DOCUMENTO TÉCNICO

# Requisitos

## Especificación

#RESUMEN DEL PROBLEMA (opcional)

Encontrar el superdigit de un numero

### Entrada:

### La primera línea contiene dos números enteros separados por espacios y

### Salida:

# Devuelve el super dígito de p , donde p se crea como la suma de los dígitos que lo conforman y este proceso se repite hasta que dé como resultado un número menor a 10

# Diseño

## Estrategia

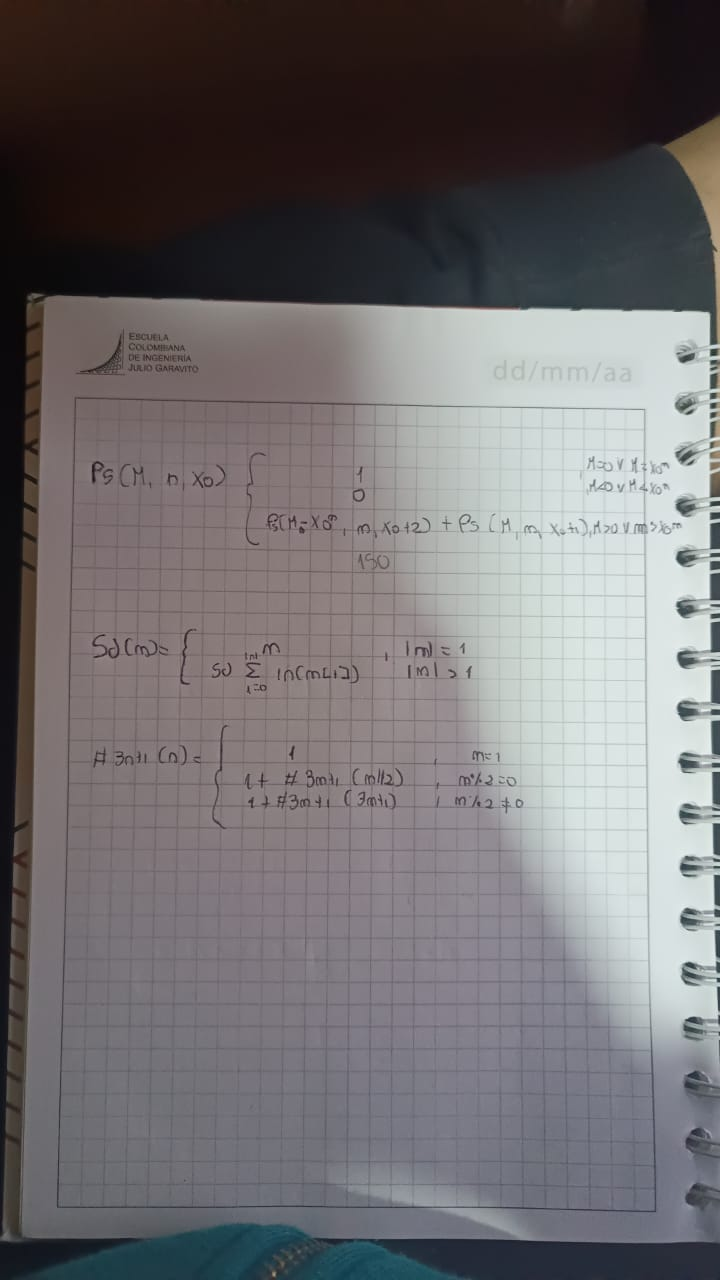
### La solución al problema no es del todo complicada. Primero se toma el numero dado, se toma uno a uno los dígitos que lo componen, se realiza la suma y si el resultado tiene mas de 1 numero , se repite el proceso hasta que sea menor a 10.

### #Estructuras de datos

### Se usaron las listas para dividir los dígitos del numero y realizar el proceso de la suma mediante el comando sum

#ALGORITMO

Programa adjunto



#INVARIANTE

* Iniciación: No hay casos
* Estabilidad: Por cada recursión se va sumando los dígitos del numero o si es menor a 10 se retorna el resultado
* Terminación: Hasta que el numero sea menor a 10 o tenga una longitud de 1 .

## Casos de prueba

|  |  |  |
| --- | --- | --- |
| **Entrada** | **Justificación** | **Salida** |
|  |  |  |
| 1 1 | Caso eficiente | 1 |
| *9875 100000000* | Caso extenso | 2 |
| 9875 4 | Caso común | *8* |

# Análisis

# Texto Descripción generada automáticamente

# Interfaz de usuario gráfica Descripción generada automáticamente

# 

## Temporal

La complejidad de este código es aproximadamente de log(n) ya que, si un elemento tiene n elementos, aproximadamente va a realizar log(n) sumas, no va realizar n sumas necesariamente. Si el numero es muy grande , podría haber problemas en el rendimiento del computador , por esto se aplica memorización y la complejidad pasa a ser constante.

# Código

*Al igual que el punto anterior pueden guiarse con este ejemplo*

## Documentación

Dentro del código.

## Fuentes

*Las fuentes son el código sobre el cual escribieron el documento*

/arena\_1\_a