



# TALLER DE TECNOLOGÍAS DE PRODUCCIÓN DE SOFTWARE

Milla, Andrés



# AGENDA

1

## PROGRAMACIÓN DINÁMICA

- ▶ Desarrollo sintético del tema teórico sobre el que se basa el problema.

2

## UVA 12146 - CANDY

- ▶ Planteo y explicación del problema.

3

## SOLUCIONES

- ▶ Proceso de desarrollo.
- ▶ Soluciones alternativas.

4

## CASOS DE PRUEBA

- ▶ Diferentes casos de prueba que responden a situaciones diferentes.





# **PROGRAMACIÓN DINÁMICA**

# PROGRAMACIÓN DINÁMICA - DEFINICIONES

- ➔ **Estrategia** de resolución de problemas:
  - ➔ Comenzar resolviendo las partes o problemas más sencillos posibles hasta la solución final.
  
- ➔ **Objetivos:**
  - ➔ Optimizar el cómputo
  - ➔ Reducir el tiempo de ejecución
  
- ➔ El algoritmo debe cumplir con las siguientes **condiciones:**
  - ➔ *Overlapping subproblems* → El algoritmo puede ser dividido en subproblemas y reutilizarlos múltiple veces para llegar a la solución final.
  - ➔ *Optimal substructure* → La solución óptima puede ser construida mediante las soluciones óptimas de los subproblemas.





# PROGRAMACIÓN DINÁMICA - VARIANTES

→ **Top-Down:**

- Solución recursiva.
- Se comienza desde el problema original y se divide en subproblemas más chicos.
- Reutilizar resultados intermedios → *memoization*.

→ **Bottom-Up:** Se reformula el problema iterativamente.

- Solución iterativa.
- Se comienza de los casos chicos hasta llegar a la solución final.
- Reutilizar resultados intermedios → *tablas "DP"*.

→ **¿Cuál elegir?** → Depende del problema, pero en la mayoría de los casos:

- **Top-Down:** Menor esfuerzo de programación.
- **Bottom-Up:** Menor tiempo de ejecución.






**UVA 12146  
CANDY**

# CANDY - PROBLEMA

- ➡ Un niño participa en una competencia de caramelos.
- ➡ Las **cajas de caramelos** se depositan en una grilla de M filas y N columnas.
- ➡ Pero... al agarrar una caja de caramelos, se anulan las cajas de la derecha y la izquierda, como también **todas** las cajas de arriba y abajo.

1	8	2	1	9
1	7	3	5	2
1	2	10	3	10
8	4	7	9	1
7	1	3	1	6



1	8	2	1	9
0	0	0	0	0
1	0	0	0	10
0	0	0	0	0
7	1	3	1	6

- ➡ **Objetivo:** Encontrar la máxima cantidad de caramelos posible.



# CANDY - FORMATO DE ENTRADA/SALIDA

➔ **Input** - Por cada test:

- ➔ En la primer línea se especifica **N** y **M** tq.  $1 \leq M \times N \leq 10^5$ .
- ➔ En las restantes líneas se especifican las filas de cajas de caramelos.

➔ **Output** - Por cada test:

- ➔ Imprimir una línea indicando la máxima cantidad de caramelos que el niño puede tomar.

1	8	2	1	9
1	7	3	5	2
1	2	10	3	10
8	4	7	9	1
7	1	3	1	6



**INPUT**

5 5  
⋮  
1 8 2 1 9  
1 7 3 5 2  
1 2 10 3 10  
8 4 7 9 1  
7 1 3 1 6



**OUTPUT:**

54







# CANDY - ESQUELETO DE LA SOLUCIÓN

```
from sys import stdin, stdout
```

```
def main():  
    for line in stdin:  
        N, M = map(int, line.split())  
  
        if (N, M) == (0, 0):  
            break  
  
        # Read the matrix  
        boxes = [read_row() for _ in range(N)]  
  
        max_candies = calculate_max_candies(boxes)  
  
        stdout.write("{}\n".format(max_candies))  
  
def read_row():  
    return list(map(int, stdin.readline().split()))  
  
if __name__ == "__main__":  
    main()
```

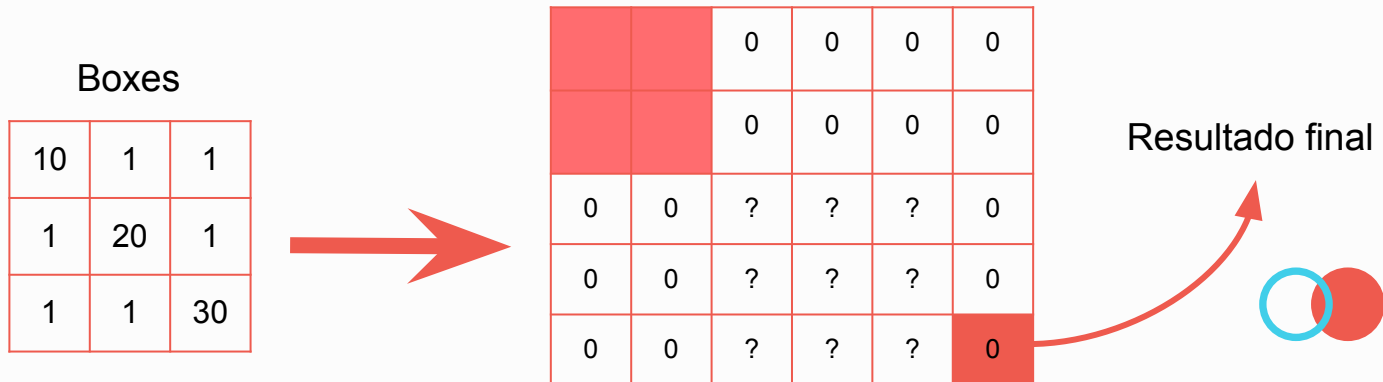
**Lectura**

**Escritura**

# CANDY - SOLUCIÓN

- ➔ Se arma una tabla adicional (denominada DP) de la siguiente forma:
  - ➡ Cada celda representará la máxima cantidad de caramelos para cada subproblema.
  - ➡ Columnas/filas adicionales [ $\text{dimensión}(\text{DP}) = (N + 2 \times M + 3)$ ]
    - ➡ Las primeras 2 filas y columnas serán rellenas con 0 → **Caso base.**
    - ➡ La última columna de cada fila contendrá la máxima cantidad de caramelos de la fila correspondiente.

```
def calculate_max_candies(boxes):  
    N, M = len(boxes), len(boxes[0])  
    DP = [[0 for _ in range(M + 3)] for _ in range(N + 2)]
```



# CANDY - SOLUCIÓN

```
def calculate_max_candies(boxes):  
    N, M = len(boxes), len(boxes[0])  
  
    DP = [[0 for _ in range(M + 3)] for _ in range(N + 2)]  
  
    for row, box in enumerate(boxes, start=2):  
        for col, candies in enumerate(box, start=2):  
            ...
```

Boxes

10	1	1
1	20	1
1	1	30

Comienzo del recorrido

DP

		0	0	0	0
		0	0	0	0
0	0	?	?	?	0
0	0	?	?	?	0
0	0	?	?	?	0



# PROGRAMACIÓN DINÁMICA - SOLUCIÓN

```
def calculate_max_candies(boxes):  
    N, M = len(boxes), len(boxes[0])  
  
    DP = [[0 for _ in range(M + 3)] for _ in range(N + 2)]  
  
    for row, box in enumerate(boxes, start=2):  
        for col, candies in enumerate(box, start=2):  
            DP[row][col] = max(  
                # Alternative 1: Don't take the candy  
                DP[row][col - 1],  
                # Alternative 2: Take the candy  
                candies + DP[row][col - 2],  
            )  
    ...
```

Boxes

10	1	1
1	20	1
1	1	30

**Alternativa 2: 10 + 0**

DP[2,0] = 0	DP[2,1] = 0	DP[2,2] = $\max(0, 10 + 0) = 10$
-------------	-------------	----------------------------------

**Alternativa 1: 0**



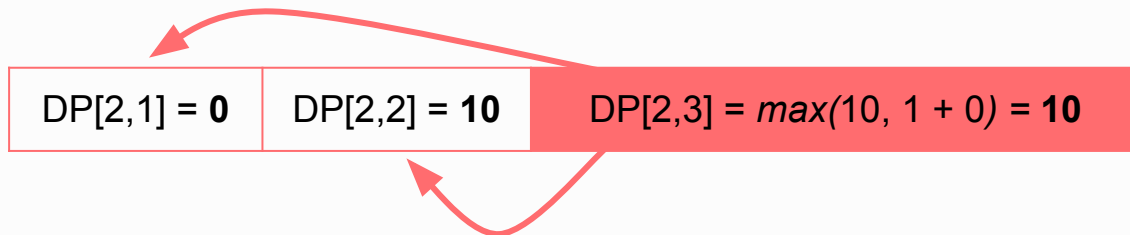
# CANDY - SOLUCIÓN

```
def calculate_max_candies(boxes):  
    N, M = len(boxes), len(boxes[0])  
  
    DP = [[0 for _ in range(M + 3)] for _ in range(N + 2)]  
  
    for row, box in enumerate(boxes, start=2):  
        for col, candies in enumerate(box, start=2):  
            DP[row][col] = max(  
                # Alternative 1: Don't take the candy  
                DP[row][col - 1],  
                # Alternative 2: Take the candy  
                candies + DP[row][col - 2],  
            )  
    ...
```

Boxes

10	1	1
1	20	1
1	1	30

**Alternativa 2: 1 + 0**



**Alternativa 1: 10**



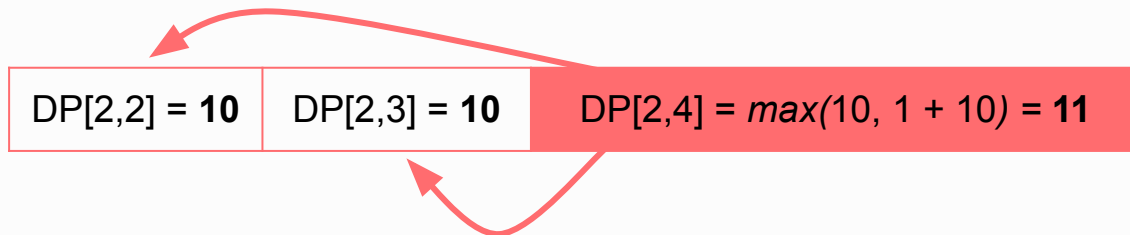
# CANDY - SOLUCIÓN

```
def calculate_max_candies(boxes):  
    N, M = len(boxes), len(boxes[0])  
  
    DP = [[0 for _ in range(M + 3)] for _ in range(N + 2)]  
  
    for row, box in enumerate(boxes, start=2):  
        for col, candies in enumerate(box, start=2):  
            DP[row][col] = max(  
                # Alternative 1: Don't take the candy  
                DP[row][col - 1],  
                # Alternative 2: Take the candy  
                candies + DP[row][col - 2],  
            )  
    ...
```

Boxes

10	1	1
1	20	1
1	1	30

**Alternativa 2: 1 + 10**



**Alternativa 1: 10**



# CANDY - SOLUCIÓN

```
def calculate_max_candies(boxes):  
    ...  
    for row, box in enumerate(boxes, start=2):  
        for col, candies in enumerate(box, start=2):  
            ...  
  
            DP[row][-1] = max(  
                # Alternative 1: Don't take the row  
                DP[row - 1][-1],  
                # Alternative 2: Take the row  
                DP[row][-2] + DP[row - 2][-1],  
            )  
  
    return DP[-1][-1]
```

Boxes

10	1	1
1	20	1
1	1	30

DP

		0	0	0	0
		0	0	0	0
0	0	10	10	11	$\max(0, 11) = 11$

Alternativa 2:  $11 + 0$

Alternativa 1: 0





# CANDY - SOLUCIONES ALTERNATIVAS

- ➡ **Solución Top-Down:** *solución recursiva*
  - ➡ Matrices → Recorrer las matrices resulta más común hacerlo iterativamente.
  - ➡ *Memoization*.
  - ➡ Múltiples llamadas a funciones.
- ➡ **Fuerza bruta:** *sin utilizar programación dinámica*
  - ➡ La complejidad aumenta → Tener en cuenta que  $1 \leq M \times N \leq 10^5$ .
- ➡ **Reducir la complejidad espacial a de  $N + 2 \times M + 3$  a  $N \times M$ :**
  - ➡ Complejiza el algoritmo y la explicación
  - ➡ Mayor esfuerzo de programación

## My Submissions

#	Problem	Verdict	Language	Run Time	Submission Date
26956229	12146 Candy	Accepted	PYTH3	0.410	2021-11-10 14:02:04





# CASOS DE PRUEBA

# 1º CASO DE PRUEBA - CASO INICIAL

```
for row, box in enumerate(boxes, start=2):
    for col, candies in enumerate(box, start=2):
        DP[row][col] = max(
            # Alternative 1: Don't take the candy
            DP[row][col - 1],
            # Alternative 2: Take the candy
            candies + DP[row][col - 2],
        )

DP[row][-1] = max(
    # Alternative 1: Don't take the row
    DP[row - 1][-1],
    # Alternative 2: Take the row
    DP[row][-2] + DP[row - 2][-1],
)
```

Boxes

10	1	1
1	20	1
1	1	30

DP

		0	0	0	0
		0	0	0	0
0	0	10	?	?	0
0	0	?	?	?	0
0	0	?	?	?	0

$$DP[2,2] = \max(DP[2,1], 10 + DP[2,0]) = \max(0, 10 + 0) = 10$$



# 1º CASO DE PRUEBA - CASO INICIAL

```
for row, box in enumerate(boxes, start=2):
    for col, candies in enumerate(box, start=2):
        DP[row][col] = max(
            # Alternative 1: Don't take the candy
            DP[row][col - 1],
            # Alternative 2: Take the candy
            candies + DP[row][col - 2],
        )

DP[row][-1] = max(
    # Alternative 1: Don't take the row
    DP[row - 1][-1],
    # Alternative 2: Take the row
    DP[row][-2] + DP[row - 2][-1],
)
```

Boxes

10	1	1
1	20	1
1	1	30

DP

		0	0	0	0
		0	0	0	0
0	0	10	10	?	0
0	0	?	?	?	0
0	0	?	?	?	0

$$DP[2,3] = \max(DP[2,2], 10 + DP[2,1]) = \max(10, 1 + 0) = 10$$



# 1º CASO DE PRUEBA - CASO INICIAL

```
for row, box in enumerate(boxes, start=2):
    for col, candies in enumerate(box, start=2):
        DP[row][col] = max(
            # Alternative 1: Don't take the candy
            DP[row][col - 1],
            # Alternative 2: Take the candy
            candies + DP[row][col - 2],
        )

DP[row][-1] = max(
    # Alternative 1: Don't take the row
    DP[row - 1][-1],
    # Alternative 2: Take the row
    DP[row][-2] + DP[row - 2][-1],
)
```

Boxes

10	1	1
1	20	1
1	1	30

DP

		0	0	0	0
		0	0	0	0
0	0	10	10	11	0
0	0	?	?	?	0
0	0	?	?	?	0

$$DP[2,4] = \max(DP[2,3], 10 + DP[2,2]) = \max(10, 1 + 10) = 11$$



# 1º CASO DE PRUEBA - CASO INICIAL

```
for row, box in enumerate(boxes, start=2):
    for col, candies in enumerate(box, start=2):
        DP[row][col] = max(
            # Alternative 1: Don't take the candy
            DP[row][col - 1],
            # Alternative 2: Take the candy
            candies + DP[row][col - 2],
        )

DP[row][-1] = max(
    # Alternative 1: Don't take the row
    DP[row - 1][-1],
    # Alternative 2: Take the row
    DP[row][-2] + DP[row - 2][-1],
)
```

Boxes

10	1	1
1	20	1
1	1	30

DP

		0	0	0	0
		0	0	0	0
0	0	10	10	11	11
0	0	?	?	?	0
0	0	?	?	?	0

$$DP[2,-1] = \max(DP[1,-1], 11 + DP[0,-1]) = \max(0, 11 + 0) = 11$$



# 1º CASO DE PRUEBA - CASO INICIAL

```
for row, box in enumerate(boxes, start=2):
    for col, candies in enumerate(box, start=2):
        DP[row][col] = max(
            # Alternative 1: Don't take the candy
            DP[row][col - 1],
            # Alternative 2: Take the candy
            candies + DP[row][col - 2],
        )

DP[row][-1] = max(
    # Alternative 1: Don't take the row
    DP[row - 1][-1],
    # Alternative 2: Take the row
    DP[row][-2] + DP[row - 2][-1],
)
```

Boxes

10	1	1
1	20	1
1	1	30

DP

		0	0	0	0
		0	0	0	0
0	0	10	10	11	11
0	0	1	?	?	0
0	0	?	?	?	0

$$DP[3,2] = \max(DP[3,1], 1 + DP[3,0]) = \max(0, 1 + 0) = 1$$



# 1º CASO DE PRUEBA - CASO INICIAL

```
for row, box in enumerate(boxes, start=2):
    for col, candies in enumerate(box, start=2):
        DP[row][col] = max(
            # Alternative 1: Don't take the candy
            DP[row][col - 1],
            # Alternative 2: Take the candy
            candies + DP[row][col - 2],
        )

DP[row][-1] = max(
    # Alternative 1: Don't take the row
    DP[row - 1][-1],
    # Alternative 2: Take the row
    DP[row][-2] + DP[row - 2][-1],
)
```

Boxes

10	1	1
1	20	1
1	1	30

DP

		0	0	0	0
		0	0	0	0
0	0	10	10	11	11
0	0	1	20	?	0
0	0	?	?	?	0

$$DP[3,3] = \max(DP[3,2], 20 + DP[3,1]) = \max(1, 20 + 0) = 20$$





# 1º CASO DE PRUEBA - CASO INICIAL

```
for row, box in enumerate(boxes, start=2):
    for col, candies in enumerate(box, start=2):
        DP[row][col] = max(
            # Alternative 1: Don't take the candy
            DP[row][col - 1],
            # Alternative 2: Take the candy
            candies + DP[row][col - 2],
        )

DP[row][-1] = max(
    # Alternative 1: Don't take the row
    DP[row - 1][-1],
    # Alternative 2: Take the row
    DP[row][-2] + DP[row - 2][-1],
)
```

Boxes

10	1	1
1	20	1
1	1	30

DP

		0	0	0	0
		0	0	0	0
0	0	10	10	11	11
0	0	1	20	20	0
0	0	?	?	?	0

$$DP[3,4] = \max(DP[3,3], 1 + DP[3,2]) = \max(20, 1 + 1) = 20$$



# 1º CASO DE PRUEBA - CASO INICIAL

```
for row, box in enumerate(boxes, start=2):
    for col, candies in enumerate(box, start=2):
        DP[row][col] = max(
            # Alternative 1: Don't take the candy
            DP[row][col - 1],
            # Alternative 2: Take the candy
            candies + DP[row][col - 2],
        )

DP[row][-1] = max(
    # Alternative 1: Don't take the row
    DP[row - 1][-1],
    # Alternative 2: Take the row
    DP[row][-2] + DP[row - 2][-1],
)
```

Boxes

10	1	1
1	20	1
1	1	30

DP

		0	0	0	0
		0	0	0	0
0	0	10	10	11	11
0	0	1	20	20	20
0	0	?	?	?	0

$$DP[3,-1] = \max(DP[2,-1], 20 + DP[1,-1]) = \max(11, 20 + 0) = 20$$



# 1º CASO DE PRUEBA - CASO INICIAL

```
for row, box in enumerate(boxes, start=2):
    for col, candies in enumerate(box, start=2):
        DP[row][col] = max(
            # Alternative 1: Don't take the candy
            DP[row][col - 1],
            # Alternative 2: Take the candy
            candies + DP[row][col - 2],
        )

DP[row][-1] = max(
    # Alternative 1: Don't take the row
    DP[row - 1][-1],
    # Alternative 2: Take the row
    DP[row][-2] + DP[row - 2][-1],
)
```

Boxes

10	1	1
1	20	1
1	1	30

DP

		0	0	0	0
		0	0	0	0
0	0	10	10	11	11
0	0	1	20	20	20
0	0	1	?	?	0

$$DP[4,2] = \max(DP[4,1], 1 + DP[4,0]) = \max(0, 1 + 0) = 1$$



# 1º CASO DE PRUEBA - CASO INICIAL

```
for row, box in enumerate(boxes, start=2):
    for col, candies in enumerate(box, start=2):
        DP[row][col] = max(
            # Alternative 1: Don't take the candy
            DP[row][col - 1],
            # Alternative 2: Take the candy
            candies + DP[row][col - 2],
        )

DP[row][-1] = max(
    # Alternative 1: Don't take the row
    DP[row - 1][-1],
    # Alternative 2: Take the row
    DP[row][-2] + DP[row - 2][-1],
)
```

Boxes

10	1	1
1	20	1
1	1	30

DP

		0	0	0	0
		0	0	0	0
0	0	10	10	11	11
0	0	1	20	20	20
0	0	1	1	?	0

$$DP[4,3] = \max(DP[4,2], 1 + DP[4,1]) = \max(1, 1 + 0) = 1$$



# 1º CASO DE PRUEBA - CASO INICIAL

```
for row, box in enumerate(boxes, start=2):
    for col, candies in enumerate(box, start=2):
        DP[row][col] = max(
            # Alternative 1: Don't take the candy
            DP[row][col - 1],
            # Alternative 2: Take the candy
            candies + DP[row][col - 2],
        )

DP[row][-1] = max(
    # Alternative 1: Don't take the row
    DP[row - 1][-1],
    # Alternative 2: Take the row
    DP[row][-2] + DP[row - 2][-1],
)
```

Boxes

10	1	1
1	20	1
1	1	30

DP

		0	0	0	0
		0	0	0	0
0	0	10	10	11	11
0	0	1	20	20	20
0	0	1	1	31	0

$$DP[4,4] = \max(DP[4,3], 30 + DP[4,2]) = \max(1, 30 + 1) = 31$$



# 1º CASO DE PRUEBA - CASO INICIAL

```
for row, box in enumerate(boxes, start=2):
    for col, candies in enumerate(box, start=2):
        DP[row][col] = max(
            # Alternative 1: Don't take the candy
            DP[row][col - 1],
            # Alternative 2: Take the candy
            candies + DP[row][col - 2],
        )

DP[row][-1] = max(
    # Alternative 1: Don't take the row
    DP[row - 1][-1],
    # Alternative 2: Take the row
    DP[row][-2] + DP[row - 2][-1],
)
```

Boxes

10	1	1
1	20	1
1	1	30

DP

		0	0	0	0
		0	0	0	0
0	0	10	10	11	11
0	0	1	20	20	20
0	0	1	1	31	42

Resultado final

$$DP[4,-1] = \max(DP[3,-1], 31 + DP[2,-1]) = \max(20, 31 + 11) = 42$$



## 2º CASO DE PRUEBA - CASO CON SOLO 1 COLUMNA

```
for row, box in enumerate(boxes, start=2):
    for col, candies in enumerate(box, start=2):
        DP[row][col] = max(
            # Alternative 1: Don't take the candy
            DP[row][col - 1],
            # Alternative 2: Take the candy
            candies + DP[row][col - 2],
        )
    DP[row][-1] = max(
        # Alternative 1: Don't take the row
        DP[row - 1][-1],
        # Alternative 2: Take the row
        DP[row][-2] + DP[row - 2][-1],
    )
```

Boxes

5
8
6
10
1

$$DP[2,2] = \max(DP[2,1], 5 + DP[2,0]) = \max(0, 5 + 0) = 5$$

DP

		0	0
		0	0
0	0	5	0
0	0	?	0
0	0	?	0
0	0	?	0
0	0	?	0



## 2º CASO DE PRUEBA - CASO CON SOLO 1 COLUMNA

```
for row, box in enumerate(boxes, start=2):
    for col, candies in enumerate(box, start=2):
        DP[row][col] = max(
            # Alternative 1: Don't take the candy
            DP[row][col - 1],
            # Alternative 2: Take the candy
            candies + DP[row][col - 2],
        )
    DP[row][-1] = max(
        # Alternative 1: Don't take the row
        DP[row - 1][-1],
        # Alternative 2: Take the row
        DP[row][-2] + DP[row - 2][-1],
    )
```

Boxes

5
8
6
10
1

$$DP[2,-1] = \max(DP[1,-1], 5 + DP[0,-1]) = \max(0, 5 + 0) = 5$$

DP

		0	0
		0	0
0	0	5	5
0	0	?	0
0	0	?	0
0	0	?	0
0	0	?	0





## 2º CASO DE PRUEBA - CASO CON SOLO 1 COLUMNA

```
for row, box in enumerate(boxes, start=2):
    for col, candies in enumerate(box, start=2):
        DP[row][col] = max(
            # Alternative 1: Don't take the candy
            DP[row][col - 1],
            # Alternative 2: Take the candy
            candies + DP[row][col - 2],
        )
    DP[row][-1] = max(
        # Alternative 1: Don't take the row
        DP[row - 1][-1],
        # Alternative 2: Take the row
        DP[row][-2] + DP[row - 2][-1],
    )
```

Boxes

5
8
6
10
1

$$DP[3,2] = \max(DP[3,1], 8 + DP[3,0]) = \max(0, 8 + 0) = 8$$

DP

		0	0
		0	0
0	0	5	5
0	0	8	0
0	0	?	0
0	0	?	0
0	0	?	0



## 2º CASO DE PRUEBA - CASO CON SOLO 1 COLUMNA

```
for row, box in enumerate(boxes, start=2):
    for col, candies in enumerate(box, start=2):
        DP[row][col] = max(
            # Alternative 1: Don't take the candy
            DP[row][col - 1],
            # Alternative 2: Take the candy
            candies + DP[row][col - 2],
        )
    DP[row][-1] = max(
        # Alternative 1: Don't take the row
        DP[row - 1][-1],
        # Alternative 2: Take the row
        DP[row][-2] + DP[row - 2][-1],
    )
```

Boxes

5
8
6
10
1

$$DP[3,-1] = \max(DP[2,-1], 8 + DP[1,-1]) = \max(5, 8 + 0) = 8$$

DP

		0	0
		0	0
0	0	5	5
0	0	8	8
0	0	?	0
0	0	?	0
0	0	?	0



## 2º CASO DE PRUEBA - CASO CON SOLO 1 COLUMNA

```
for row, box in enumerate(boxes, start=2):
    for col, candies in enumerate(box, start=2):
        DP[row][col] = max(
            # Alternative 1: Don't take the candy
            DP[row][col - 1],
            # Alternative 2: Take the candy
            candies + DP[row][col - 2],
        )
    DP[row][-1] = max(
        # Alternative 1: Don't take the row
        DP[row - 1][-1],
        # Alternative 2: Take the row
        DP[row][-2] + DP[row - 2][-1],
    )
```

Boxes

5
8
6
10
1

$$DP[4,2] = \max(DP[4,1], 6 + DP[4,0]) = \max(0, 6 + 0) = 6$$

DP

		0	0
		0	0
0	0	5	5
0	0	8	8
0	0	6	0
0	0	?	0
0	0	?	0



## 2º CASO DE PRUEBA - CASO CON SOLO 1 COLUMNA

```
for row, box in enumerate(boxes, start=2):
    for col, candies in enumerate(box, start=2):
        DP[row][col] = max(
            # Alternative 1: Don't take the candy
            DP[row][col - 1],
            # Alternative 2: Take the candy
            candies + DP[row][col - 2],
        )
    DP[row][-1] = max(
        # Alternative 1: Don't take the row
        DP[row - 1][-1],
        # Alternative 2: Take the row
        DP[row][-2] + DP[row - 2][-1],
    )
```

Boxes

5
8
6
10
1

$$DP[4,-1] = \max(DP[3,-1], 6 + DP[2,-1]) = \max(8, 6 + 5) = 11$$

DP

		0	0
		0	0
0	0	5	5
0	0	8	8
0	0	6	11
0	0	?	0
0	0	?	0



## 2º CASO DE PRUEBA - CASO CON SOLO 1 COLUMNA

```
for row, box in enumerate(boxes, start=2):
    for col, candies in enumerate(box, start=2):
        DP[row][col] = max(
            # Alternative 1: Don't take the candy
            DP[row][col - 1],
            # Alternative 2: Take the candy
            candies + DP[row][col - 2],
        )
    DP[row][-1] = max(
        # Alternative 1: Don't take the row
        DP[row - 1][-1],
        # Alternative 2: Take the row
        DP[row][-2] + DP[row - 2][-1],
    )
```

Boxes

5
8
6
10
1

$$DP[5,2] = \max(DP[5,1], 10 + DP[5,0]) = \max(0, 10 + 0) = 10$$

DP

		0	0
		0	0
0	0	5	5
0	0	8	8
0	0	6	11
0	0	10	0
0	0	?	0



## 2º CASO DE PRUEBA - CASO CON SOLO 1 COLUMNA

```
for row, box in enumerate(boxes, start=2):
    for col, candies in enumerate(box, start=2):
        DP[row][col] = max(
            # Alternative 1: Don't take the candy
            DP[row][col - 1],
            # Alternative 2: Take the candy
            candies + DP[row][col - 2],
        )
    DP[row][-1] = max(
        # Alternative 1: Don't take the row
        DP[row - 1][-1],
        # Alternative 2: Take the row
        DP[row][-2] + DP[row - 2][-1],
    )
```

Boxes

5
8
6
10
1

$$DP[5,-1] = \max(DP[4,-1], 10 + DP[3,-1]) = \max(11, 10 + 8) = 18$$

DP

		0	0
		0	0
0	0	5	5
0	0	8	8
0	0	6	11
0	0	10	18
0	0	?	0



## 2º CASO DE PRUEBA - CASO CON SOLO 1 COLUMNA

```
for row, box in enumerate(boxes, start=2):
    for col, candies in enumerate(box, start=2):
        DP[row][col] = max(
            # Alternative 1: Don't take the candy
            DP[row][col - 1],
            # Alternative 2: Take the candy
            candies + DP[row][col - 2],
        )
    DP[row][-1] = max(
        # Alternative 1: Don't take the row
        DP[row - 1][-1],
        # Alternative 2: Take the row
        DP[row][-2] + DP[row - 2][-1],
    )
```

Boxes

5
8
6
10
1

$$DP[6,2] = \max(DP[6,1], 1 + DP[6,0]) = \max(0, 1 + 0) = 1$$

DP

		0	0
		0	0
0	0	5	5
0	0	8	8
0	0	6	11
0	0	10	18
0	0	1	0



## 2º CASO DE PRUEBA - CASO CON SOLO 1 COLUMNA

```
for row, box in enumerate(boxes, start=2):
    for col, candies in enumerate(box, start=2):
        DP[row][col] = max(
            # Alternative 1: Don't take the candy
            DP[row][col - 1],
            # Alternative 2: Take the candy
            candies + DP[row][col - 2],
        )
    DP[row][-1] = max(
        # Alternative 1: Don't take the row
        DP[row - 1][-1],
        # Alternative 2: Take the row
        DP[row][-2] + DP[row - 2][-1],
    )
```

Boxes

5
8
6
10
1

$$DP[6,-1] = \max(DP[5,-1], 1 + DP[4,-1]) = \max(18, 1 + 11) = 18$$

DP

		0	0
		0	0
0	0	5	5
0	0	8	8
0	0	6	11
0	0	10	18
0	0	1	18

Resultado final





### 3º CASO DE PRUEBA - CASO CON SOLO 1 FILA

```
for row, box in enumerate(boxes, start=2):
    for col, candies in enumerate(box, start=2):
        DP[row][col] = max(
            # Alternative 1: Don't take the candy
            DP[row][col - 1],
            # Alternative 2: Take the candy
            candies + DP[row][col - 2],
        )

DP[row][-1] = max(
    # Alternative 1: Don't take the row
    DP[row - 1][-1],
    # Alternative 2: Take the row
    DP[row - 2] + DP[row - 2][-1],
)
```

Boxes

5	8	6	10	1
---	---	---	----	---

$$DP[2,2] = \max(DP[2,1], 5 + DP[2,0]) = \max(0, 5 + 0) = 5$$

DP

		0	0	0	0	0	0
		0	0	0	0	0	0
0	0	5	?	?	?	?	0

### 3º CASO DE PRUEBA - CASO CON SOLO 1 FILA

```
for row, box in enumerate(boxes, start=2):
    for col, candies in enumerate(box, start=2):
        DP[row][col] = max(
            # Alternative 1: Don't take the candy
            DP[row][col - 1],
            # Alternative 2: Take the candy
            candies + DP[row][col - 2],
        )

DP[row][-1] = max(
    # Alternative 1: Don't take the row
    DP[row - 1][-1],
    # Alternative 2: Take the row
    DP[row][-2] + DP[row - 2][-1],
)
```

Boxes

5	8	6	10	1
---	---	---	----	---

$$DP[2,3] = \max(DP[2,2], 8 + DP[2,1]) = \max(5, 8 + 0) = 8$$

DP

		0	0	0	0	0	0
		0	0	0	0	0	0
0	0	5	8	?	?	?	0

### 3º CASO DE PRUEBA - CASO CON SOLO 1 FILA

```
for row, box in enumerate(boxes, start=2):
    for col, candies in enumerate(box, start=2):
        DP[row][col] = max(
            # Alternative 1: Don't take the candy
            DP[row][col - 1],
            # Alternative 2: Take the candy
            candies + DP[row][col - 2],
        )
    DP[row][-1] = max(
        # Alternative 1: Don't take the row
        DP[row - 1][-1],
        # Alternative 2: Take the row
        DP[row - 2] + DP[row - 2][-1],
    )
```

Boxes

5	8	6	10	1
---	---	---	----	---

$$DP[2,4] = \max(DP[2,3], 6 + DP[2,2]) = \max(8, 6 + 5) = 11$$

DP

		0	0	0	0	0	0
		0	0	0	0	0	0
0	0	5	8	11	?	?	0

### 3º CASO DE PRUEBA - CASO CON SOLO 1 FILA

```
for row, box in enumerate(boxes, start=2):
    for col, candies in enumerate(box, start=2):
        DP[row][col] = max(
            # Alternative 1: Don't take the candy
            DP[row][col - 1],
            # Alternative 2: Take the candy
            candies + DP[row][col - 2],
        )
    DP[row][-1] = max(
        # Alternative 1: Don't take the row
        DP[row - 1][-1],
        # Alternative 2: Take the row
        DP[row - 2] + DP[row - 2][-1],
    )
```

Boxes

5	8	6	10	1
---	---	---	----	---

$$DP[2,5] = \max(DP[2,4], 10 + DP[2,3]) = \max(6, 10 + 8) = 18$$

DP

		0	0	0	0	0	0
		0	0	0	0	0	0
0	0	5	8	11	18	?	0

### 3º CASO DE PRUEBA - CASO CON SOLO 1 FILA

```
for row, box in enumerate(boxes, start=2):
    for col, candies in enumerate(box, start=2):
        DP[row][col] = max(
            # Alternative 1: Don't take the candy
            DP[row][col - 1],
            # Alternative 2: Take the candy
            candies + DP[row][col - 2],
        )
    DP[row][-1] = max(
        # Alternative 1: Don't take the row
        DP[row - 1][-1],
        # Alternative 2: Take the row
        DP[row - 2] + DP[row - 2][-1],
    )
```

Boxes

5	8	6	10	1
---	---	---	----	---

$$DP[2,6] = \max(DP[2,5], 1 + DP[2,4]) = \max(18, 1 + 11) = 18$$

DP

		0	0	0	0	0	0
		0	0	0	0	0	0
0	0	5	8	11	18	18	0

### 3º CASO DE PRUEBA - CASO CON SOLO 1 FILA

```
for row, box in enumerate(boxes, start=2):
    for col, candies in enumerate(box, start=2):
        DP[row][col] = max(
            # Alternative 1: Don't take the candy
            DP[row][col - 1],
            # Alternative 2: Take the candy
            candies + DP[row][col - 2],
        )

DP[row][-1] = max(
    # Alternative 1: Don't take the row
    DP[row - 1][-1],
    # Alternative 2: Take the row
    DP[row][-2] + DP[row - 2][-1],
)
```

Boxes

5	8	6	10	1
---	---	---	----	---

$$DP[2,-1] = \max(DP[1,-1], 18 + DP[0,-1]) = \max(0, 18 + 0) = 18$$

DP

		0	0	0	0	0	0
		0	0	0	0	0	0
0	0	5	8	11	18	18	18

Resultado final

## 4º CASO DE PRUEBA - CASO MÁS COMPLEJO

```
for row, box in enumerate(boxes, start=2):
    for col, candies in enumerate(box, start=2):
        DP[row][col] = max(
            # Alternative 1: Don't take the candy
            DP[row][col - 1],
            # Alternative 2: Take the candy
            candies + DP[row][col - 2],
        )

DP[row][-1] = max(
    # Alternative 1: Don't take the row
    DP[row - 1][-1],
    # Alternative 2: Take the row
    DP[row][-2] + DP[row - 2][-1],
)
```

Boxes

1	8	2	1	9
1	7	3	5	2
1	2	10	3	10
8	4	7	9	1
7	1	3	1	6

$$DP[2,2] = \max(DP[2,1], 1 + DP[2,0]) = \max(0, 1 + 0) = 1$$

DP

		0	0	0	0	0	0
		0	0	0	0	0	0
0	0	1	?	?	?	?	0
0	0	?	?	?	?	?	0
0	0	?	?	?	?	?	0
0	0	?	?	?	?	?	0
0	0	?	?	?	?	?	0

## 4º CASO DE PRUEBA - CASO MÁS COMPLEJO

```
for row, box in enumerate(boxes, start=2):
    for col, candies in enumerate(box, start=2):
        DP[row][col] = max(
            # Alternative 1: Don't take the candy
            DP[row][col - 1],
            # Alternative 2: Take the candy
            candies + DP[row][col - 2],
        )

DP[row][-1] = max(
    # Alternative 1: Don't take the row
    DP[row - 1][-1],
    # Alternative 2: Take the row
    DP[row][-2] + DP[row - 2][-1],
)
```

Boxes

1	8	2	1	9
1	7	3	5	2
1	2	10	3	10
8	4	7	9	1
7	1	3	1	6

$$DP[2,3] = \max(DP[2,2], 8 + DP[2,1]) = \max(1, 8 + 0) = 8$$

DP

		0	0	0	0	0	0
		0	0	0	0	0	0
0	0	1	8	?	?	?	0
0	0	?	?	?	?	?	0
0	0	?	?	?	?	?	0
0	0	?	?	?	?	?	0
0	0	?	?	?	?	?	0



## 4º CASO DE PRUEBA - CASO MÁS COMPLEJO

```
for row, box in enumerate(boxes, start=2):
    for col, candies in enumerate(box, start=2):
        DP[row][col] = max(
            # Alternative 1: Don't take the candy
            DP[row][col - 1],
            # Alternative 2: Take the candy
            candies + DP[row][col - 2],
        )

DP[row][-1] = max(
    # Alternative 1: Don't take the row
    DP[row - 1][-1],
    # Alternative 2: Take the row
    DP[row][-2] + DP[row - 2][-1],
)
```

Boxes

1	8	2	1	9
1	7	3	5	2
1	2	10	3	10
8	4	7	9	1
7	1	3	1	6

$$DP[2,4] = \max(DP[2,3], 2 + DP[2,2]) = \max(8, 2 + 1) = 8$$

DP

		0	0	0	0	0	0
		0	0	0	0	0	0
0	0	1	8	8	?	?	0
0	0	?	?	?	?	?	0
0	0	?	?	?	?	?	0
0	0	?	?	?	?	?	0
0	0	?	?	?	?	?	0

## 4º CASO DE PRUEBA - CASO MÁS COMPLEJO

```
for row, box in enumerate(boxes, start=2):
    for col, candies in enumerate(box, start=2):
        DP[row][col] = max(
            # Alternative 1: Don't take the candy
            DP[row][col - 1],
            # Alternative 2: Take the candy
            candies + DP[row][col - 2],
        )

DP[row][-1] = max(
    # Alternative 1: Don't take the row
    DP[row - 1][-1],
    # Alternative 2: Take the row
    DP[row][-2] + DP[row - 2][-1],
)
```

Boxes

1	8	2	1	9
1	7	3	5	2
1	2	10	3	10
8	4	7	9	1
7	1	3	1	6

$$DP[2,5] = \max(DP[2,4], 1 + DP[2,3]) = \max(8, 1 + 8) = 9$$

DP

		0	0	0	0	0	0
		0	0	0	0	0	0
0	0	1	8	8	9	?	0
0	0	?	?	?	?	?	0
0	0	?	?	?	?	?	0
0	0	?	?	?	?	?	0
0	0	?	?	?	?	?	0

## 4º CASO DE PRUEBA - CASO MÁS COMPLEJO

```
for row, box in enumerate(boxes, start=2):
    for col, candies in enumerate(box, start=2):
        DP[row][col] = max(
            # Alternative 1: Don't take the candy
            DP[row][col - 1],
            # Alternative 2: Take the candy
            candies + DP[row][col - 2],
        )
    DP[row][-1] = max(
        # Alternative 1: Don't take the row
        DP[row - 1][-1],
        # Alternative 2: Take the row
        DP[row][-2] + DP[row - 2][-1],
    )
```

Boxes

1	8	2	1	9
1	7	3	5	2
1	2	10	3	10
8	4	7	9	1
7	1	3	1	6

$$DP[2,6] = \max(DP[2,5], 9 + DP[2,4]) = \max(8, 9 + 8) = 17$$

DP

		0	0	0	0	0	0
		0	0	0	0	0	0
0	0	1	8	8	9	17	0
0	0	?	?	?	?	?	0
0	0	?	?	?	?	?	0
0	0	?	?	?	?	?	0
0	0	?	?	?	?	?	0

## 4º CASO DE PRUEBA - CASO MÁS COMPLEJO

```
for row, box in enumerate(boxes, start=2):
    for col, candies in enumerate(box, start=2):
        DP[row][col] = max(
            # Alternative 1: Don't take the candy
            DP[row][col - 1],
            # Alternative 2: Take the candy
            candies + DP[row][col - 2],
        )

DP[row][-1] = max(
    # Alternative 1: Don't take the row
    DP[row - 1][-1],
    # Alternative 2: Take the row
    DP[row][-2] + DP[row - 2][-1],
)
```

Boxes

1	8	2	1	9
1	7	3	5	2
1	2	10	3	10
8	4	7	9	1
7	1	3	1	6

$$DP[2,-1] = \max(DP[1,-1], 17 + DP[0,-1]) = \max(0, 17 + 0) = 17$$

DP

		0	0	0	0	0	0
		0	0	0	0	0	0
0	0	1	8	8	9	17	17
0	0	?	?	?	?	?	0
0	0	?	?	?	?	?	0
0	0	?	?	?	?	?	0
0	0	?	?	?	?	?	0

## 4º CASO DE PRUEBA - CASO MÁS COMPLEJO

```
for row, box in enumerate(boxes, start=2):
    for col, candies in enumerate(box, start=2):
        DP[row][col] = max(
            # Alternative 1: Don't take the candy
            DP[row][col - 1],
            # Alternative 2: Take the candy
            candies + DP[row][col - 2],
        )
    DP[row][-1] = max(
        # Alternative 1: Don't take the row
        DP[row - 1][-1],
        # Alternative 2: Take the row
        DP[row][-2] + DP[row - 2][-1],
    )
```

Boxes

1	8	2	1	9
1	7	3	5	2
1	2	10	3	10
8	4	7	9	1
7	1	3	1	6

$$DP[3,2] = \max(DP[3,1], 1 + DP[3,0]) = \max(0, 1 + 0) = 1$$

DP

		0	0	0	0	0	0
		0	0	0	0	0	0
0	0	1	8	8	9	17	17
0	0	1	?	?	?	?	0
0	0	?	?	?	?	?	0
0	0	?	?	?	?	?	0
0	0	?	?	?	?	?	0

## 4º CASO DE PRUEBA - CASO MÁS COMPLEJO

```
for row, box in enumerate(boxes, start=2):
    for col, candies in enumerate(box, start=2):
        DP[row][col] = max(
            # Alternative 1: Don't take the candy
            DP[row][col - 1],
            # Alternative 2: Take the candy
            candies + DP[row][col - 2],
        )
    DP[row][-1] = max(
        # Alternative 1: Don't take the row
        DP[row - 1][-1],
        # Alternative 2: Take the row
        DP[row][-2] + DP[row - 2][-1],
    )
```

Boxes

1	8	2	1	9
1	7	3	5	2
1	2	10	3	10
8	4	7	9	1
7	1	3	1	6

$$DP[3,3] = \max(DP[3,2], 7 + DP[3,1]) = \max(1, 7 + 0) = 7$$

DP

		0	0	0	0	0	0
		0	0	0	0	0	0
0	0	1	8	8	9	17	17
0	0	1	7	?	?	?	0
0	0	?	?	?	?	?	0
0	0	?	?	?	?	?	0
0	0	?	?	?	?	?	0

## 4º CASO DE PRUEBA - CASO MÁS COMPLEJO

```
for row, box in enumerate(boxes, start=2):
    for col, candies in enumerate(box, start=2):
        DP[row][col] = max(
            # Alternative 1: Don't take the candy
            DP[row][col - 1],
            # Alternative 2: Take the candy
            candies + DP[row][col - 2],
        )
    DP[row][-1] = max(
        # Alternative 1: Don't take the row
        DP[row - 1][-1],
        # Alternative 2: Take the row
        DP[row][-2] + DP[row - 2][-1],
    )
```

Boxes

1	8	2	1	9
1	7	3	5	2
1	2	10	3	10
8	4	7	9	1
7	1	3	1	6

$$DP[3,4] = \max(DP[3,3], 3 + DP[3,2]) = \max(7, 3 + 1) = 7$$

DP

		0	0	0	0	0	0
		0	0	0	0	0	0
0	0	1	8	8	9	17	17
0	0	1	7	7	?	?	0
0	0	?	?	?	?	?	0
0	0	?	?	?	?	?	0
0	0	?	?	?	?	?	0

## 4º CASO DE PRUEBA - CASO MÁS COMPLEJO

```
for row, box in enumerate(boxes, start=2):
    for col, candies in enumerate(box, start=2):
        DP[row][col] = max(
            # Alternative 1: Don't take the candy
            DP[row][col - 1],
            # Alternative 2: Take the candy
            candies + DP[row][col - 2],
        )

DP[row][-1] = max(
    # Alternative 1: Don't take the row
    DP[row - 1][-1],
    # Alternative 2: Take the row
    DP[row][-2] + DP[row - 2][-1],
)
```

Boxes

1	8	2	1	9
1	7	3	5	2
1	2	10	3	10
8	4	7	9	1
7	1	3	1	6

$$DP[3,5] = \max(DP[3,4], 5 + DP[3,3]) = \max(7, 5 + 7) = 12$$

DP

		0	0	0	0	0	0
		0	0	0	0	0	0
0	0	1	8	8	9	17	17
0	0	1	7	7	12	?	0
0	0	?	?	?	?	?	0
0	0	?	?	?	?	?	0
0	0	?	?	?	?	?	0



## 4º CASO DE PRUEBA - CASO MÁS COMPLEJO

```
for row, box in enumerate(boxes, start=2):
    for col, candies in enumerate(box, start=2):
        DP[row][col] = max(
            # Alternative 1: Don't take the candy
            DP[row][col - 1],
            # Alternative 2: Take the candy
            candies + DP[row][col - 2],
        )

DP[row][-1] = max(
    # Alternative 1: Don't take the row
    DP[row - 1][-1],
    # Alternative 2: Take the row
    DP[row][-2] + DP[row - 2][-1],
)
```

Boxes

1	8	2	1	9
1	7	3	5	2
1	2	10	3	10
8	4	7	9	1
7	1	3	1	6

$$DP[3,6] = \max(DP[3,5], 2 + DP[3,4]) = \max(12, 2 + 7) = 12$$

DP

		0	0	0	0	0	0
		0	0	0	0	0	0
0	0	1	8	8	9	17	17
0	0	1	7	7	12	12	0
0	0	?	?	?	?	?	0
0	0	?	?	?	?	?	0
0	0	?	?	?	?	?	0

## 4º CASO DE PRUEBA - CASO MÁS COMPLEJO

```
for row, box in enumerate(boxes, start=2):
    for col, candies in enumerate(box, start=2):
        DP[row][col] = max(
            # Alternative 1: Don't take the candy
            DP[row][col - 1],
            # Alternative 2: Take the candy
            candies + DP[row][col - 2],
        )

DP[row][-1] = max(
    # Alternative 1: Don't take the row
    DP[row - 1][-1],
    # Alternative 2: Take the row
    DP[row][-2] + DP[row - 2][-1],
)
```

Boxes

1	8	2	1	9
1	7	3	5	2
1	2	10	3	10
8	4	7	9	1
7	1	3	1	6

$$DP[3,-1] = \max(DP[2,-1], 12 + DP[1,-1]) = \max(17, 12 + 0) = 17$$

DP

		0	0	0	0	0	0
		0	0	0	0	0	0
0	0	1	8	8	9	17	17
0	0	1	7	7	12	12	17
0	0	?	?	?	?	?	0
0	0	?	?	?	?	?	0
0	0	?	?	?	?	?	0

## 4º CASO DE PRUEBA - CASO MÁS COMPLEJO

```
for row, box in enumerate(boxes, start=2):
    for col, candies in enumerate(box, start=2):
        DP[row][col] = max(
            # Alternative 1: Don't take the candy
            DP[row][col - 1],
            # Alternative 2: Take the candy
            candies + DP[row][col - 2],
        )
    DP[row][-1] = max(
        # Alternative 1: Don't take the row
        DP[row - 1][-1],
        # Alternative 2: Take the row
        DP[row][-2] + DP[row - 2][-1],
    )
```

Boxes

1	8	2	1	9
1	7	3	5	2
1	2	10	3	10
8	4	7	9	1
7	1	3	1	6

$$DP[4,2] = \max(DP[4,1], 1 + DP[4,0]) = \max(0, 1 + 0) = 1$$

DP

		0	0	0	0	0	0
		0	0	0	0	0	0
0	0	1	8	8	9	17	17
0	0	1	7	7	12	12	17
0	0	1	?	?	?	?	0
0	0	?	?	?	?	?	0
0	0	?	?	?	?	?	0

## 4º CASO DE PRUEBA - CASO MÁS COMPLEJO

```
for row, box in enumerate(boxes, start=2):
    for col, candies in enumerate(box, start=2):
        DP[row][col] = max(
            # Alternative 1: Don't take the candy
            DP[row][col - 1],
            # Alternative 2: Take the candy
            candies + DP[row][col - 2],
        )
    DP[row][-1] = max(
        # Alternative 1: Don't take the row
        DP[row - 1][-1],
        # Alternative 2: Take the row
        DP[row][-2] + DP[row - 2][-1],
    )
```

Boxes

1	8	2	1	9
1	7	3	5	2
1	2	10	3	10
8	4	7	9	1
7	1	3	1	6

$$DP[4,3] = \max(DP[4,2], 2 + DP[4,1]) = \max(1, 2 + 0) = 2$$

DP

		0	0	0	0	0	0
		0	0	0	0	0	0
0	0	1	8	8	9	17	17
0	0	1	7	7	12	12	17
0	0	1	2	?	?	?	0
0	0	?	?	?	?	?	0
0	0	?	?	?	?	?	0

## 4º CASO DE PRUEBA - CASO MÁS COMPLEJO

```
for row, box in enumerate(boxes, start=2):
    for col, candies in enumerate(box, start=2):
        DP[row][col] = max(
            # Alternative 1: Don't take the candy
            DP[row][col - 1],
            # Alternative 2: Take the candy
            candies + DP[row][col - 2],
        )

DP[row][-1] = max(
    # Alternative 1: Don't take the row
    DP[row - 1][-1],
    # Alternative 2: Take the row
    DP[row][-2] + DP[row - 2][-1],
)
```

Boxes

1	8	2	1	9
1	7	3	5	2
1	2	10	3	10
8	4	7	9	1
7	1	3	1	6

$$DP[4,4] = \max(DP[4,3], 10 + DP[4,2]) = \max(2, 10 + 1) = 11$$

DP

		0	0	0	0	0	0
		0	0	0	0	0	0
0	0	1	8	8	9	17	17
0	0	1	7	7	12	12	17
0	0	1	2	11	?	?	0
0	0	?	?	?	?	?	0
0	0	?	?	?	?	?	0

## 4º CASO DE PRUEBA - CASO MÁS COMPLEJO

```
for row, box in enumerate(boxes, start=2):
    for col, candies in enumerate(box, start=2):
        DP[row][col] = max(
            # Alternative 1: Don't take the candy
            DP[row][col - 1],
            # Alternative 2: Take the candy
            candies + DP[row][col - 2],
        )

DP[row][-1] = max(
    # Alternative 1: Don't take the row
    DP[row - 1][-1],
    # Alternative 2: Take the row
    DP[row][-2] + DP[row - 2][-1],
)
```

Boxes

1	8	2	1	9
1	7	3	5	2
1	2	10	3	10
8	4	7	9	1
7	1	3	1	6

$$DP[4,5] = \max(DP[4,4], 3 + DP[4,3]) = \max(11, 3 + 2) = 11$$

DP

		0	0	0	0	0	0
		0	0	0	0	0	0
0	0	1	8	8	9	17	17
0	0	1	7	7	12	17	17
0	0	1	2	11	11	?	0
0	0	?	?	?	?	?	0
0	0	?	?	?	?	?	0

## 4º CASO DE PRUEBA - CASO MÁS COMPLEJO

```
for row, box in enumerate(boxes, start=2):
    for col, candies in enumerate(box, start=2):
        DP[row][col] = max(
            # Alternative 1: Don't take the candy
            DP[row][col - 1],
            # Alternative 2: Take the candy
            candies + DP[row][col - 2],
        )

DP[row][-1] = max(
    # Alternative 1: Don't take the row
    DP[row - 1][-1],
    # Alternative 2: Take the row
    DP[row][-2] + DP[row - 2][-1],
)
```

Boxes

1	8	2	1	9
1	7	3	5	2
1	2	10	3	10
8	4	7	9	1
7	1	3	1	6

$$DP[4,6] = \max(DP[4,5], 10 + DP[4,4]) = \max(11, 10 + 11) = 21$$

DP

		0	0	0	0	0	0
		0	0	0	0	0	0
0	0	1	8	8	9	17	17
0	0	1	7	7	12	12	17
0	0	1	2	11	11	21	0
0	0	?	?	?	?	?	0
0	0	?	?	?	?	?	0

## 4º CASO DE PRUEBA - CASO MÁS COMPLEJO

```
for row, box in enumerate(boxes, start=2):
    for col, candies in enumerate(box, start=2):
        DP[row][col] = max(
            # Alternative 1: Don't take the candy
            DP[row][col - 1],
            # Alternative 2: Take the candy
            candies + DP[row][col - 2],
        )

DP[row][-1] = max(
    # Alternative 1: Don't take the row
    DP[row - 1][-1],
    # Alternative 2: Take the row
    DP[row][-2] + DP[row - 2][-1],
)
```

Boxes

1	8	2	1	9
1	7	3	5	2
1	2	10	3	10
8	4	7	9	1
7	1	3	1	6

$$DP[4,-1] = \max(DP[3,-1], 21 + DP[2,-1]) = \max(17, 21 + 17) = 38$$

DP

		0	0	0	0	0	0
		0	0	0	0	0	0
0	0	1	8	8	9	17	17
0	0	1	7	7	12	12	17
0	0	1	2	11	11	21	38
0	0	?	?	?	?	?	0
0	0	?	?	?	?	?	0



## 4º CASO DE PRUEBA - CASO MÁS COMPLEJO

```
for row, box in enumerate(boxes, start=2):
    for col, candies in enumerate(box, start=2):
        DP[row][col] = max(
            # Alternative 1: Don't take the candy
            DP[row][col - 1],
            # Alternative 2: Take the candy
            candies + DP[row][col - 2],
        )

DP[row][-1] = max(
    # Alternative 1: Don't take the row
    DP[row - 1][-1],
    # Alternative 2: Take the row
    DP[row][-2] + DP[row - 2][-1],
)
```

Boxes

1	8	2	1	9
1	7	3	5	2
1	2	10	3	10
8	4	7	9	1
7	1	3	1	6

$$DP[5,2] = \max(DP[5,1], 8 + DP[5,0]) = \max(0, 8 + 0) = 8$$

DP

		0	0	0	0	0	0
		0	0	0	0	0	0
0	0	1	8	8	9	17	17
0	0	1	7	7	12	12	17
0	0	1	2	11	11	21	38
0	0	8	?	?	?	?	0
0	0	?	?	?	?	?	0

## 4º CASO DE PRUEBA - CASO MÁS COMPLEJO

```
for row, box in enumerate(boxes, start=2):
    for col, candies in enumerate(box, start=2):
        DP[row][col] = max(
            # Alternative 1: Don't take the candy
            DP[row][col - 1],
            # Alternative 2: Take the candy
            candies + DP[row][col - 2],
        )
    DP[row][-1] = max(
        # Alternative 1: Don't take the row
        DP[row - 1][-1],
        # Alternative 2: Take the row
        DP[row][-2] + DP[row - 2][-1],
    )
```

Boxes

1	8	2	1	9
1	7	3	5	2
1	2	10	3	10
8	4	7	9	1
7	1	3	1	6

$$DP[5,3] = \max(DP[5,2], 4 + DP[5,1]) = \max(8, 4 + 0) = 8$$

DP

		0	0	0	0	0	0
		0	0	0	0	0	0
0	0	1	8	8	9	17	17
0	0	1	7	7	12	12	17
0	0	1	2	11	11	21	38
0	0	8	8	?	?	?	0
0	0	?	?	?	?	?	0

## 4º CASO DE PRUEBA - CASO MÁS COMPLEJO

```
for row, box in enumerate(boxes, start=2):
    for col, candies in enumerate(box, start=2):
        DP[row][col] = max(
            # Alternative 1: Don't take the candy
            DP[row][col - 1],
            # Alternative 2: Take the candy
            candies + DP[row][col - 2],
        )

DP[row][-1] = max(
    # Alternative 1: Don't take the row
    DP[row - 1][-1],
    # Alternative 2: Take the row
    DP[row][-2] + DP[row - 2][-1],
)
```

Boxes

1	8	2	1	9
1	7	3	5	2
1	2	10	3	10
8	4	7	9	1
7	1	3	1	6

$$DP[5,4] = \max(DP[5,3], 7 + DP[5,2]) = \max(8, 7 + 8) = 15$$

DP

		0	0	0	0	0	0
		0	0	0	0	0	0
0	0	1	8	8	9	17	17
0	0	1	7	7	12	12	17
0	0	1	2	11	11	21	38
0	0	8	8	15	?	?	0
0	0	?	?	?	?	?	0

## 4º CASO DE PRUEBA - CASO MÁS COMPLEJO

```
for row, box in enumerate(boxes, start=2):
    for col, candies in enumerate(box, start=2):
        DP[row][col] = max(
            # Alternative 1: Don't take the candy
            DP[row][col - 1],
            # Alternative 2: Take the candy
            candies + DP[row][col - 2],
        )
    DP[row][-1] = max(
        # Alternative 1: Don't take the row
        DP[row - 1][-1],
        # Alternative 2: Take the row
        DP[row][-2] + DP[row - 2][-1],
    )
```

Boxes

1	8	2	1	9
1	7	3	5	2
1	2	10	3	10
8	4	7	9	1
7	1	3	1	6

$$DP[5,5] = \max(DP[5,4], 9 + DP[5,3]) = \max(15, 9 + 8) = 17$$

DP

		0	0	0	0	0	0
		0	0	0	0	0	0
0	0	1	8	8	9	17	17
0	0	1	7	7	12	12	17
0	0	1	2	11	11	21	38
0	0	8	8	15	17	?	0
0	0	?	?	?	?	?	0

## 4º CASO DE PRUEBA - CASO MÁS COMPLEJO

```
for row, box in enumerate(boxes, start=2):
    for col, candies in enumerate(box, start=2):
        DP[row][col] = max(
            # Alternative 1: Don't take the candy
            DP[row][col - 1],
            # Alternative 2: Take the candy
            candies + DP[row][col - 2],
        )

DP[row][-1] = max(
    # Alternative 1: Don't take the row
    DP[row - 1][-1],
    # Alternative 2: Take the row
    DP[row][-2] + DP[row - 2][-1],
)
```

Boxes

1	8	2	1	9
1	7	3	5	2
1	2	10	3	10
8	4	7	9	1
7	1	3	1	6

$$DP[5,6] = \max(DP[5,5], 1 + DP[5,4]) = \max(17, 1 + 15) = 17$$

DP

		0	0	0	0	0	0
		0	0	0	0	0	0
0	0	1	8	8	9	17	17
0	0	1	7	7	12	12	17
0	0	1	2	11	11	21	38
0	0	8	8	15	17	17	0
0	0	?	?	?	?	?	0

## 4º CASO DE PRUEBA - CASO MÁS COMPLEJO

```
for row, box in enumerate(boxes, start=2):
    for col, candies in enumerate(box, start=2):
        DP[row][col] = max(
            # Alternative 1: Don't take the candy
            DP[row][col - 1],
            # Alternative 2: Take the candy
            candies + DP[row][col - 2],
        )

DP[row][-1] = max(
    # Alternative 1: Don't take the row
    DP[row - 1][-1],
    # Alternative 2: Take the row
    DP[row][-2] + DP[row - 2][-1],
)
```

Boxes

1	8	2	1	9
1	7	3	5	2
1	2	10	3	10
8	4	7	9	1
7	1	3	1	6

$$DP[5,-1] = \max(DP[4,-1], 17 + DP[3,-1]) = \max(38, 17 + 17) = 38$$

DP

		0	0	0	0	0	0
		0	0	0	0	0	0
0	0	1	8	8	9	17	17
0	0	1	7	7	12	12	17
0	0	1	2	11	11	21	38
0	0	8	8	15	17	17	38
0	0	?	?	?	?	?	0

## 4º CASO DE PRUEBA - CASO MÁS COMPLEJO

```
for row, box in enumerate(boxes, start=2):
    for col, candies in enumerate(box, start=2):
        DP[row][col] = max(
            # Alternative 1: Don't take the candy
            DP[row][col - 1],
            # Alternative 2: Take the candy
            candies + DP[row][col - 2],
        )

DP[row][-1] = max(
    # Alternative 1: Don't take the row
    DP[row - 1][-1],
    # Alternative 2: Take the row
    DP[row][-2] + DP[row - 2][-1],
)
```

Boxes

1	8	2	1	9
1	7	3	5	2
1	2	10	3	10
8	4	7	9	1
7	1	3	1	6

$$DP[6,2] = \max(DP[6,1], 7 + DP[6,0]) = \max(0, 7 + 0) = 7$$

DP

		0	0	0	0	0	0
		0	0	0	0	0	0
0	0	1	8	8	9	17	17
0	0	1	7	7	12	12	17
0	0	1	2	11	11	21	38
0	0	8	8	15	17	17	38
0	0	7	?	?	?	?	0

## 4º CASO DE PRUEBA - CASO MÁS COMPLEJO

```
for row, box in enumerate(boxes, start=2):
    for col, candies in enumerate(box, start=2):
        DP[row][col] = max(
            # Alternative 1: Don't take the candy
            DP[row][col - 1],
            # Alternative 2: Take the candy
            candies + DP[row][col - 2],
        )

DP[row][-1] = max(
    # Alternative 1: Don't take the row
    DP[row - 1][-1],
    # Alternative 2: Take the row
    DP[row][-2] + DP[row - 2][-1],
)
```

Boxes

1	8	2	1	9
1	7	3	5	2
1	2	10	3	10
8	4	7	9	1
7	1	3	1	6

$$DP[6,3] = \max(DP[6,2], 1 + DP[6,1]) = \max(7, 1 + 0) = 7$$

DP

		0	0	0	0	0	0
		0	0	0	0	0	0
0	0	1	8	8	9	17	17
0	0	1	7	7	12	12	17
0	0	1	2	11	11	21	38
0	0	8	8	15	17	17	38
0	0	7	7	?	?	?	0



## 4º CASO DE PRUEBA - CASO MÁS COMPLEJO

```
for row, box in enumerate(boxes, start=2):
    for col, candies in enumerate(box, start=2):
        DP[row][col] = max(
            # Alternative 1: Don't take the candy
            DP[row][col - 1],
            # Alternative 2: Take the candy
            candies + DP[row][col - 2],
        )

DP[row][-1] = max(
    # Alternative 1: Don't take the row
    DP[row - 1][-1],
    # Alternative 2: Take the row
    DP[row][-2] + DP[row - 2][-1],
)
```

Boxes

1	8	2	1	9
1	7	3	5	2
1	2	10	3	10
8	4	7	9	1
7	1	3	1	6

$$DP[6,4] = \max(DP[6,3], 3 + DP[6,2]) = \max(7, 3 + 7) = 10$$

DP

		0	0	0	0	0	0
		0	0	0	0	0	0
0	0	1	8	8	9	17	17
0	0	1	7	7	12	12	17
0	0	1	2	11	11	21	38
0	0	8	8	15	17	17	38
0	0	7	7	10	?	?	0

## 4º CASO DE PRUEBA - CASO MÁS COMPLEJO

```
for row, box in enumerate(boxes, start=2):
    for col, candies in enumerate(box, start=2):
        DP[row][col] = max(
            # Alternative 1: Don't take the candy
            DP[row][col - 1],
            # Alternative 2: Take the candy
            candies + DP[row][col - 2],
        )
    DP[row][-1] = max(
        # Alternative 1: Don't take the row
        DP[row - 1][-1],
        # Alternative 2: Take the row
        DP[row][-2] + DP[row - 2][-1],
    )
```

Boxes

1	8	2	1	9
1	7	3	5	2
1	2	10	3	10
8	4	7	9	1
7	1	3	1	6

$$DP[6,5] = \max(DP[6,4], 1 + DP[6,3]) = \max(10, 1 + 7) = 10$$

DP

		0	0	0	0	0	0
		0	0	0	0	0	0
0	0	1	8	8	9	17	17
0	0	1	7	7	12	12	17
0	0	1	2	11	11	21	38
0	0	8	8	15	17	17	38
0	0	7	7	10	10	?	0

## 4º CASO DE PRUEBA - CASO MÁS COMPLEJO

```
for row, box in enumerate(boxes, start=2):
    for col, candies in enumerate(box, start=2):
        DP[row][col] = max(
            # Alternative 1: Don't take the candy
            DP[row][col - 1],
            # Alternative 2: Take the candy
            candies + DP[row][col - 2],
        )

DP[row][-1] = max(
    # Alternative 1: Don't take the row
    DP[row - 1][-1],
    # Alternative 2: Take the row
    DP[row][-2] + DP[row - 2][-1],
)
```

Boxes

1	8	2	1	9
1	7	3	5	2
1	2	10	3	10
8	4	7	9	1
7	1	3	1	6

$$DP[6,6] = \max(DP[6,5], 6 + DP[6,4]) = \max(10, 6 + 10) = 16$$

DP

		0	0	0	0	0	0
		0	0	0	0	0	0
0	0	1	8	8	9	17	17
0	0	1	7	7	12	12	17
0	0	1	2	11	11	21	38
0	0	8	8	15	17	17	38
0	0	7	7	10	10	16	0

## 4º CASO DE PRUEBA - CASO MÁS COMPLEJO

```
for row, box in enumerate(boxes, start=2):
    for col, candies in enumerate(box, start=2):
        DP[row][col] = max(
            # Alternative 1: Don't take the candy
            DP[row][col - 1],
            # Alternative 2: Take the candy
            candies + DP[row][col - 2],
        )

DP[row][-1] = max(
    # Alternative 1: Don't take the row
    DP[row - 1][-1],
    # Alternative 2: Take the row
    DP[row][-2] + DP[row - 2][-1],
)
```

Boxes

1	8	2	1	9
1	7	3	5	2
1	2	10	3	10
8	4	7	9	1
7	1	3	1	6

$$DP[6,-1] = \max(DP[5,-1], 16 + DP[4,-1]) = \max(38, 16 + 38) = 54$$

DP

		0	0	0	0	0	0
		0	0	0	0	0	0
0	0	1	8	8	9	17	17
0	0	1	7	7	12	12	17
0	0	1	2	11	11	21	38
0	0	8	8	15	17	17	38
0	0	7	7	10	10	16	54

Resultado final



## 5º CASO DE PRUEBA - CASO ATÓMICO

```
for row, box in enumerate(boxes, start=2):
    for col, candies in enumerate(box, start=2):
        DP[row][col] = max(
            # Alternative 1: Don't take the candy
            DP[row][col - 1],
            # Alternative 2: Take the candy
            candies + DP[row][col - 2],
        )
    DP[row][-1] = max(
        # Alternative 1: Don't take the row
        DP[row - 1][-1],
        # Alternative 2: Take the row
        DP[row][-2] + DP[row - 2][-1],
    )
```

Boxes

10

$$DP[2,2] = \max(DP[2,1], 10 + DP[2,0]) = \max(0, 10 + 0) = 10$$

DP

		0	0
		0	0
0	0	10	0

## 5º CASO DE PRUEBA - CASO ATÓMICO

```
for row, box in enumerate(boxes, start=2):
    for col, candies in enumerate(box, start=2):
        DP[row][col] = max(
            # Alternative 1: Don't take the candy
            DP[row][col - 1],
            # Alternative 2: Take the candy
            candies + DP[row][col - 2],
        )
    DP[row][-1] = max(
        # Alternative 1: Don't take the row
        DP[row - 1][-1],
        # Alternative 2: Take the row
        DP[row][-2] + DP[row - 2][-1],
    )
```

Boxes

10
----

$$DP[2,-1] = \max(DP[1,-1], 10 + DP[0,-1]) = \max(0, 10 + 0) = 10$$

DP

		0	0
		0	0
0	0	10	10

Resultado final



**¡GRACIAS!**  
**¿PREGUNTAS?**