Pontificia Universidad Católica de Chile
Departamento de Ciencia de la Computación
IIC2133 - Estructuras de Datos y Algoritmos

Ayudantía 5

Cristóbal Berríos: crisberrios@uc.cl
Nicholas Mc-Donnell: namcdonnell@uc.cl

Problema 1

A partir del arreglo A = [1, 50, -35, 68, 98, 5, -57, 33, 25, -45, -90], construye un Segment Tree para encontrar el número máximo dentro de un subarreglo de A y responde a las siguientes queries: (1, 7), (3, 8), (8, 10).

1	50	-35	68	98	5	-57	33	25	-45	-90

Problema 1

```
build(A): //retorna la raíz del ST a hecho partir de un arreglo A
    n = largo de A
    return build(A, 1, n)

build(A, L, R):
    if L = R:
        return nodo(L, R, A[L], null, null)   // crea un nodo con el rango
    else:                                      // (L, R) valor A[L] y sin hijos
        B_left = build(A, L, (L + R)/2)
        B_right = build(A, (L + R)/2 + 1, R)
        v = min(B_left.value , B_right.value)
        return nodo(L, R, v, B_left, B_right)
```

Problema 1

1	50	-35	68	98	5	-57	33	25	-45	-90

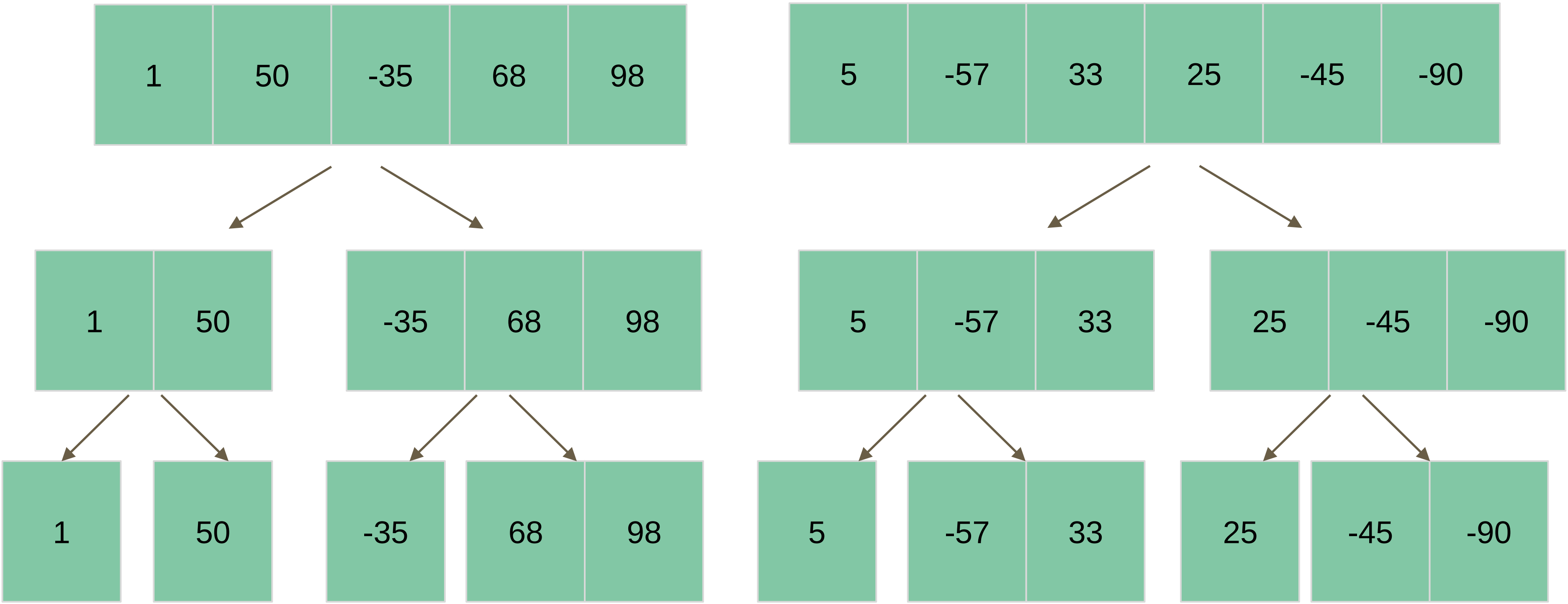
$\boldsymbol{B}_{left} = \mathrm{build}(A, L, (L + R)/2)$

$\boldsymbol{B}_{right} = \mathrm{build}(A, (L + R)/2 + 1, R)$

1	50	-35	68	98

5	-57	33	25	-45	-90

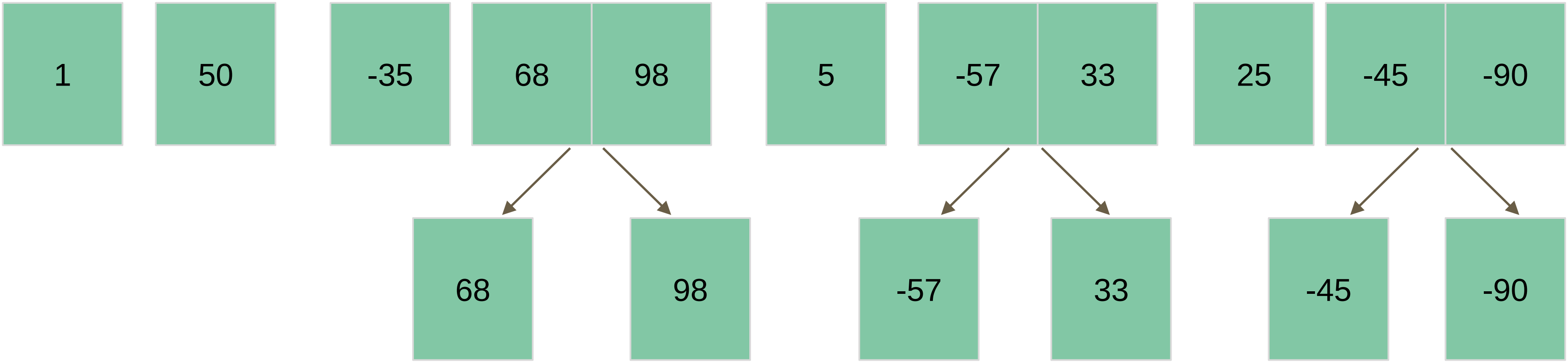
Problema 1


1 50 -35 68 98
5 -57 33 25 -45 -90
1 50
-35 68 98
5 -57 33
25 -45 -90
1
50
-35
68 98
5
-57 33
25
-45 -90

Problema 1

1	50	-35	68	98	5	-57	33	25	-45	-90

Problema 1



Problema 1

1	50	-35	68	98	5	-57	33	25	-45	-90

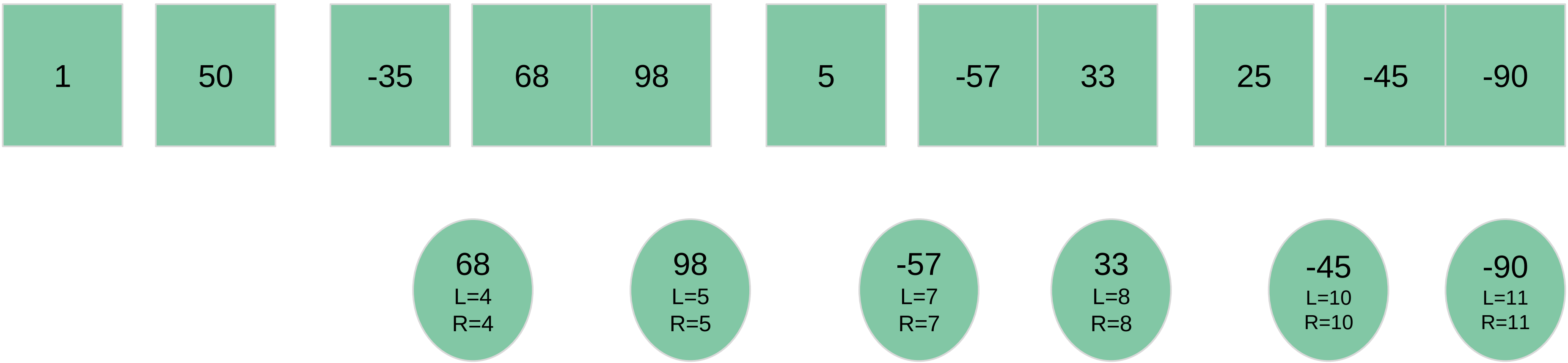
68	98	-57	33	-45	-90

if $L = R$:

return nodo(L, R, $A[L]$, **null**, **null**)

Llegamos al Caso Base!

Problema 1

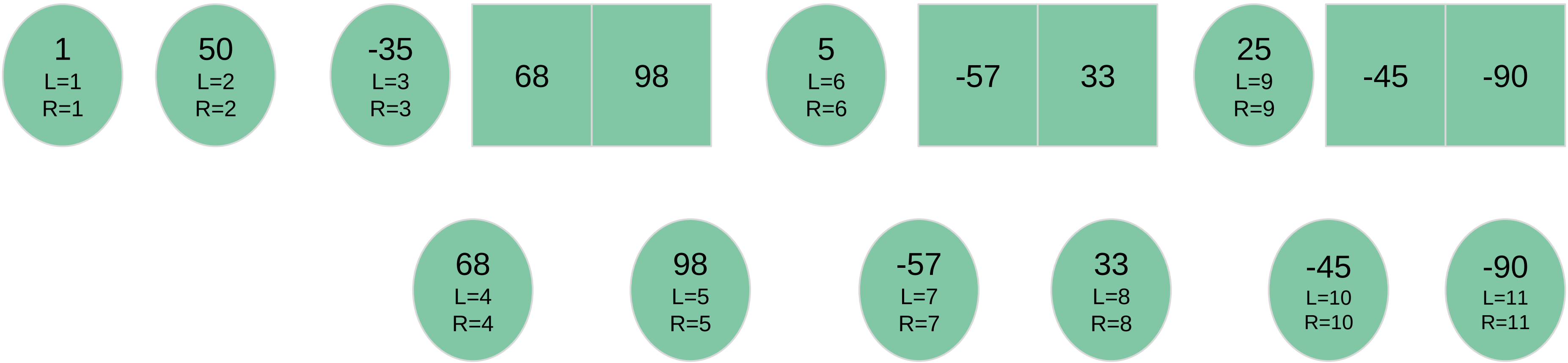


if $L = R$:

return nodo(L, R, $A[L]$, **null**, **null**)

Ahora se empiezan a instanciar los nodos del Segment Tree

Problema 1

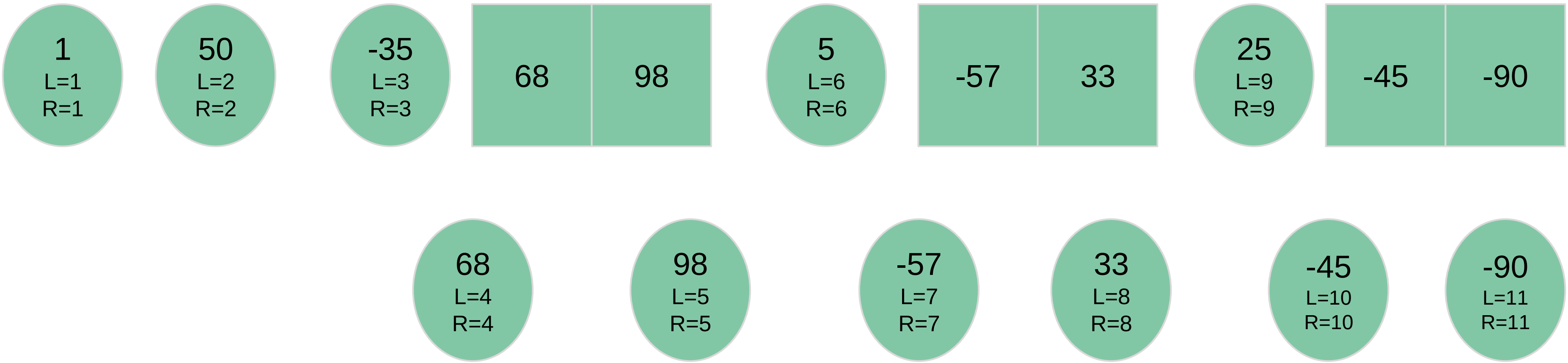


if $L = R$:

return nodo(L, R, $A[L]$, **null**, **null**)

Ahora se empiezan a instanciar los nodos del Segment Tree

Problema 1

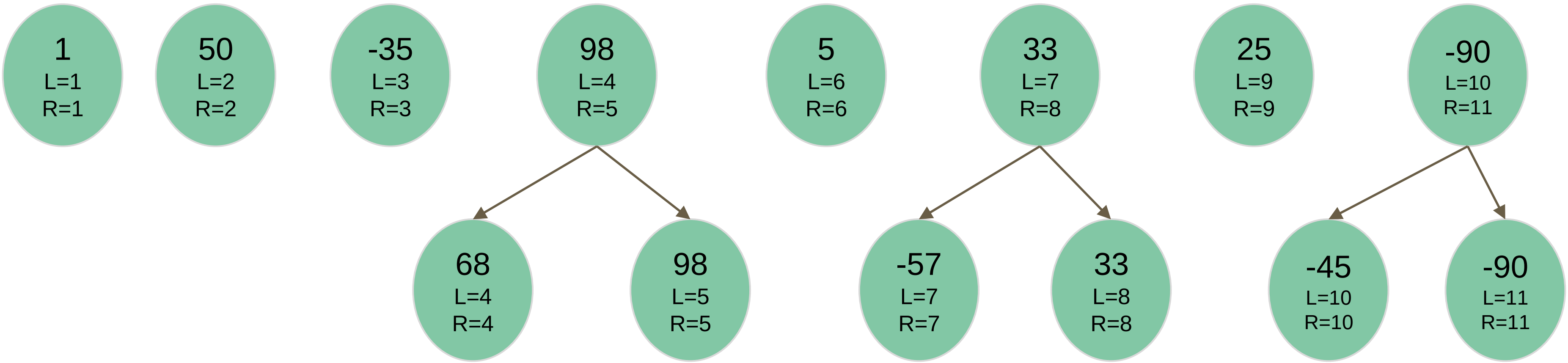


$$v = \min(\boldsymbol{B_{left}}.\boldsymbol{value}\ , \boldsymbol{B_{right}}.\boldsymbol{value})$$

$$\textbf{return}\ \text{nodo}(L, R, v, \boldsymbol{B_{left}}, \boldsymbol{B_{right}})$$

Y ahora los que no son caso base

Problema 1

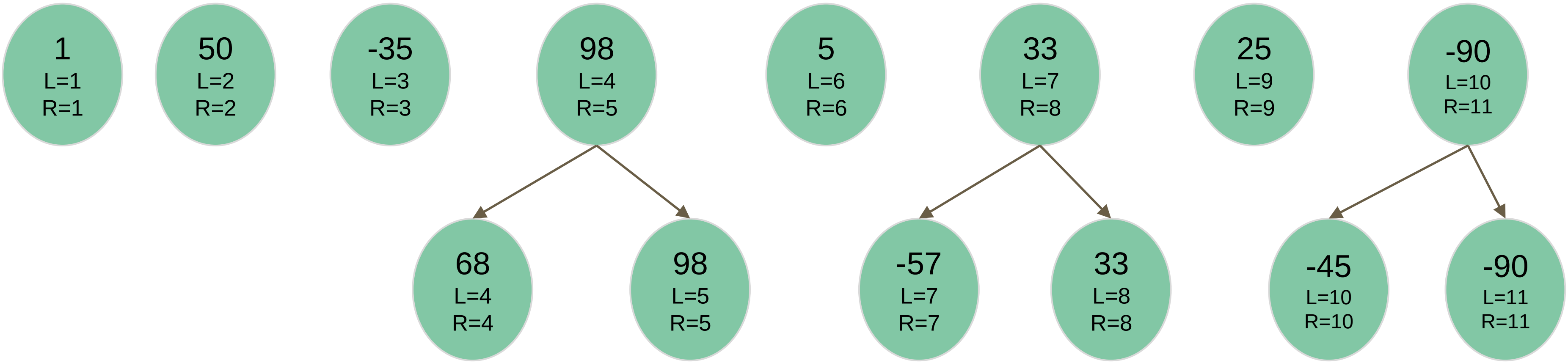


$$v = \min(\boldsymbol{B_{left}}.\boldsymbol{value}\ ,\ \boldsymbol{B_{right}}.\boldsymbol{value})$$

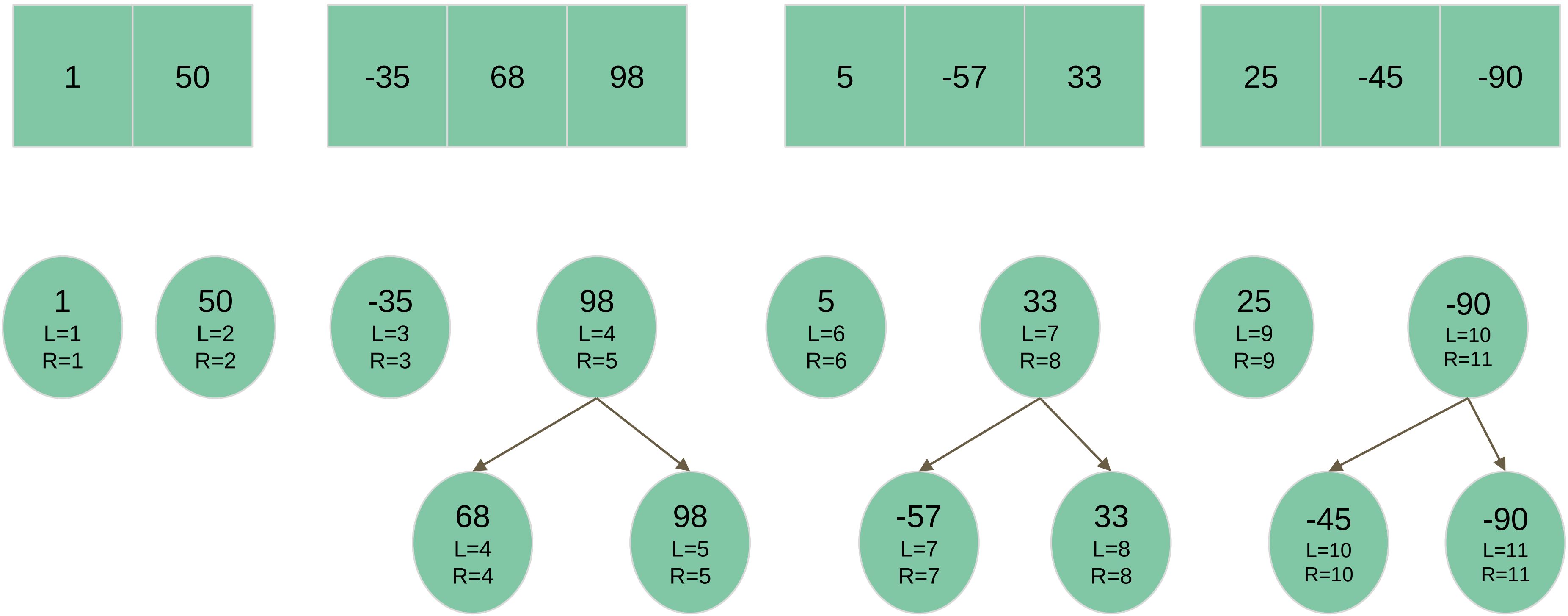
$$\textbf{return}\ \text{nodo}(L, R, v, \boldsymbol{B_{left}}, \boldsymbol{B_{right}})$$

Y ahora los que no son caso base
(solo que en nuestro caso buscamos los máximos, no los mínimos)

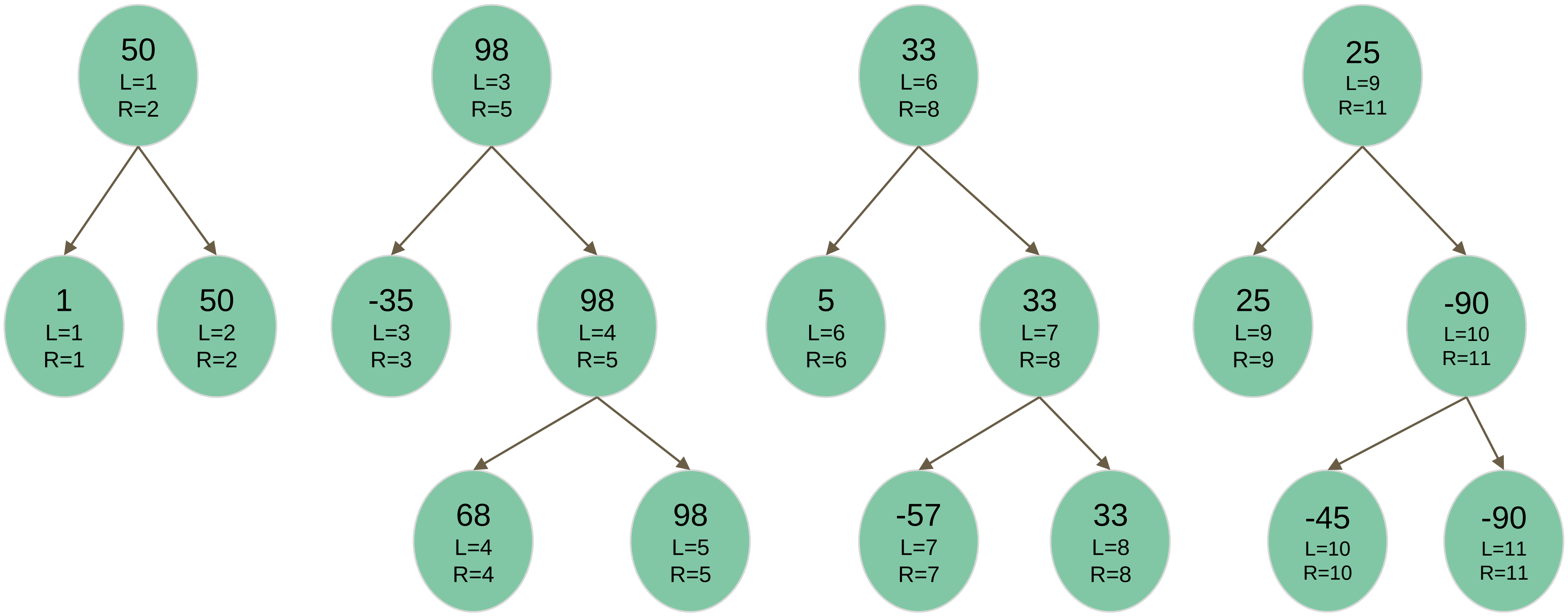
Problema 1


1
L=1
R=1
50
L=2
R=2
-35
L=3
R=3
98
L=4
R=5
5
L=6
R=6
33
L=7
R=8
25
L=9
R=9
-90
L=10
R=11
68
L=4
R=4
98
L=5
R=5
-57
L=7
R=7
33
L=8
R=8
-45
L=10
R=10
-90
L=11
R=11

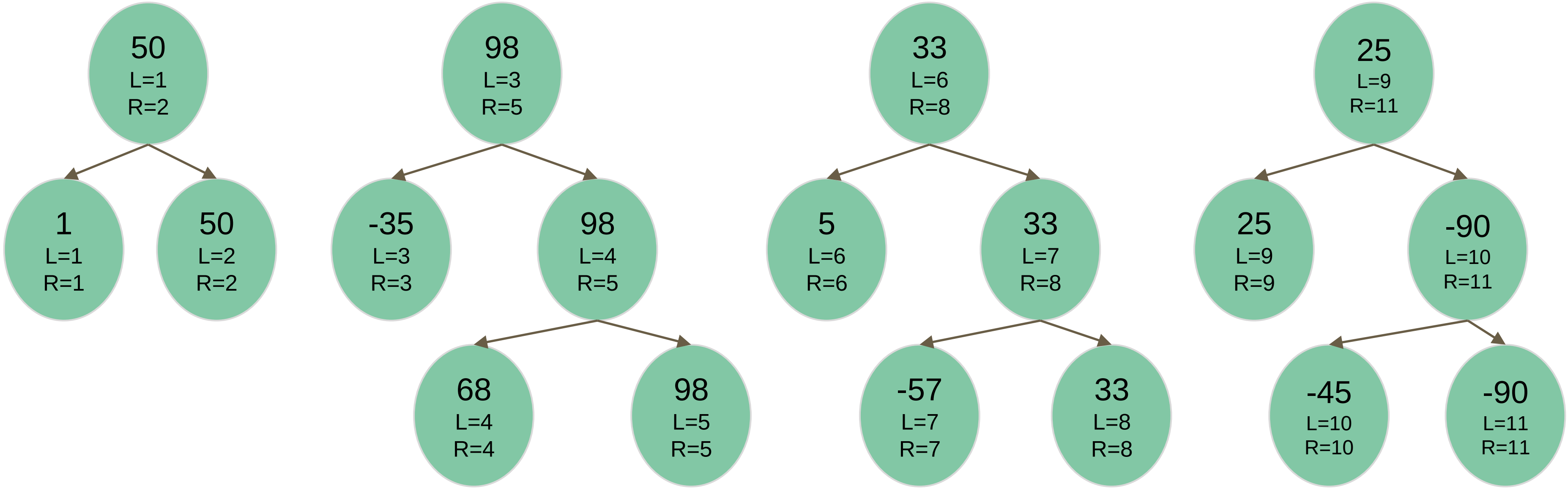
Problema 1


1
50
-35
68
98
5
-57
33
25
-45
-90
1 L=1 R=1
50 L=2 R=2
-35 L=3 R=3
98 L=4 R=5
5 L=6 R=6
33 L=7 R=8
25 L=9 R=9
-90 L=10 R=11
68 L=4 R=4
98 L=5 R=5
-57 L=7 R=7
33 L=8 R=8
-45 L=10 R=10
-90 L=11 R=11

Problema 1


50
L=1
R=2
1
L=1
R=1
50
L=2
R=2
98
L=3
R=5
-35
L=3
R=3
98
L=4
R=5
68
L=4
R=4
98
L=5
R=5
33
L=6
R=8
5
L=6
R=6
33
L=7
R=8
-57
L=7
R=7
33
L=8
R=8
25
L=9
R=11
25
L=9
R=9
-90
L=10
R=11
-45
L=10
R=10
-90
L=11
R=11

Problema 1

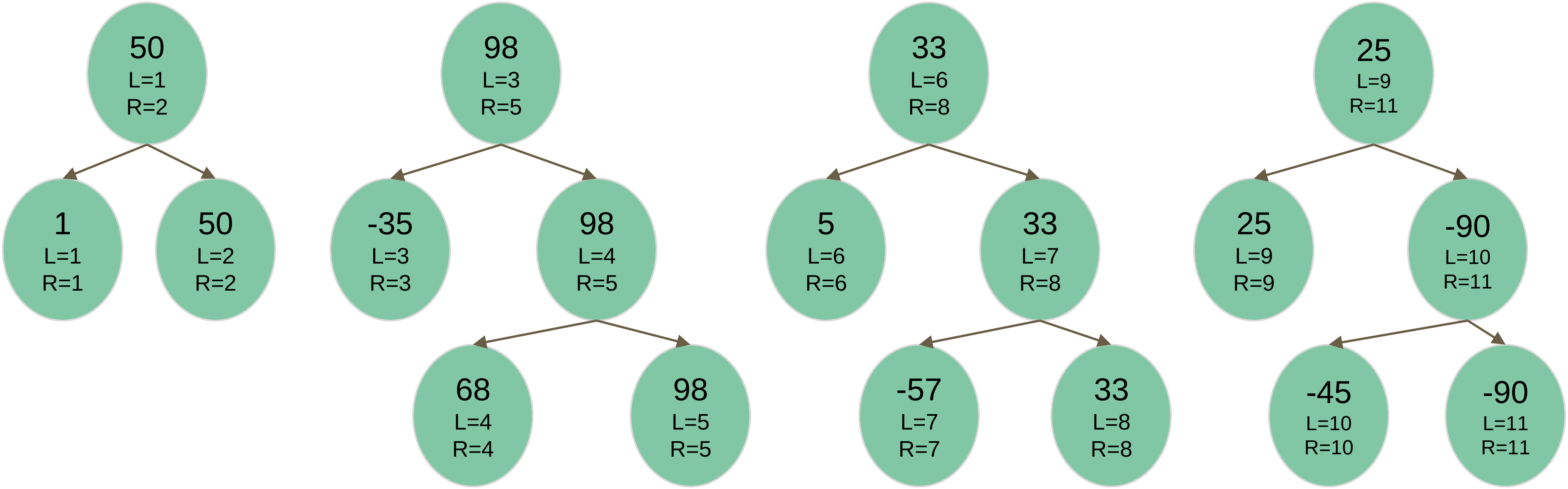


50 L=1 R=2
1 L=1 R=1
50 L=2 R=2
98 L=3 R=5
-35 L=3 R=3
98 L=4 R=5
68 L=4 R=4
98 L=5 R=5
33 L=6 R=8
5 L=6 R=6
33 L=7 R=8
-57 L=7 R=7
33 L=8 R=8
25 L=9 R=11
25 L=9 R=9
-90 L=10 R=11
-45 L=10 R=10
-90 L=11 R=11

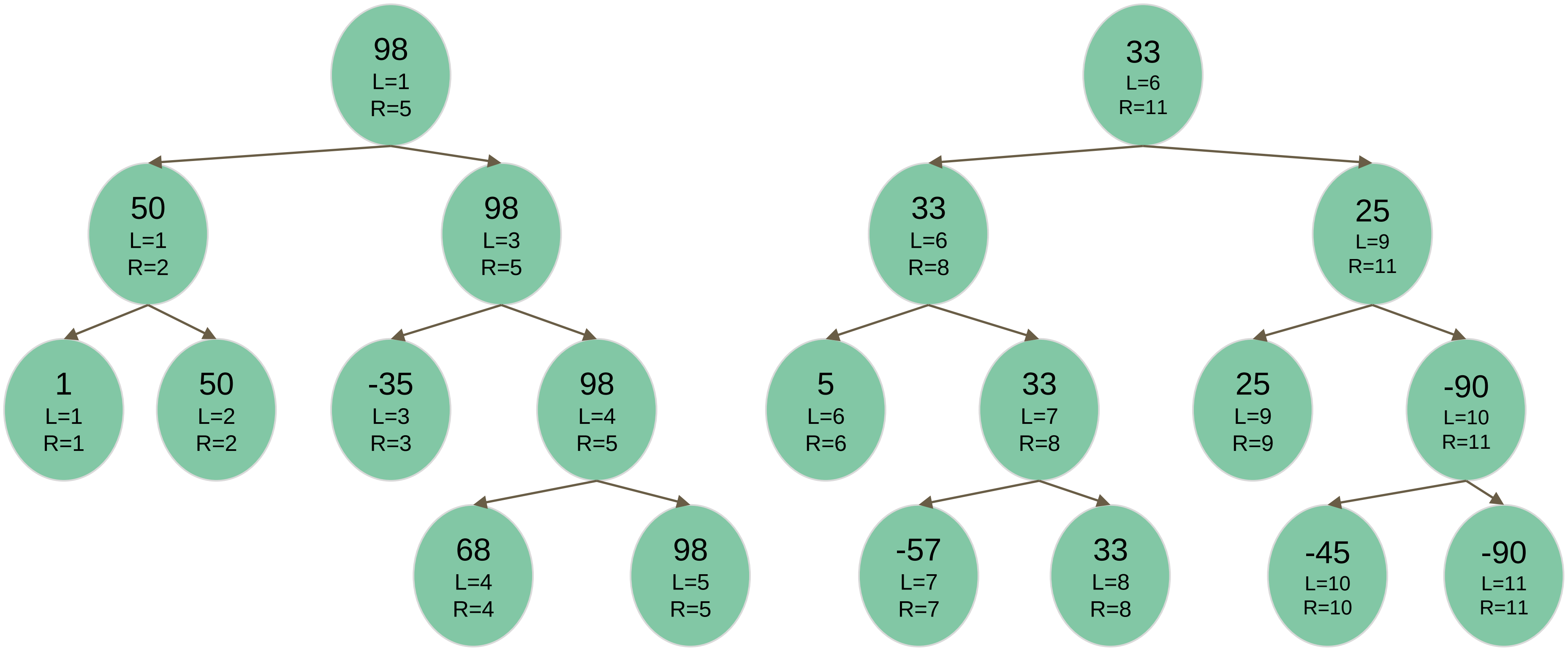
Problema 1

1	50	-35	68	98

5	-57	33	25	-45	-90

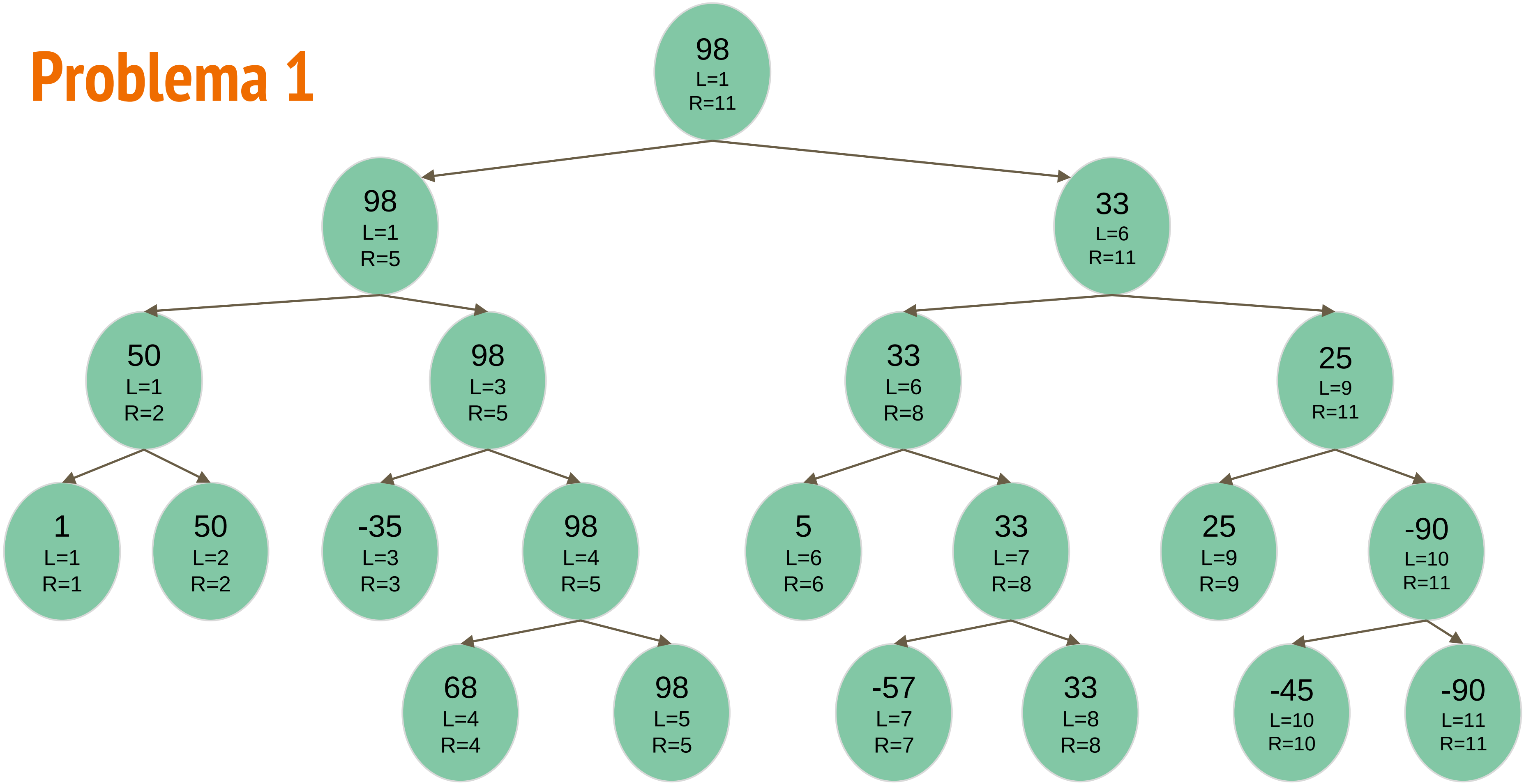


Problema 1



98
L=1
R=5
50
L=1
R=2
98
L=3
R=5
1
L=1
R=1
50
L=2
R=2
-35
L=3
R=3
98
L=4
R=5
68
L=4
R=4
98
L=5
R=5
33
L=6
R=11
33
L=6
R=8
25
L=9
R=11
5
L=6
R=6
33
L=7
R=8
25
L=9
R=9
-90
L=10
R=11
-57
L=7
R=7
33
L=8
R=8
-45
L=10
R=10
-90
L=11
R=11

Problema 1

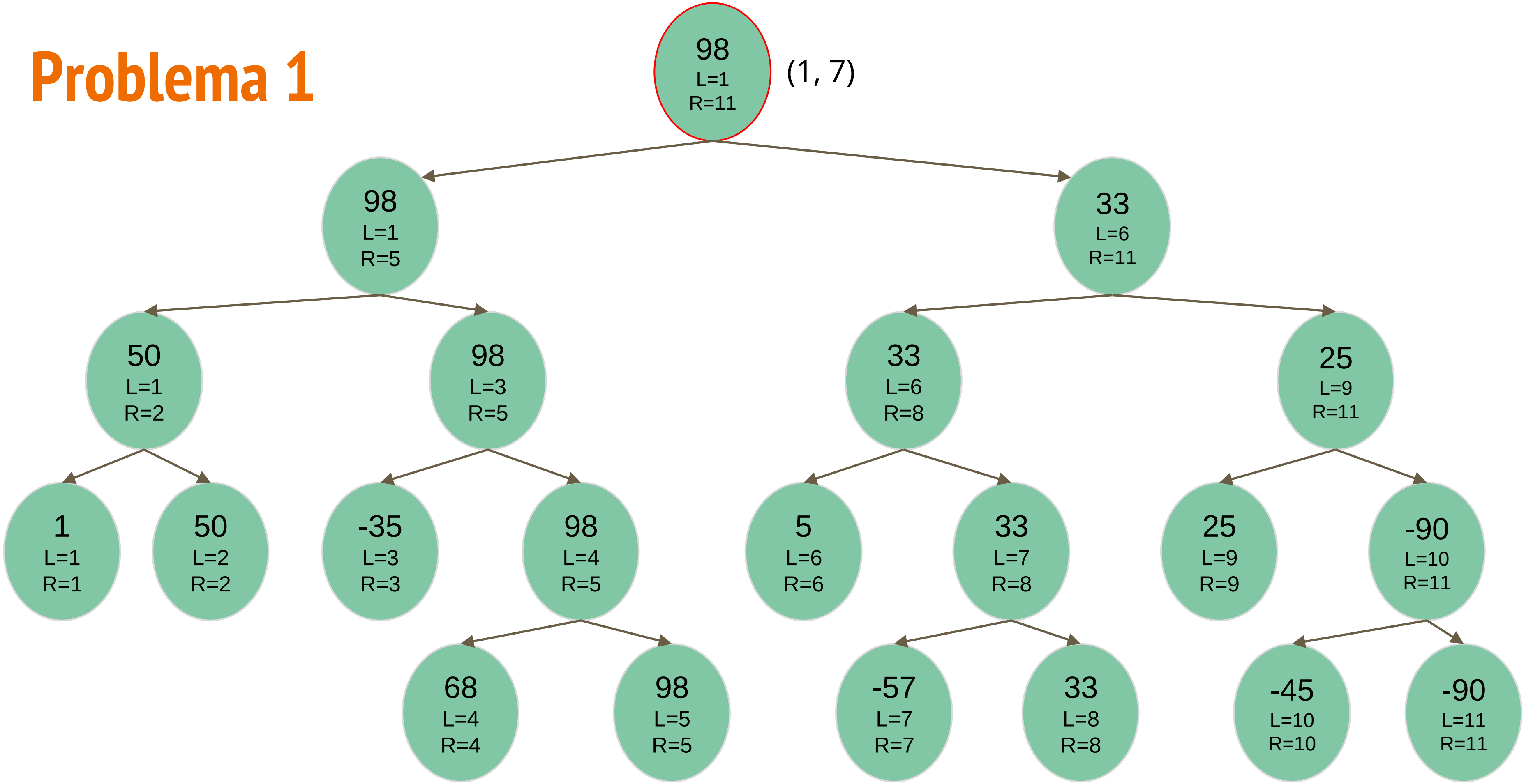


98 L=1 R=11
98 L=1 R=5
33 L=6 R=11
50 L=1 R=2
98 L=3 R=5
33 L=6 R=8
25 L=9 R=11
1 L=1 R=1
50 L=2 R=2
-35 L=3 R=3
98 L=4 R=5
5 L=6 R=6
33 L=7 R=8
25 L=9 R=9
-90 L=10 R=11
68 L=4 R=4
98 L=5 R=5
-57 L=7 R=7
33 L=8 R=8
-45 L=10 R=10
-90 L=11 R=11

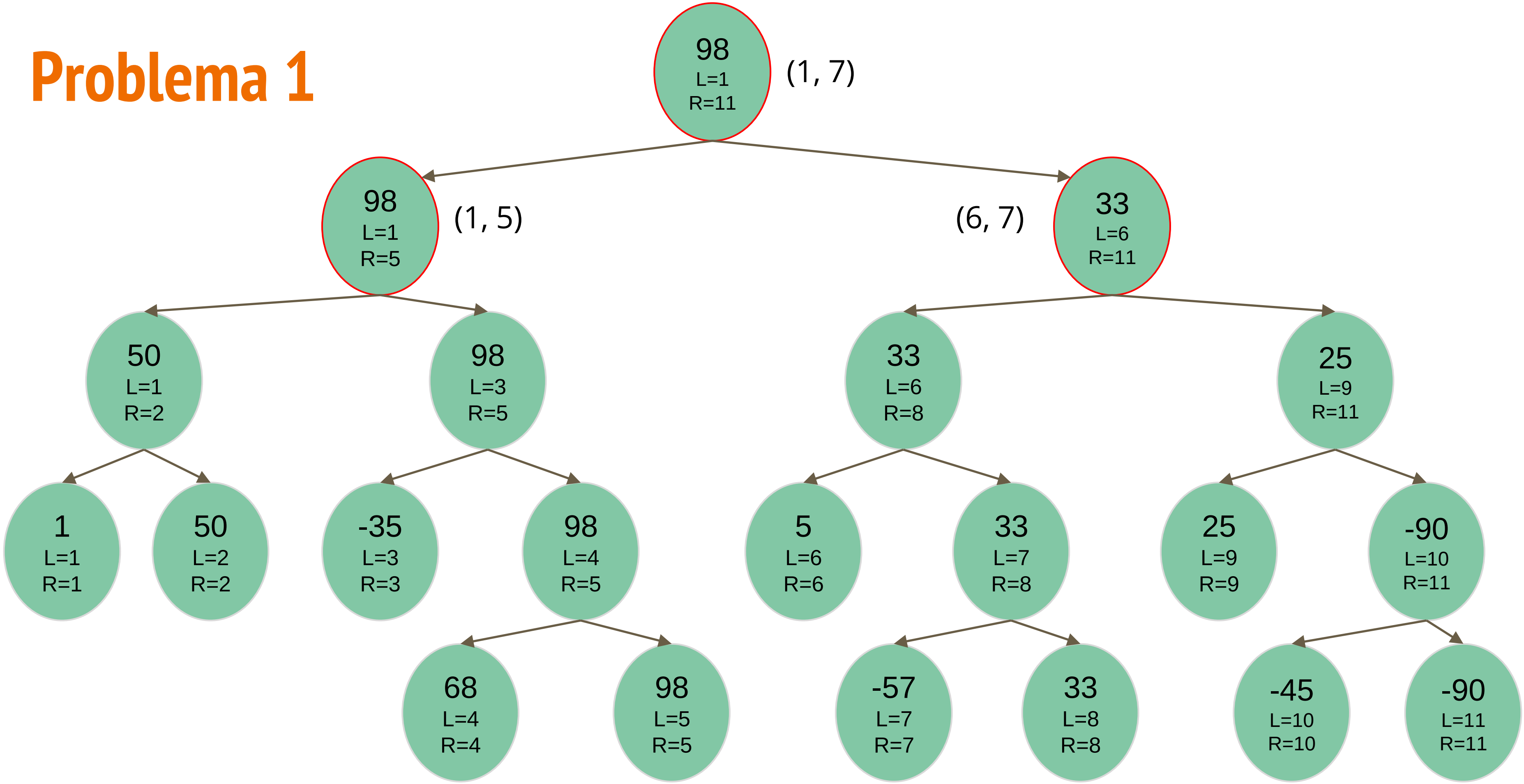
Problema 1

A partir del arreglo A = [1, 50, -35, 68, 98, 5, -57, 33, 25, -45, -90], construye un Segment Tree para encontrar el número máximo dentro de un subarreglo de A y responde a las siguientes queries: (1, 7), (3, 8), (8, 10).

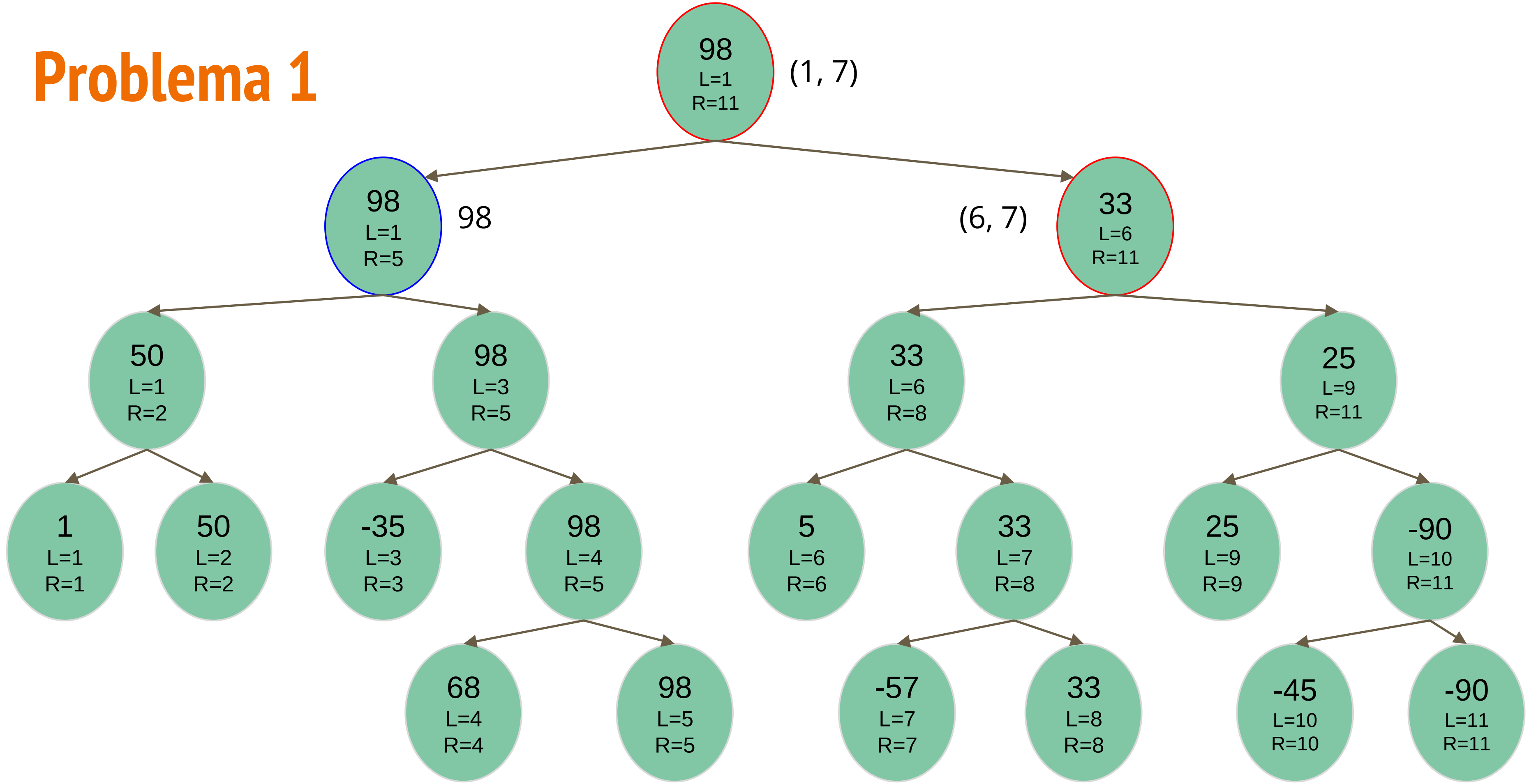
Problema 1


98
L=1
R=11
(1, 7)
98
L=1
R=5
33
L=6
R=11
50
L=1
R=2
98
L=3
R=5
33
L=6
R=8
25
L=9
R=11
1
L=1
R=1
50
L=2
R=2
-35
L=3
R=3
98
L=4
R=5
5
L=6
R=6
33
L=7
R=8
25
L=9
R=9
-90
L=10
R=11
68
L=4
R=4
98
L=5
R=5
-57
L=7
R=7
33
L=8
R=8
-45
L=10
R=10
-90
L=11
R=11

Problema 1

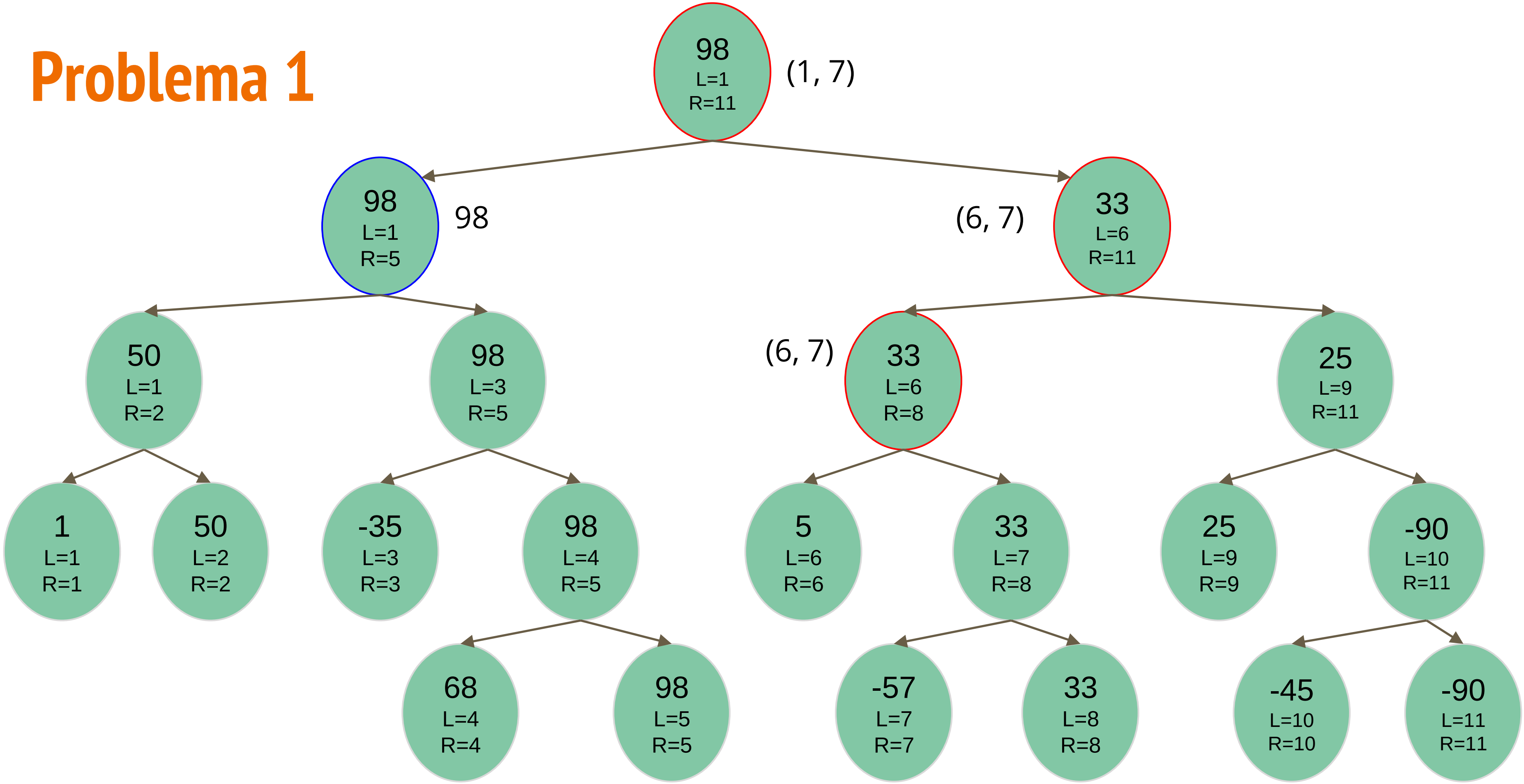

98 L=1 R=11
(1, 7)
98 L=1 R=5
(1, 5)
(6, 7)
33 L=6 R=11
50 L=1 R=2
98 L=3 R=5
33 L=6 R=8
25 L=9 R=11
1 L=1 R=1
50 L=2 R=2
-35 L=3 R=3
98 L=4 R=5
5 L=6 R=6
33 L=7 R=8
25 L=9 R=9
-90 L=10 R=11
68 L=4 R=4
98 L=5 R=5
-57 L=7 R=7
33 L=8 R=8
-45 L=10 R=10
-90 L=11 R=11

Problema 1



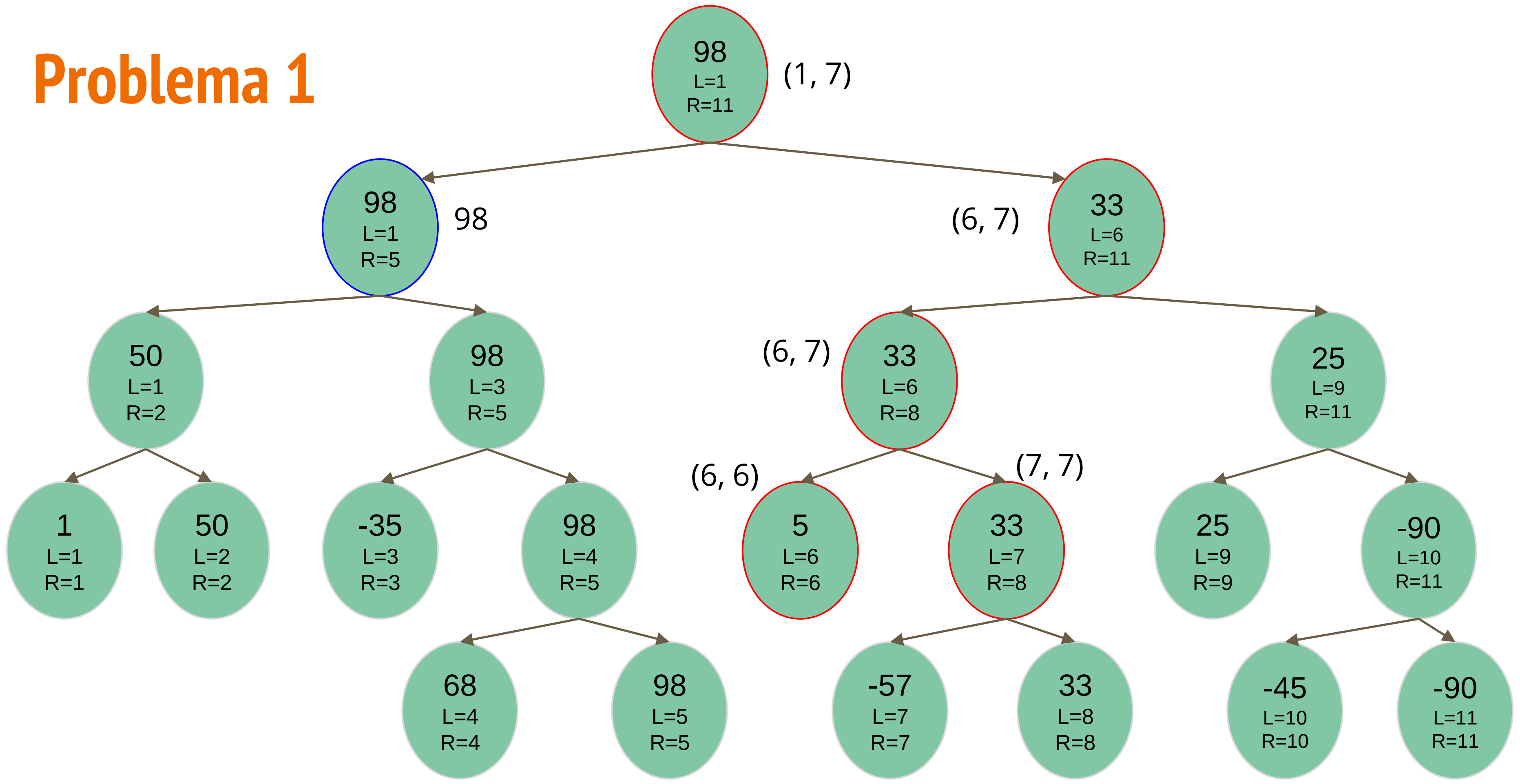
98 L=1 R=11
(1, 7)
98 L=1 R=5
98
(6, 7)
33 L=6 R=11
50 L=1 R=2
98 L=3 R=5
33 L=6 R=8
25 L=9 R=11
1 L=1 R=1
50 L=2 R=2
-35 L=3 R=3
98 L=4 R=5
5 L=6 R=6
33 L=7 R=8
25 L=9 R=9
-90 L=10 R=11
68 L=4 R=4
98 L=5 R=5
-57 L=7 R=7
33 L=8 R=8
-45 L=10 R=10
-90 L=11 R=11

Problema 1



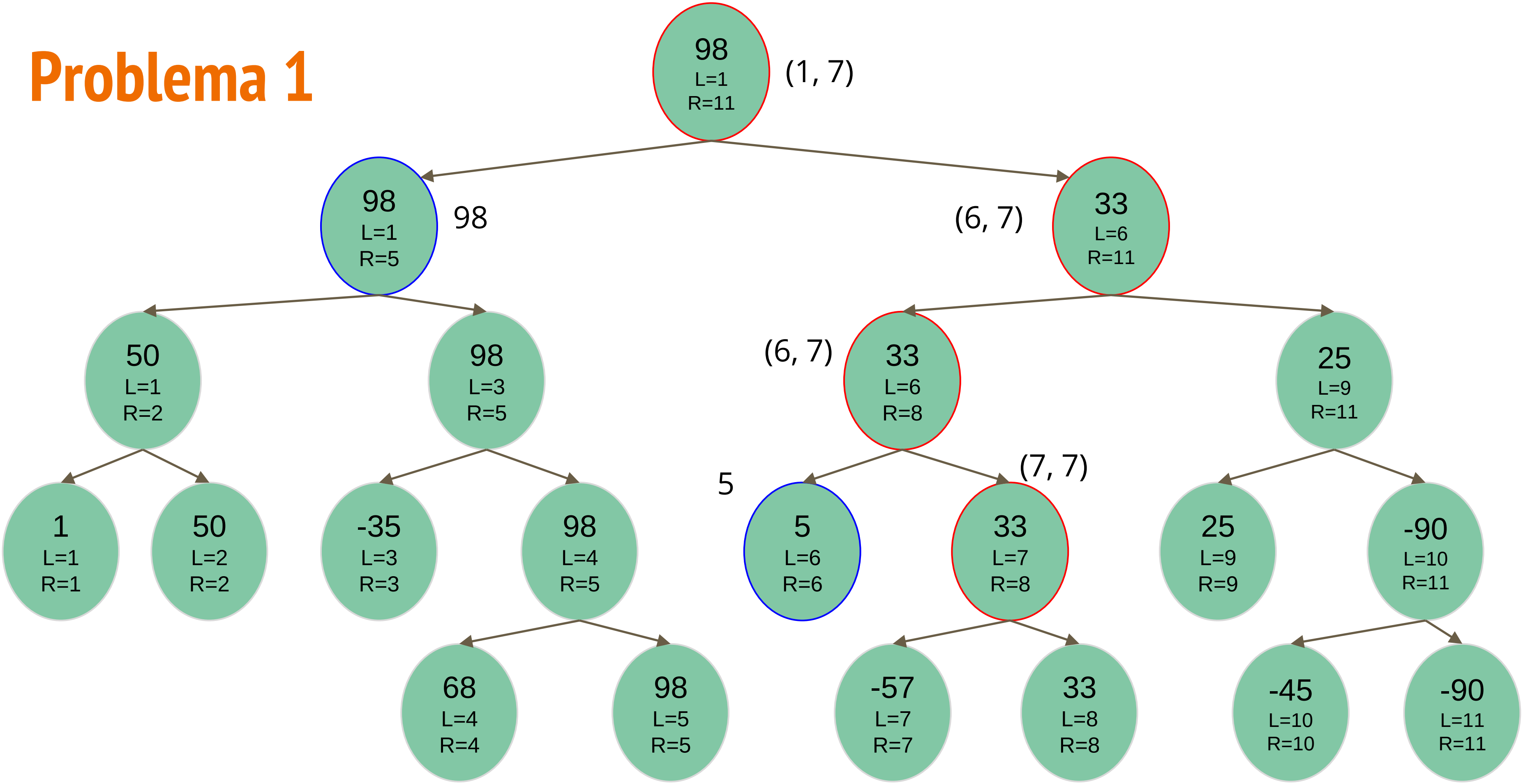
98
L=1
R=11
(1, 7)
98
L=1
R=5
98
(6, 7)
33
L=6
R=11
50
L=1
R=2
98
L=3
R=5
(6, 7)
33
L=6
R=8
25
L=9
R=11
1
L=1
R=1
50
L=2
R=2
-35
L=3
R=3
98
L=4
R=5
5
L=6
R=6
33
L=7
R=8
25
L=9
R=9
-90
L=10
R=11
68
L=4
R=4
98
L=5
R=5
-57
L=7
R=7
33
L=8
R=8
-45
L=10
R=10
-90
L=11
R=11

Problema 1

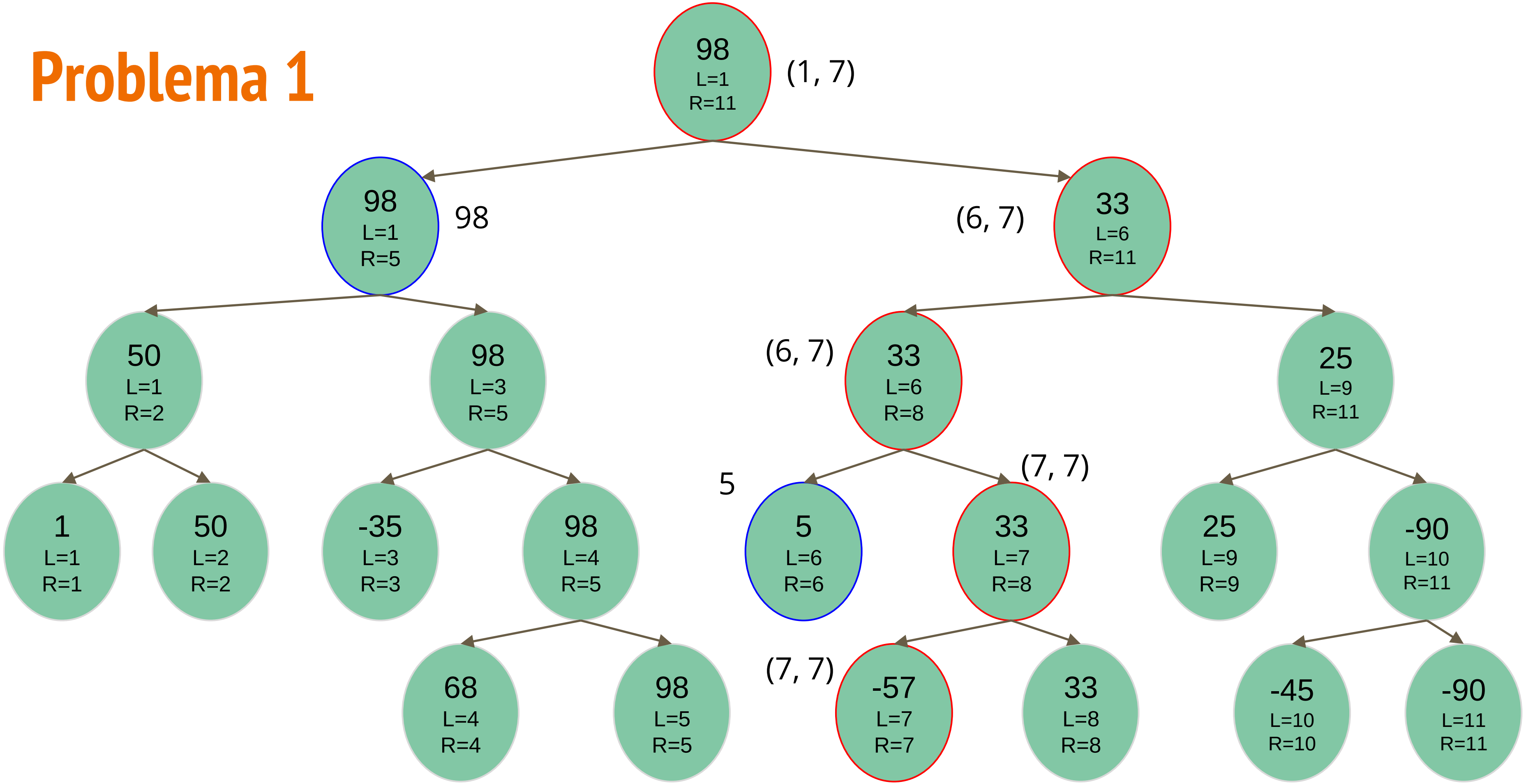


98 L=1 R=11
(1, 7)
98 L=1 R=5
98
(6, 7)
33 L=6 R=11
50 L=1 R=2
98 L=3 R=5
(6, 7)
33 L=6 R=8
25 L=9 R=11
(6, 6)
(7, 7)
1 L=1 R=1
50 L=2 R=2
-35 L=3 R=3
98 L=4 R=5
5 L=6 R=6
33 L=7 R=8
25 L=9 R=9
-90 L=10 R=11
68 L=4 R=4
98 L=5 R=5
-57 L=7 R=7
33 L=8 R=8
-45 L=10 R=10
-90 L=11 R=11

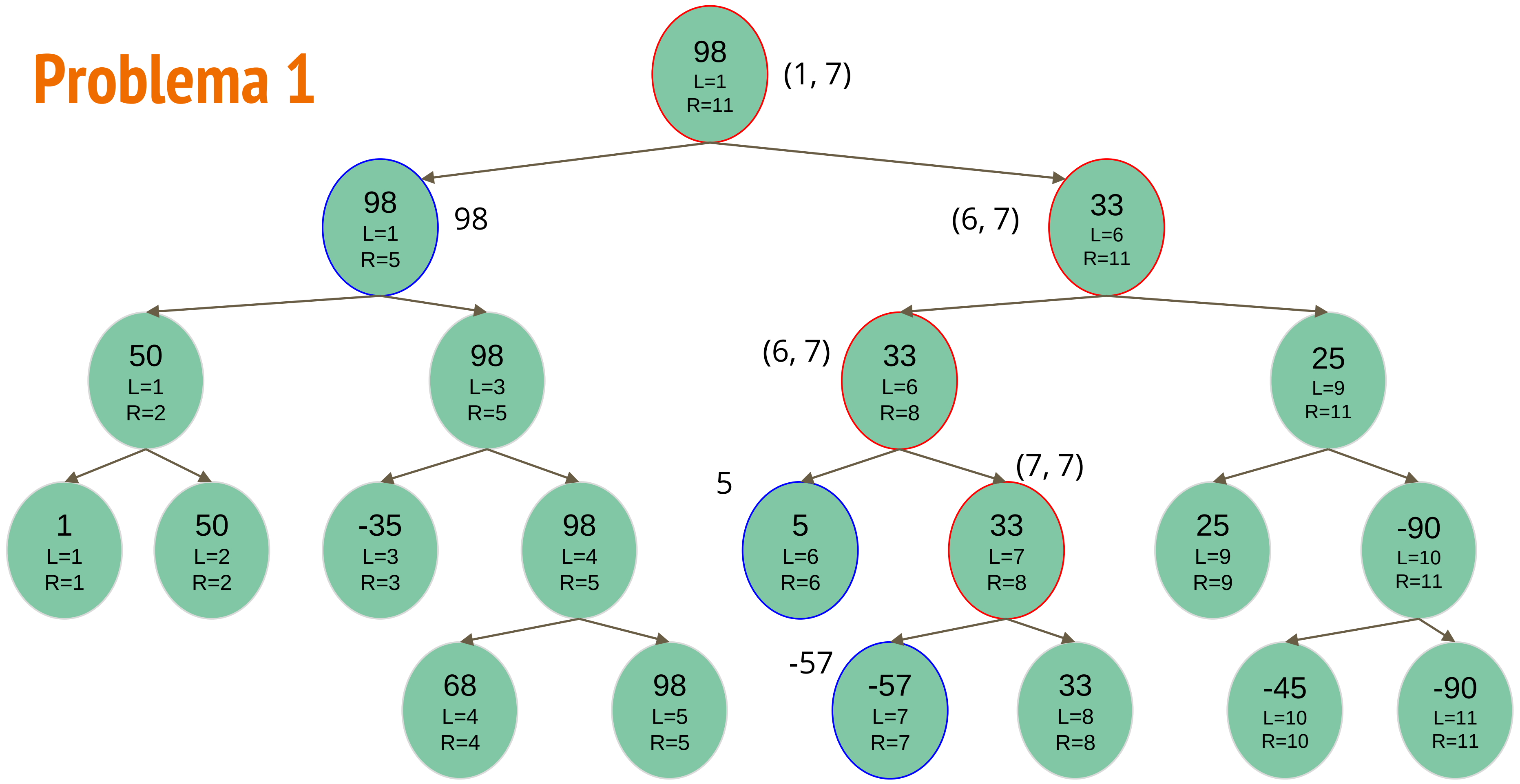
Problema 1


98
L=1
R=11
(1, 7)
98
L=1
R=5
98
(6, 7)
33
L=6
R=11
50
L=1
R=2
98
L=3
R=5
(6, 7)
33
L=6
R=8
25
L=9
R=11
1
L=1
R=1
50
L=2
R=2
-35
L=3
R=3
98
L=4
R=5
5
5
L=6
R=6
(7, 7)
33
L=7
R=8
25
L=9
R=9
-90
L=10
R=11
68
L=4
R=4
98
L=5
R=5
-57
L=7
R=7
33
L=8
R=8
-45
L=10
R=10
-90
L=11
R=11

Problema 1

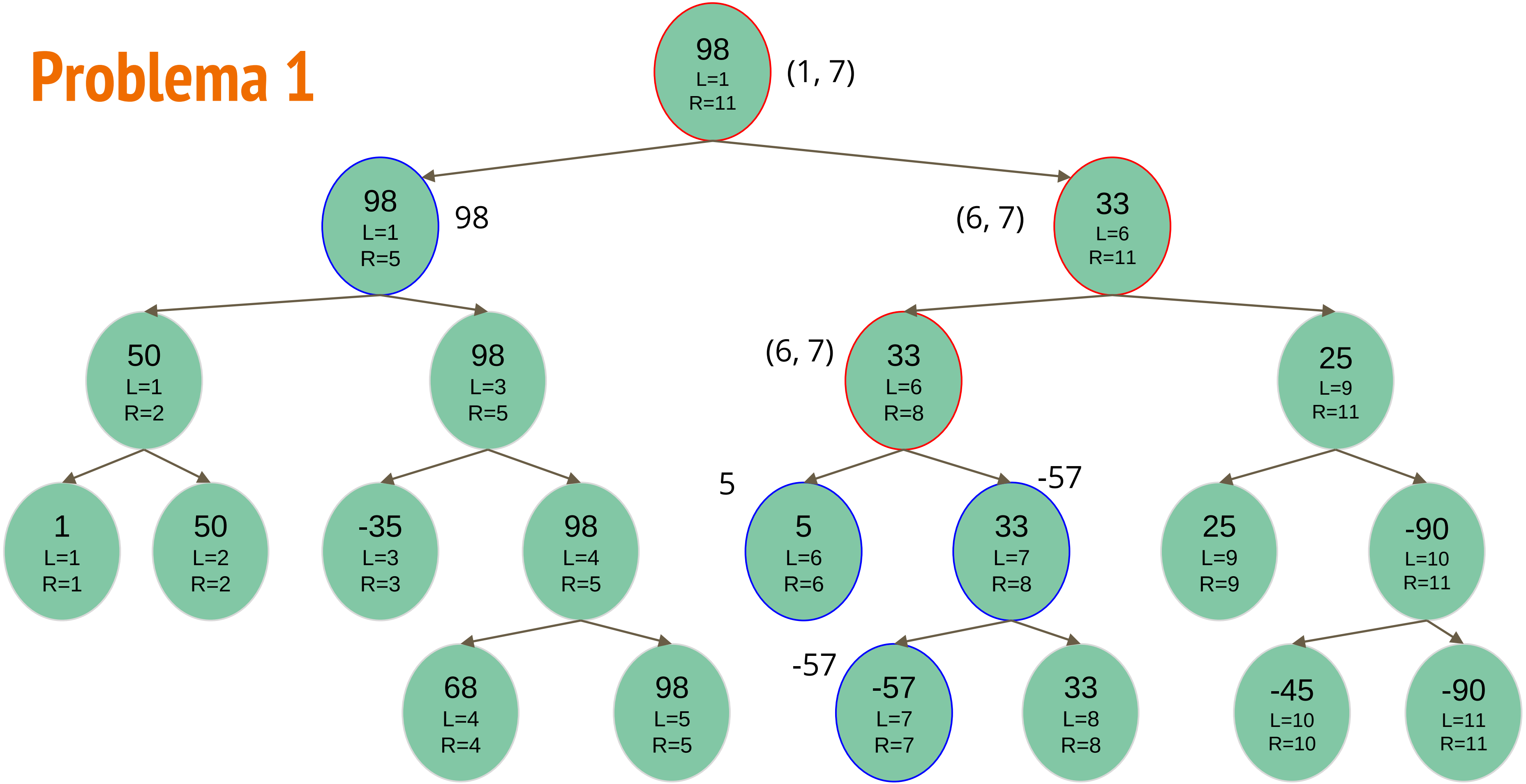

98
L=1
R=11
(1, 7)
98
L=1
R=5
98
33
L=6
R=11
(6, 7)
50
L=1
R=2
98
L=3
R=5
33
L=6
R=8
(6, 7)
25
L=9
R=11
1
L=1
R=1
50
L=2
R=2
-35
L=3
R=3
98
L=4
R=5
5
L=6
R=6
5
33
L=7
R=8
(7, 7)
25
L=9
R=9
-90
L=10
R=11
68
L=4
R=4
98
L=5
R=5
-57
L=7
R=7
(7, 7)
33
L=8
R=8
-45
L=10
R=10
-90
L=11
R=11

Problema 1

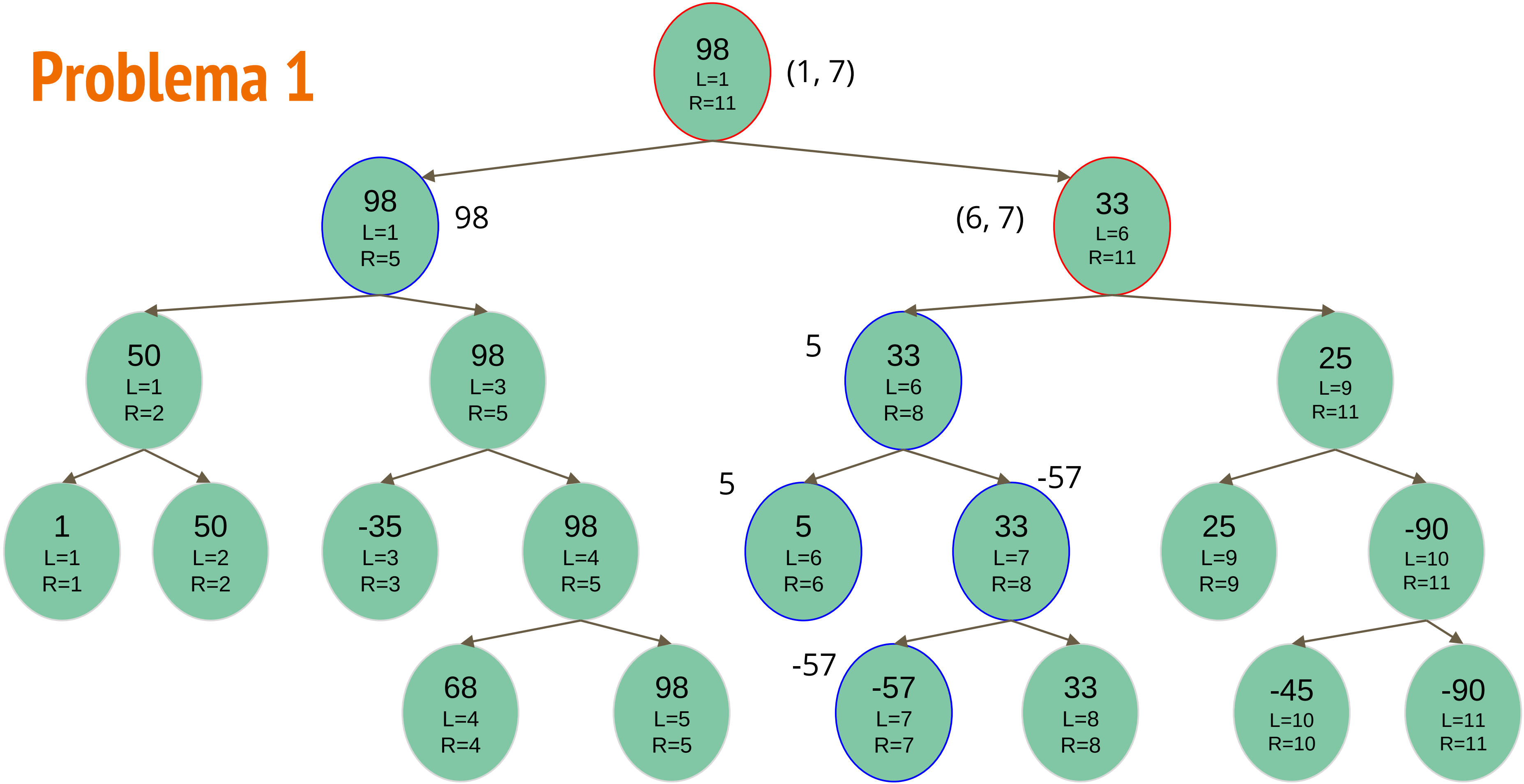


98
L=1
R=11
(1, 7)
98
L=1
R=5
98
(6, 7)
33
L=6
R=11
50
L=1
R=2
98
L=3
R=5
(6, 7)
33
L=6
R=8
25
L=9
R=11
1
L=1
R=1
50
L=2
R=2
-35
L=3
R=3
98
L=4
R=5
5
5
L=6
R=6
(7, 7)
33
L=7
R=8
25
L=9
R=9
-90
L=10
R=11
68
L=4
R=4
98
L=5
R=5
-57
-57
L=7
R=7
33
L=8
R=8
-45
L=10
R=10
-90
L=11
R=11

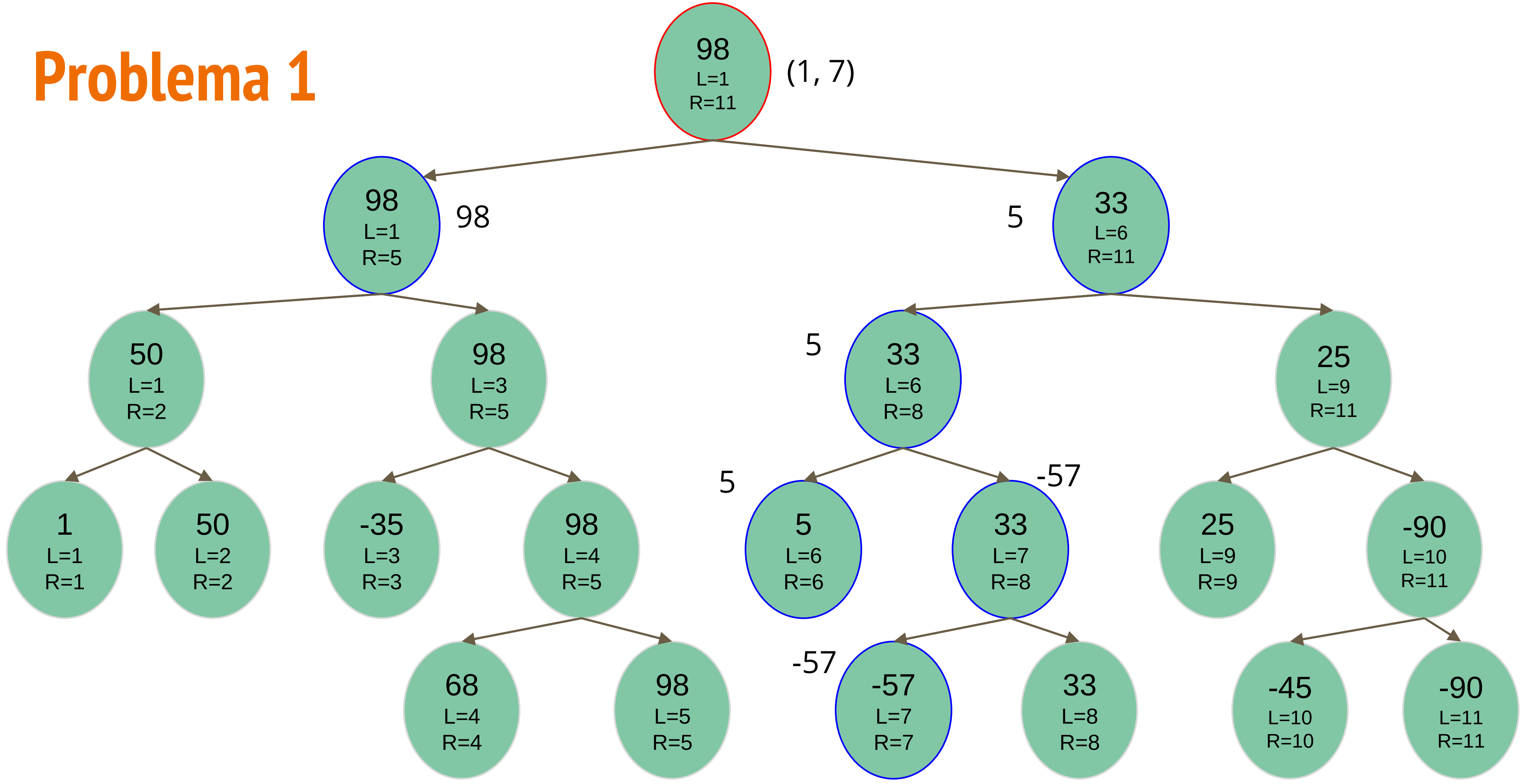
Problema 1


98 L=1 R=11
(1, 7)
98 L=1 R=5
98
(6, 7)
33 L=6 R=11
50 L=1 R=2
98 L=3 R=5
(6, 7)
33 L=6 R=8
25 L=9 R=11
1 L=1 R=1
50 L=2 R=2
-35 L=3 R=3
98 L=4 R=5
5
5 L=6 R=6
-57
33 L=7 R=8
25 L=9 R=9
-90 L=10 R=11
68 L=4 R=4
98 L=5 R=5
-57
-57 L=7 R=7
33 L=8 R=8
-45 L=10 R=10
-90 L=11 R=11

Problema 1

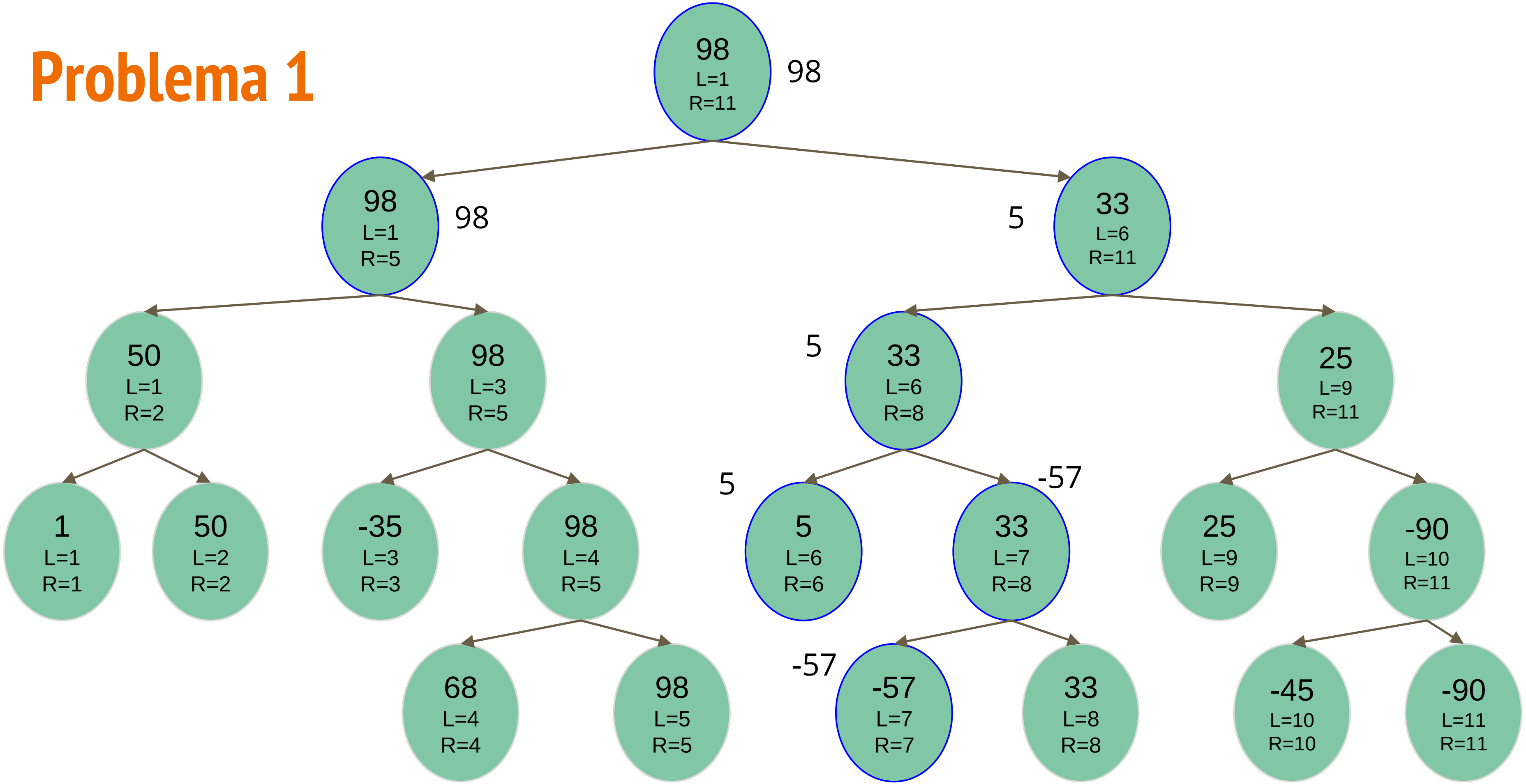

98
L=1
R=11
(1, 7)
98
L=1
R=5
98
(6, 7)
33
L=6
R=11
50
L=1
R=2
98
L=3
R=5
5
33
L=6
R=8
25
L=9
R=11
1
L=1
R=1
50
L=2
R=2
-35
L=3
R=3
98
L=4
R=5
5
5
L=6
R=6
-57
33
L=7
R=8
25
L=9
R=9
-90
L=10
R=11
68
L=4
R=4
98
L=5
R=5
-57
-57
L=7
R=7
33
L=8
R=8
-45
L=10
R=10
-90
L=11
R=11

Problema 1



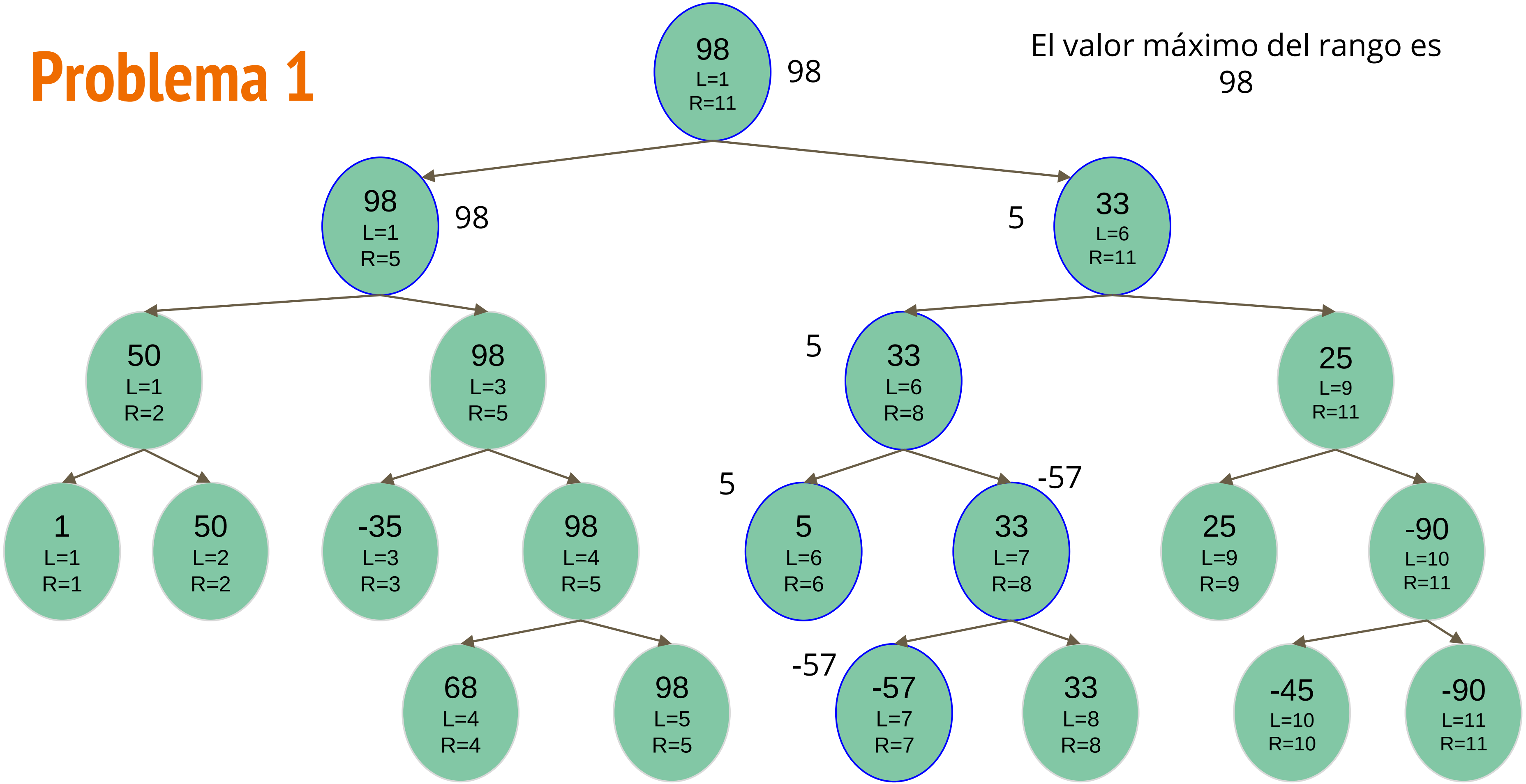
98
L=1
R=11
(1, 7)
98
L=1
R=5
98
5
33
L=6
R=11
50
L=1
R=2
98
L=3
R=5
5
33
L=6
R=8
25
L=9
R=11
1
L=1
R=1
50
L=2
R=2
-35
L=3
R=3
98
L=4
R=5
5
5
L=6
R=6
-57
33
L=7
R=8
25
L=9
R=9
-90
L=10
R=11
68
L=4
R=4
98
L=5
R=5
-57
-57
L=7
R=7
33
L=8
R=8
-45
L=10
R=10
-90
L=11
R=11

Problema 1


98
L=1
R=11
98
98
L=1
R=5
98
5
33
L=6
R=11
50
L=1
R=2
98
L=3
R=5
5
33
L=6
R=8
25
L=9
R=11
1
L=1
R=1
50
L=2
R=2
-35
L=3
R=3
98
L=4
R=5
5
5
L=6
R=6
-57
33
L=7
R=8
25
L=9
R=9
-90
L=10
R=11
68
L=4
R=4
98
L=5
R=5
-57
-57
L=7
R=7
33
L=8
R=8
-45
L=10
R=10
-90
L=11
R=11

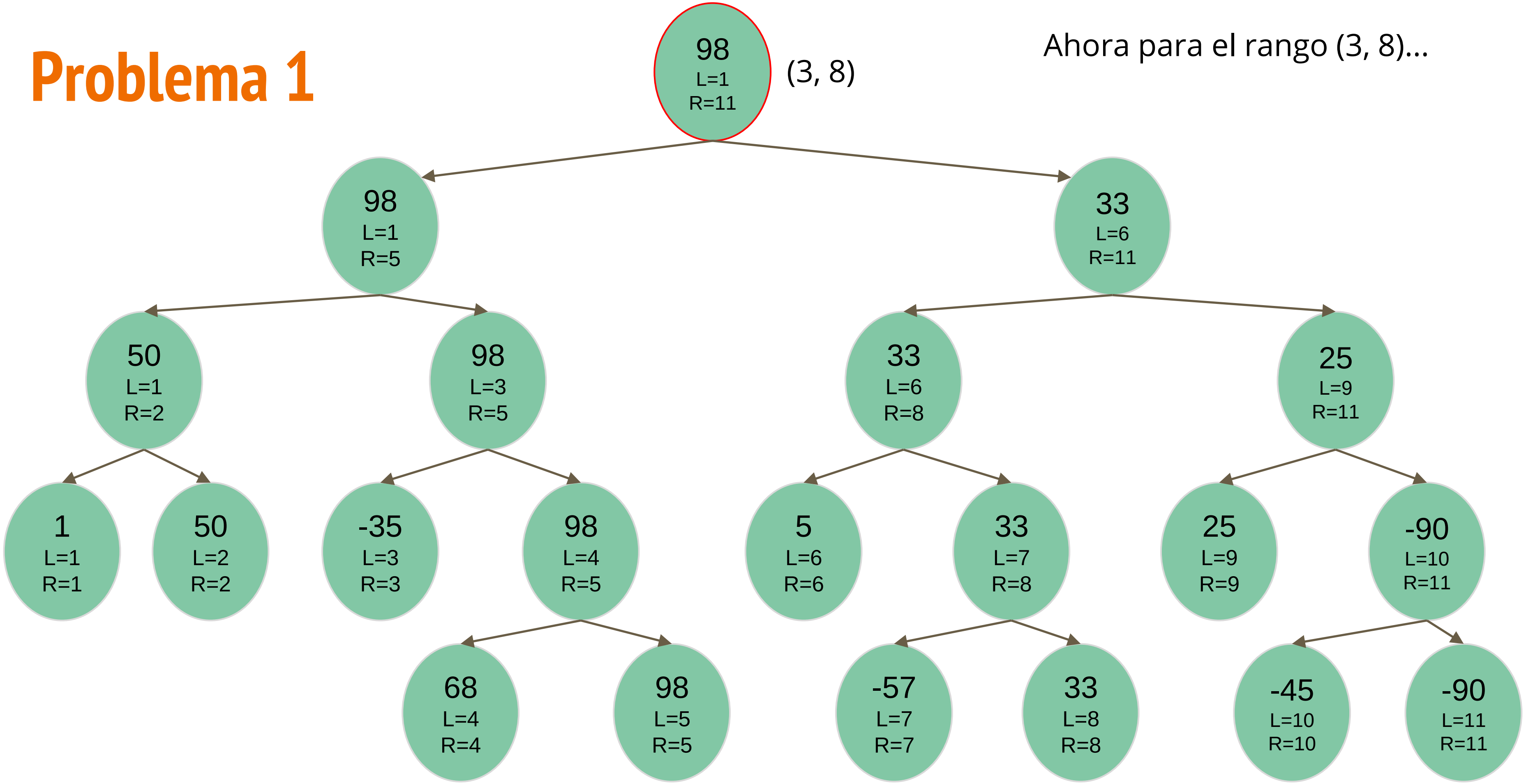
Problema 1

El valor máximo del rango es
98



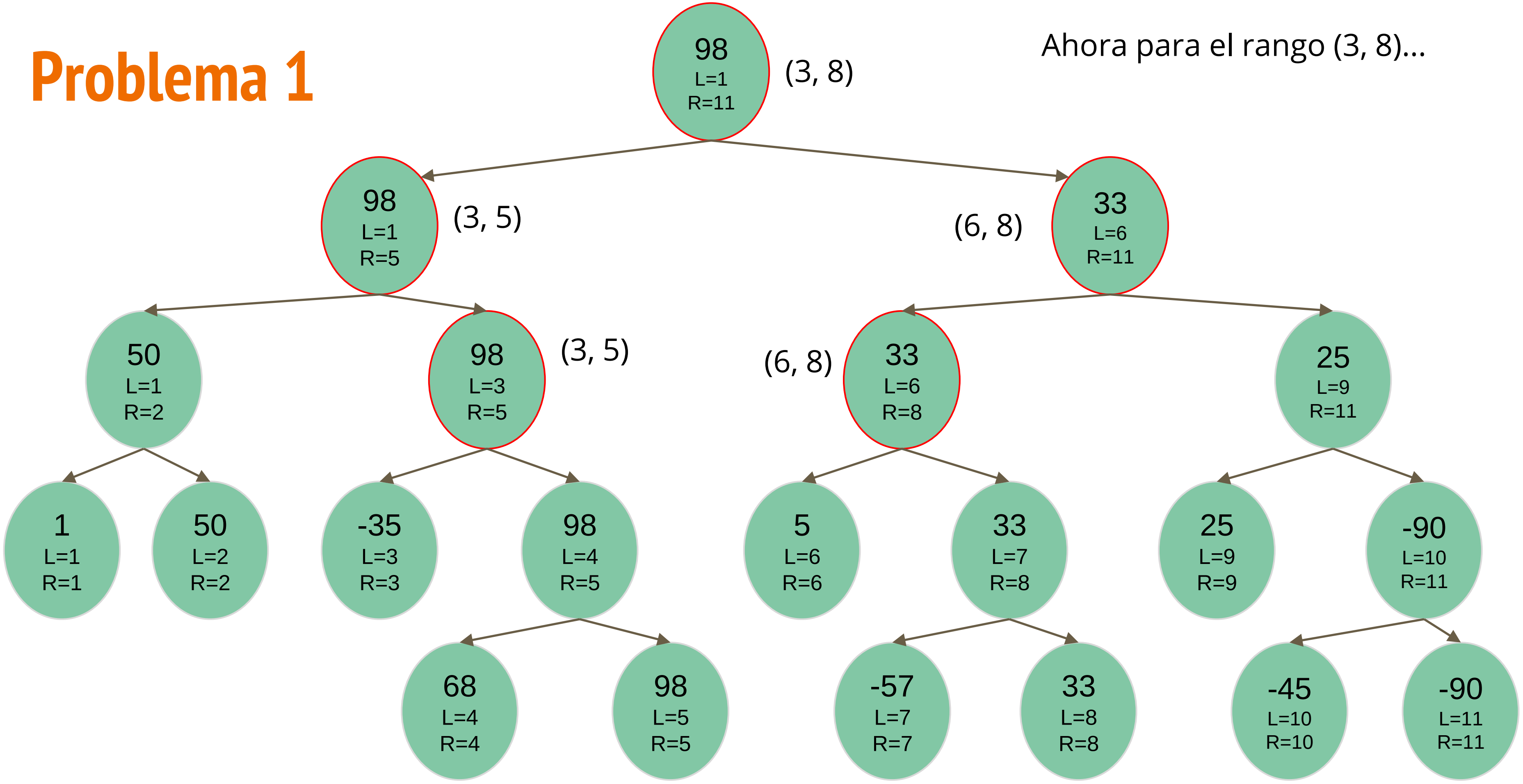
Problema 1

Ahora para el rango (3, 8)...



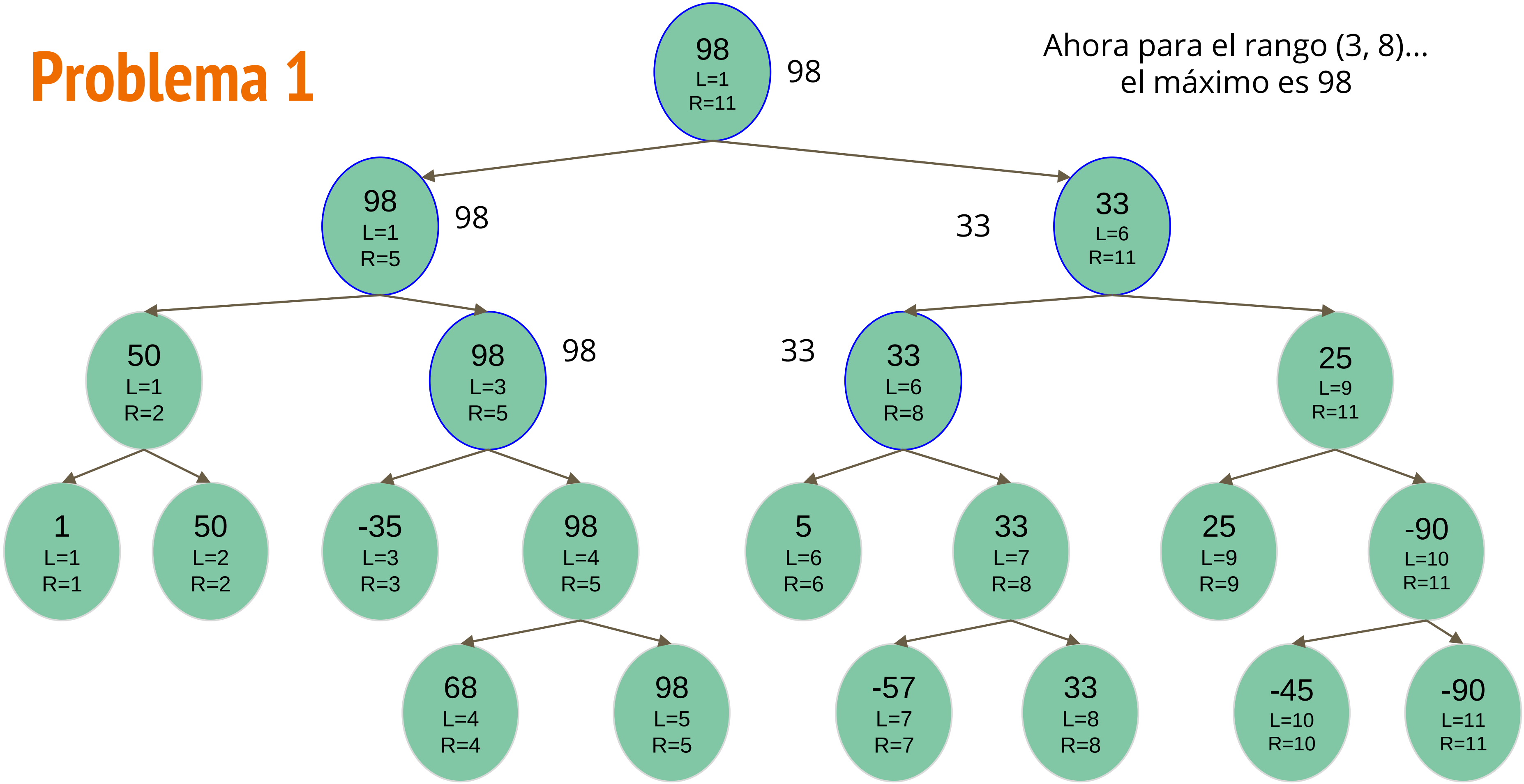
Problema 1

Ahora para el rango (3, 8)...



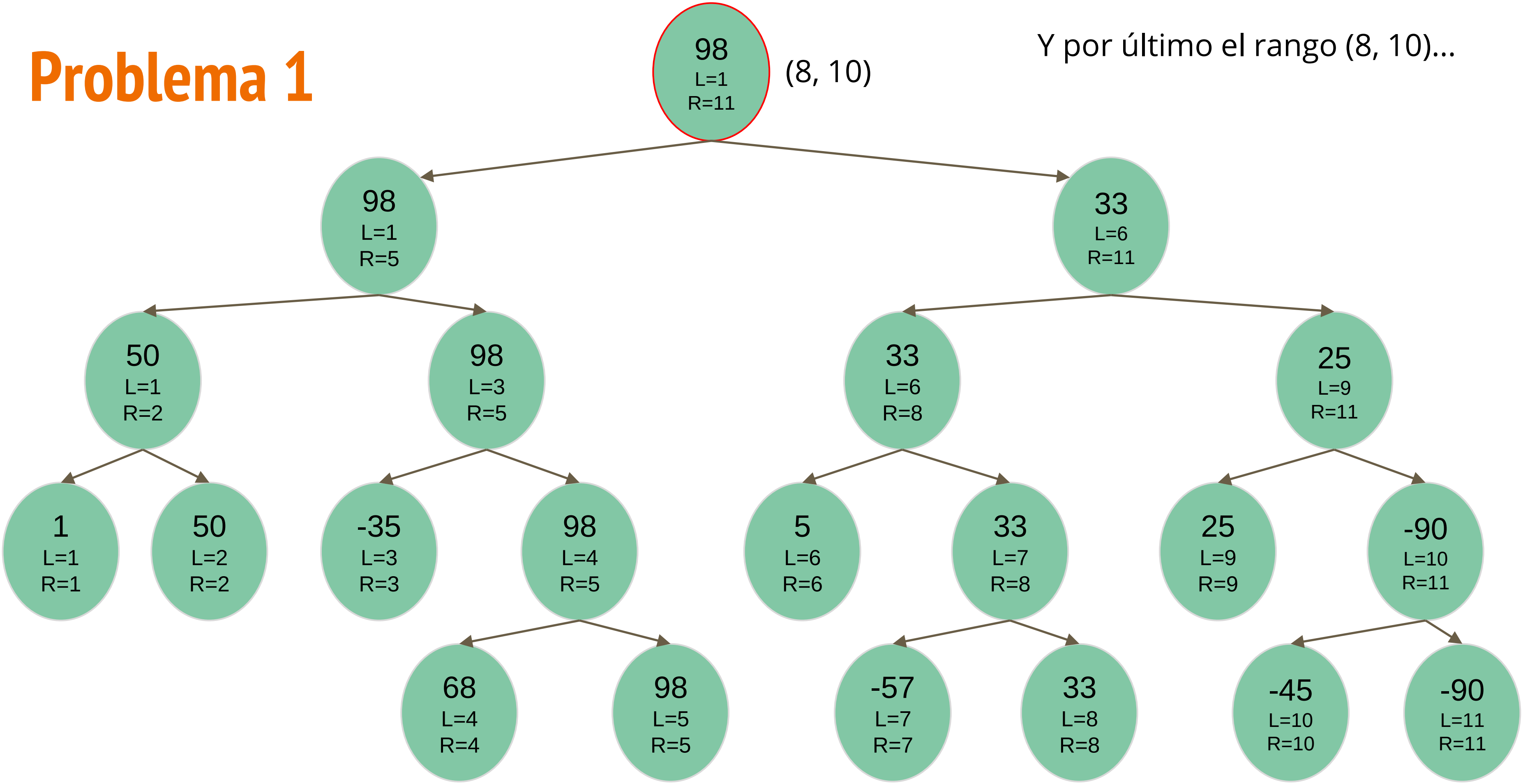
Problema 1

Ahora para el rango (3, 8)...
el máximo es 98



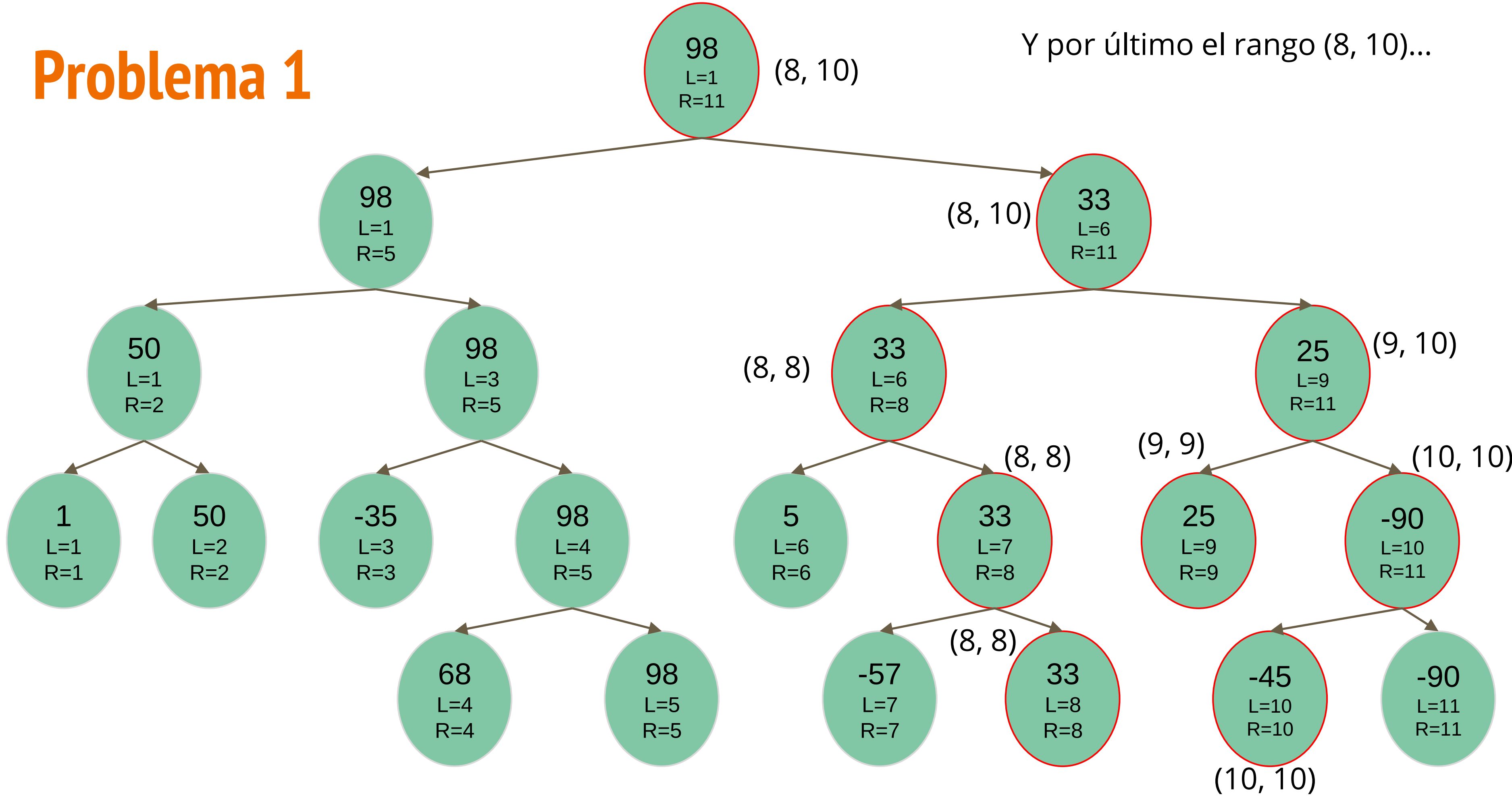
Problema 1

Y por último el rango (8, 10)...



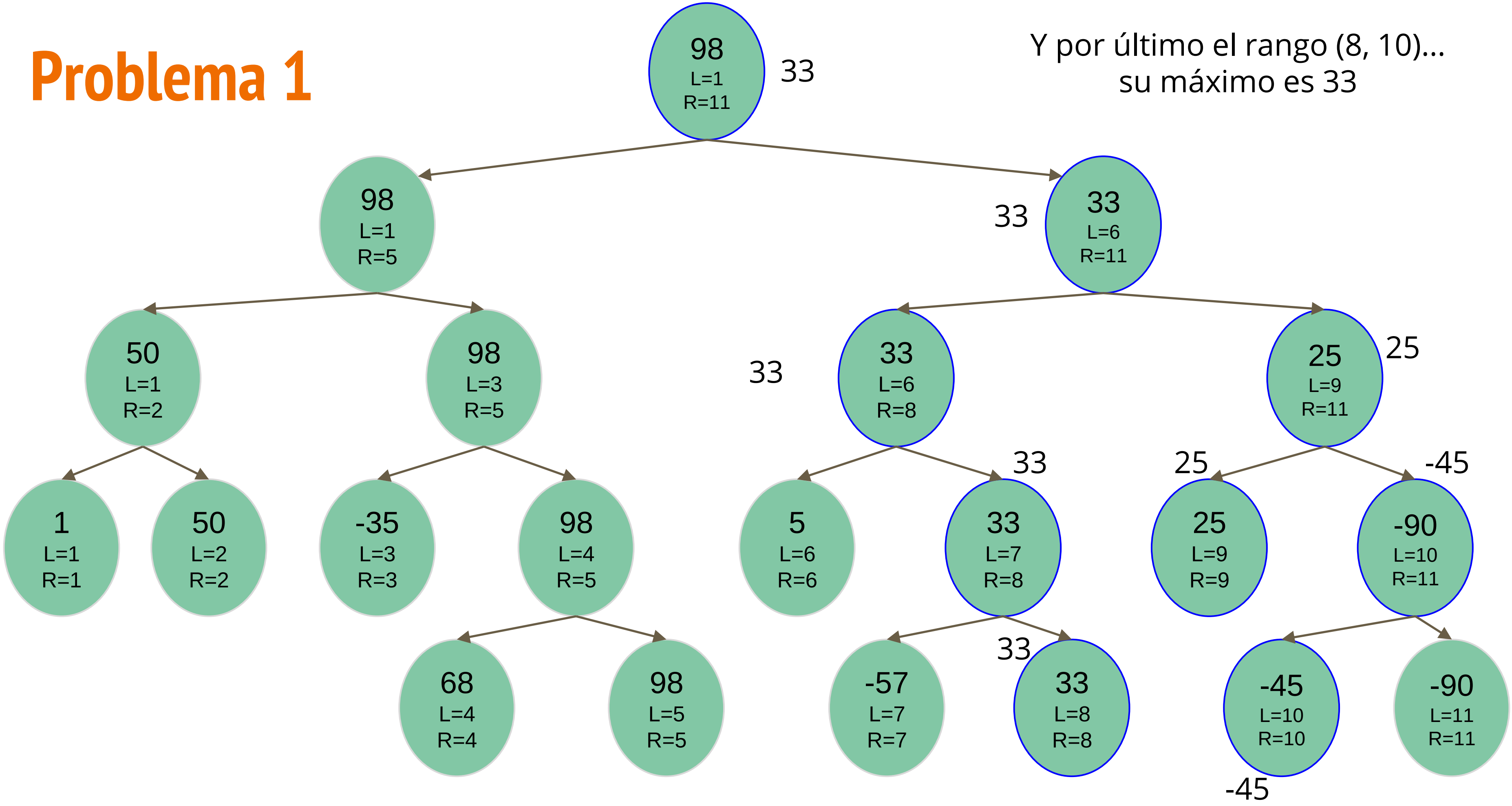
Problema 1

Y por último el rango (8, 10)...



Problema 1

Y por último el rango (8, 10)...
su máximo es 33



Problema 1

A partir del arreglo A = [1, 50, -35, 68, 98, 5, -57, 33, 25, -45, -90], construye un Segment Tree para encontrar el número máximo dentro de un subarreglo de A y responde a las siguientes queries: (1, 7), (3, 8), (8, 10).

- El máximo en (1, 7) es 98
- El máximo en (3, 8) es 98
- El máximo en (8, 10) es 33

Problema 1

A partir del arreglo A = [1, 50, -35, 68, 98, 5, -57, 33, 25, -45, -90], construye un Segment Tree para encontrar el número máximo dentro de un subarreglo de A y responde a las siguientes queries: (1, 7), (3, 8), (8, 10).

- El máximo en (1, 7) es 98
- El máximo en (3, 8) es 98
- El máximo en (8, 10) es 33

Problema 2

A partir del arreglo $A = [a_1 , a_2 , \dots , a_n]$ tales que $a_i \geq 0$, construye un Segment Tree que pueda responder el siguiente tipo de consultas:

1. La posición del k-ésimo 0 en el arreglo
2. El prefijo (subarreglo) más corto tal que la suma de sus elementos sean al menos un número c

Problema 2.1

1. La posición del k-ésimo 0 en el arreglo

Problema 2.1

1. La posición del k-ésimo 0 en el arreglo

   ¿Cómo resolvemos este problema?

Problema 2.1

1. La posición del k-ésimo 0 en el arreglo

   ¿Cómo resolvemos este problema?
   En lugar de guardar el mínimo o el máximo, podemos guardar la cantidad de 0s

Problema 2.1

1. La posición del k-ésimo 0 en el arreglo

   ¿Cómo resolvemos este problema?
   En lugar de guardar el mínimo o el máximo, podemos guardar la cantidad de 0s

   ¿Pero de qué nos sirve esto?

Problema 2.1

Supongamos que tenemos la raíz de un Segment Tree de cantidad de 0s, y buscamos el k-ésimo 0 del arreglo.

Problema 2.1

Supongamos que tenemos la raíz de un Segment Tree de cantidad de 0s, y buscamos el k-ésimo 0 del arreglo.

Tenemos dos posibilidades:

1. El k-ésimo 0 está en el hijo izquierdo
2. el k-ésimo 0 está en el hijo derecho

Problema 2.1

Supongamos que tenemos la raíz de un Segment Tree de cantidad de 0s, y buscamos el k-ésimo 0 del arreglo.

Tenemos dos posibilidades:

1. El k-ésimo 0 está en el hijo izquierdo
2. El k-ésimo 0 está en el hijo derecho

¿Cómo sabemos en cuál está?

Problema 2.1

Supongamos que tenemos la raíz de un Segment Tree de cantidad de 0s, y buscamos el k-ésimo 0 del arreglo.

Tenemos dos posibilidades:

1. El k-ésimo 0 está en el hijo izquierdo
2. El k-ésimo 0 está en el hijo derecho

¿Cómo sabemos en cuál está?

Según la cantidad de 0s del hijo izquierdo

Problema 2.1

Si la cantidad de 0s bajo el hijo izquierdo es n:

Problema 2.1

Si la cantidad de 0s bajo el hijo izquierdo es n:

1. Si $k \leq n$, entonces el k-ésimo 0 está en el hijo izquierdo
2. Si $n < k$, entonces está en el hijo derecho

Luego, seguimos buscando recursivamente

Problema 2.1

Si la cantidad de 0s bajo el hijo izquierdo es n:

1. Si $k \le n$, entonces el k-ésimo 0 está en el hijo izquierdo
2. Si $n < k$, entonces está en el hijo derecho

Luego, seguimos buscando recursivamente

Algo importante, si buscamos en el hijo derecho, ahora vamos a buscar el (k-n)-ésimo 0 bajo el Segment Tree derecho

Problema 2.1

Si la cantidad de 0s bajo el hijo izquierdo es n:

1. Si $k \leq n$, entonces el k-ésimo 0 está en el hijo izquierdo
2. Si $n < k$, entonces está en el hijo derecho

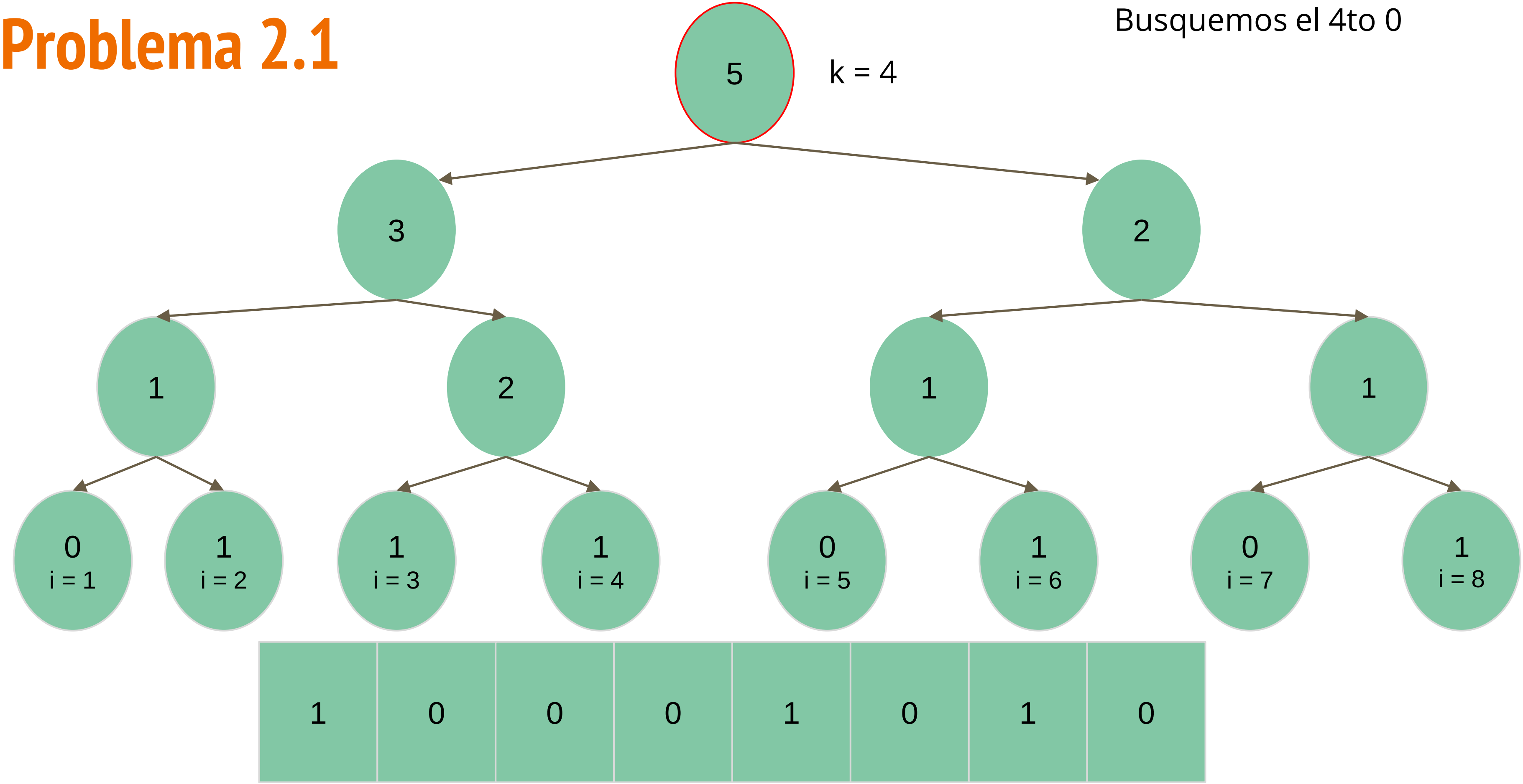
Luego, seguimos buscando recursivamente

Algo importante, si buscamos en el hijo derecho, ahora vamos a buscar el (k-n)-ésimo 0 bajo el Segment Tree derecho

Veamos un ejemplo:

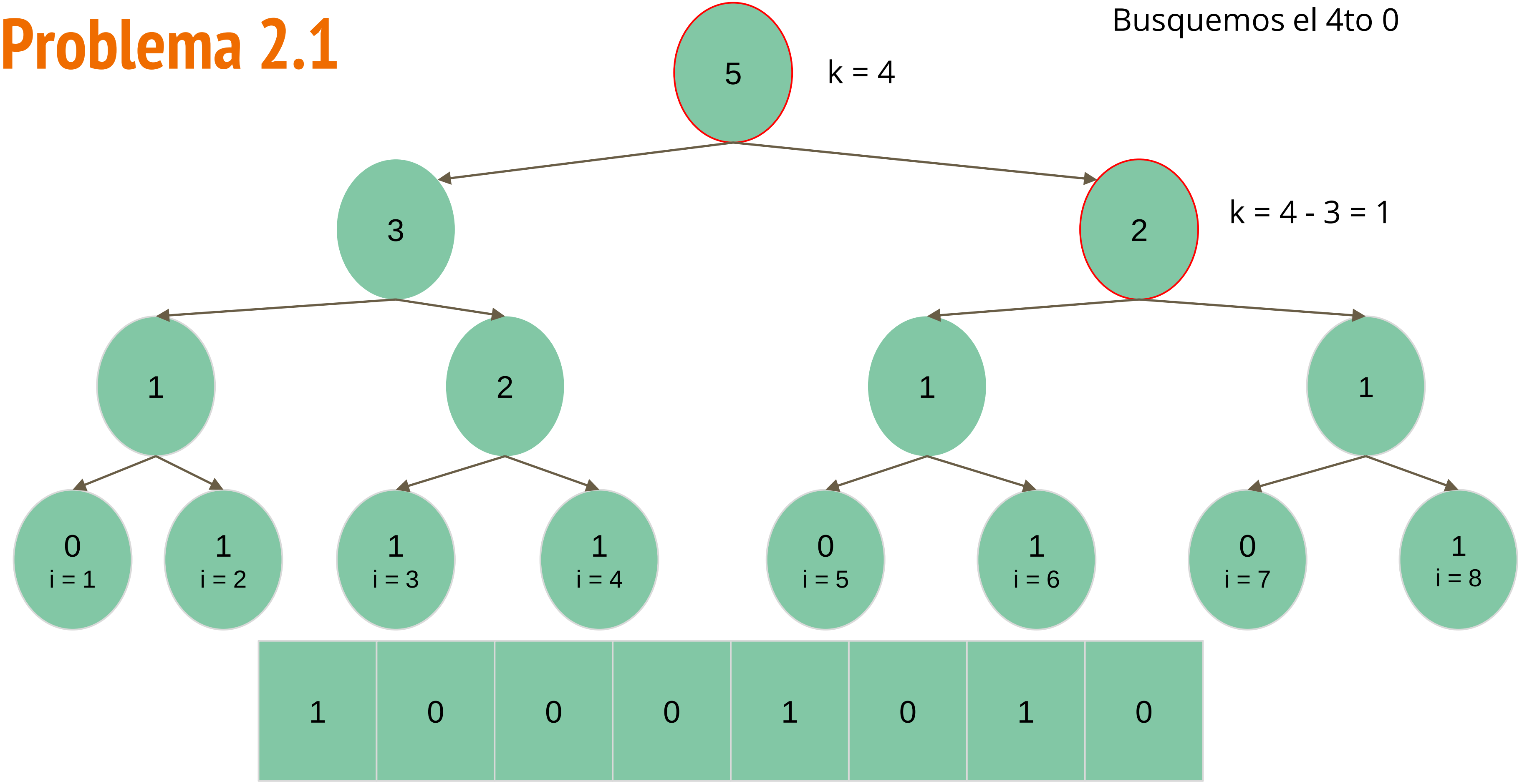
Problema 2.1

Busquemos el 4to 0



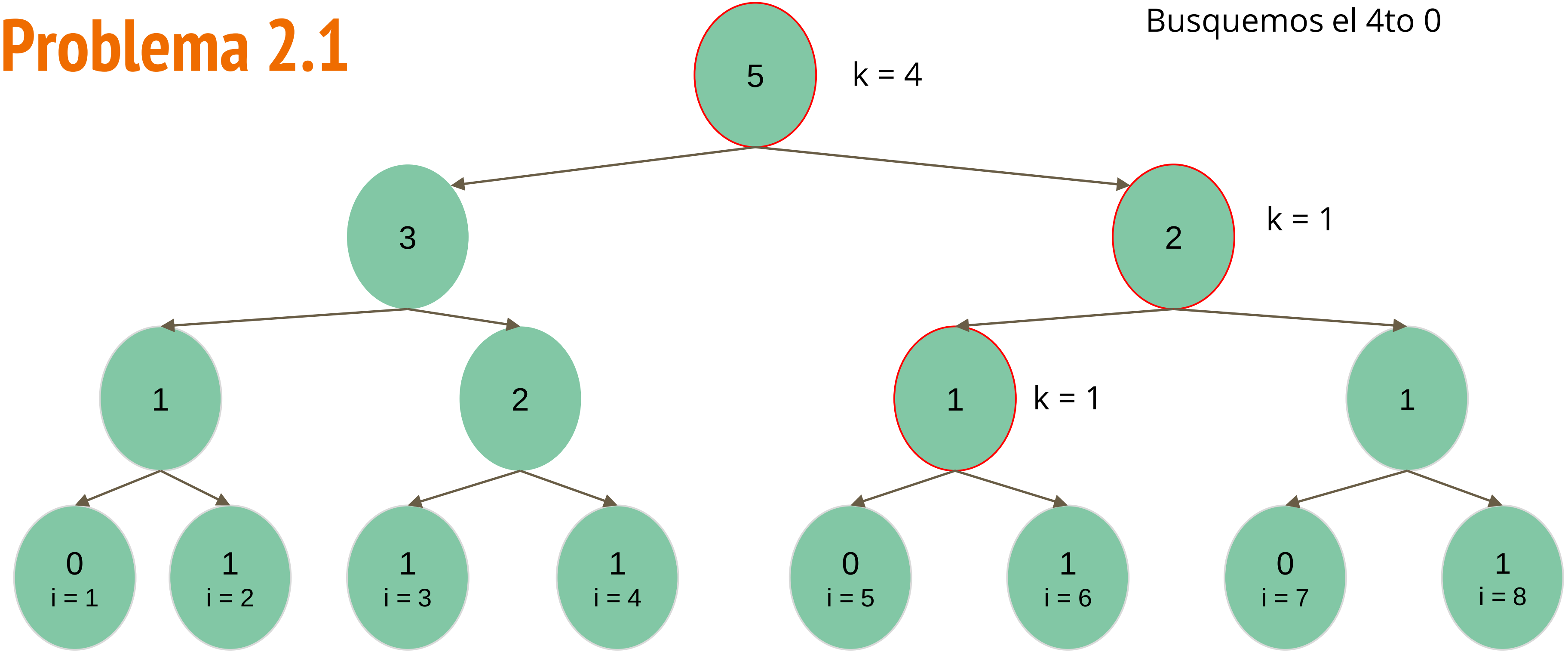
Problema 2.1

Busquemos el 4to 0



Problema 2.1

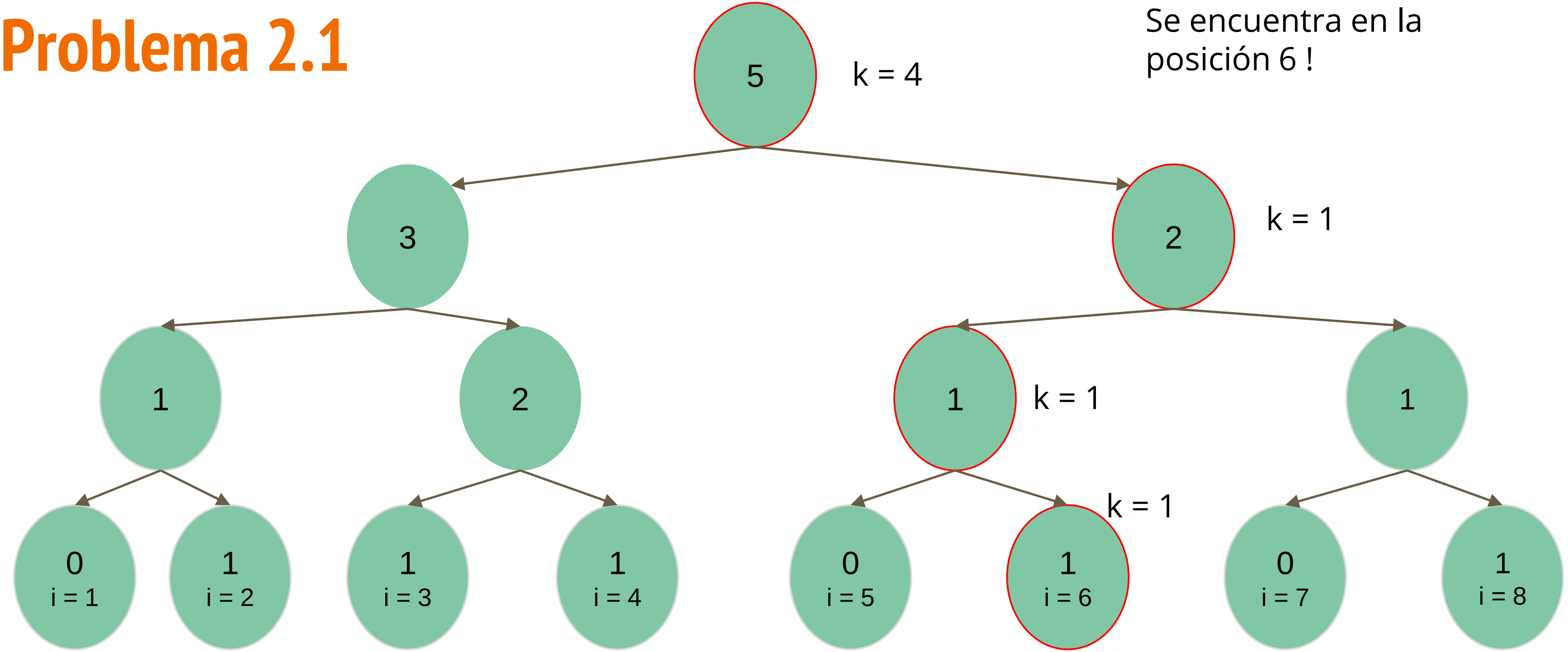
Busquemos el 4to 0



1	0	0	0	1	0	1	0

Problema 2.1

Se encuentra en la posición 6 !



1	0	0	0	1	0	1	0

Problema 2.2

2. El prefijo más corto que suma al menos c (en un arreglo de elementos no negativos)

Vamos a reusar ideas de 2.1

- Vamos a tener la suma del segmento en cada nodo
- En cada nodo preguntamos la suma del nodo izquierdo, digamos que es s
    - Si $s \geq c$ entonces el prefijo no está contenido en el nodo derecho
    - Si $s < c$ entonces el prefijo si está parcialmente contenido en el nodo derecho

Notar que si $s < c$ entonces se tiene en el caso recursivo se toma $c' = c - s$

Problema 2.2 (Ejemplo)

Problema 3

A partir del arreglo de números $A=[a_1,a_2,\ldots,a_n]$ (notar que los a_i pueden ser negativos), construya un Segment Tree que pueda responder a la consulta: ¿Cúal es el subsegmento de suma máxima en el subarreglo [L,R]?

Problema 3

Usando un Segment Tree podemos reducir el problema al siguiente: dado dos subarreglos [L,i] y [i+1,R], ¿qué información de los subarreglos necesitamos para calcular la solución para el subarreglo [L,R]?

Problema 3

Dado dos subarreglos [L,i] y [i+1,R] vemos que el subsegmento de suma máxima del subarreglo [L,R] puede cumplir uno de los siguientes 3:

1. Puede estar completamente contenido en [L,i]
2. Puede estar completamente contenido en [i+1,R]
3. Puede estar parcialmente contenido en ambos

Notemos que para 1 y 2 solo necesitamos el subsegmento de suma máxima en cada subarreglo. Veremos 3 con más detalle.

Problema 3

Para el caso 3 (el subsegmento está contenido parcialmente en [L,i] y en [i+1,R]), veamos que el subsegmento es la unión de un sufijo de [L,i] y un prefijo de [i+1,R], más específicamente es la unión del sufijo de suma máxima de [L,i] y del prefijo de suma máxima de [i+1,R].

Problema 3

Juntando lo anterior vemos que hasta ahora necesitamos guardar la siguiente información:

- Prefijo de suma máxima
- Sufijo de suma máxima
- Subsegmento de suma máxima

Pero nos falta poder calcular el prefijo/sufijo de suma máxima.

Problema 3

Queremos poder calcular el prefijo/sufijo de suma máxima de un subarreglo [L,R] dado dos subarreglos [L,i] y [i+1,R]. Para calcular el prefijo/sufijo vemos que hay dos casos:

- El prefijo está totalmente contenido en [L,i] (similarmente el sufijo en [i+1,R])
- El prefijo está parcialmente contenido en [i+1,R] (similarmente el sufijo en [L,i])

Problema 3

Volviendo a la información que necesitamos:

1. Prefijo de suma máxima
2. Sufijo de suma máxima
3. Subsegmento de suma máxima
4. Suma del subarreglo

Problema 3

Veamos como calcular la información dado dos nodos de un Segment Tree u y v

1. max(u.prefix, u.sum+v.prefix)
2. max(v.suffix, v.sum+u.suffix)
3. max(u.ans,v.ans,u.suffix+v.prefix)
4. u.sum+v.sum

Problema 3 (Ejemplo)

Problema Propuesto

El segment tree visto en clases tiene updates puntuales y consultas por rango, ¿existirá alguna forma de modificarlo para que tengo updates por rango y consultas por rango?

Spoiler: Sí

Hint: dejen el trabajo para último momento