7. Astro Void se dedica a la compra de asteroides. Sea $p \in \mathbb{N}^n$ tal que p_i es el precio de un asteroide el i-ésimo día en una secuencia de n días. Astro Void quiere comprar y vender asteroides durante esos n días de manera tal de obtener la mayor ganancia neta posible. Debido a las dificultades que existen en el transporte y almacenamiento de asteroides, Astro Void puede comprar a lo sumo un asteroide cada día, puede vender a lo sumo un asteroide cada día y comienza sin asteroides. Además, el Ente Regulador Asteroidal impide que Astro Void venda un asteroide que no haya comprado. Queremos encontrar la máxima ganancia neta que puede obtener Astro Void respetando las restricciones indicadas. Por ejemplo, si $p = (3, 2, 5, 6)$ el resultado es 6 y si $p = (3, 6, 10)$ el resultado es 7. Notar que en una solución óptima, Astro Void debe terminar sin asteroides.

   a) Convencerse de que la máxima ganancia neta (m.g.n.), si Astro Void tiene c asteroides al fin del día j, es:

   - indefinido (i.e., $-\infty$) si $c < 0$ o $c > j$, o
   - el máximo entre:
     - la m.g.n. de finalizar el día $j-1$ con $c-1$ asteroides y comprar uno en el día j,
     - la m.g.n. de finalizar el día $j-1$ con $c+1$ asteroides y vender uno en el día j,
     - la m.g.n. de finalizar el día $j-1$ con c asteroides y no operar el día j.

* Si $c<0$ → vendí ⊕ de 1 asteroide en un día
$c>j$ → compré ⊕ de 1 asteroide en un día

* Si tengo c asteroides al final del día j → o terminé el día $j-1$ con $c-1$ y compré uno el día j

hay que ver cual maximizó mi ganancia
→ o terminé el día $j-1$ con $c+1$ y vendí uno el día j
→ o terminé el día $j-1$ con c y no hice nada el día j

* Estamos moviendo a j de n a 0

No me queda claro el tema de las estructuras de los asteroides y su precio

↳ el vector p tiene el precio de los asteroides. De todos!

b) Escribir matemáticamente la formulación recursiva enunciada en *a*). Dar los valores de los casos base en función de la restricción de que comienza sin asteroides.

Caso base: $j=1$, en el primer día o no compro o compro → al final del día o tengo 1 o 0

entonces mi caso base va a ser $j=0$ y va a devolver 0

$$
mgn(a,j) = \begin{cases} -\infty & \text{si } a<0 \lor a>j \\ 0 & \text{si } j=0 \\ \max\begin{cases} mgn(a-1,j-1) \overset{\text{compro}}{-} p_{aj} \\ mgn(a+1,j-1) + p_{aj} \\ mgn(a,j-1) \end{cases} & \text{c.c.} \end{cases}
$$

resto de piezas faltantes

En esta función falta algo que me permita sacar el precio. Como en el del CD pongo P_i, acá P_a

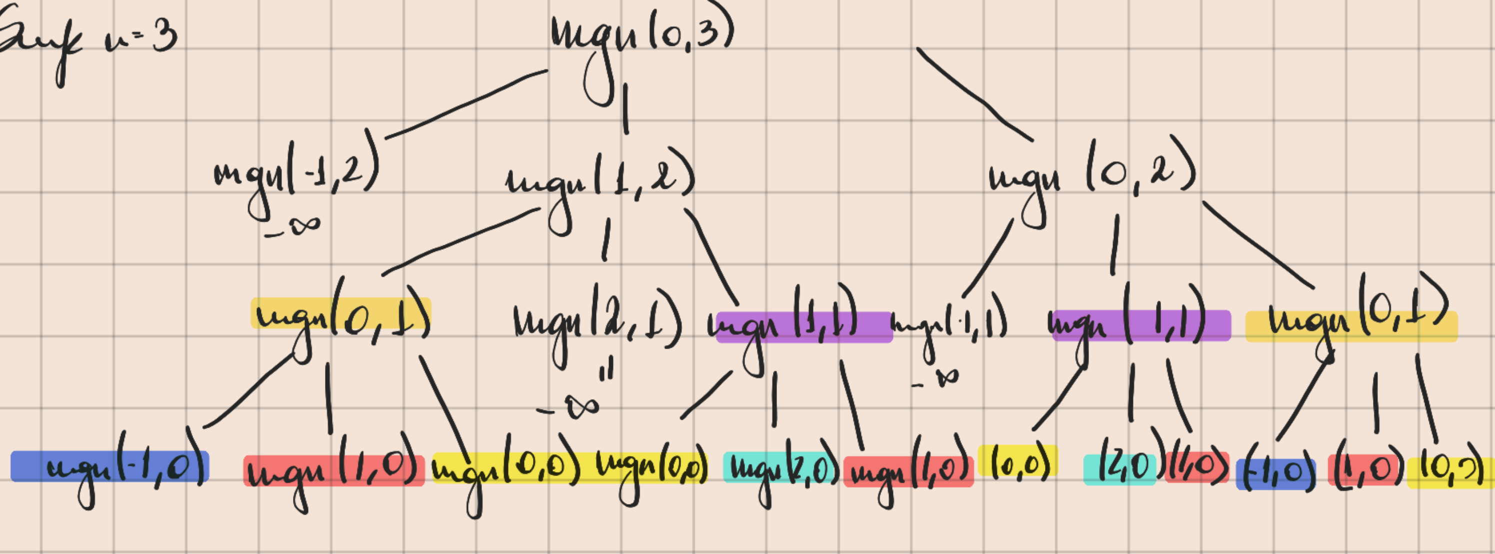
c) Indicar qué dato es la respuesta al problema con esa formulación recursiva.

El dato que resuelve es un número

Tengo que llamar a la función con $mgn(0,n)$

al final del día n no tengo asteroides

Sup $n=3$



Calculo varias veces el mismo problema

$d)$ Diseñar un algoritmo de PD *top-down* que resuelva el problema y explicar su complejidad temporal y espacial auxiliar.

① Mi función resuelve mi problema, básicamente me lo dicen

③ ↗

② Tenemos que ver que tiene superposición de subproblemas

¿Cuántas llamadas recursivas hace mi función? No tenemos un nivel completo más allá del primero pero siempre x nivel 2 nodos tienen al menos 3 hijos

$\Omega(2^n)$

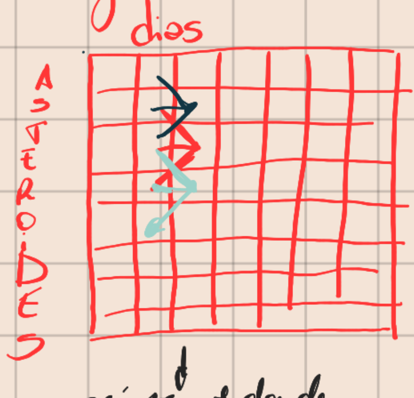
¿Cuántos subproblemas distintos resolvemos?

día $\in [0, n+1]$ $\rightarrow$ $O(n^2)$ problemas

asteroides $\in [-1, n-1]$

$\rightarrow$ $n^2 << 2^n$ siempre $\Rightarrow$ hay siempre sup de subpro

Tengo sup de subproblemas porque calculo varias veces lo que pasó en días anteriores. Tenemos un matriz



¡acá vemos donde se superponen!

$\longleftarrow$ hacia allá se llena mi matriz m.g.n (a,d)   sin p.d

if $a<0 \vee a>d$

    return +inf

endif

if d=0

    return 0

else

    return $\max\{mgn(a-1,d-1)-p_d,\ mgn(a+1,d-1)+p_d,\ mgn(a,d-1)\}$

④ Defino un estructura de memoización: Matriz de$(n+1)\times n$ inicializada en $\perp$

falta un vector de precios

m.g.n (a,d)   p.d

if $a<0 \vee a>d$

    return +inf

if d=0

    return 0

else

    if $M[d][a]=\perp$ -1

        compré = mgn(a-1,d-1) - p_d

        vendí = mgn(a+1,d-1) + p_d

        nohicenada = mgn (a ,d-1)

        $M[d][a]$ = max{compré, vendí, nohicenada}   -1

    endif

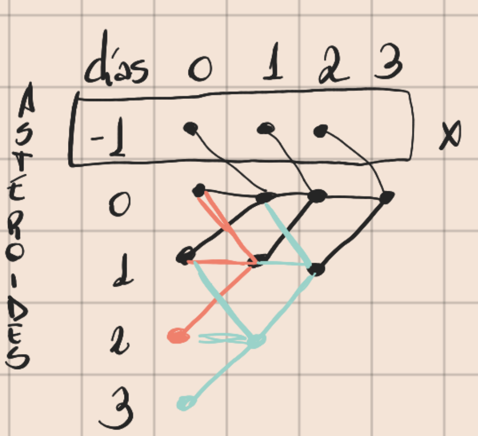
    return $M[d][a]$ -1

Complejidad espacial $\Theta(n^2)$

Complejidad temporal $\Theta(n^2)$

e) (Opcional) Diseñar un algoritmo de PD *bottom-up*, reduciendo la complejidad espacial.

Veamos como se llama minmatg para $d=3$ $p_1=(1,2,3)$



} puedo no mantenerla total si ya es -inf

no estamos usando todo este espacio. Podríamos mejorar con un algo BU

$$m.g.u(a,j) = \begin{cases} -\infty & \text{si } a<0 \lor a>j \\ 0 & \text{si } j=0 \\ \text{máx}\begin{cases} m.g.u(a-1,j-1) - p_{a_j} \;(\text{compro}) \\ m.g.u(a+1,j-1) + p_{a_j} \\ m.g.u(a,j-1) \end{cases} \end{cases}$$

$$mgu(p_1,0,3) = \max\{\overbrace{m.g.u(p,-1,2)}^{-inf}, \overbrace{m.gu(p,1,2)+3}^{-1+3=2}, \overbrace{mgu(p,0,2)}^{1}\} = 2$$

$$mgu(p_1,1,2) = \max\{m.gu(p,0,1)-2, \underbrace{mgu(p,2,1)}_{-inf}+2, mgu(p,1,1)\} = \max\{-2,-inf,-1\} = -1$$

$$mgu(p,0,1) = \max\{\underbrace{mgu(p,-1,0)}_{-inf}, \underbrace{mgu(p,1,0)}_{-inf}+1, \underbrace{mgu(p,0,0)}_{0}\} = 0$$

$$mgu(p,1,1) = \max\{mgu(p,0,0)-1, \underbrace{mgu(p,2,0)}_{-inf}, \underbrace{mgu(p,1,0)}_{-inf}\} = -1$$

$$mgu(p,0,2) = \max\{\overbrace{mgu(p,-1,1)}^{-inf}, \underbrace{mgu(p,1,1)}_{=-1}+2, \underbrace{mgu(p,0,1)}_{0}\} = 1$$

Vamos a hacer un algo BU que mejore complejidad

Inicializamos 2 vectores $N[0,\dots,n]$ y $M[0,\dots,n]$, N en 0 y $M[0]=0$, $M[1:n]=-\inf$

$p = 1,2,3$

$d = 3$

M	0	-inf	-inf	-inf
N	0	-1	-inf	0
	0	1	2	
M	0	-1	-inf	0
N	1	-1	~~X~~	0

```
AT(p,a,d)
 vector N de tamaño d+1 inicializado en 0
 vector M de tamaño d+1 con su primera pos en 0, el resto en -inf
 for(i=1; i≤d; i++){
   for(j=0; j≤d-i  j++){
     if j=0 then
       N[j]=max(M[j], M[j+1]+p[i])
     else
       N[j]=max(M[j], M[j-1]-p[j], M[j+1]+p[j])
     endif
     swap(N,M)
   endfor
 endfor
 return N[0]
```

$\underbrace{\Theta(n^2)}_{T}$ $\underbrace{\Theta(n)}_{\text{espacial}}$