

## Trabajo Práctico 2 Programación Dinámica For The Win

TEORÍA DE ALGORITMOS  
(75.29) CURSO BUCHWALD - GENENDER

Nombre	Padrón
Denise Dall'Acqua	108645
Martín Alejo Polese	106808
Nicolás Agustín Riedel	102130

## Índice

1. Algoritmo por programación dinámica	4
2. Demostración del Algoritmo	4
3. Algoritmo planteado y complejidad	4
4. Variabilidad	4
5. Ejemplos de ejecución	4
6. Medición empírica	4
7. Conclusiones	4

## Consigna

### Introducción

Seguimos con la misma situación planteada en el trabajo práctico anterior, pero ahora pasaron varios años. Mateo ahora tiene 7 años. Los mismos años que tenía Sophia cuando comenzaron a jugar al juego de las monedas. Eso quiere decir que Mateo también ya aprendió sobre algoritmos greedy, y lo comenzó a aplicar. Esto hace que ahora quién gane dependa más de quién comience y un tanto de suerte.

Esto no le gusta nada a Sophia. Ella quiere estar segura de ganar siempre. Lo bueno es que ella comenzó a aprender sobre programación dinámica. Ahora va a aplicar esta nueva técnica para asegurarse ganar siempre que pueda.

### Consigna

1. Hacer un análisis del problema, plantear la ecuación de recurrencia correspondiente y proponer un algoritmo por programación dinámica que obtenga la solución óptima al problema planteado: Dada la secuencia de monedas  $m_1, m_2, \dots, m_n$ , sabiendo que Sophia empieza el juego y que Mateo siempre elegirá la moneda más grande para sí entre la primera y la última moneda en sus respectivos turnos, definir qué monedas debe elegir Sophia para asegurarse obtener el máximo valor acumulado posible. Esto no necesariamente le asegurará a Sophia ganar, ya que puede ser que esto no sea obtenible, dado por cómo juega Mateo. Por ejemplo, para  $[1, 10, 5]$ , no importa lo que haga Sophia, Mateo ganará.
2. Demostrar que la ecuación de recurrencia planteada en el punto anterior en efecto nos lleva a obtener el máximo valor acumulado posible.
3. Escribir el algoritmo planteado. Describir y justificar la complejidad de dicho algoritmo. Analizar si (y cómo) afecta a los tiempos del algoritmo planteado la variabilidad de los valores de las monedas.
4. Realizar ejemplos de ejecución para encontrar soluciones y corroborar lo encontrado. Adicionalmente, el curso proveerá con algunos casos particulares que deben cumplirse su optimalidad también.
5. Hacer mediciones de tiempos para corroborar la complejidad teórica indicada. Agregar los casos de prueba necesarios para dicha corroboración (generando sus propios sets de datos). Esta corroboración empírica debe realizarse confeccionando gráficos correspondientes, y utilizando la técnica de cuadrados mínimos. Para esto, proveemos una explicación detallada, en conjunto de ejemplos.
6. Agregar cualquier conclusión que les parezca relevante.

## Resolución

1. Algoritmo por programación dinámica
2. Demostración del Algoritmo
3. Algoritmo planteado y complejidad
4. Variabilidad
5. Ejemplos de ejecución
6. Medición empírica
7. Conclusiones