Tarea 4
Algoritmos y Complejidad
2021-1

Camilo Contreras
201873063-7

27 de mayo de 2021

Concepto	**Tiempo [min]**
Investigación	4[h]
Desarrollo	3[h]
Informe	2[h]

1. Branch and Bound

En la perdida colonia antártica de Nadiria la moneda venía en denominaciones de $1, $4, $7, $13, $28, $52, $91 y $365. Como el papel era extremadamente escaso, por ley cualquier transacción en dinero debía usar el mínimo número de billetes.

Escriba un programa que lea de la entrada estándar un número n de cantidades a procesar, seguido por n cantidades de la entrada estándar. Puede suponer que los datos son correctos, no hace falta validar. Para cada cantidad escriba esta, el número mínimo de billetes y cuántos de cada denominación entregar, en orden decreciente. Su rutina principal debe tomar como datos el arreglo denom de denominaciones a manejar (no se limite al caso de Nadiria) y la cantidad c a entregar, y retornar el arreglo de cantidades de cada moneda. Por ejemplo:

```
change([5, 4, 1], 9)
--> [1, 1, 0]
change([5, 4, 1], 8)
--> [0, 2, 0]
change([365, 91, 52, 28, 13, 7, 4, 1], 416)
--> [0, 4, 1, 0, 0, 0, 0, 0]
```

2. Solución

En nuestro programa, utilizamos una cota superior muy grande, para evitar conflictos. Lo que hace el algortimo es recorrer la lista de billetes, revisa el primer billete, y si este sirve para nuestro monto, lo agregamos a la lista de monedas, restamos la moneda a nuestro valor inicial, y asi ahora llamamos a la funcion nuevamente, con este nuevo valor, aplciando recursividad.

```
for i in billetes:
    if i > n:
        pass
    else:
        totalMon = 1 + changePrincipal(billetes, n - i)[0]
        if totalMon < minMon:
            minMon = totalMon
            arr = changePrincipal(billetes, n - i)[1]+ [i]
```

El punto critico de la funcion es nuestro break, es este el cual nos permite descartar ramas que no son eficientes, agilizando el proceso mucho mas. Se compara si el minimo que llevamos es menor que el minimo global, de no ser asi, aplica el break, rompiendo el ciclo for y terminando con esa rama.

```
if totalMon < minMon:
    minMon = totalMon
        arr = changePrincipal(billetes, n - i)[1]+ [i]
else:
    break
```

Finalmente tenemos una funcion cuyo objetivo es arreglar los arreglos para una salida mas amena a la vista. Como podemos ver en la siguiente imagen:

```
------------Money-------------
[365, 91, 52, 28, 13, 7, 4, 1]
------------------------------
Cambio: 416
[0, 4, 1, 0, 0, 0, 0, 0]
```