DOCUMENTO TÉCNICO

Integrantes:

- Diego Cardenas
- Zayra Gutiérrez
- Felipe Calvache

Solución al Algoritmo Super Digit

Requisitos

Especificación

Definimos el SuperDigit de un entero usando las siguientes reglas: Dado un número entero, necesitamos encontrar el superdígito del número entero. Si x solo tiene un dígito, entonces su superdígito es x. De lo contrario, el superdígito de x es igual al superdígito de la suma de los dígitos de x. Por ejemplo, el superdígito de 9875 se calculará como: 9 + 8 + 7 + 5. Se le dan dos números n y k. El numero p se crea concatenando los strings de la cadena n veces.

Entrada:

La primera línea contiene dos espacios separando enteros, n y k. Donde n es un String que representa un entero k que es un entero que es las veces que se concatena n para generar p.

Salida:

Retorna el superdígito de p, donde p es creado según lo descrito

Estrategia

A los dos valores de la línea ingresada se les realiza la concatenación para crear el número p, luego se crea la función que separa los dígitos y los suma, repite iterativamente este proceso de descomponer en dígitos hasta que solo tenga un dígito.}

Resumen del problema

se trata de un desafío de programación que se presenta en el sitio web HackerRank.

La tarea consiste en encontrar la suma recursiva de los dígitos de un número entero dado. Para ello, primero se debe sumar los dígitos del número inicial y, si el resultado es mayor que 9, se debe repetir el proceso hasta que se obtenga un solo dígito.

Por ejemplo, si se tiene el número entero 148, la suma de sus dígitos sería 1 + 4 + 8 = 13. Luego, como 13 es mayor que 9, se debe sumar sus dígitos nuevamente, es decir, 1 + 3 = 4.

El objetivo del problema es implementar una función que calcule la suma recursiva de dígitos de un número entero dado y retornar el resultado final.

Entrada:

Recibe una constante la cual es el numero x y k la cual es las veces que se repite el numero x.

Salida:
Retorna la sumatoria de valor a valor de x hasta que esta sea menor a 10.

Diseño

Estrategia
Implementar una función que calcule la suma de los dígitos de un número entero. Esta función debe tomar como entrada un número entero y retornar la suma de sus dígitos.

Utilizar la función del paso anterior para calcular la suma de dígitos del número dado. Si el resultado es menor o igual a 9, retornar el resultado.

Si el resultado de la suma de dígitos es mayor que 9, llamar recursivamente a la misma función con el resultado de la suma de dígitos como entrada.

Repetir el paso anterior hasta que se obtenga un resultado menor o igual a 9.

Retornar el resultado final.

En resumen, se puede usar una función recursiva que llame a sí misma hasta que la suma de los dígitos sea menor o igual a 9. Cada llamada recursiva debe calcular la suma de los dígitos y pasar el resultado a la siguiente llamada recursiva.

Algoritmo
Se realiza un algoritmo recursivo el cual es el óptimo para este problema

```
from sys import stdin
def digit(n):                          # costos      pasos
    resto = 0                           # 1           1
    while n != 0:                       # 1           n
        resto += n%10                   # 1           n
        n = n//10                       # 1           n
    if resto > 10:                      # 1           n
        return digit(resto)             # 1           n
    return resto                        # 1           1

def repe(n, k):                        # costos      pasos
    list = []                           # 1           1
    nums = []                           # 1           1
    entr = ""                           # 1           1
    for i in str(n):                    # 1           n-1
        nums.append(i)                  # 1           n-1
    for j in range(k):                  # 1           n-1
        list.extend(nums)               # 1           n-1
    for l in range(len(list)):          # 1           n-1
        entr = str(entr)+str(list[l])   # 1           n-1
    return int(entr)                    # 1           1
```

```
def main():                                  # costos      pasos
   x = stdin.readline().strip()                 # 1          1
   while x:                                     # 1          n
      x = x.split(" ")                          # 1          n
      print(digit(repe(int(x[0]), int(x[1]))))  # 1          n
      x = stdin.readline().strip()              # 1          n
main()
```

Invariantes

Invariante #1:

Saca los datos de uno a uno y los suma y si la suma sigue siendo más mayor a 9 entonces vuelve a sumar los valores

- Iniciación: Recibe el dato de n el cual si es diferente a 0 crea un ciclo que se repite hasta que b sea diferente a 0 y adjunta la suma de n%10 y actualiza n a n//10, después suma los datos hasta que el resultado de la suma de n%10 sea < 10.
- Estabilidad: Es estable en el sentido de no tener complejidad a escoger sumar los datos y recibirlos y devolver la suma
- terminación: Termina cuando n <10.

Invariante #2:

Repite el numero x k veces

- Iniciación: sin problemas y de manera óptima el cual no pretende ser y lee los datos con listas si nada de condicionales avanzados.
- Estabilidad: Estable en el sentido de capturar y devolver los datos correctos sin complicaciones.
- Terminación: termina cuando repite las k veces.

Invariante #3

Recibe los datos e imprime el resultado

- Iniciación: Recibe dos datos tanto x como k
- Estabilidad: debido a que estos datos se envían directamente a repe solamente. imprime el resultado de repe.
- terminación: Termina cuando se ingresa algo vacío.

Casos prueba

Entrada	Justificación	Salida
100 1000	Verificar la respuesta en este rango	1
23 27	Verificar respuesta con valores 23, 27	9
0 0	Probar con los dos valores siendo 0	0

Fuentes

/Arena_2_b