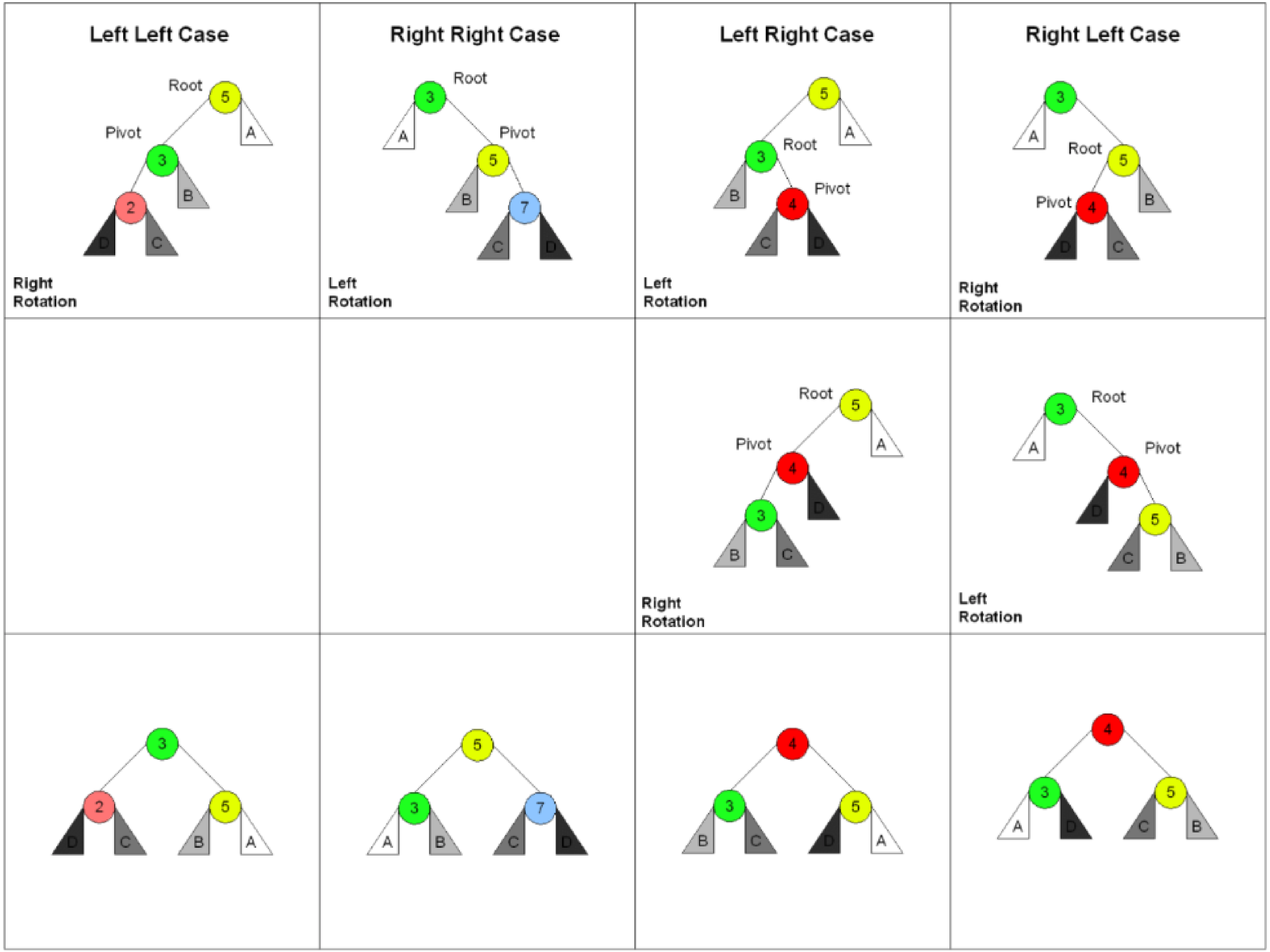
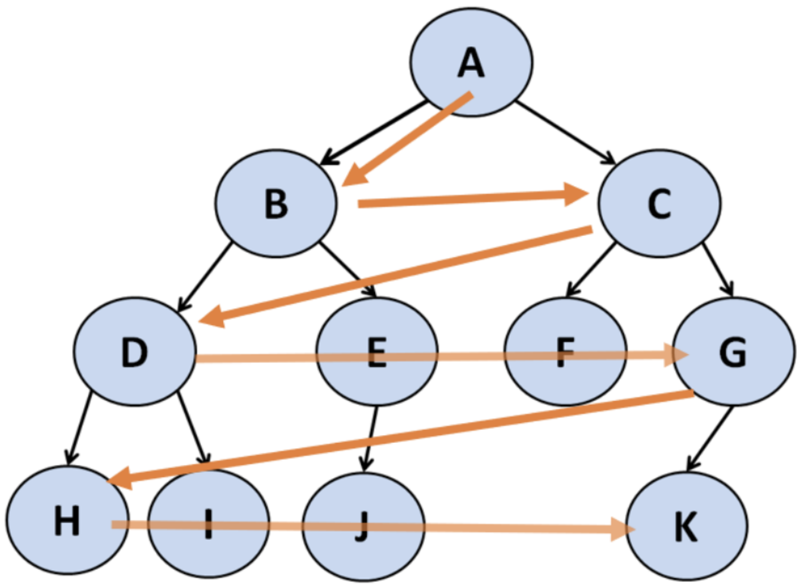

Left Left Case
Root
Pivot
5
3
2
A
B
C
D
Right Rotation
Right Right Case
Root
Pivot
3
5
7
A
B
C
D
Left Rotation
Left Right Case
Root
Pivot
5
3
4
A
B
C
D
Left Rotation
Right Left Case
Root
Pivot
3
5
4
A
B
C
D
Right Rotation
Root
Pivot
Right Rotation
Root
Pivot
Left Rotation

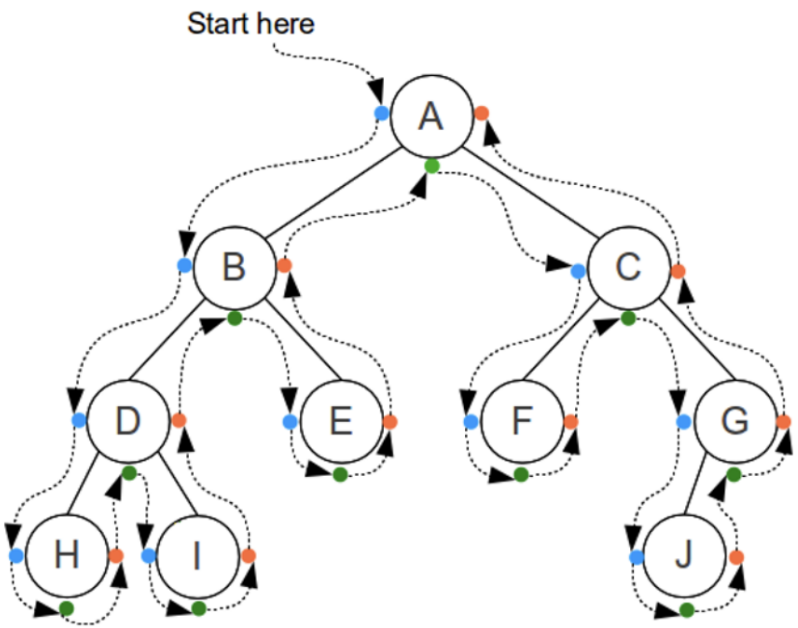
“Depth First” vs “Breadth First”
illustrated with arrows!

<u>Breadth First Traversal (BFT)</u>



Level-order: ABCDEFGHIJK

<u>3 types of Depth First Traversal (DFT)</u>

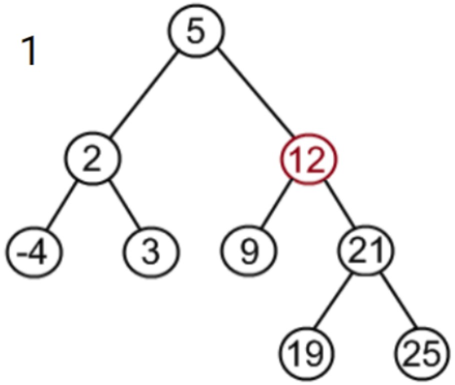


Pre-Order ABDHIECFGJ
In-Order HDIBEAFCJG
Post-Order HIDEBFJGCA

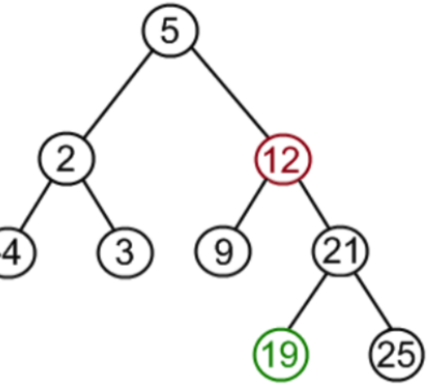
Node to be removed has two children - case IV

- find a minimum value in the right subtree;
- replace value of the node to be removed with found minimum. Now, right subtree contains a duplicate!
- apply remove to the right subtree to remove a duplicate.
- Notice, that the node with minimum value has no left child and, therefore, it's removal may result in first or second cases only.

1

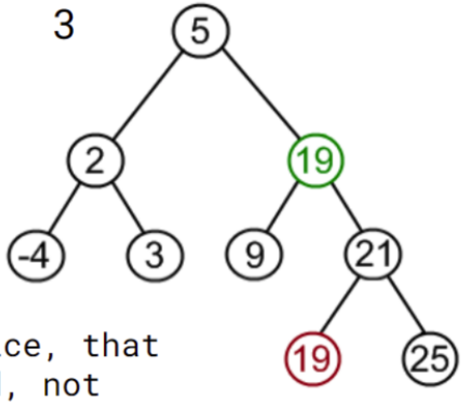


2



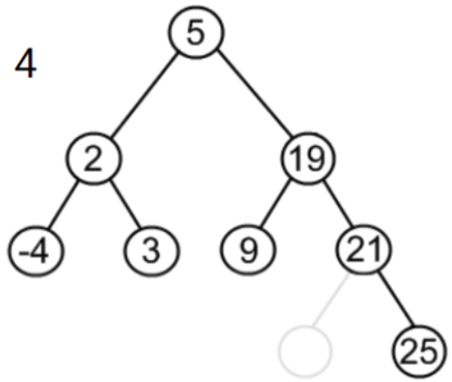
Find min in the right sub: 19.

3



Replace 12 with 19. Notice, that only values are replaced, not nodes. Now we have two nodes with the same value.

4



Remove 19 from the left subtree.

Inorder Traversal

Go to left-subtree

Visit Node

Go to right-subtree

Postorder Traversal

Go to left-subtree

Go to right-subtree

Visit Node

2^1	=	2	2^{11}	=	2,048	2^{21}	=	2,097,152
2^2	=	4	2^{12}	=	4,096	2^{22}	=	4,194,304
2^3	=	8	2^{13}	=	8,192	2^{23}	=	8,388,608
2^4	=	16	2^{14}	=	16,384	2^{24}	=	16,777,216
2^5	=	32	2^{15}	=	32,768	2^{25}	=	33,554,432
2^6	=	64	2^{16}	=	65,536	2^{26}	=	67,108,864
2^7	=	128	2^{17}	=	131,072	2^{27}	=	134,217,728
2^8	=	256	2^{18}	=	262,144	2^{28}	=	268,435,456
2^9	=	512	2^{19}	=	524,288	2^{29}	=	536,870,912
2^{10}	=	1,024	2^{20}	=	1,048,576	2^{30}	=	1,073,741,824

Preorder Traversal

Visit Node

Go to left-subtree

Go to right-subtree

There are 3 different types of Depth First Traversal (DFT)

- **Preorder**: visit the root, then traverse the subtrees from left to right

```
Visit node
Traverse (left child)
Traverse (right child)
```

$$+ +a * b\ c\ * d + e\ f$$

- **Inorder**: traverse the left subtree, then visit the root, then traverse the right subtree

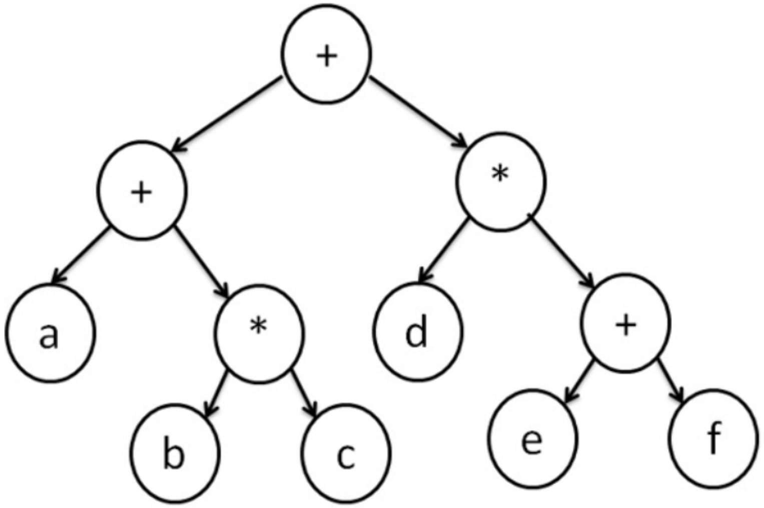
```
Traverse (left child)
Visit node
Traverse (right child)
```

$$a + b * c + d * (e + f)$$

- **Postorder**: traverse the subtrees from left to right, then visit the root

```
Traverse (left child)
Traverse (right child)
Visit node
```

$$a\ b\ c * +d\ e\ f + * +$$



Changing Limits

$$\sum_{j=1}^{n} a_j = \sum_{i=0}^{n-1} a_{i+1}$$

TABLE 2 Some Useful Summation Formulae.

Sum	*Closed Form*
$\sum_{k=0}^{n} ar^k \ (r \neq 0)$	$\frac{ar^{n+1} - a}{r - 1}, r \neq 1$
$\sum_{k=1}^{n} k$	$\frac{n(n+1)}{2}$
$\sum_{k=1}^{n} k^2$	$\frac{n(n+1)(2n+1)}{6}$
$\sum_{k=1}^{n} k^3$	$\frac{n^2(n+1)^2}{4}$
$\sum_{k=0}^{\infty} x^k, \lvert x \rvert < 1$	$\frac{1}{1-x}$
$\sum_{k=1}^{\infty} kx^{k-1}, \lvert x \rvert < 1$	$\frac{1}{(1-x)^2}$

The Master theorem

When the structure of the recursion is of the form $T\left(\frac{n}{2}\right)$ - as it was for the function we just considered – solving the recurrence relation is quite hard!

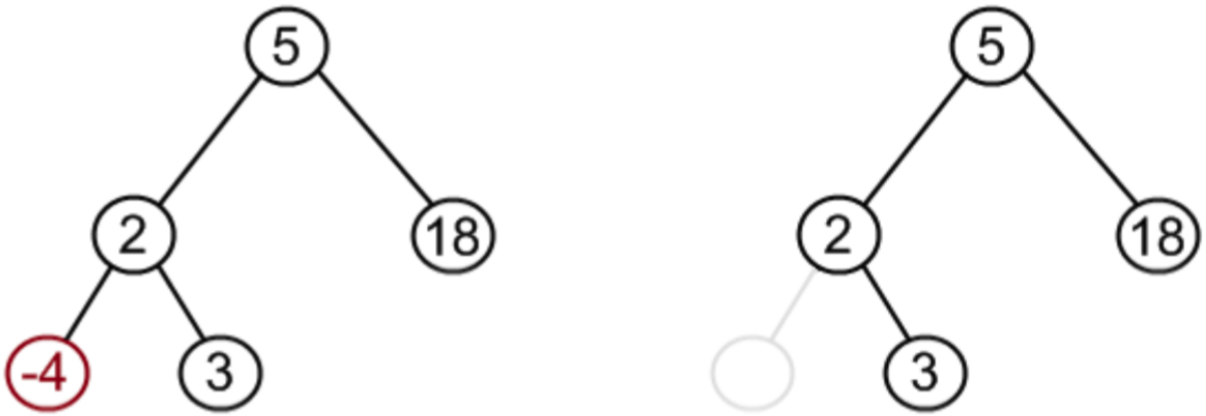
Luckily there's another method: We can use "The Master Theorem" ("Divide and conquer") in such cases:

$$T(N) = a \cdot T\left(\frac{N}{b}\right) + \overset{f(n)}{\Theta\left(N^k\right)}$$

$$T(N) = \begin{cases} \Theta\left(N^{\log_b(a)}\right) & \text{if } a > b^k \\ \Theta\left(N^k \cdot \log N\right) & \text{if } a = b^k \\ \Theta\left(N^k\right) & \text{if } a < b^k \end{cases}$$

Node to be removed has no children - Case I

This case is quite simple. Algorithm sets corresponding link of the parent to NULL and disposes the node.



Example – exponential growth

Imagine that you have a balance of 93 cents on your bank account on January 1^{st} year 2000. The account has an interest of 2.25% which is compounded annually,

a) After how many years will the account balance exceed 2$?

b) What will the account balance be on January 1^{st} year 3000?

<u>Solution</u>

We first find a functional expression for the account balance after x years:

$$f(x) = 0.93 \cdot 1.0225^x$$

a) We solve the equation $2 = 0.93 \cdot 1.0225^x$:

$$a)\quad 2 = 0.93 \cdot 1.0225^x \Leftrightarrow \frac{2}{0.93} = 1.0225^x \Leftrightarrow x = \log_{1.0225}\left(\frac{2}{0.93}\right)$$

We can evaluate the last expression by using the base-conversion formula $\log_b(m) = \frac{\log_q(m)}{\log_q(b)}$

with $b = 1.0225, m = \frac{2}{0.93}$ and (e.g.) $q = 10$: $\quad x = \log_{1.0225}\left(\frac{2}{0.93}\right) = \frac{\log(2/0.93)}{\log(1.0225)} = 34.4$

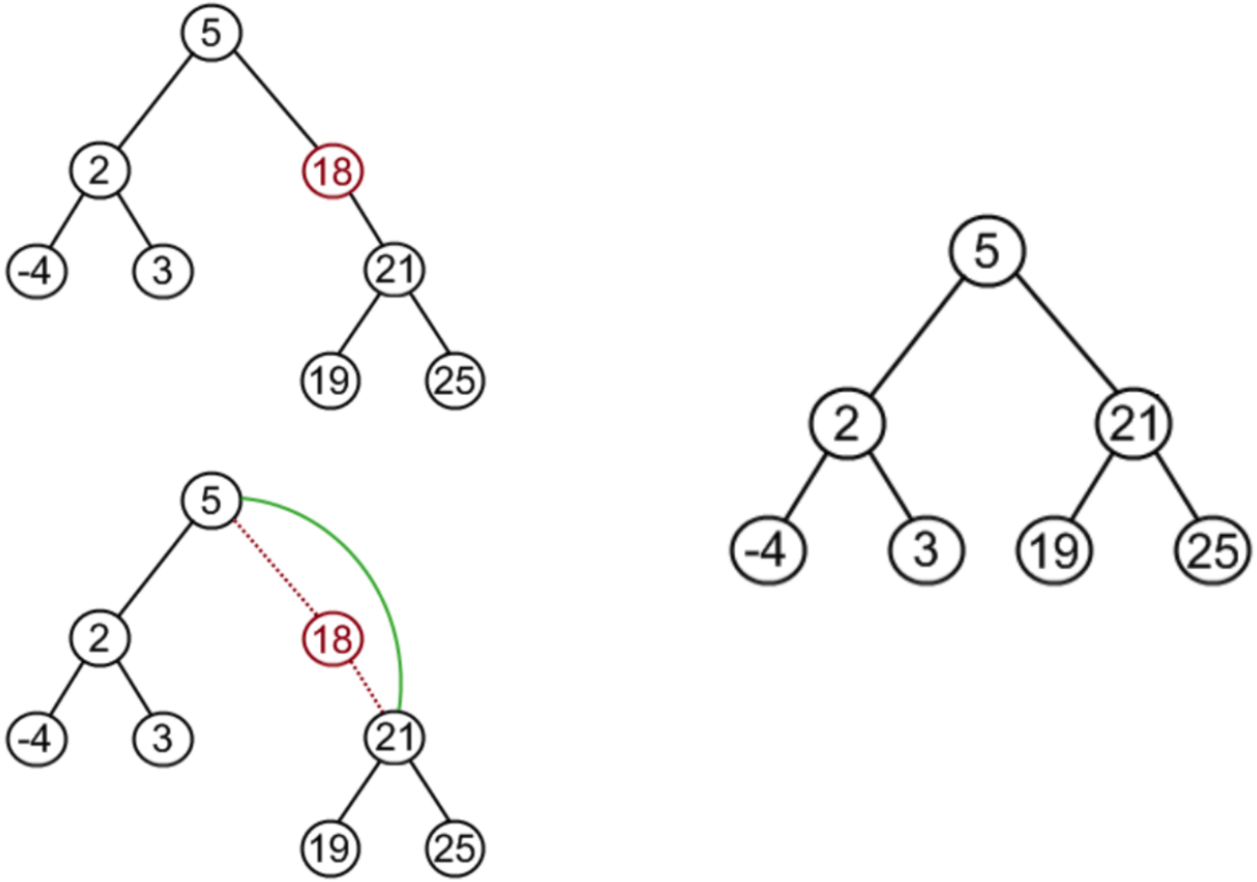
So the balance will exceed 2$ after 35 years.

b) We insert $x = 3000 - 2000 = 1000$: $f(1000) = 0.93 \cdot 1.0225^{1000} = 4.28 \cdot 10^9$

So the balance on the 1^{st} of January year 3000 will be 4.28 billion dollars.

Node to be removed has one child - cases II & III

It this case, node is cut from the tree and algorithm links single child (with it's subtree) directly to the parent of the removed node.



Quadratic probing

Example: $P(i) = i^2$

Insert the number 76, 40, 48 and 5 into a hash table of size 7 using the hash function $H(k) = k \text{ mod } 7$. In case of collisions, use quadratic probing with $P(i) = i^2$

$H(76) = (76 \text{ mod } 7 + P(0)) \text{ mod } 7 = 6$

$H(40) = (40 \text{ mod } 7 + P(0)) \text{ mod } 7 = 5$

$H(48) = (48 \text{ mod } 7 + P(0)) \text{ mod } 7 = 6$

Collision! Increase i and try again:

$H(48) = (48 \text{ mod } 7 + P(1)) \text{ mod } 7 = 0$

$H(5) = (5 \text{ mod } 7 + P(0)) \text{ mod } 7 = 5$

Collision! Increase i and try again:

$H(5) = (5 \text{ mod } 7 + P(1)) \text{ mod } 7 = 6$

Collision! Increase i and try again:

$H(5) = (5 \text{ mod } 7 + P(2)) \text{ mod } 7 = 2$

48		5			40	76
0	1	2	3	4	5	6