final de la Unidad 1 es: %.2f\n", nota_final_unidad1);
        printf("Clasificacion: %s\n", clasificacion);



    return 0;
}
```

Resultados

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

Resultados de la Unidad 1:
Nota Ponderada ACD: 1.80
Nota Ponderada APE: 2.00
Nota Ponderada AA: 1.90

La nota final de la Unidad 1 es: 9.06
Clasificacion: Excelente
PS C:\Users\Usuario\Documents\Java_C_Python_VSC>
```

6. Preguntas de Control

- **¿Qué es una condición y cuál es su función dentro de una estructura condicional en programación?**

  Una condición es una expresión lógica que siempre resulta en un valor booleano (verdadero o falso) al ser evaluada. Su función principal dentro de una estructura condicional (if, while, etc.) es actuar como un filtro o interruptor que determina el flujo de ejecución del programa, decidiendo qué bloque de instrucciones debe ser ejecutado y cuál debe ser omitido.

- **¿Qué diferencia existe entre una estructura condicional simple, doble y múltiple?**

  La diferencia fundamental entre las estructuras condicionales radica en los caminos de ejecución que ofrecen: una estructura simple (if) solo ejecuta código si la condición es verdadera y salta al final si es falsa; una estructura doble (if-else) ofrece dos caminos obligatorios, ejecutando un bloque si es verdadera y uno alternativo si es falsa; mientras que una múltiple (if-else if-else o switch) permite evaluar una serie de condiciones secuenciales, ejecutando el bloque del primer caso verdadero encontrado y, opcionalmente, un bloque por defecto.

- **¿Qué es una estructura condicional anidada y en qué casos resulta útil emplearla dentro de un programa?**

  Una estructura condicional anidada es una estructura (if, if-else, etc.) que se coloca completamente dentro del bloque de código de otra estructura condicional externa. Resulta útil emplearla cuando la decisión final que se desea tomar depende de múltiples niveles de evaluación jerárquica, permitiendo refinar una decisión inicial y solo evaluar la segunda condición (la anidada) si la primera condición (la externa) ya se ha cumplido.

7. Conclusiones

Aplicación de estructuras condicionales: La práctica permitió comprender y aplicar eficazmente las estructuras condicionales (simples, dobles y múltiples) al diseñar y codificar un algoritmo para clasificar el nivel de desempeño de una nota final. Esto demuestra que estas estructuras son fundamentales para introducir **lógica de decisión** en el flujo de un programa.

Diseño y validación: El proceso de diseñar el algoritmo mediante un diagrama de flujo y realizar pruebas de escritorio resultó crucial para verificar la lógica antes de la implementación. La posterior codificación en C y las pruebas prácticas validaron el funcionamiento correcto del programa ante las condiciones de clasificación de notas.

8. Recomendaciones

Se recomienda implementar validaciones de entrada (if simples) al inicio del código para verificar que todas las notas ingresadas por el usuario estén dentro del rango permitido (ej. 0 a 10$), haciendo el programa más robusto en escenarios de aplicación real.