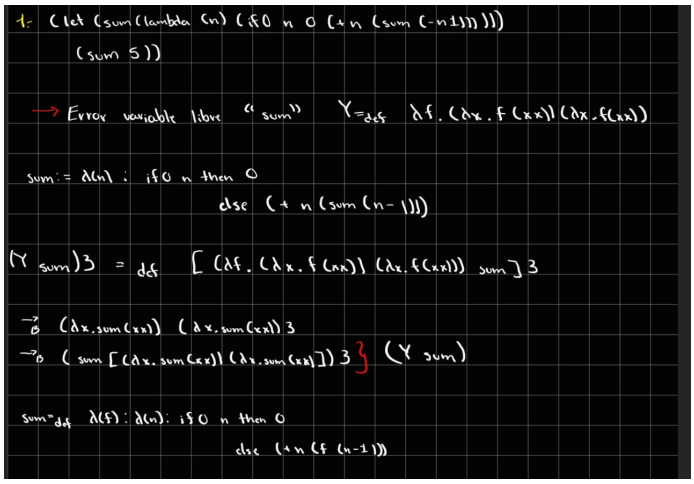
Universidad Nacional Autónoma De México
Facultad de Ciencias

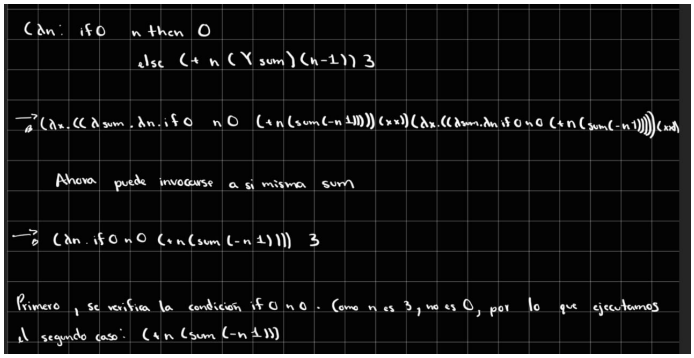
Semanal 7

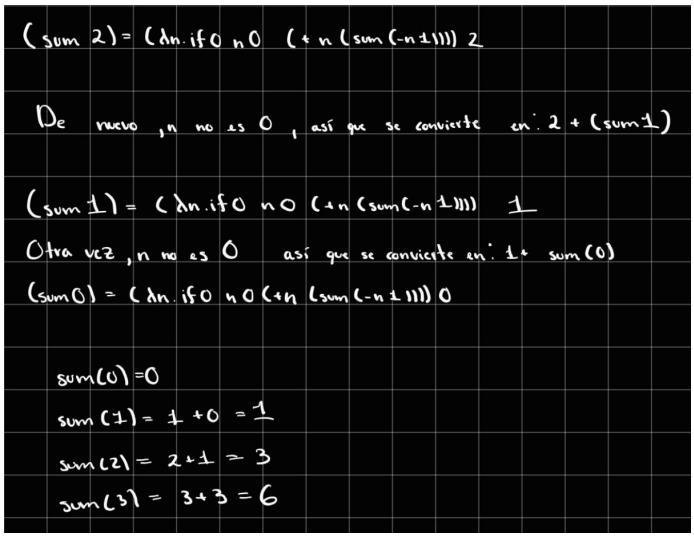
Lenguajes de programacion

Integrantes:
Emiliano Luna Campos: 320292084
☎☎☎☎☎☎☎
Jose Luis Flores Gutierrez: 320060087
☎☎☎☎☎☎☎
Oscar Tapia Sanchez: 320266076
☎☎☎☎☎☎☎

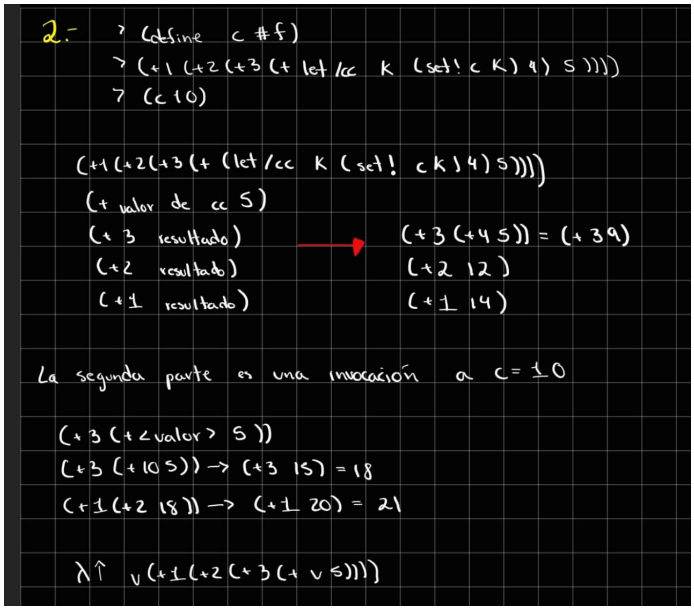
1. Ejercicio 1



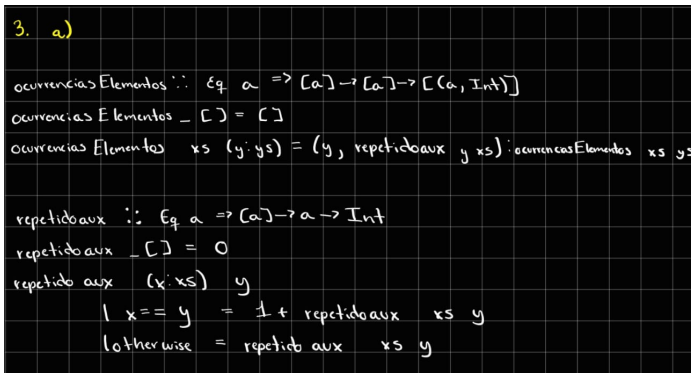




2. Ejercicio 2



3. Ejercicio 3



b) ocurrenciasElementos [1,2,3] [1,2]
L1 L2

1.- Agarramos el 1 de L2 y llamamos a repetidoAux para que vea cuantas veces se repite en L1

repetidoaux 1 [1,2,3] primero se compara 1==1 y es un True #T, asi que agrega 1.

Luego se compara 1 [2,3] 1==2 es #f y continúa

Al final solo se compara con el 3, 1==3 es #f y termina

Agarramos el 2 de L2 y llamamos otra vez a repetidoAux para que vea cuantas veces se repite en L1

Primero llama repetidoaux 2 [1,2,3] 2==1 es False #f y continúa, luego esta repetidoaux 2 [2,3] y 2==2 es True añade o le suma 1 y luego queda 2 [3] 2==3 es false termina.

Llama a ocurrenciaElementos y como ya utilizamos todos los elementos de L2 entonces ocurrencias Elementos [1,2,3] []

La salida seria [(1,1),(2,1)]

c) repetido aux :: Eq a=>[a]-> a ->Int ->Int
repetido aux [] _ acc = acc
repetido aux (x:xs) e acc
| e == x = repetido aux xs e (acc+1)
| otherwise = repetidoaux xs e acc

repetido Ocu :: Eq a => [a]-> [a] -> [a,Int)] -> [(a,Int)]
repetido Ocu _ [] acc = reverse acc
repetido Ocu xs (y:ys) acc =repetidoOcu xs ys((y repetidoaux xs y 0):acc)

OcurrenciasElementos :: Eq a => [a] -> [a] -> [(a,Int)]
ocurrenciasElementos xs ys = ocurrenciasElementos xs ys []

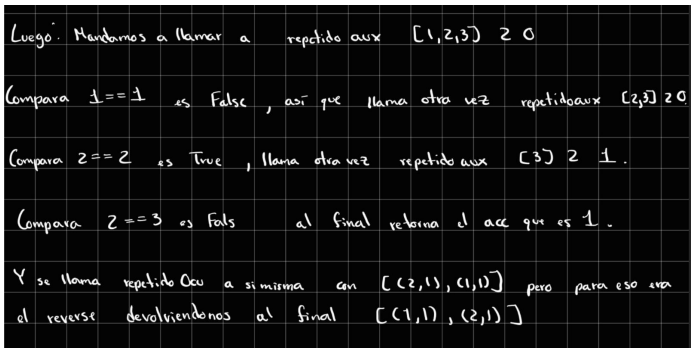
d) Es evaluado ocurrenciasElementos [1,2,3] [1,2]
L1 L2

Manda a llamar a repetidoOcu [1,2,3] [1,2] [] acc

Despues esa manda a llamar a repetidoAux [1,2,3] 1 0
elem L2

Compara 1==1 es True y manda allamar a repetido aux [2,3] 1 1.
Luego:
Compara 1==2 es False y manda a llamar a repetido aux [3] 1 1.
Despues
Compara 1==3 es False y retorna 1
La funcion repetidoOcu se llama a si misma y tiene [(1,1)]

Luego: Mandamos a llamar a repetido aux [1,2,3] 2 0


Luego. Mandamos a llamar a repetido aux [1,2,3] 2 0
Compara 1==1 es False, así que llama otra vez repetidoaux [2,3] 2 0
Compara 2==2 es True, llama otra vez repetido aux [3] 2 1.
Compara 2==3 es Fals al final retorna el acc que es 1.
Y se llama repetido Occ a si misma con [(2,1),(1,1)] pero para eso era el reverse devolviendonos al final [(1,1),(2,1)]