22-2-2026

Semana 4

Estudiantes- Programación dinámica

Laila Zareth Romano Guerrero
DDYA-7

Descripción

El problema consiste en desarrollar una herramienta que permita procesar una lista de estudiantes (nombres y notas) para identificar quiénes tienen un rendimiento sobresaliente. Sin embargo, a diferencia de un promedio fijo, el sistema ahora debe permitir una evaluación personalizada por rangos. El usuario ya no está limitado a comparar a los estudiantes contra el promedio de todo el grupo. Ahora, el usuario puede seleccionar un grupo específico (por ejemplo, del estudiante número 5 al 10) para calcular un promedio de referencia.

H01 - Procesamiento de información.

- **Descripción:** Como usuario, puede ingresar todos los nombres y notas una sola vez al inicio para que queden registrados en el sistema.
- **Caso exitoso:** El usuario ingresa la cadena "Nombre Nota Nombre Nota" y el sistema llena las listas nombres y notas.
- **Caso no exitoso**: El usuario ingresa una cadena vacía o con datos impares, impidiendo la correcta asociación nombre-nota.
- **Entrada:** Cadena de texto.
- **Salida:** Confirmación interna.

H02 - Selección de grupo para promedio de referencia.

- **Descripción:** Como usuario, puede elegir un rango de la lista para calcular un promedio específico y usarlo como base de comparación.
- **Caso exitoso:** El usuario ingresa índices válidos y el sistema obtiene el promedio de ese grupo.
- **Caso no exitoso**: El usuario ingresa un índice fuera de los límites (ej. índice 10 en una lista de 5) o un índice final menor al inicial.
- **Entrada:** Dos números enteros (Inicio y Fin).
- **Salida:** Valor del promedio calculado del rango.

H03 - Consulta rápida de rangos repetidos.

- **Descripción:** Como usuario, puede que, si vuelve a elegir un rango que ya consulto antes, el sistema me dé la respuesta sin demoras de cálculo.
- **Caso exitoso:** El usuario ingresa un rango ya procesado y el sistema recupera el promedio del diccionario memo.
- **Caso no exitoso:** El rango es nuevo, por lo que el sistema debe calcularlo y guardarlo por primera vez.
- **Entrada:** Rango (inicio-fin).
- **Salida**: Promedio recuperado o almacenado.

H04 - Identificación de estudiantes aprobados.

- **Descripción:** Como usuario, puede ver la lista de todos los estudiantes del grupo que superan el promedio del rango que elegí.
- **Caso exitoso:** El sistema descompone la lista recursivamente y muestra los nombres que cumplen la condición.

- **Caso no exitoso:** Ningún estudiante de la lista supera el promedio del rango de referencia seleccionado.
- **Entrada:** Promedio de referencia y listas de datos.
- **Salida:** Lista de nombres que superan la nota de referencia.

Diagrama de flujo

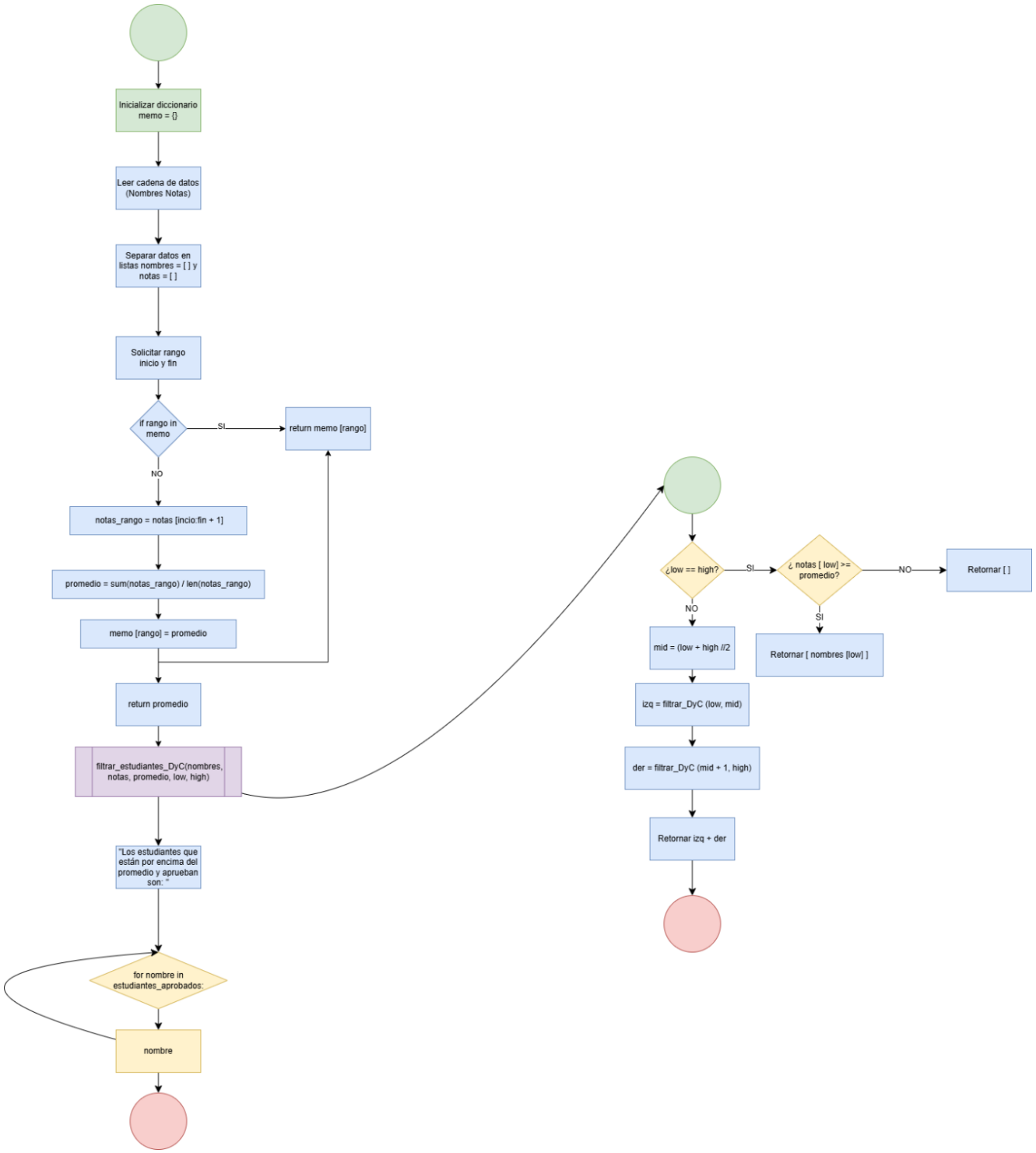


Diagrama de secuencia

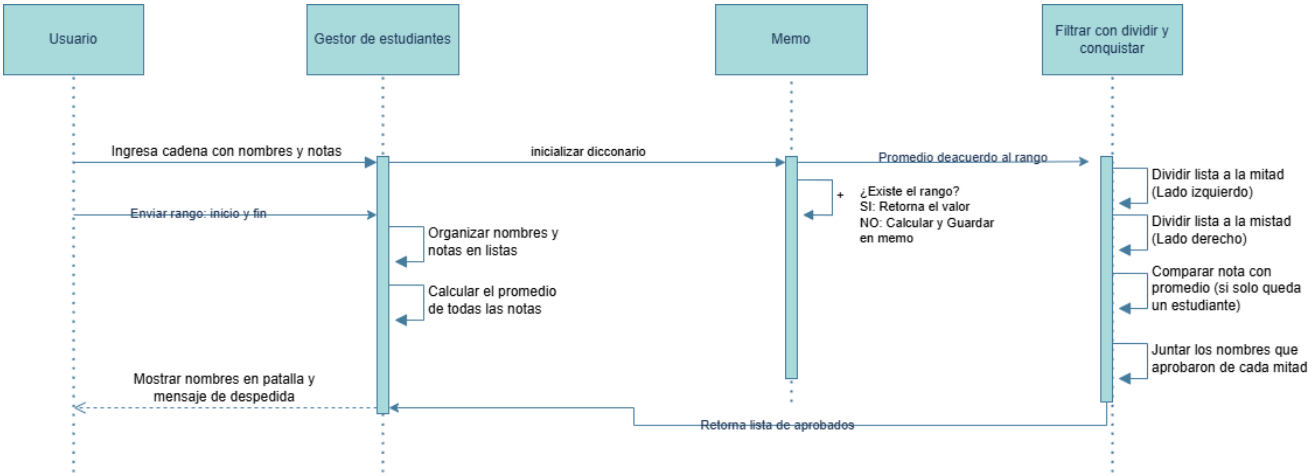


Diagrama de caso de uso

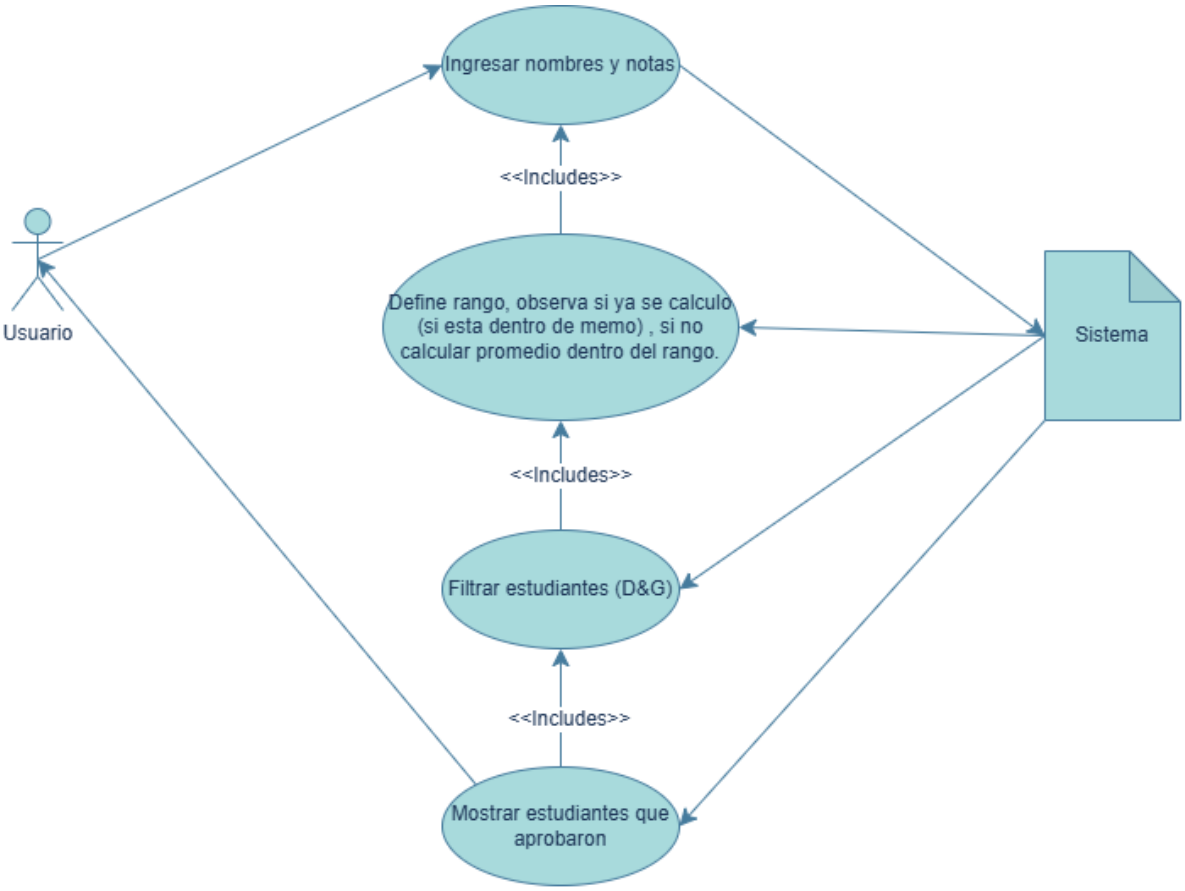


Tabla de complejidad

Línea	Cost	Times
`memo = {}`	C_1	1
`Def calcular_promedio_memo(notas, inicio, fin):`		
`    rango = (inicio, fin)`	C_2	1
`    if rango in memo:`	C_3	1(rango in memo)
`        return memo[rango]`	C_4	1
`    notas_rango = notas[inicio : fin + 1]`	C_5	n (inicio: fin)
`    promedio = sum(notas_rango) / len(notas_rango)`	C_6	n (inicio: fin)
`    memo[rango] = promedio`	C_7	1
`    return promedio`	C_8	1
`def filtrar_estudiantes_DyC(nombres, notas, promedio, low, high):`		
`    if low == high:`	C_9	1
`            return [nombres[low]]`	C_{10}	1(low == high)
`        else:`	C_{11}	1
`            return []`	C_{12}	1
`    mid = (low + high) // 2`	C_{13}	1
`    izq = filtrar_estudiantes_DyC(nombres, notas, promedio, low, mid)`	$C_{14} + T(n/2)$	1
`    der = filtrar_estudiantes_DyC(nombres, notas, promedio, mid + 1, high)`	$C_{15} + T(n/2)$	1
`    return izq + der`	C_{16}	n
`def gestor_estudiantes_dinamico():`		
`    entrada = input("Ingrese datos: ")`	C_{17}	1
`    datos = entrada.split()`	C_{18}	1
`    nombres = [datos[i] for i in range(0, len(datos), 2)]`	C_{19}	n
`    notas = [float(datos[i+1]) for i in range(0, len(datos), 2)]`	C_{20}	n
`    continuar = True`	C_{21}	1
`    while continuar:`	C_{22}	n
`        print("\nTotal estudiantes:" , len(nombres))`	C_{23}	1
`        inicio = int(input("Desde qué índice (0 a " + str(len(nombres)-1) + "): "))`	C_{24}	1

`fin = int(input("Hasta qué índice (0 a " + str(len(nombres)-1) + "): "))`	C_{25}	1
`prom_ref = calcular_promedio_memo(notas, inicio, fin)`	C_{26}	n
`aprobados = filtrar_estudiantes_DyC(nombres, notas, prom_ref, 0, len(notas)-1)`	C_{27}	n
`print("Superan el promedio de", prom_ref, ":", aprobados)`	C_{28}	1
`opcion = input("\n¿Desea probar otro rango? (si/no): ")`	C_{29}	1
`if opcion.lower() != 'si':`	C_{30}	$1(\text{opcion} != \text{"si"})$
`continuar = False`	C_{31}	1

Complejidad total

$$T(n) = Cn + 2T(n/2) + C$$

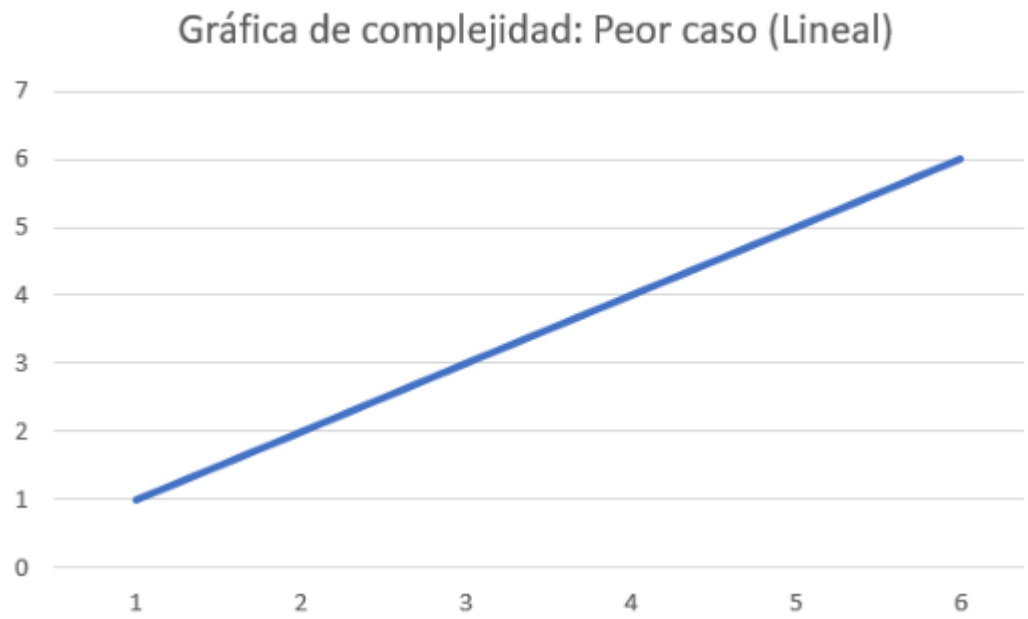
$$T(n) = O(n)$$

Mejor caso: $\boldsymbol{O(1)}$

El sistema no tiene datos que procesar o solo se ingresó un dato el cual no generada algún rango para calcular su promedio, simplemente pasa como aprobado.

Peor caso: $\boldsymbol{O(n)}$

El sistema procesa una cantidad n grande de estudiantes, debe separar los datos, pide un rango de que estudiantes serán la base para calcular el promedio, el sistema valida si ya está en la memoria calculador y si no es así lo calcula, en caso de que el usuario solicite nuevamente el mismo rango el sistema no lo calculara de nuevo ya que revisa en su memoria y ya está. Al utilizar estar herramienta dinámica hacemos que el sistema no haga pasos innecesarios y se centre en ser mas eficiente



Análisis

Anteriormente al implementar solo dividir y conquistar como estrategia, si pedíamos el promedio de los mismos datos 10 veces, el sistema sumaba y dividía 10 veces dándonos un gasto innecesario de ejecuciones. En este caso, al implementar D&G y programación dinámica si el usuario pide el mismo rango 10 veces, el programa calcula el promedio de ese rango una sola vez y las otras 9 veces simplemente busca si este ya está calculado en su memoria para entregar el resultado sin tener que calcularlo nuevamente. Convirtiendo un proceso de cálculo recursivo en una simple consulta de memoria.