Auxiliar 3 - Recursión

P1. Cree la función maxComunDiv(n,m) que entregue el máximo común divisor de los números enteros n y m entregados. Siga el algoritmo recursivo de Euclides para su construcción.

Algoritmo de Euclides:

- maxComunDiv(n,m) = n si n==m
- si n<m -> maxComunDiv(n,m-n)
- si n>m -> maxComunDiv(n-m,m)

Ej: maxComunDiv(12,40) = 4

```
#maxComunDiv: int int -> int
#Calcula el máximo común divisor de los números enteros entregados
#Ej: maxComunDiv(12,40) = 4

def maxComunDiv(n,m):
  if n == m:
    return n
  elif n>m:
    return maxComunDiv(n-m,m)
  else: return maxComunDiv(n,m-n)

assert maxComunDiv(12,40) == 4
assert maxComunDiv(18,9) == 9
assert maxComunDiv(120,360) == 120
```

P2. Defina la función elMayor(n), que a partir de un número entregado devuelva el mayor de ss dígitos.

Ej: elMayor(257926) = 9

```
#elMayor: int -> int
#Devuelve el digito mayor de un numero entregado
#Ejemplo: elMayor(257926) = 9

def elMayor(n):
  if n == 0:
    return 0
  else:
    ultimo = n%10
    return max(ultimo,elMayor(n//10))

assert elMayor(198739) == 9
assert elMayor(37863) == 8
```

```
assert elMayor(362702) == 7
```

P3. Cree la función invertir(n) cuya función es invertir el número n, el cual tiene una catidad cualquiera de digitos.

Ejemplo:

```
  invertir(123456) = 654321
```

```
#invertir(n): int -> int
#Invierte los digitos de un numero dado
#Ejemplo: invertir(123456) -> 654321

def invertir(n,numero=0):
  if n<10:
    return numero+n
  else:
    numero += n%10
    return invertir(n//10, numero*10)

assert invertir(243) == 342
assert invertir(123456) == 654321
assert invertir(46372213) == 31227364
```

P4. Cree de forma recursiva el programa interactivo calculadoraNotas() el cual le vaya pidiendo sus notas y las vaya almacenando junto con la cantidad de notas que ha ingresado. Cuando quiera que el programa deje de funcionar y le entregue el resultado, el usuario deberá ingresar el número: 0, con esto se calculará el promedio con las notas ingresadas y se le aplicará el redondeo al primer dígito, mostando en pantalla `Su nota final es: +` la nota calculada. En el caso de que el numero entregado sea menor a 1.0 o mayor a 7.0, el programa mostrará en pantalla `Nota invalida` y seguirá corriendo, sin contar la nota ingresada erroneamente.

Ejemplo:

```
  Ingrese una nota entre 1.0 y 7.0: 4.1
  Ingrese una nota entre 1.0 y 7.0: 7.8
  Nota invalida
  Ingrese una nota entre 1.0 y 7.0: 4.3
  Ingrese una nota entre 1.0 y 7.0: 0
  Su nota final es: 4.2
```

```
#calculadoraNotas():   -> float
#Solicita la nota al usuario y la almacena, junto con la cantidad de notas que se h
```

```
#cuando el usuario ingresa el número 0, se entrega el promedio

def calculadoraNotas(notasAcumuladas=0,cantidadNotas=0):
  nota = float(input('Ingrese una nota entre 1.0 y 7.0: '))
  if nota==0:
    notaFinal = round(notasAcumuladas/cantidadNotas,1)
    return 'Su nota final es: ' + str(notaFinal)
  elif (nota>0 and nota<1.0) or (nota>7.0):
    print('Nota invalida')
    return calculadoraNotas(notasAcumuladas,cantidadNotas)
  else:
    notasAcumuladas += nota
    cantidadNotas += 1
    return calculadoraNotas(notasAcumuladas,cantidadNotas)

calculadoraNotas()
```

P5. Cree la función sumaDigitos(n) la cual va a sumar todos los digitos de el número entregado.

Ejemplo:

```
  sumaDigitos(12345) = 15
```

```
#sumaDigitos(n): int -> int
#Suma los digitos de un numero
#sumaDigitos(12345) = 15

def sumaDigitos(n):
  if n<10:
    return n
  else: return n%10 + sumaDigitos(n//10)

assert sumaDigitos(12345) == 15
assert sumaDigitos(4683293874) == 54
assert sumaDigitos(12) == 3
```

P6. Existe una montaña rusa que posee un sensor de movimiento en la base, el cual va midiendo la altura con respecto al suelo cada cierto tiempo, opteniendo valores enteros entre 1 y 9, considerando que 1 coresponde a 10 metros y las medidas se redondean.

Los resultados finales entregados por este sensor son enteros que corresponden a las medidas tomas concatenadas, de forma que el entero 14562 corresponde a las alturas aproximadas de 10, 40, 50, 60 y 20, en ese orden.

Creeuna función llamada `diferenciaMax(n)` que dado un entero como el descrito anteriormente devuelva la mayor diferencia (en valor absoluto) entre dos alturas contiguas.

Ejemplo:

```
diferenciaMax(14562)
40
```

```
#diferenciaMax(n): int -> int
#A partir de un entero muestra la diferencia máx entre dos numeros contiguos
#Ej: diferenciaMax(14562) = 40

def diferenciaMax(n):
  if n<100:
    return abs(n//10-n%10)
  else:
    dif = abs(((n//10)%10)-n%10)
    return max(dif,diferenciaMax(n//10))


assert diferenciaMax(11113111) == 2
assert diferenciaMax(45012345) == 5
assert diferenciaMax(40912345) == 9
assert diferenciaMax(11111111) == 0
```