

Laboratorio N-4

Antes de comenzar ...

1. Instalar dependencias de desarrollo y creacion del `package.lock.json`

```
npm i
```

2. El codigo fue separado en carpetas `functions` su funcion tiene una relacion directa con el nombre respectivo.
3. Ejecuta el codigo con el siguiente comando

```
node index.js
```

Kadane

1. La ubicacion del archivo se encuentra en `functions/kadane.js`
2. La complejidad del algoritmo es de $O(n)$
3. Se cumple con la salida requerida, puede visualizarlo compilando el programa o mirar el resultado en el archivo `/data/kadane.txt`

Divide and Conquer

```
SEG-MAX-DV(A,p,r) =  $\Theta(n \log(n))$ 
1: si p = r entonces
2:   devolver max(0, A[p])
3: q ← [(p + r)/2]
4: maxizq ← SEG-MAX-DV(A,p,q)
5: maxder ← SEG-MAX-DV(A,q+1,r)
6: max2izq ← suma ← A[q]
7: para i ← q - 1 desc p hacer
8:   suma ← suma + A[i]
9:   max2izq ← max(max2izq, suma)
10: max2der ← suma ← A[q + 1]
11: para f ← q + 2 hasta r hacer
12:   suma ← suma + A[f]
13:   max2der ← max(max2der, suma)
14: maxcruz ← max2izq + max2der
15: devolver max(maxizq, maxcruz, maxder)
```

1. La ubicacion del archivo se encuentra en `functions/divideAndConquer.js`
2. La complejidad del algoritmo es de $\Theta(n * \log(n))$

Observaciones

- Al ser un algoritmo que ejecuta recursividad se tiene mucha dificultad para tener la salida requerida con respecto a las rutas, pero en si cumple su proposito de obtener la maxima suma.

La ubicacion de la salida esta en `/data/divideAndConquer.txt`

Profesor considerelo plis 🙏

Ejemplo:

```
La ruta 1 tiene un costo optimo de 6
La ruta 2 tiene un costo optimo de 9
La ruta 3 tiene un costo optimo de 6
La ruta 4 tiene un costo optimo de 10
```