Example

UNIVERSITY OF CALGARY

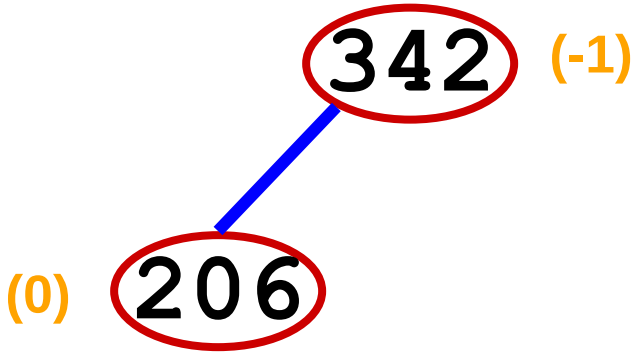
- Sketch the various stages involved in inserting the following keys, in the order given, into an AVL tree:

- 342, 206, 444, 523, 607, 301, 142, 183, 102, 157, 149

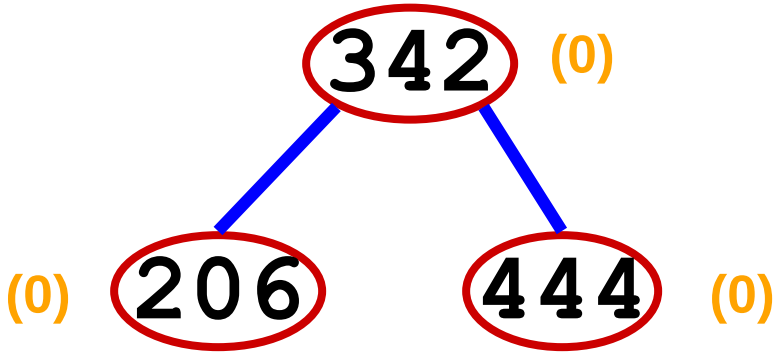
- First try to do it by yourself, and check with the soluton.

342 (0)

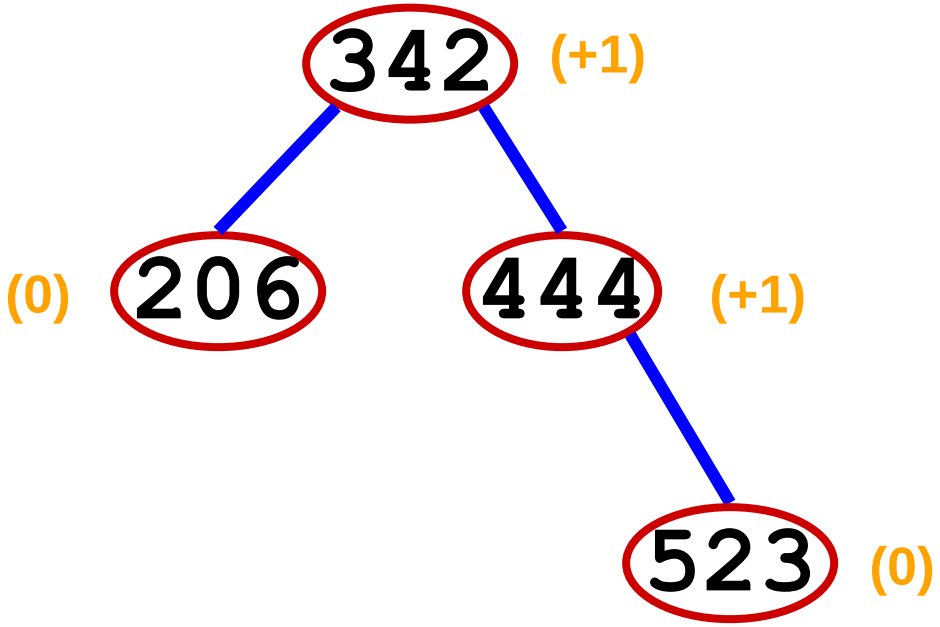
UNIVERSITY OF CALGARY



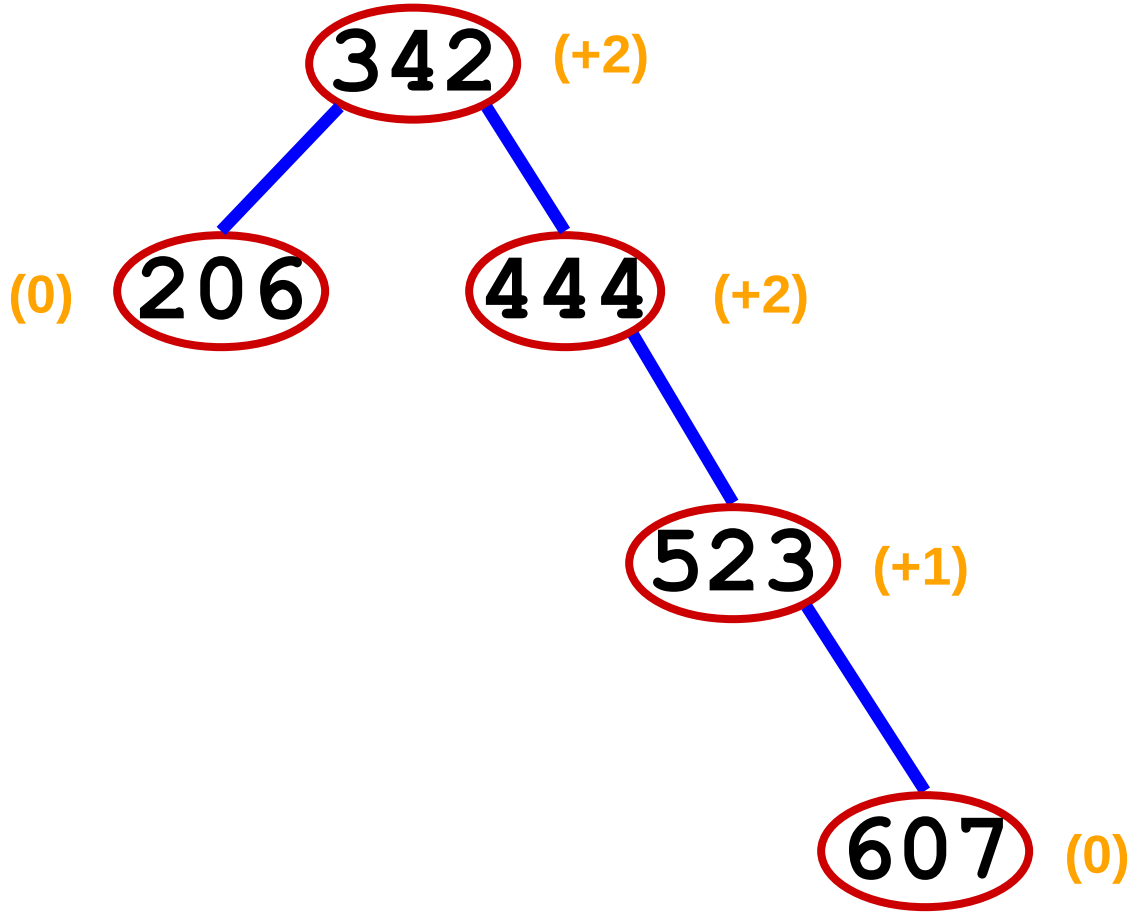
UNIVERSITY OF CALGARY



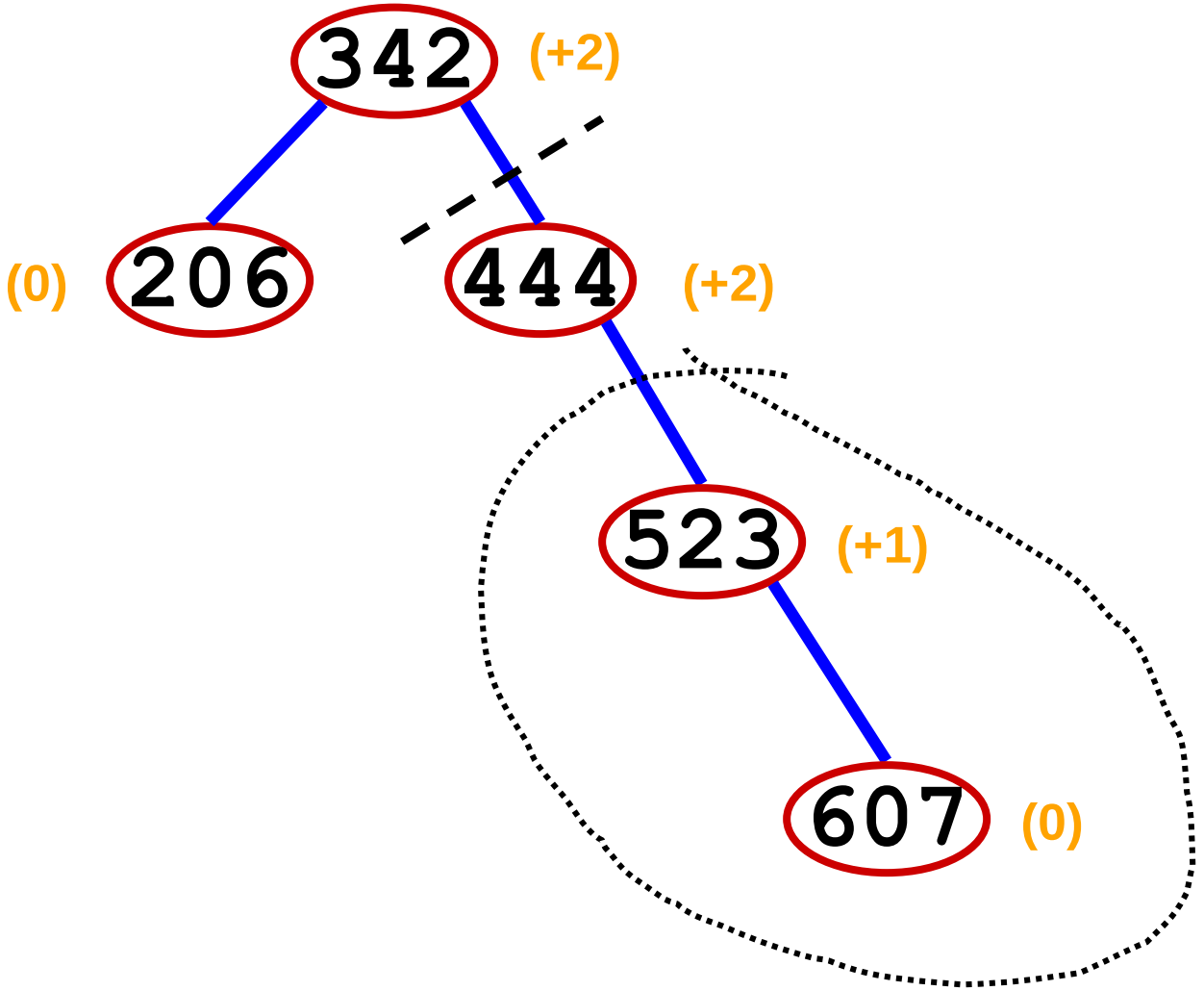
UNIVERSITY OF CALGARY



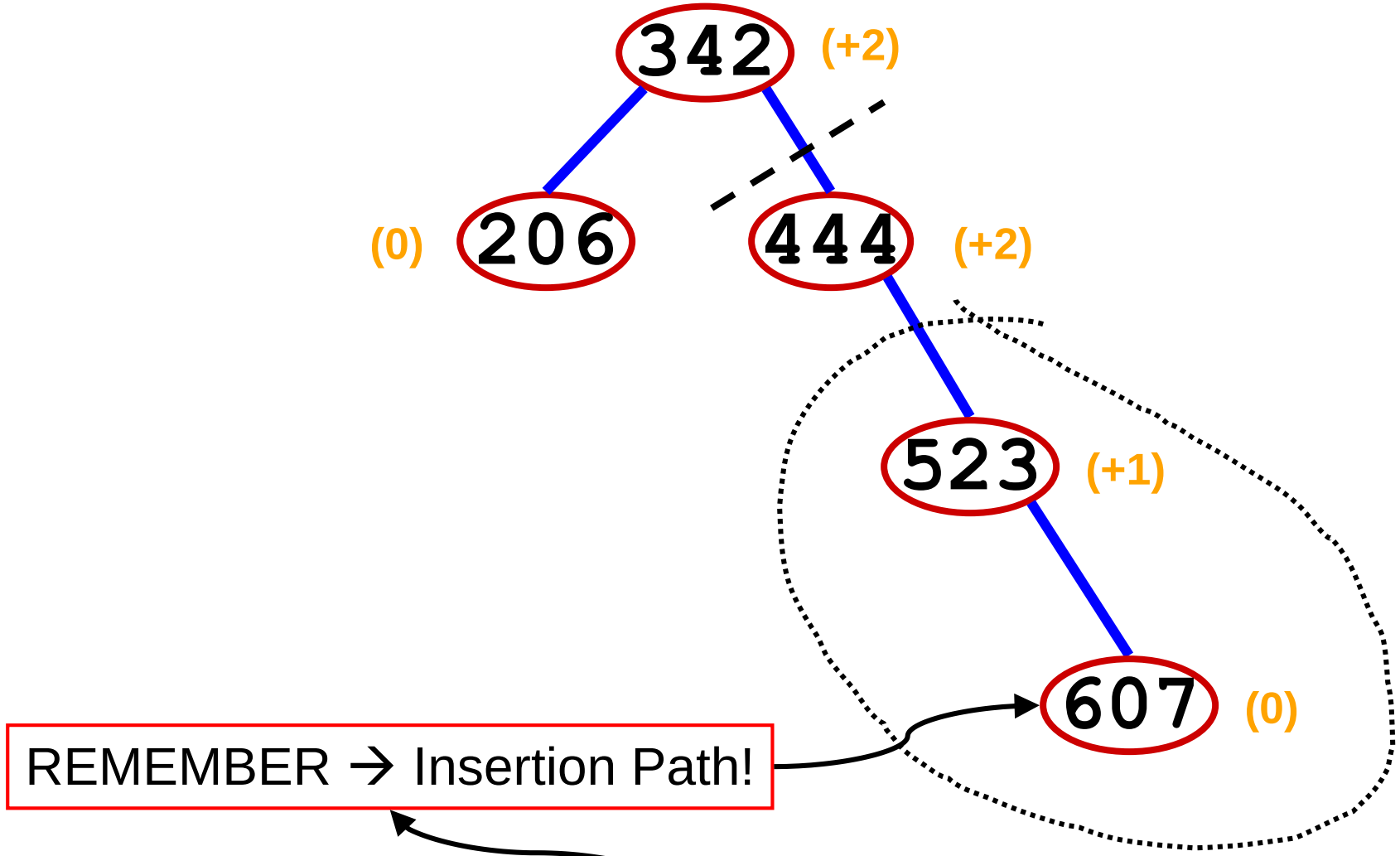
UNIVERSITY OF CALGARY



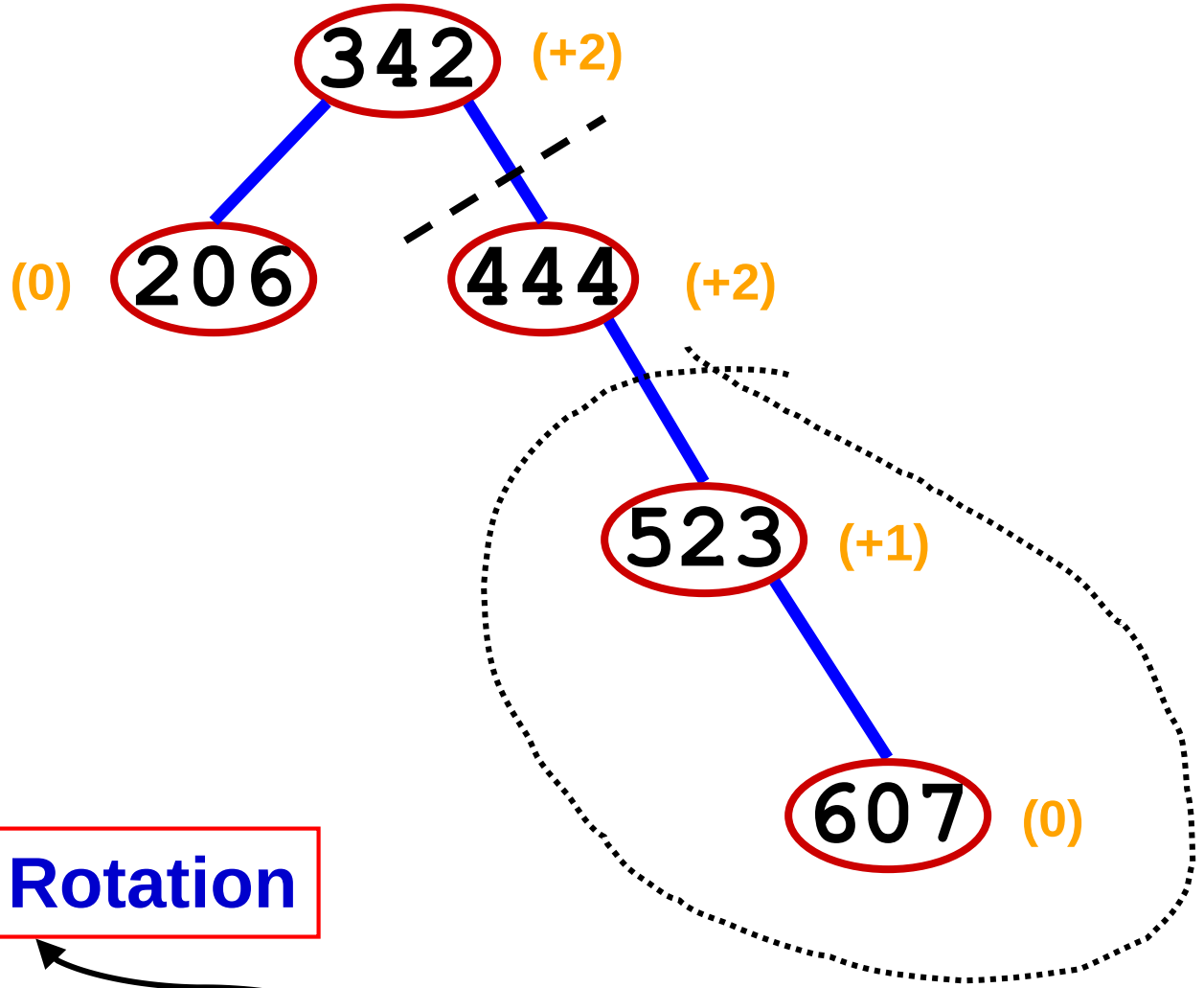
UNIVERSITY OF CALGARY



607 causes RR imbalance at 444

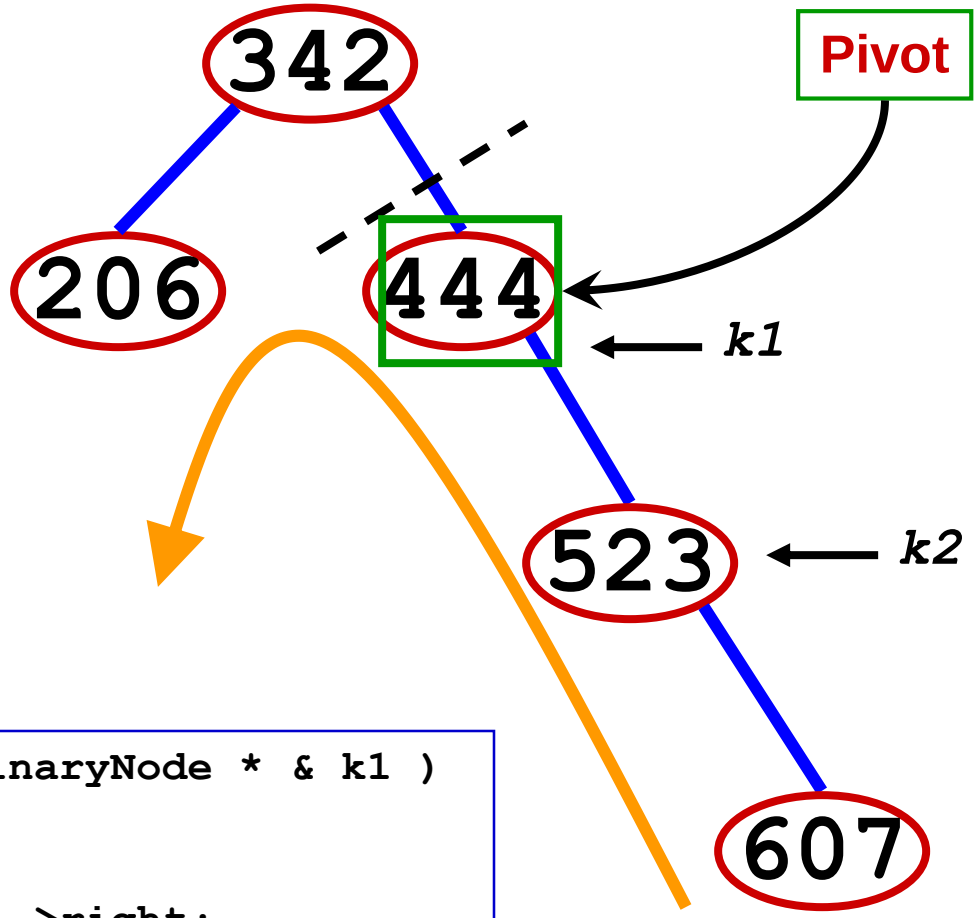


607 causes RR imbalance at 444

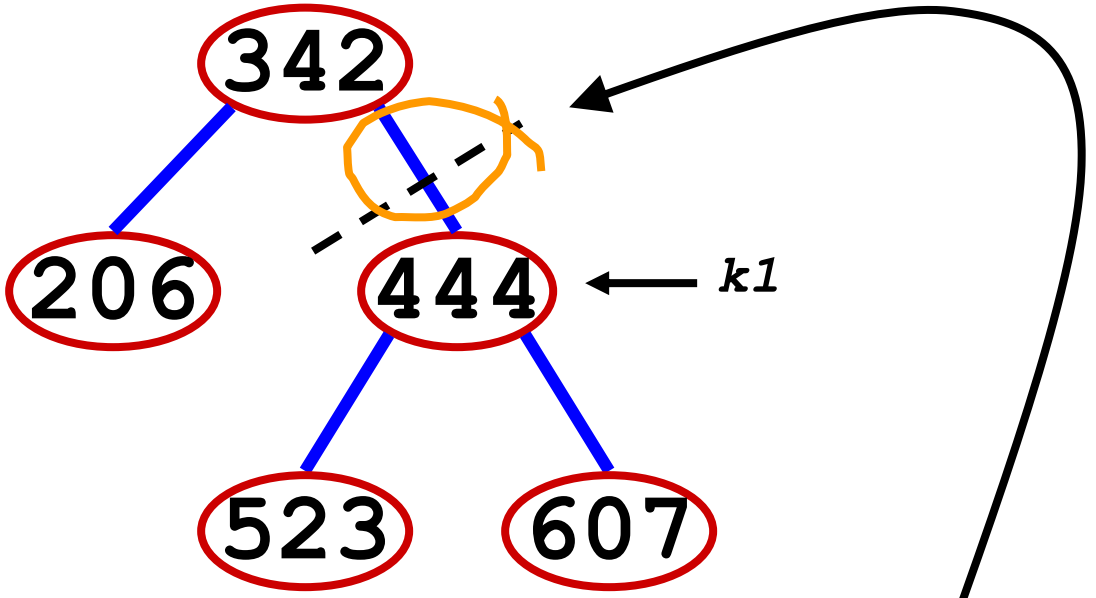


Single Rotation

607 causes RR imbalance at 444

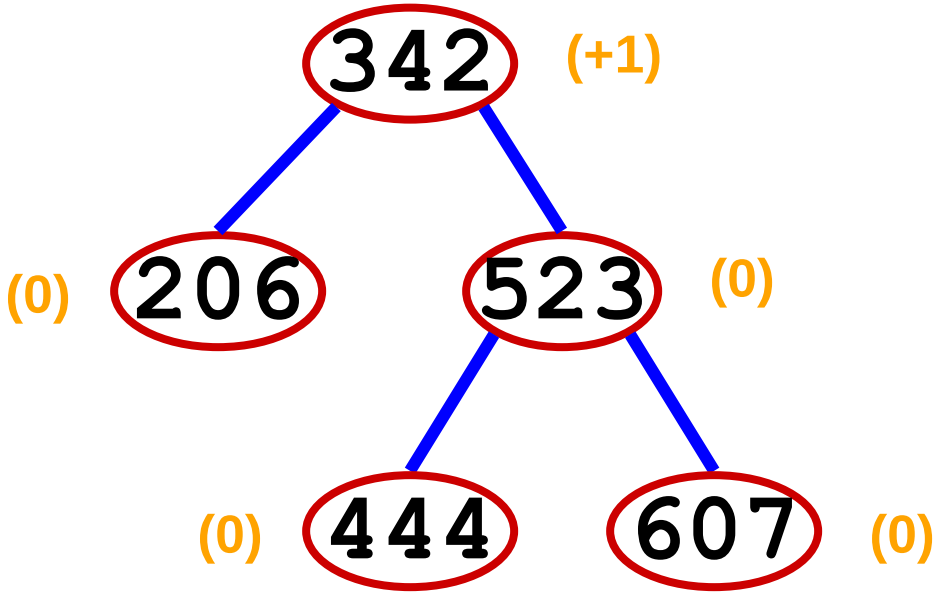



```
void LeftRotate( BinaryNode * & k1 )
{
BinaryNode *k2 = k1->right;
k1->right = k2->left;
k2->left = k1;
k1 = k2;
}
```




```
void LeftRotate( BinaryNode * & k1 )
{
BinaryNode *k2 = k1->right;
k1->right = k2->left;
k2->left = k1;
k1 = k2;
}
```

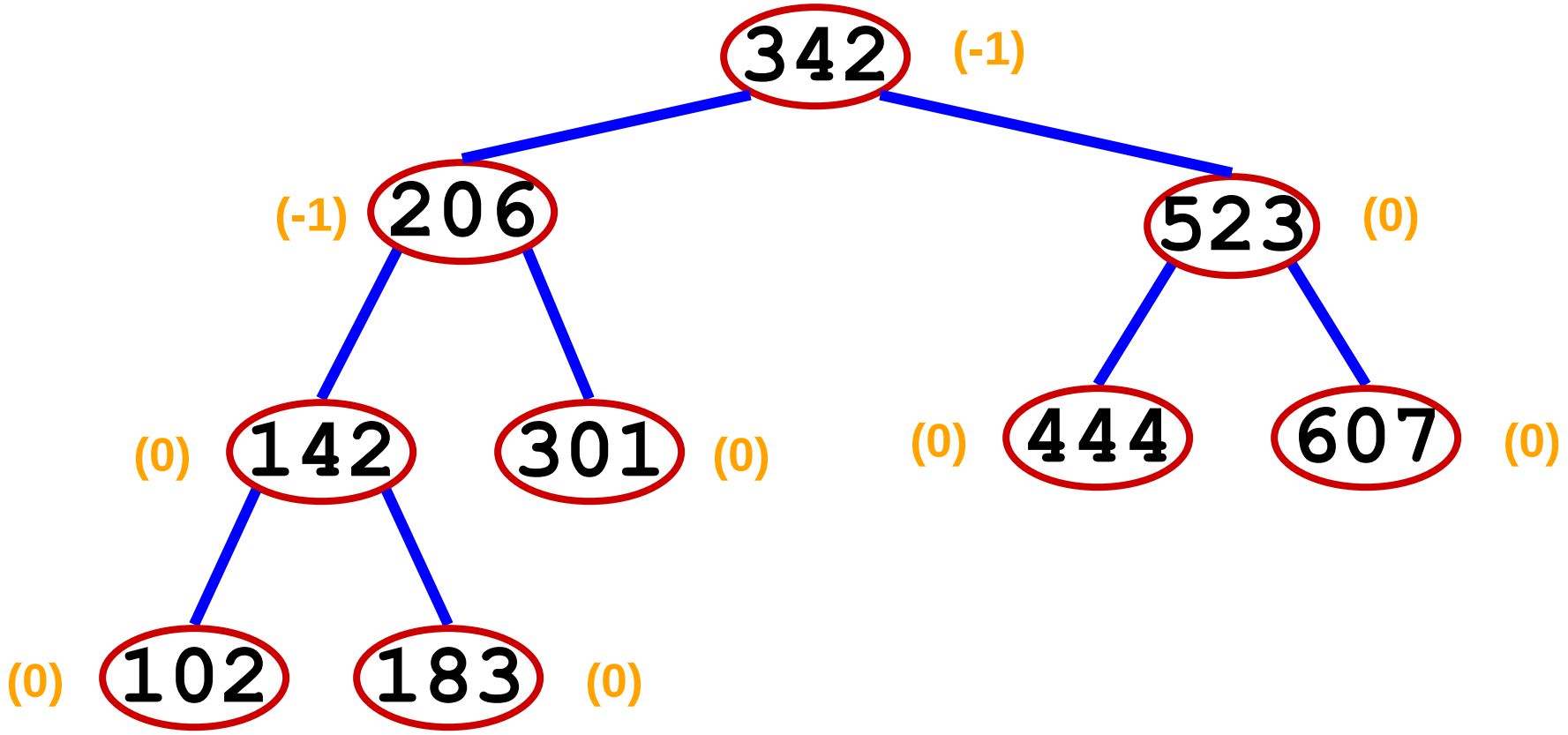
REMEMBER !
re-connect k1→node to the tree



UNIVERSITY OF CALGARY

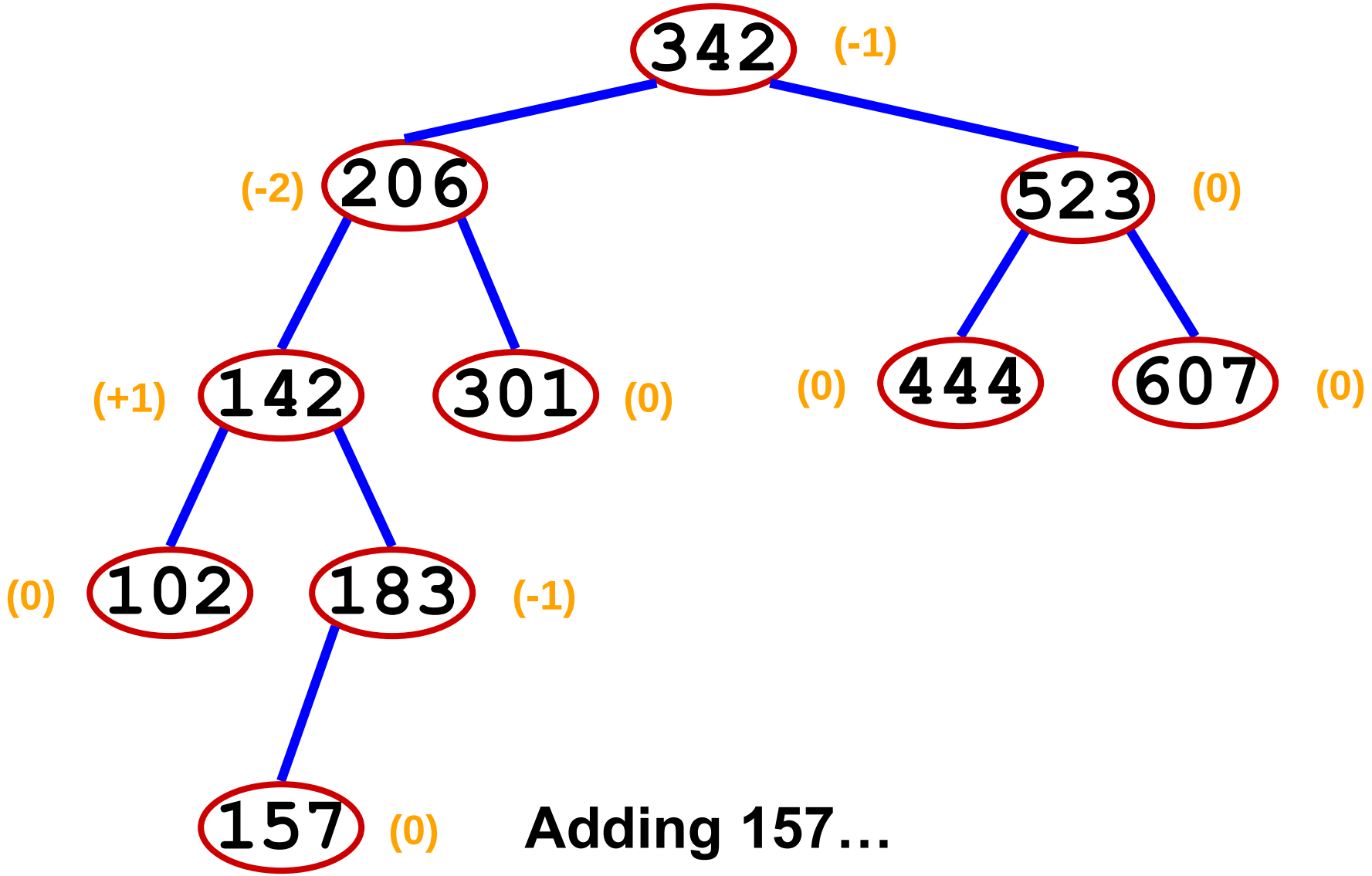
Check balance-factors…

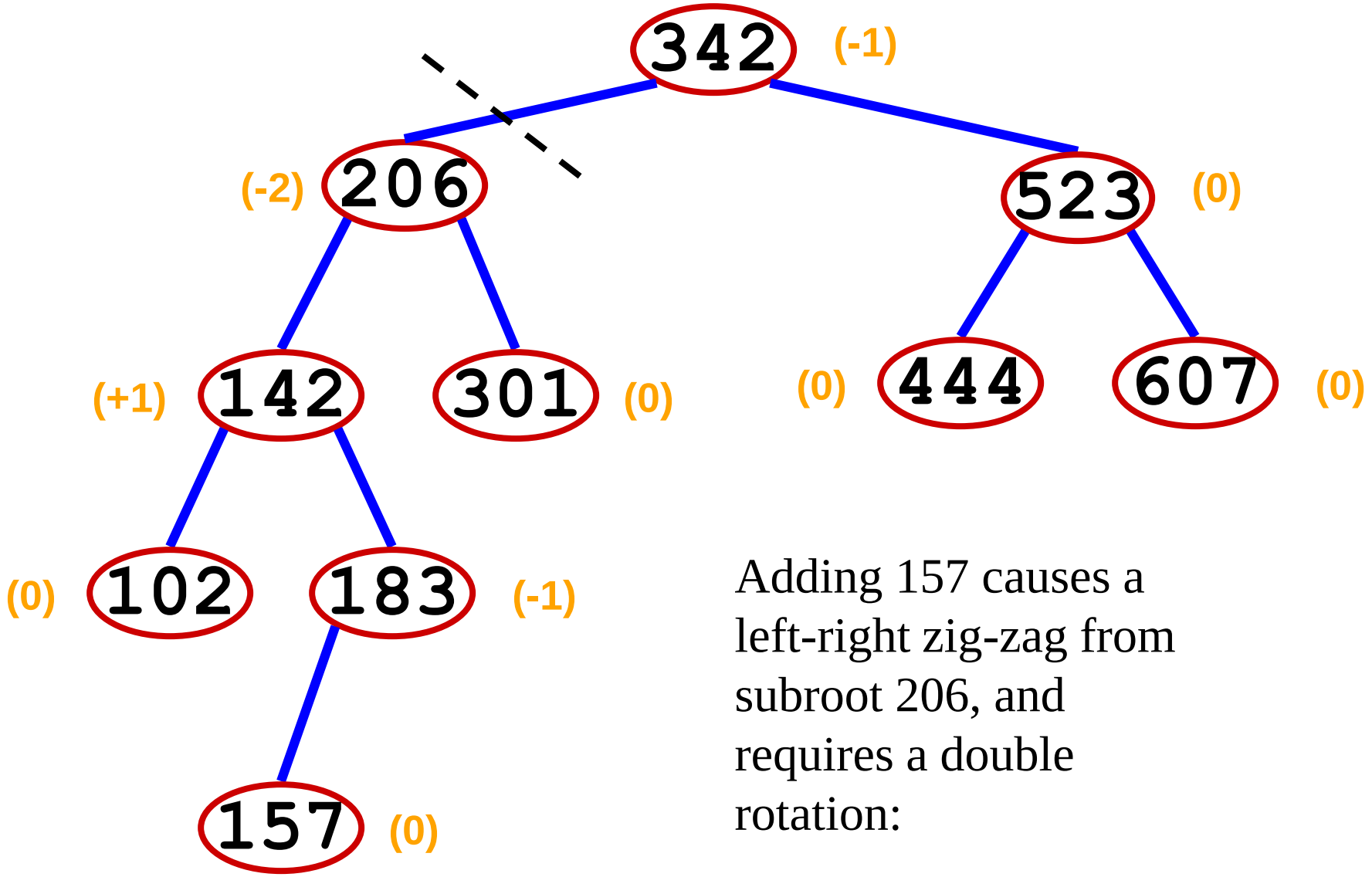
OK



UNIVERSITY OF CALGARY

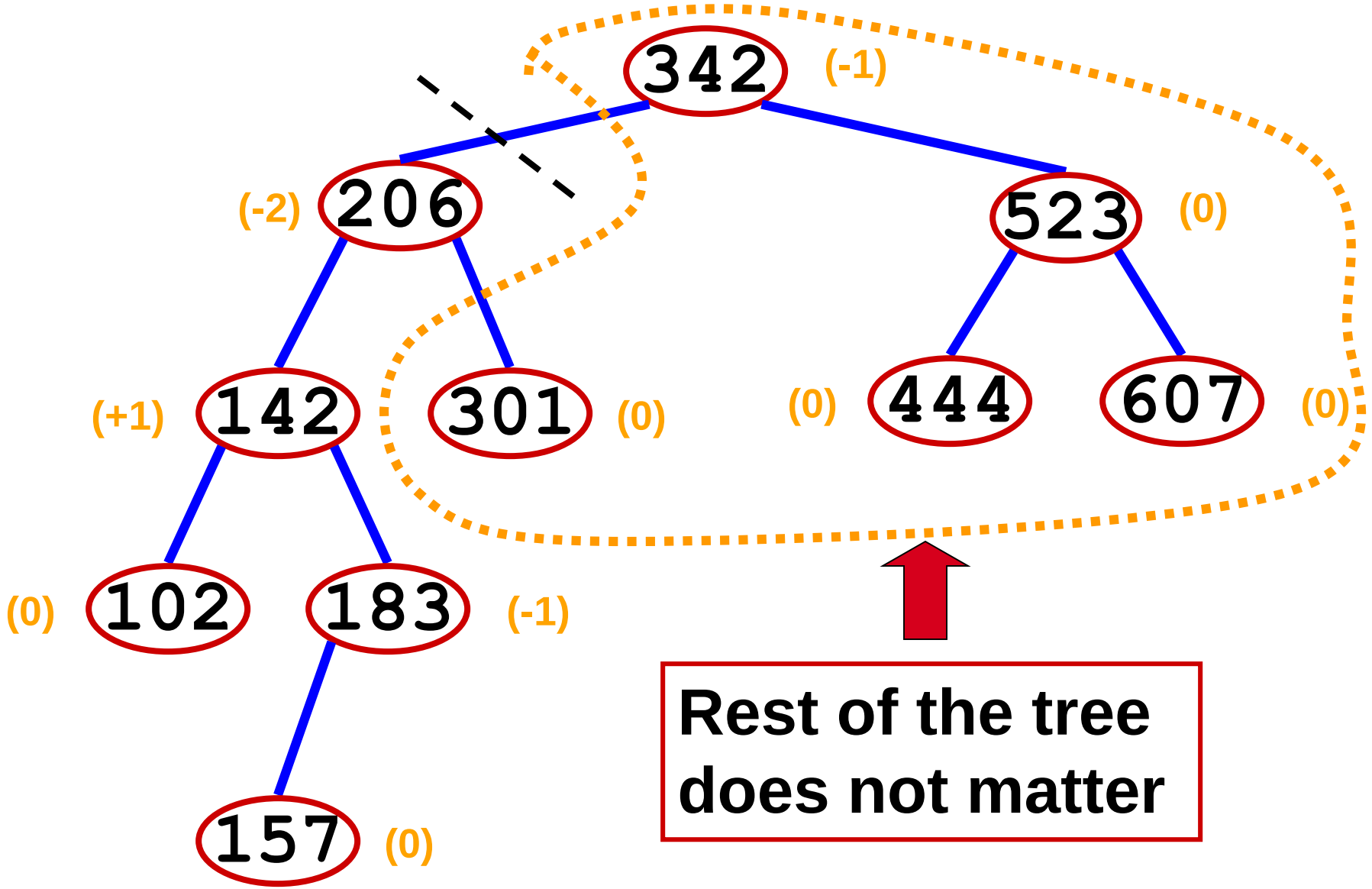
So far so good…

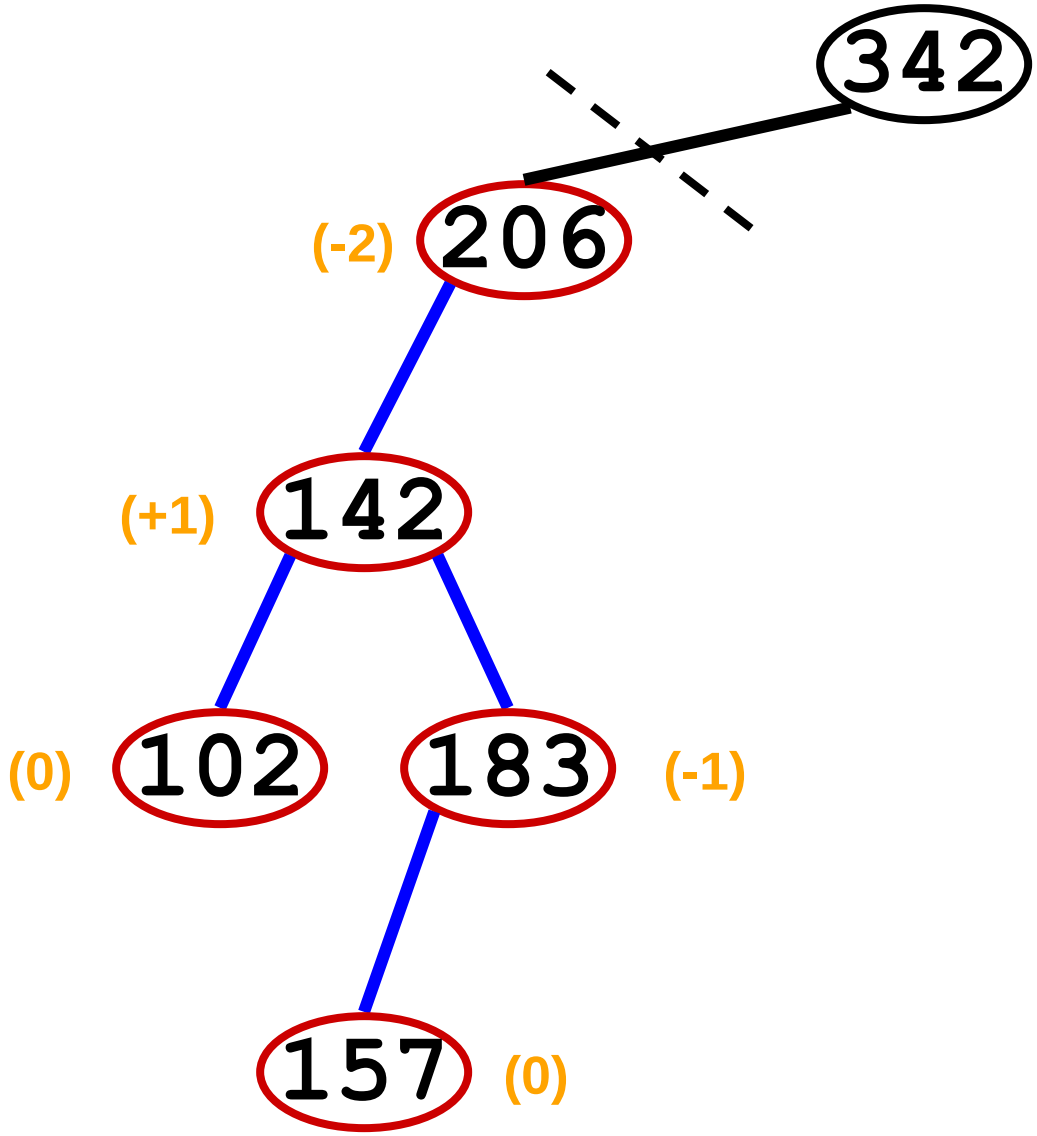

342
(-1)
(-2)
206
523
(0)
(+1)
142
301
(0)
(0)
444
607
(0)
(0)
102
183
(-1)
157
(0)
Adding 157…



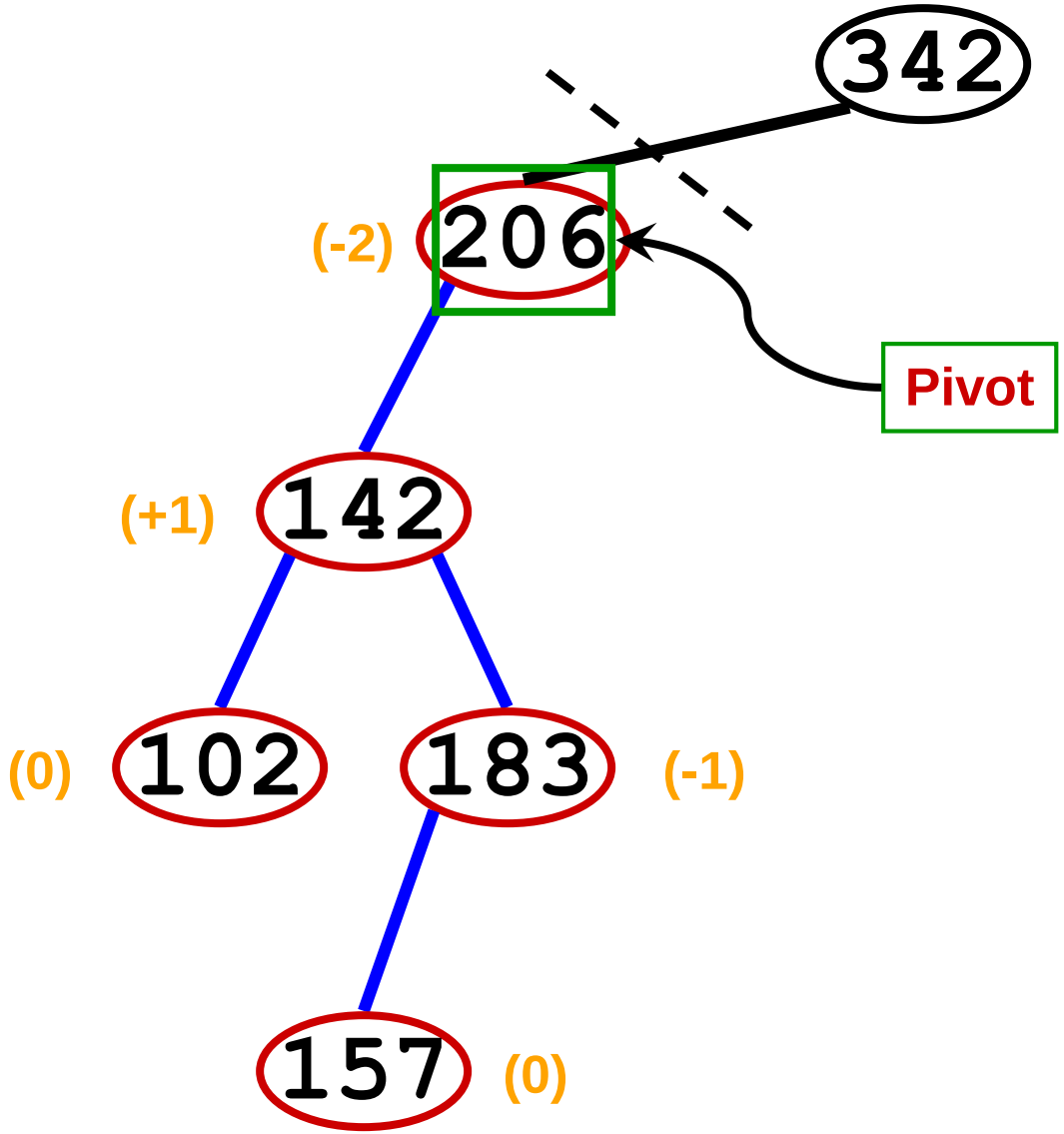
Adding 157 causes a left-right zig-zag from subroot 206, and requires a double rotation:

Make 183 the new subroot

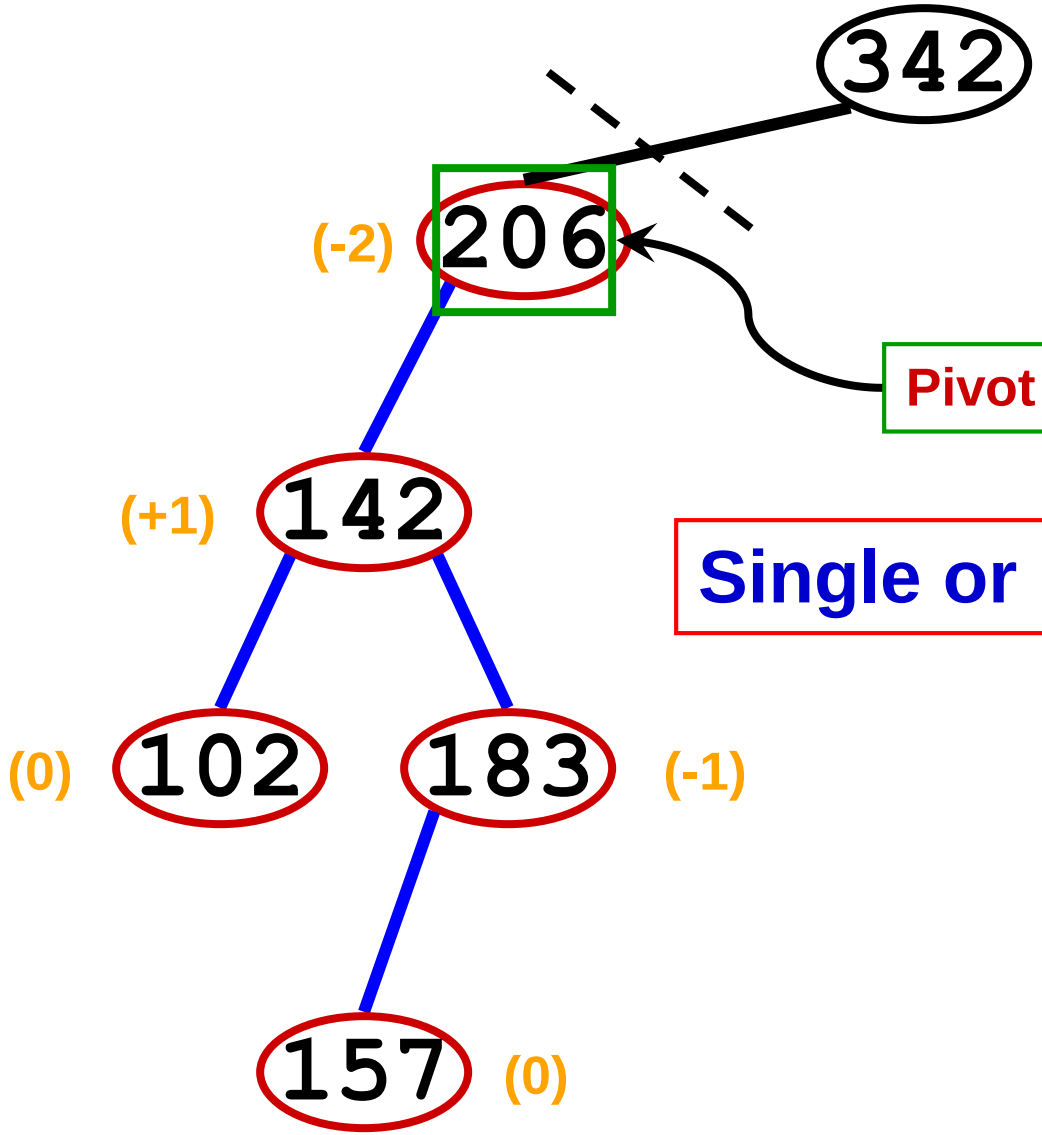

342
(-1)
(-2)
206
523
(0)
(+1)
142
301
(0)
(0)
444
607
(0)
(0)
102
183
(-1)
157
(0)
Rest of the tree does not matter



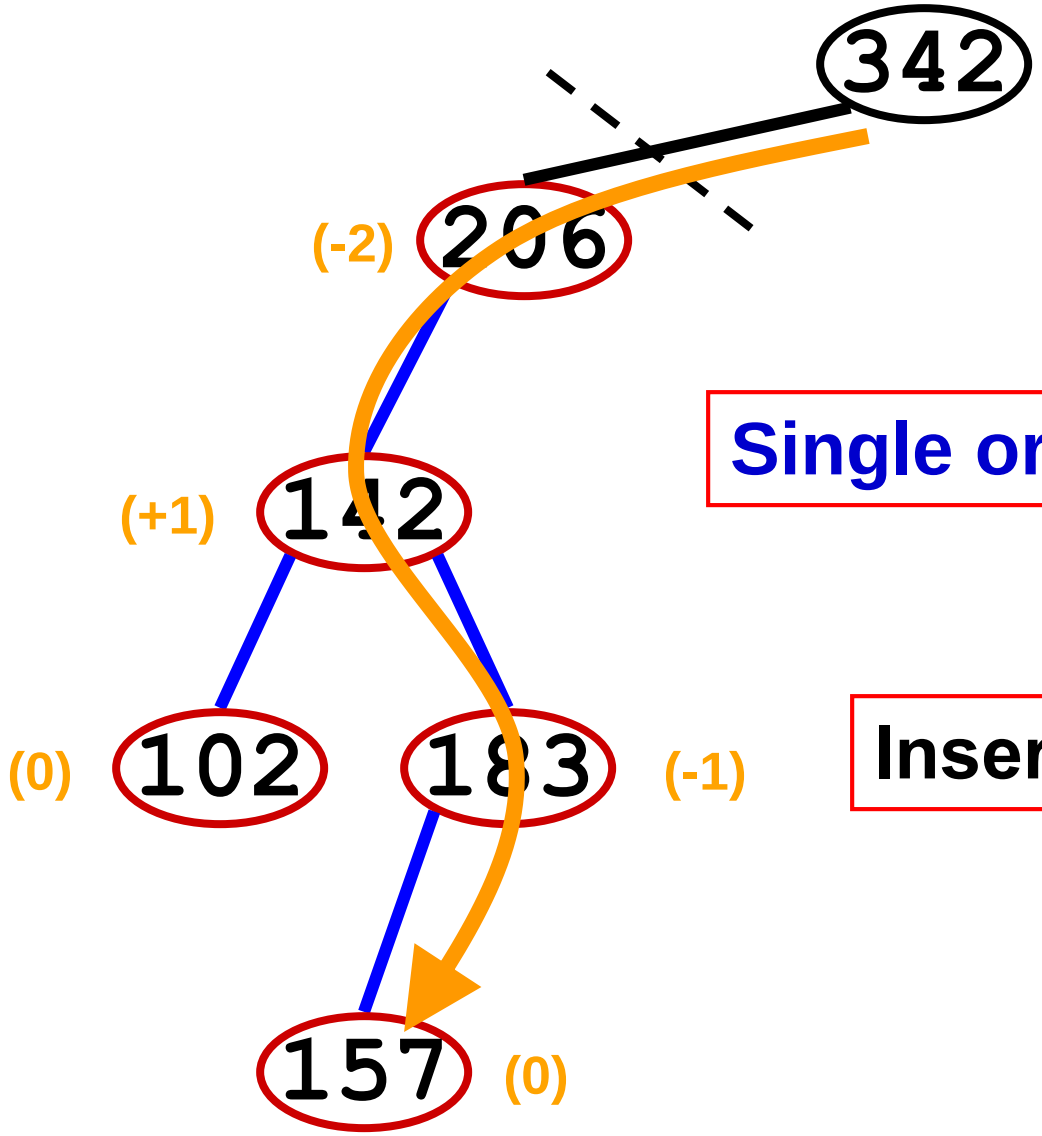
UNIVERSITY OF CALGARY



UNIVERSITY OF CALGARY

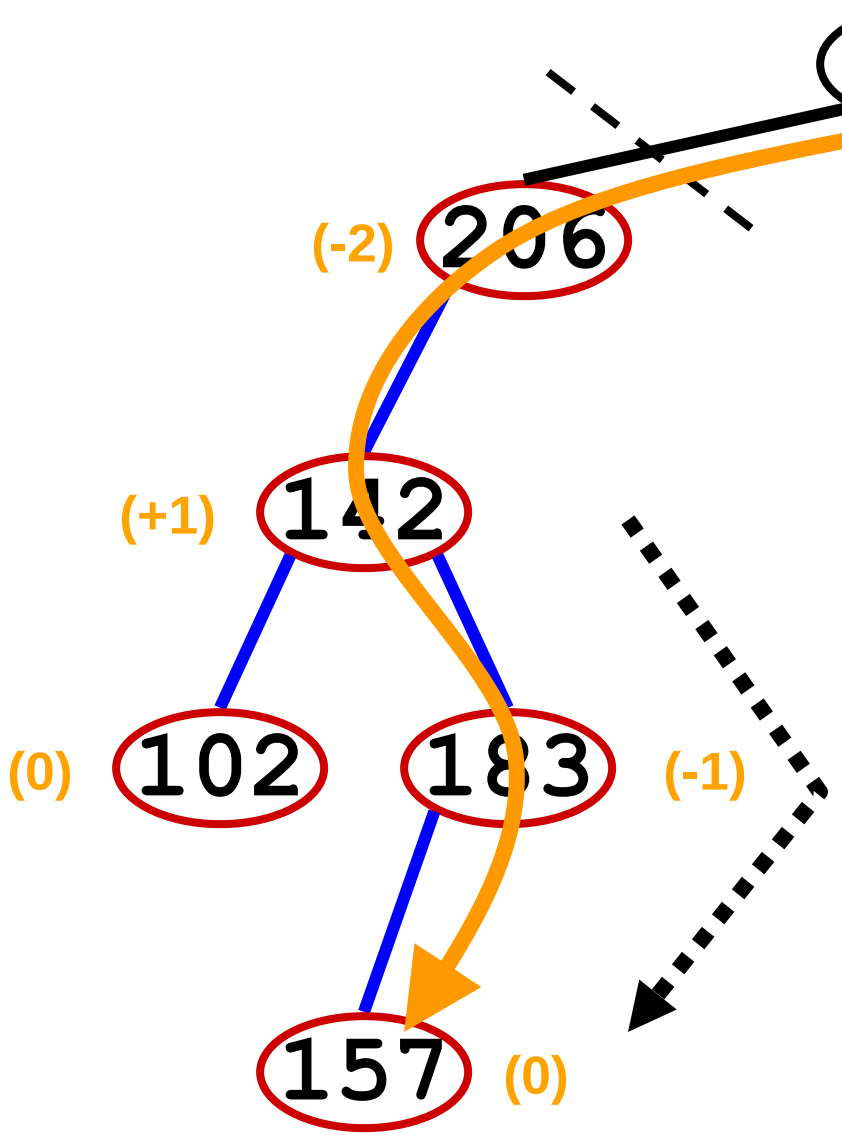


Single or Double Rotation?



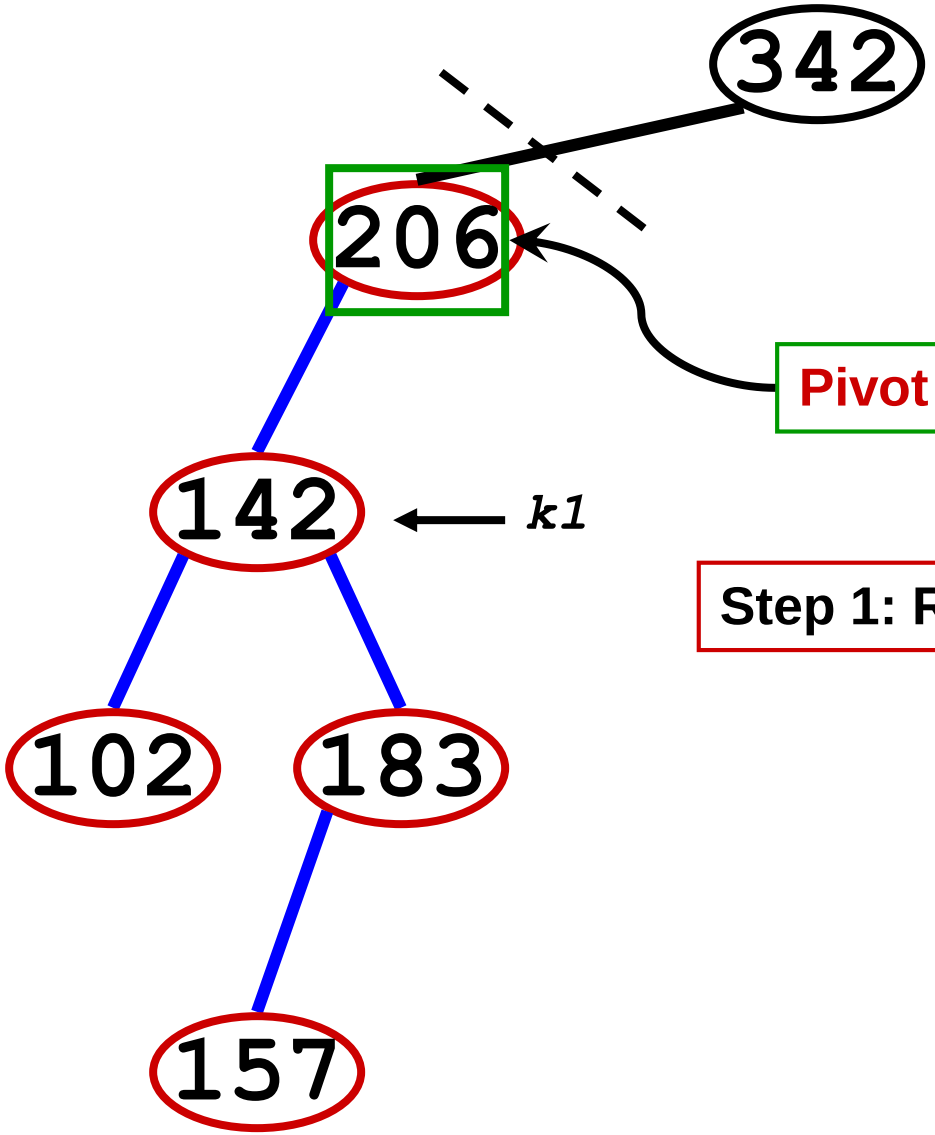
Single or Double Rotation?

Insertion Path…



zig-zag formation

Double Rotation

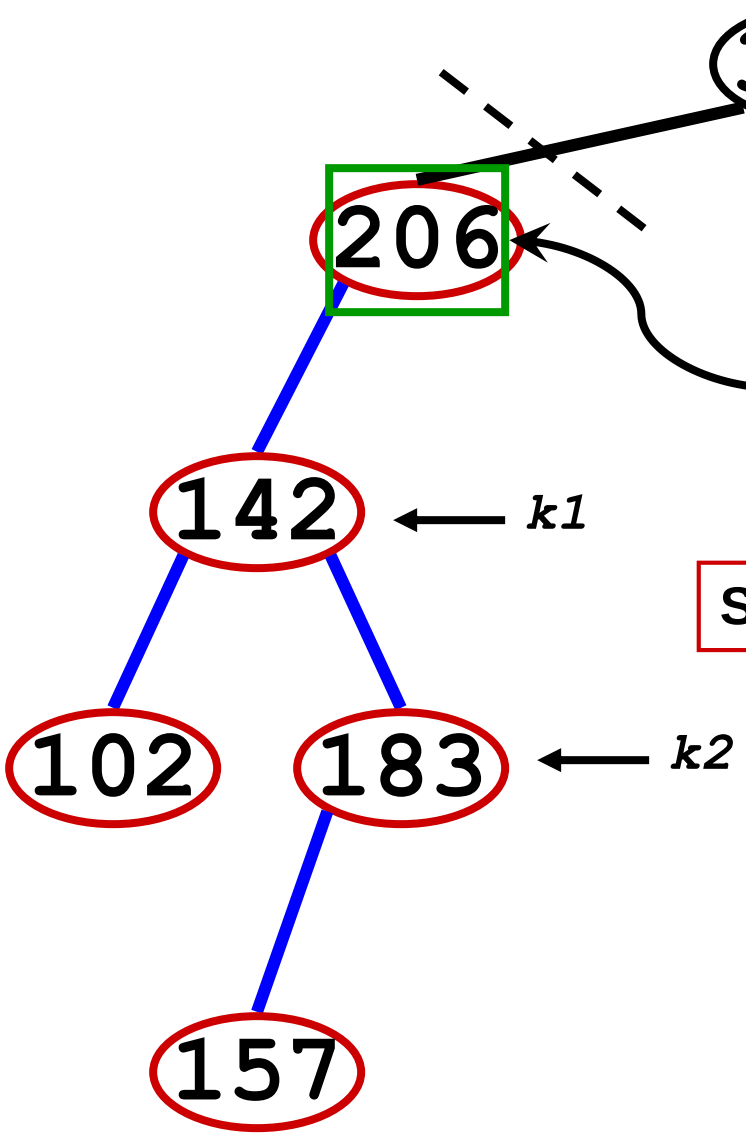


Double Rotation

Step 1: Rotate child and grandchild

RIGHT rotation

```
BinaryNode *k2 = k1->right;
k1->right = k2->left;
k2->left = k1;
k1 = k2;
```

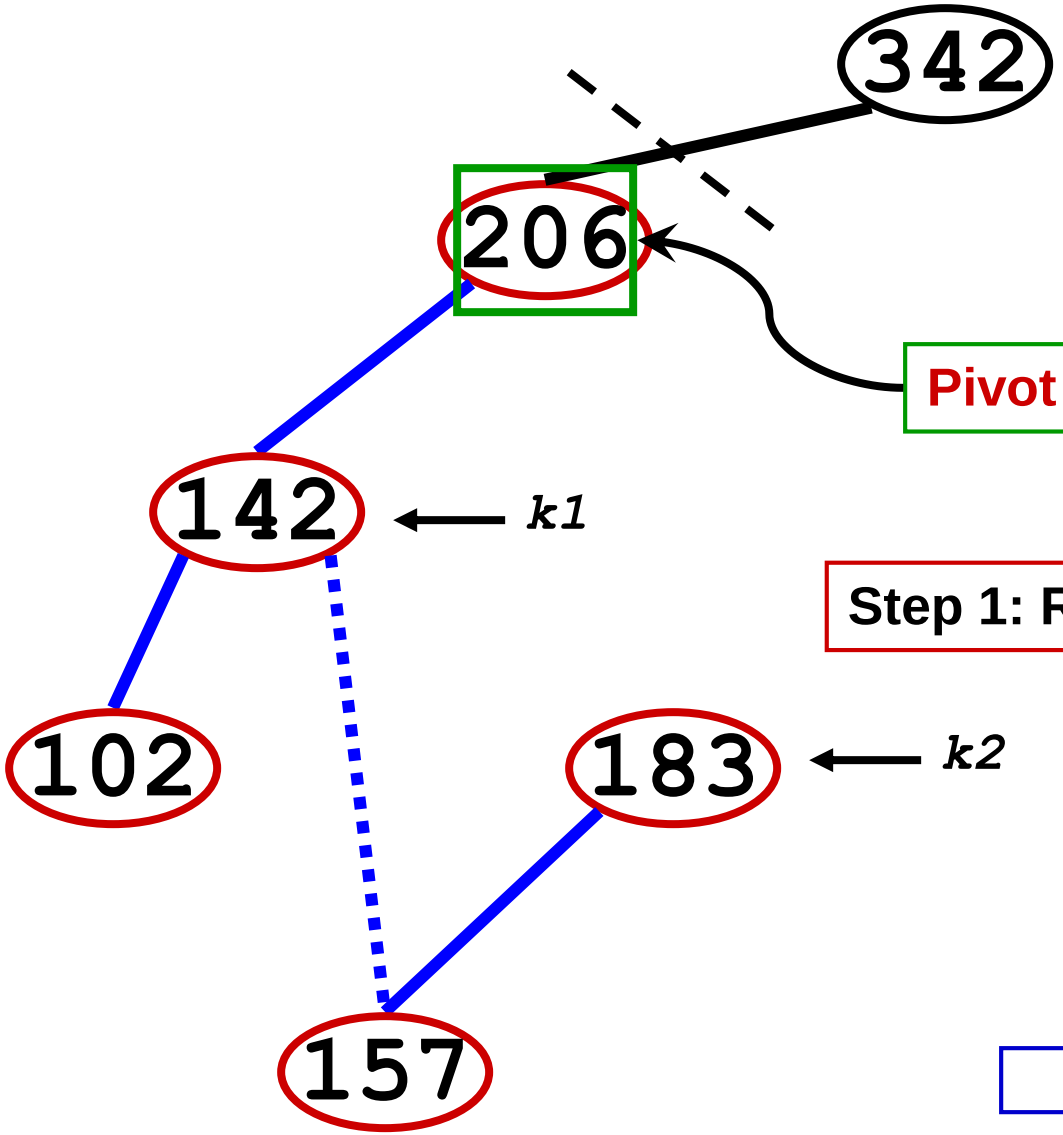


Double Rotation

Step 1: Rotate child and grandchild

RIGHT rotation

```
BinaryNode *k2 = k1->right;
k1->right = k2->left;
k2->left = k1;
k1 = k2;
```

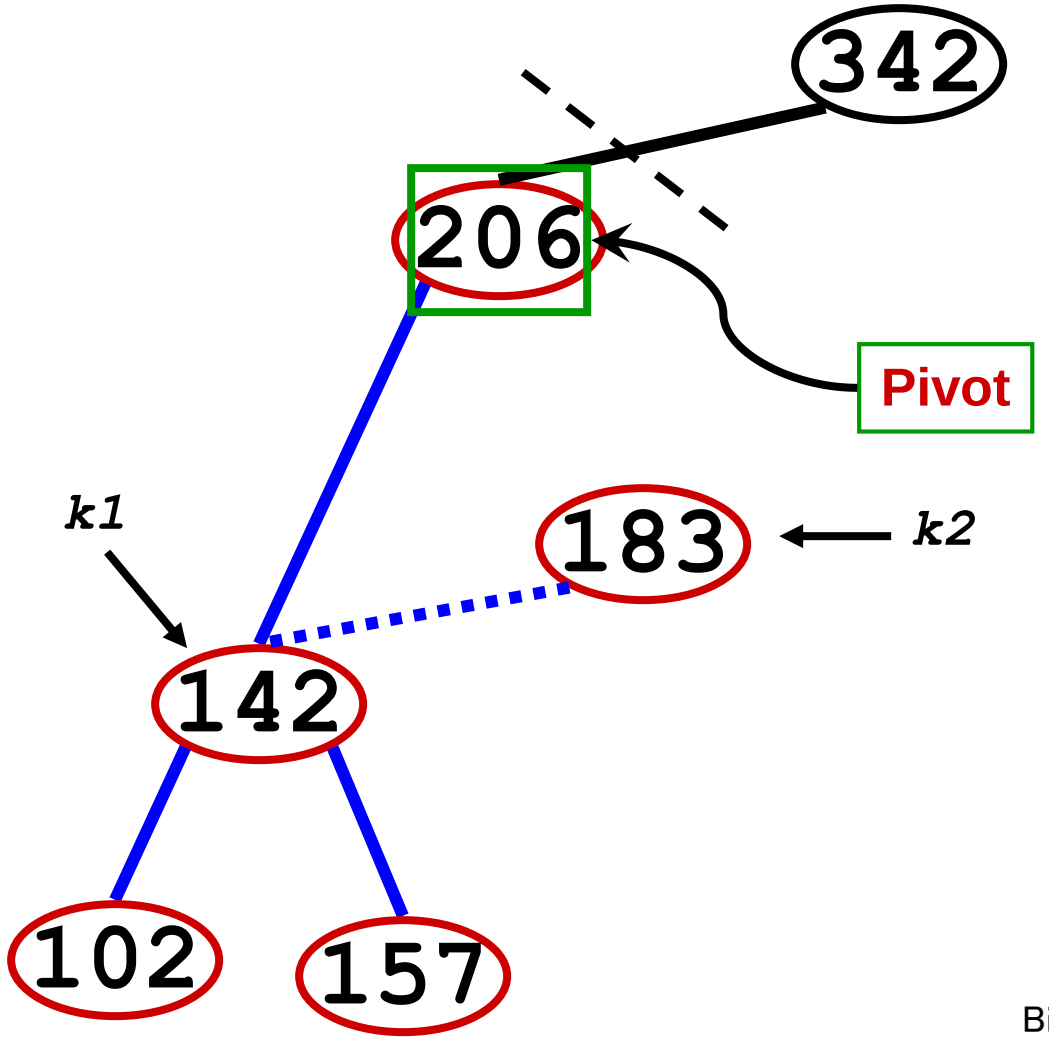


Double Rotation

Step 1: Rotate child and grandchild

RIGHT rotation

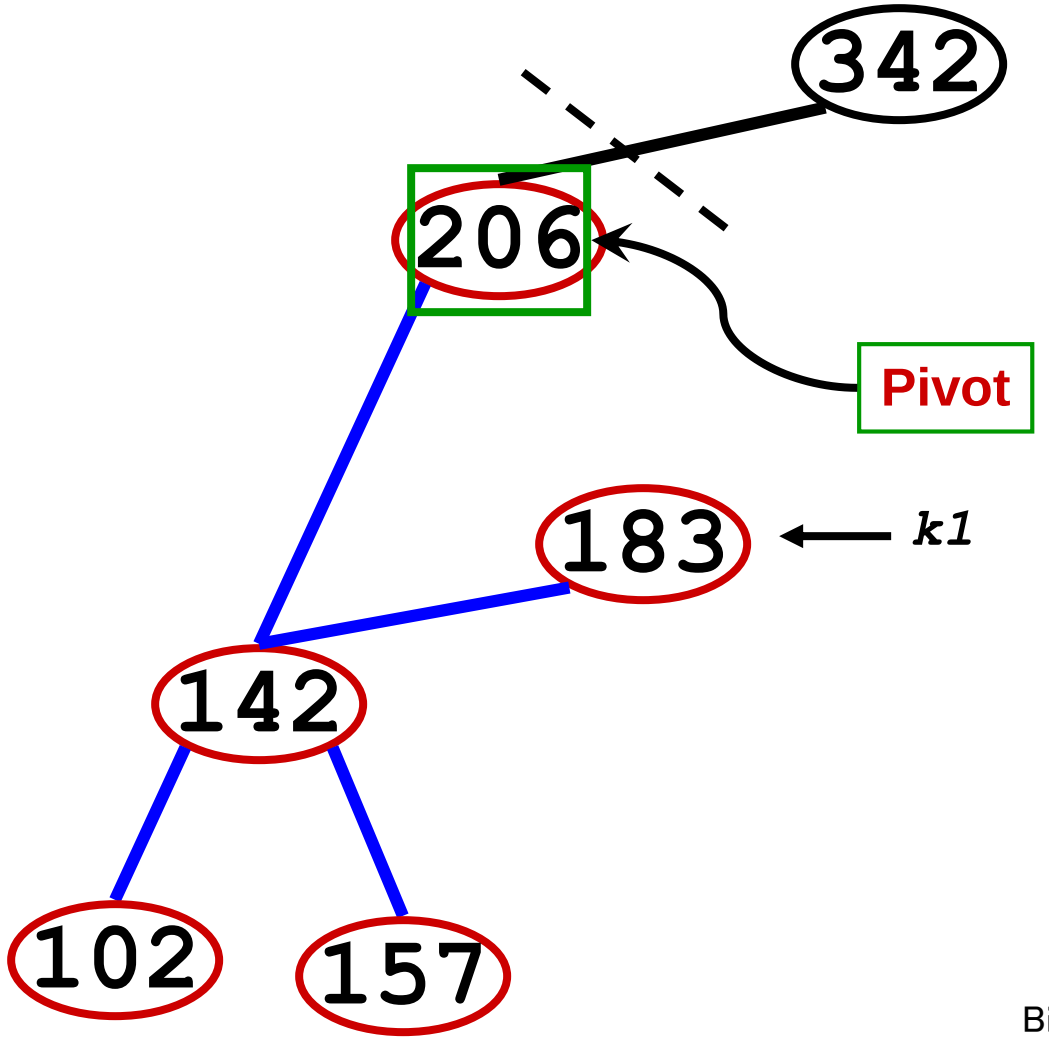
```
BinaryNode *k2 = k1->right;
k1->right = k2->left;
k2->left = k1;
k1 = k2;
```



Double Rotation

RIGHT rotation

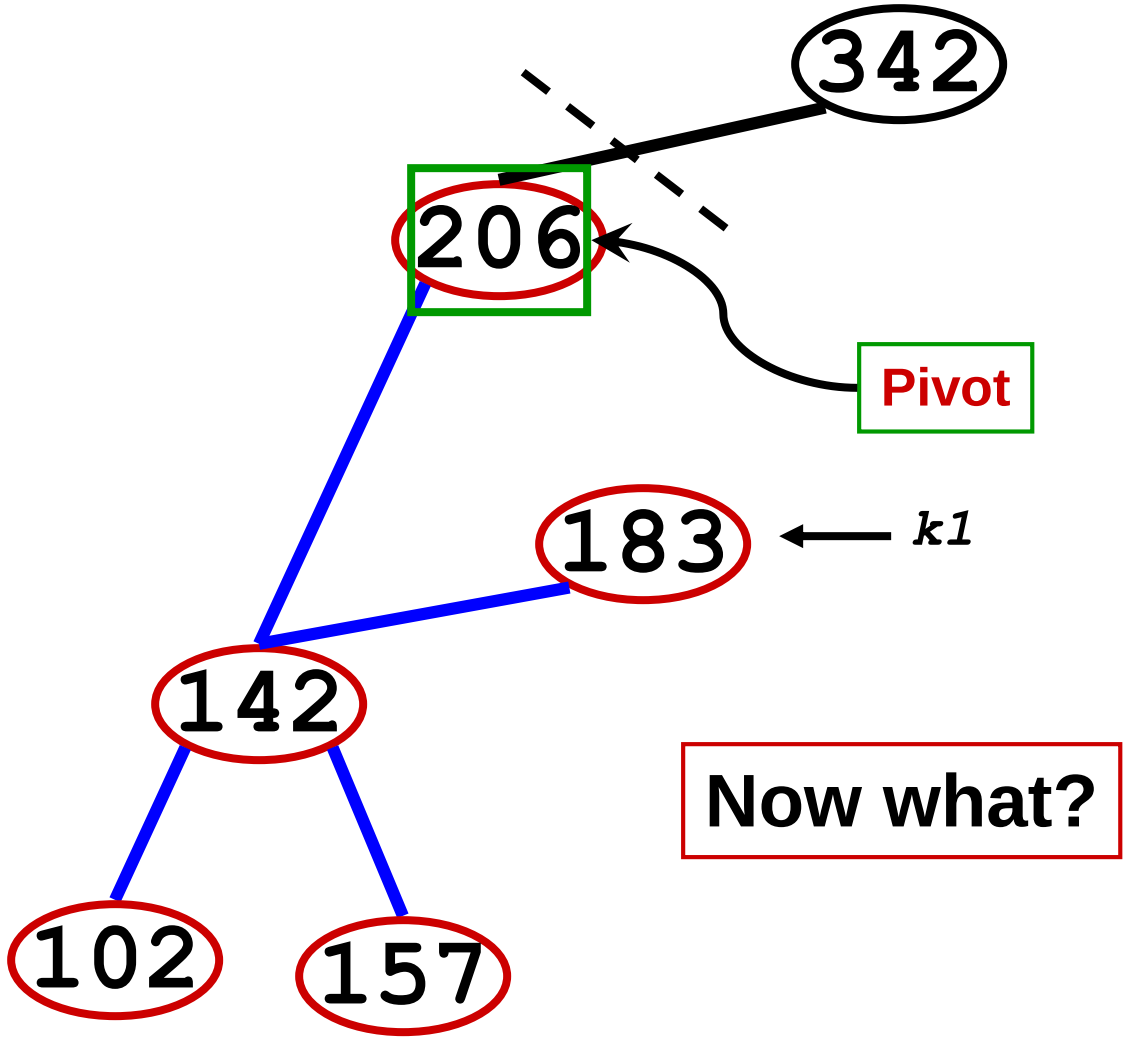
```
BinaryNode *k2 = k1->right;
k1->right = k2->left;
k2->left = k1;
k1 = k2;
```



Double Rotation

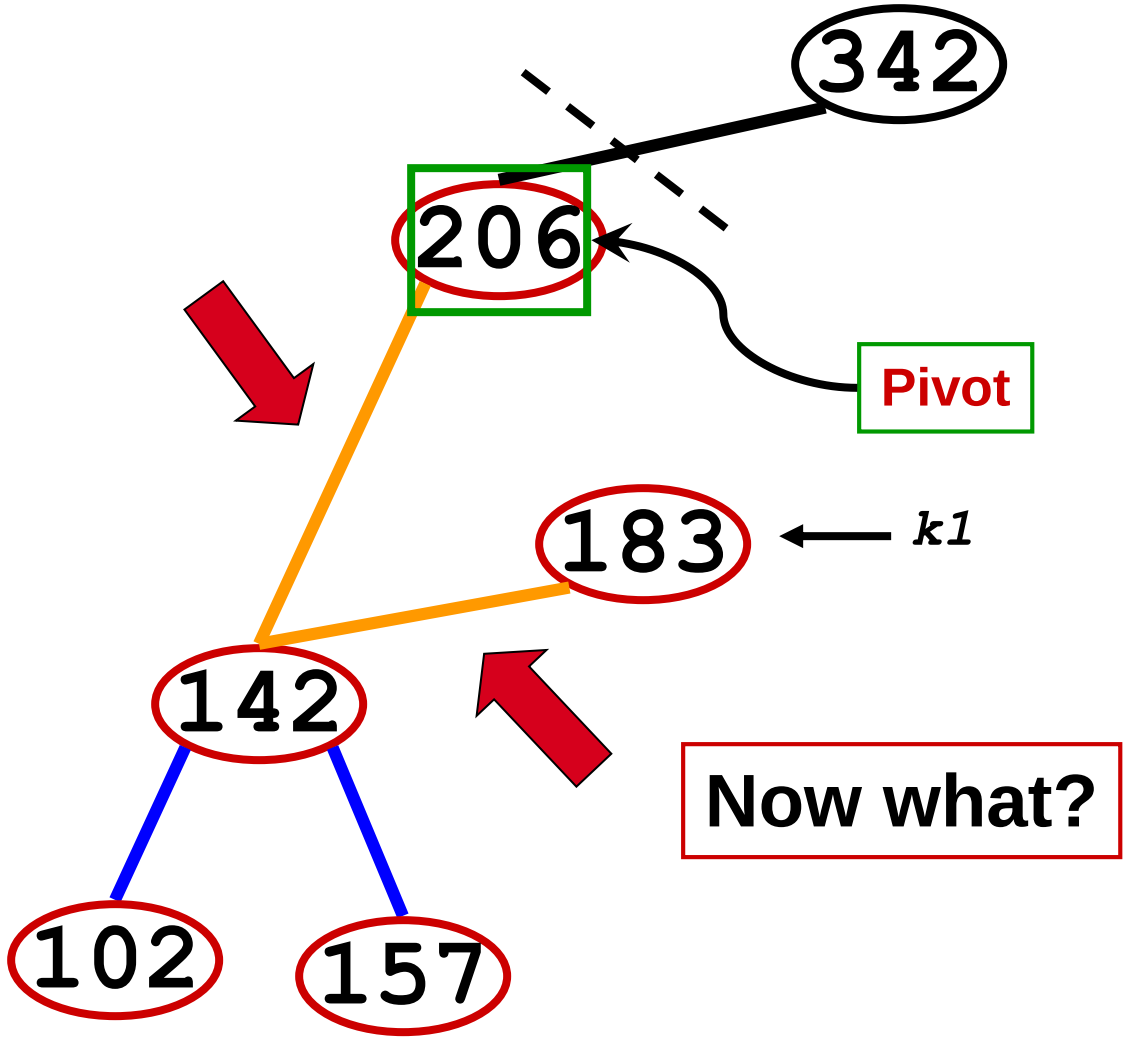
RIGHT rotation

```
BinaryNode *k2 = k1->right;
k1->right = k2->left;
k2->left = k1;
k1 = k2;
```



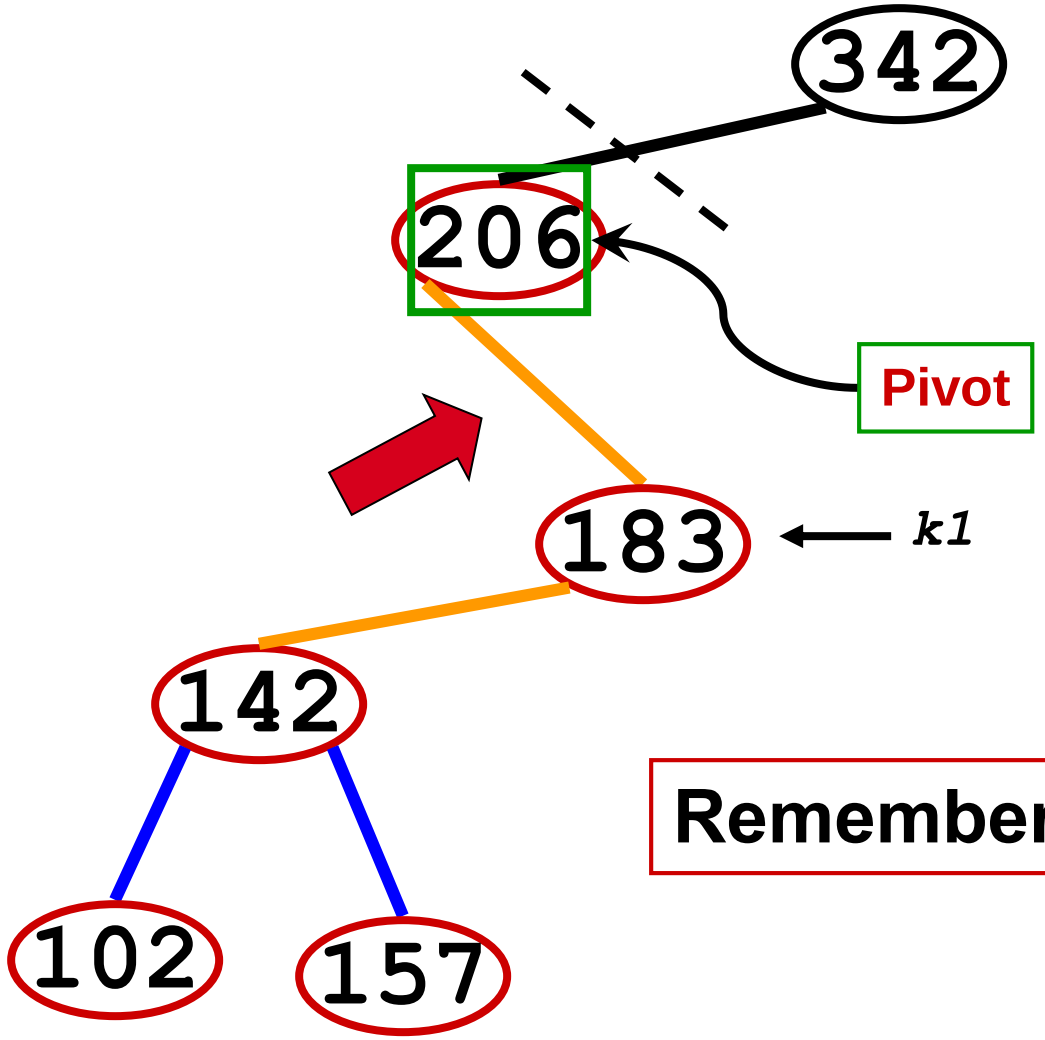
Double Rotation

Now what?



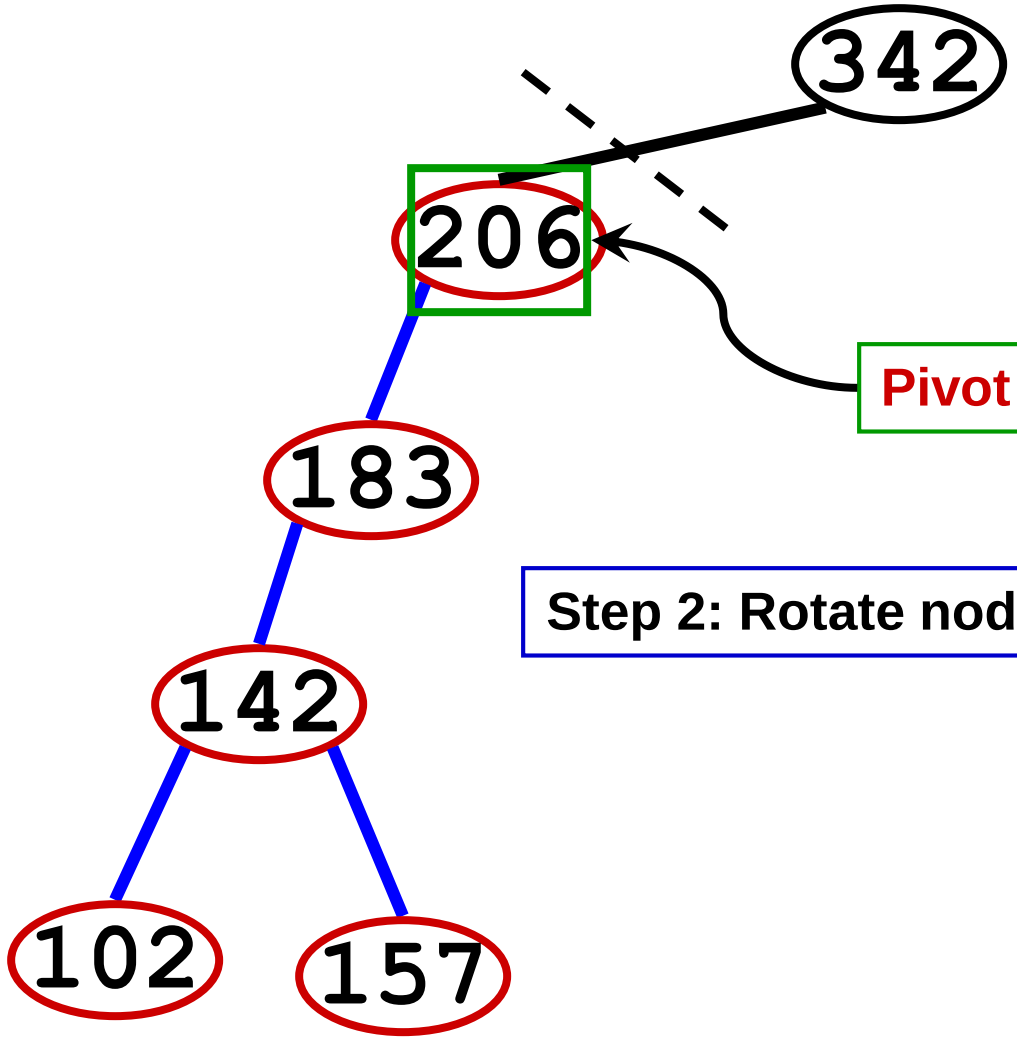
Double Rotation

Now what?



Double Rotation

Remember that…

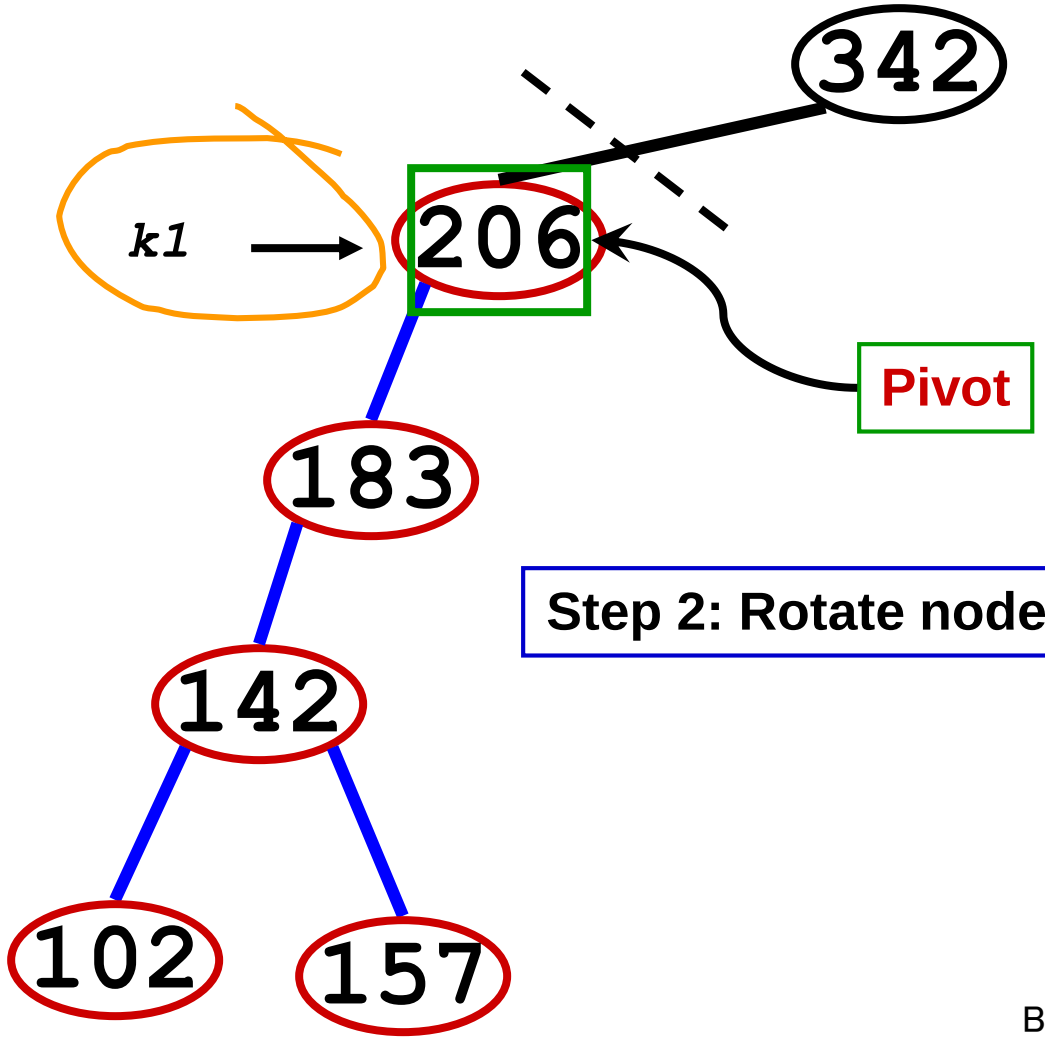


Double Rotation

Step 2: Rotate node and new child (AVL)

LEFT rotation

```
BinaryNode *k2 = k1->left;
k1->left = k2->right;
k2->right = k1;
k1 = k2;
```

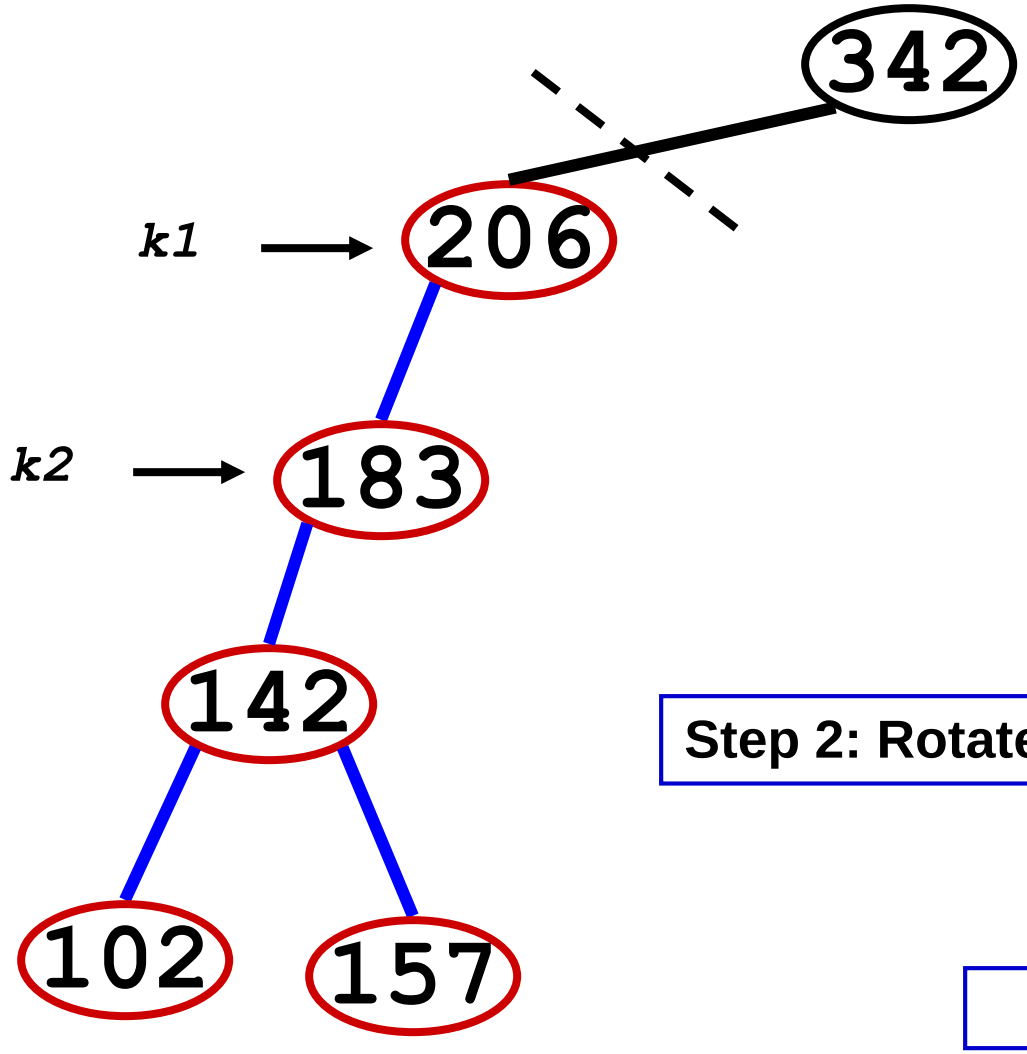


Double Rotation

Step 2: Rotate node and new child (AVL)

LEFT rotation

```
BinaryNode *k2 = k1->left;
k1->left = k2->right;
k2->right = k1;
k1 = k2;
```



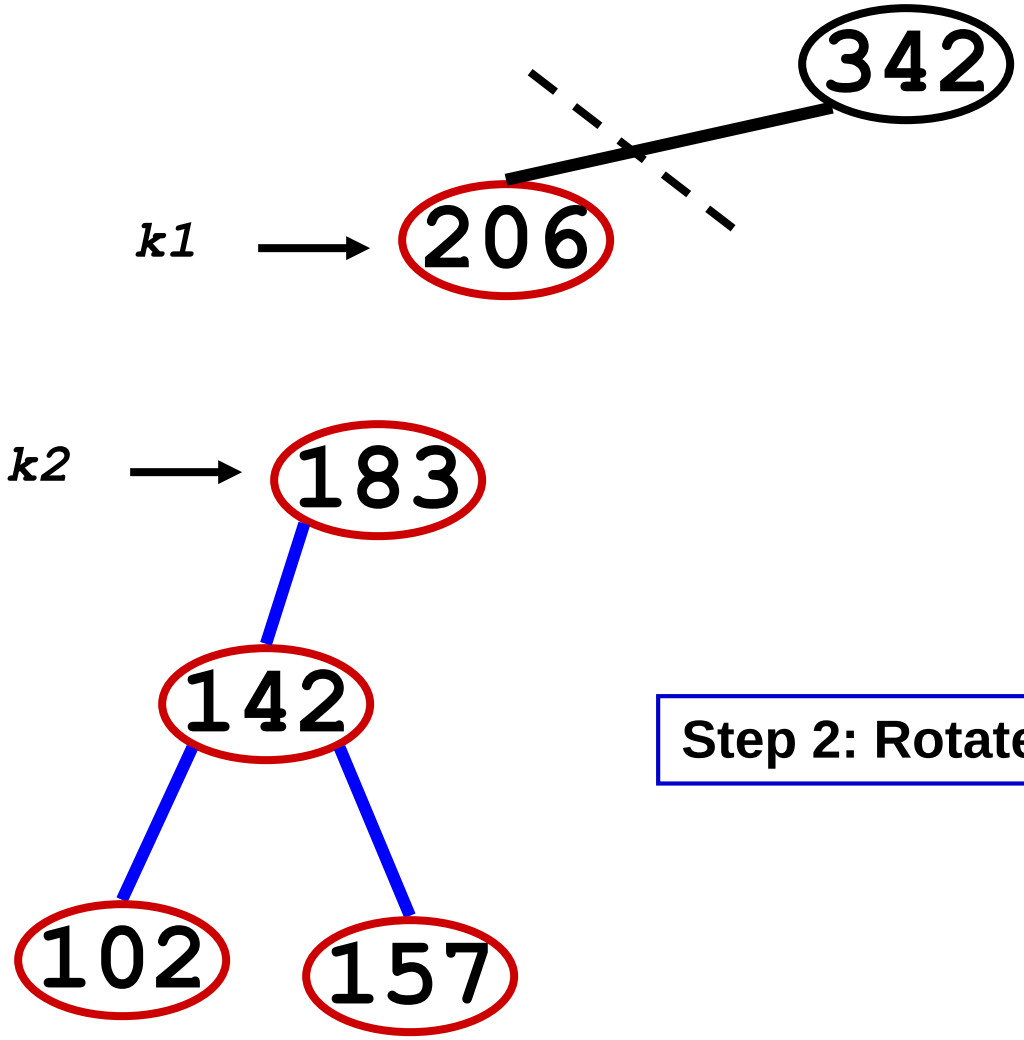
UNIVERSITY OF CALGARY

Double Rotation

Step 2: Rotate node and new child (AVL)

LEFT rotation

```
BinaryNode *k2 = k1->left;
k1->left = k2->right;
k2->right = k1;
k1 = k2;
```

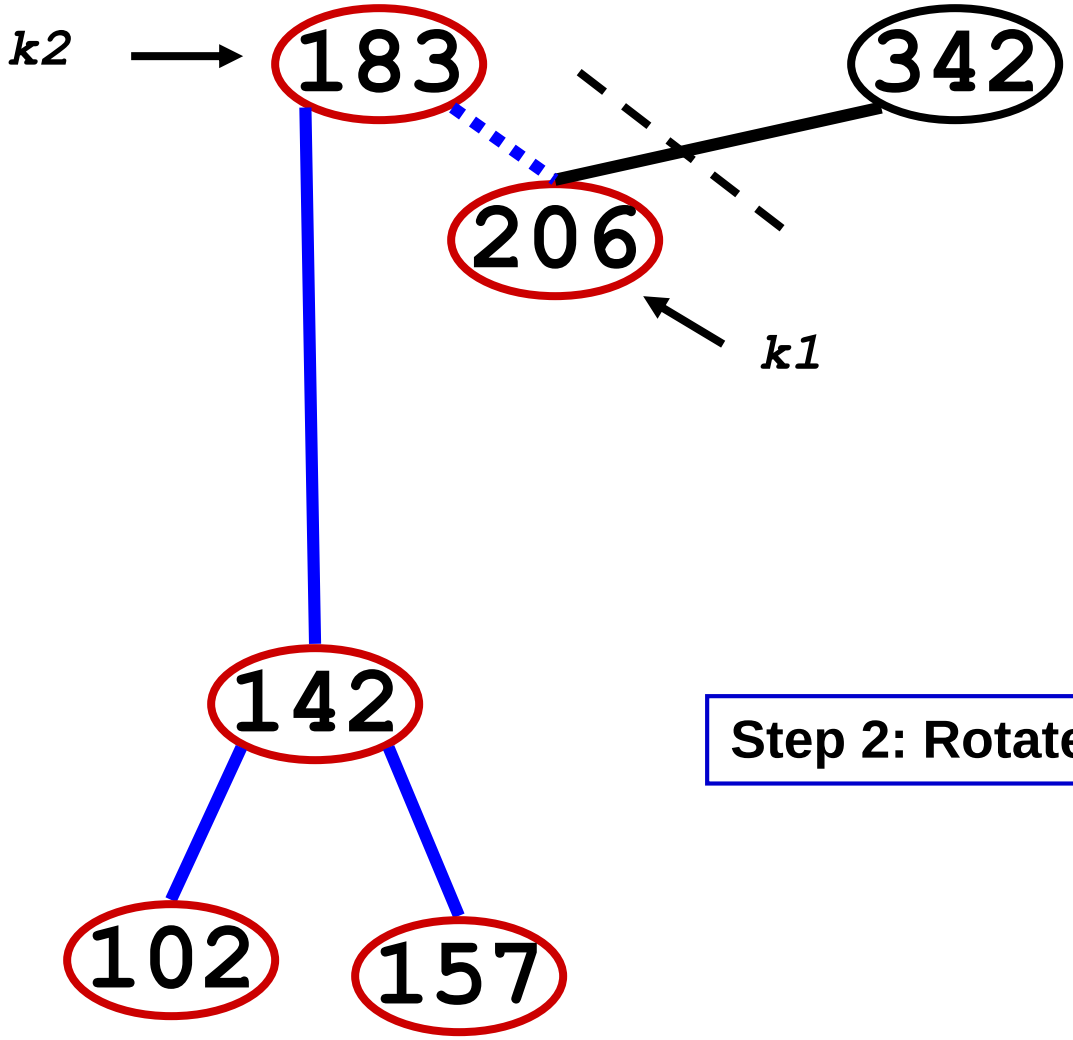


Double Rotation

Step 2: Rotate node and new child (AVL)

LEFT rotation

```
BinaryNode *k2 = k1->left;
k1->left = k2->right;
k2->right = k1;
k1 = k2;
```



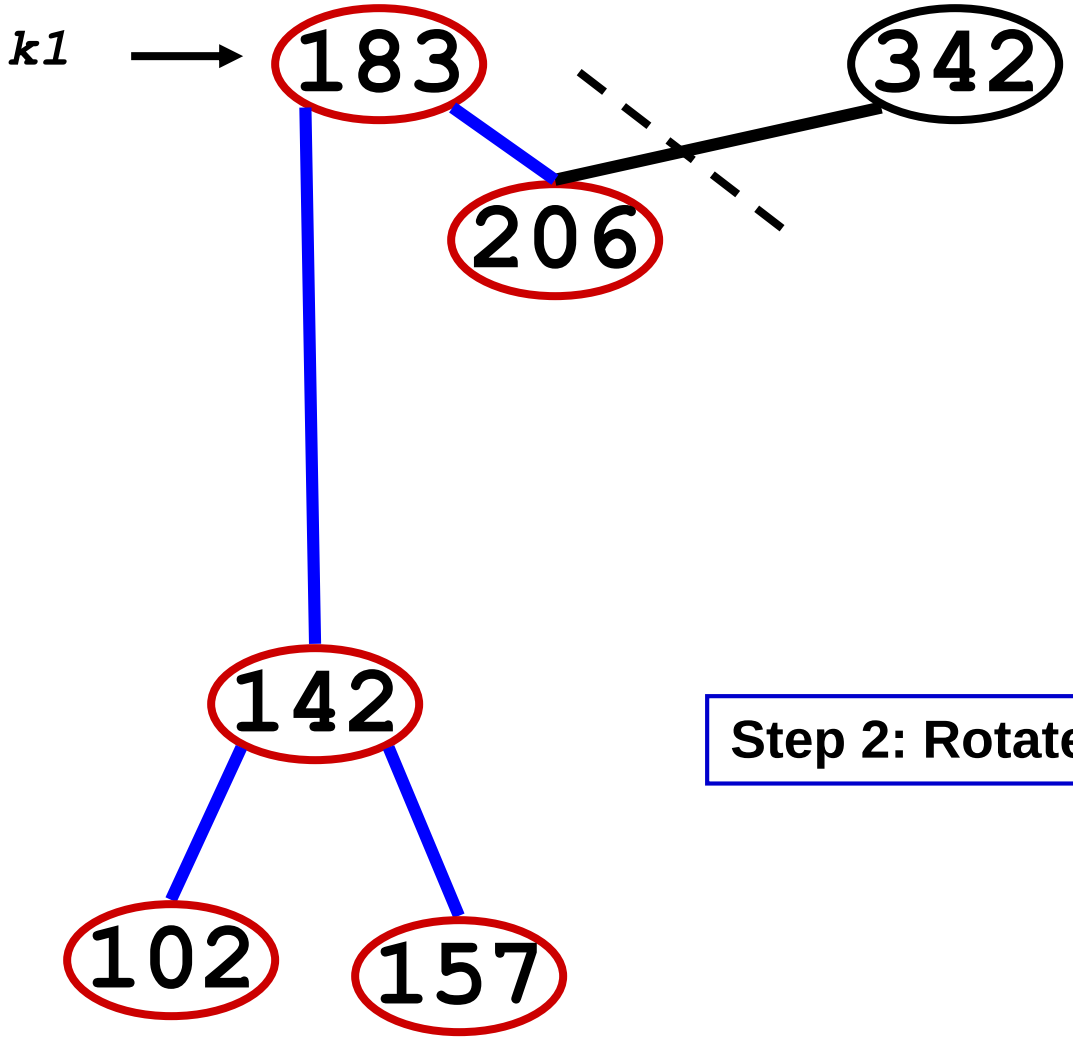
UNIVERSITY OF CALGARY

Double Rotation

Step 2: Rotate node and new child (AVL)

LEFT rotation

```
BinaryNode *k2 = k1->left;
k1->left = k2->right;
k2->right = k1;
k1 = k2;
```



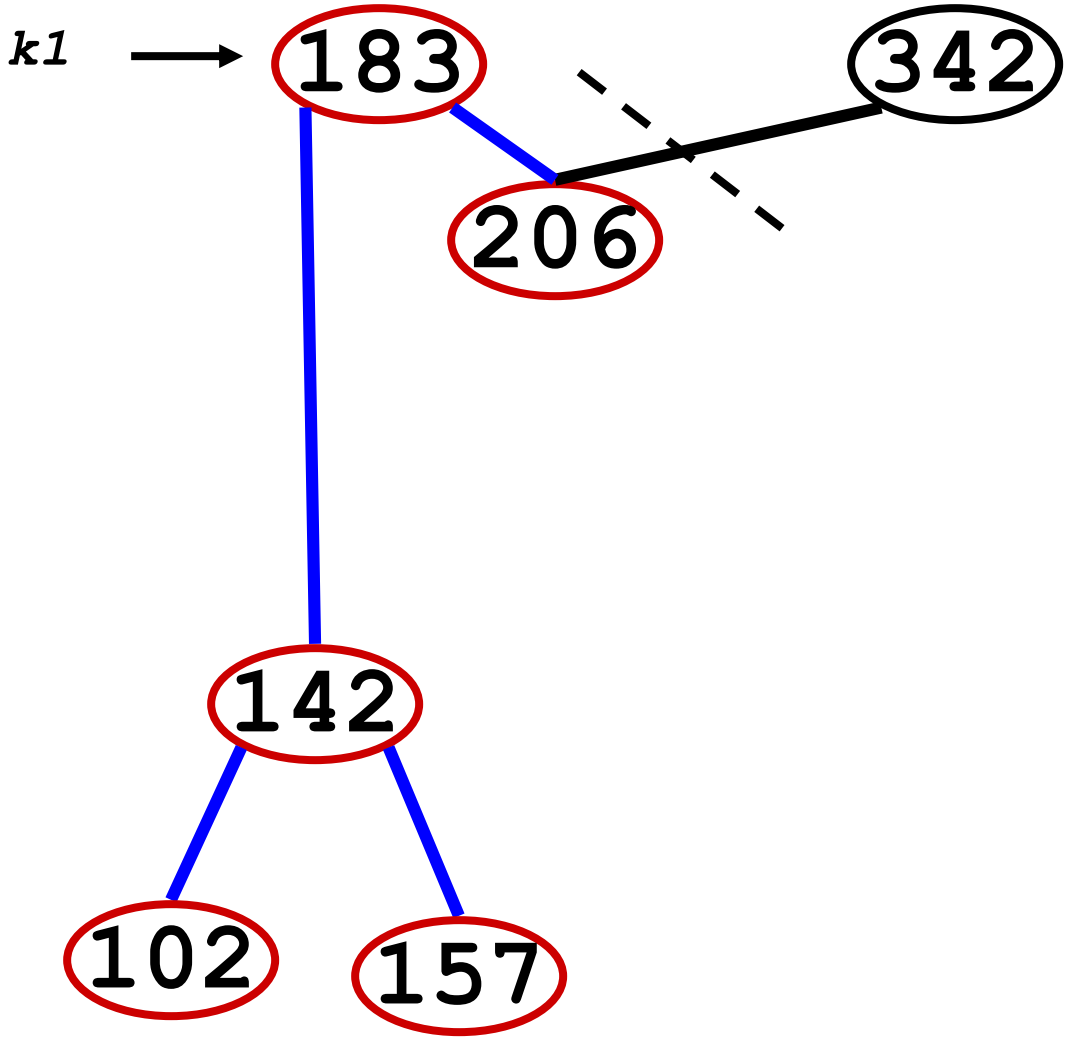
UNIVERSITY OF CALGARY

Double Rotation

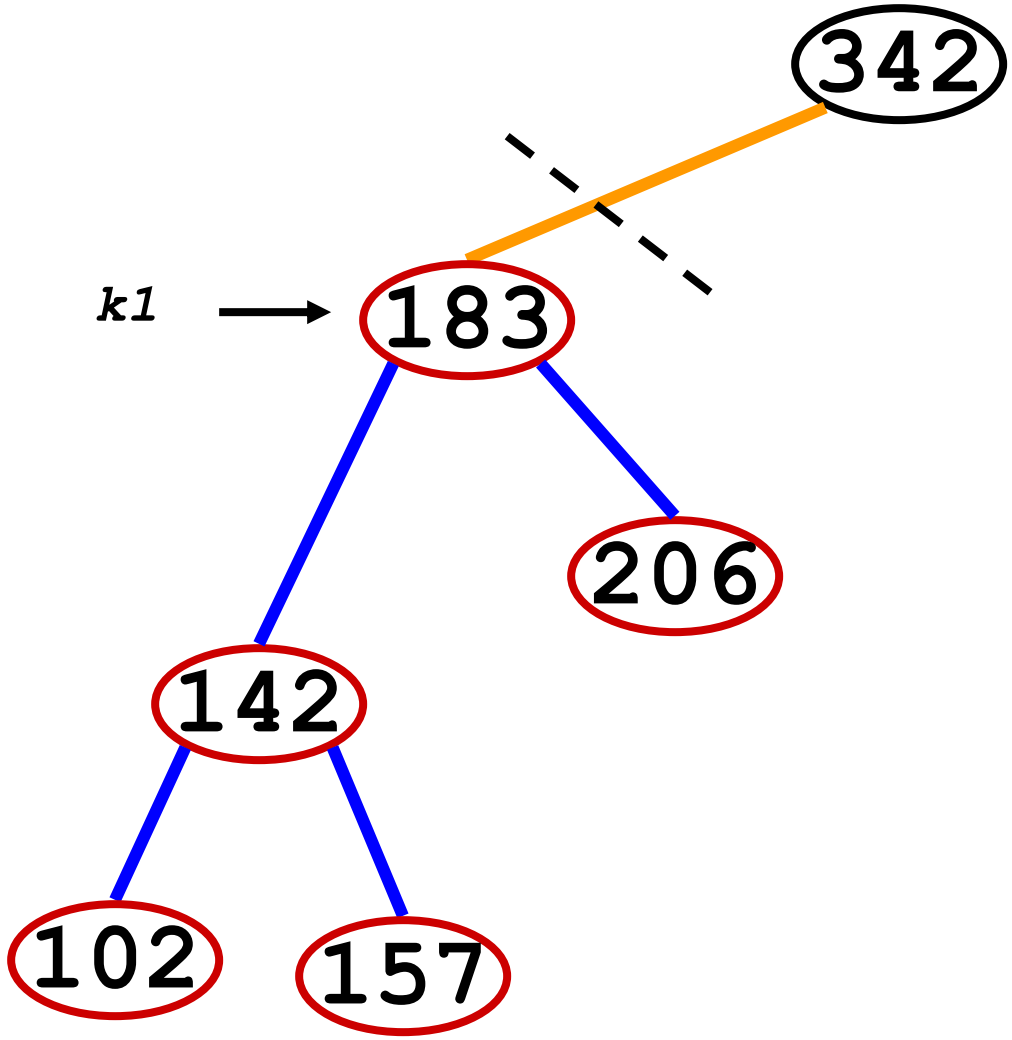
Step 2: Rotate node and new child (AVL)

LEFT rotation

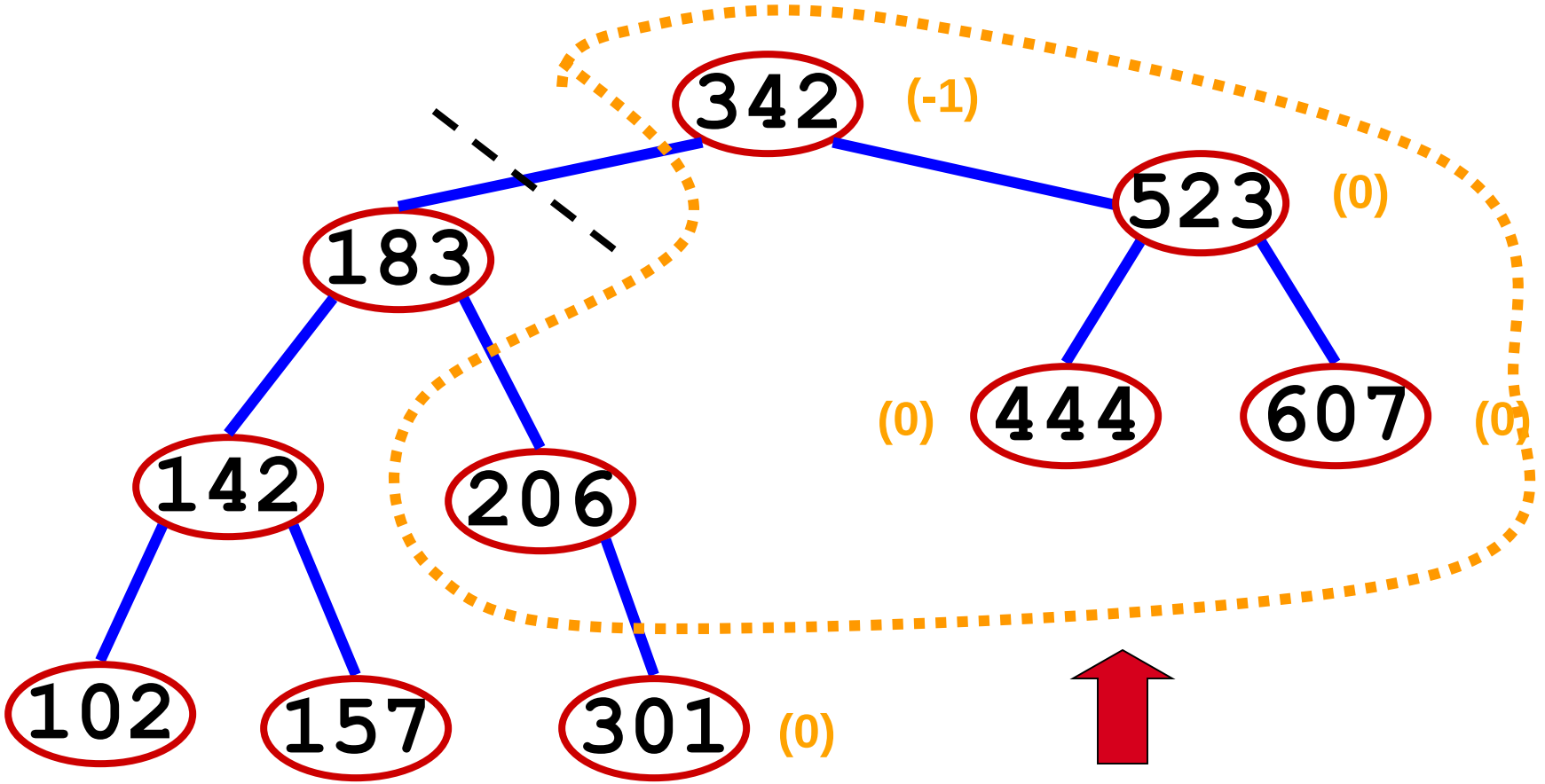
```
BinaryNode *k2 = k1->left;
k1->left = k2->right;
k2->right = k1;
k1 = k2;
```



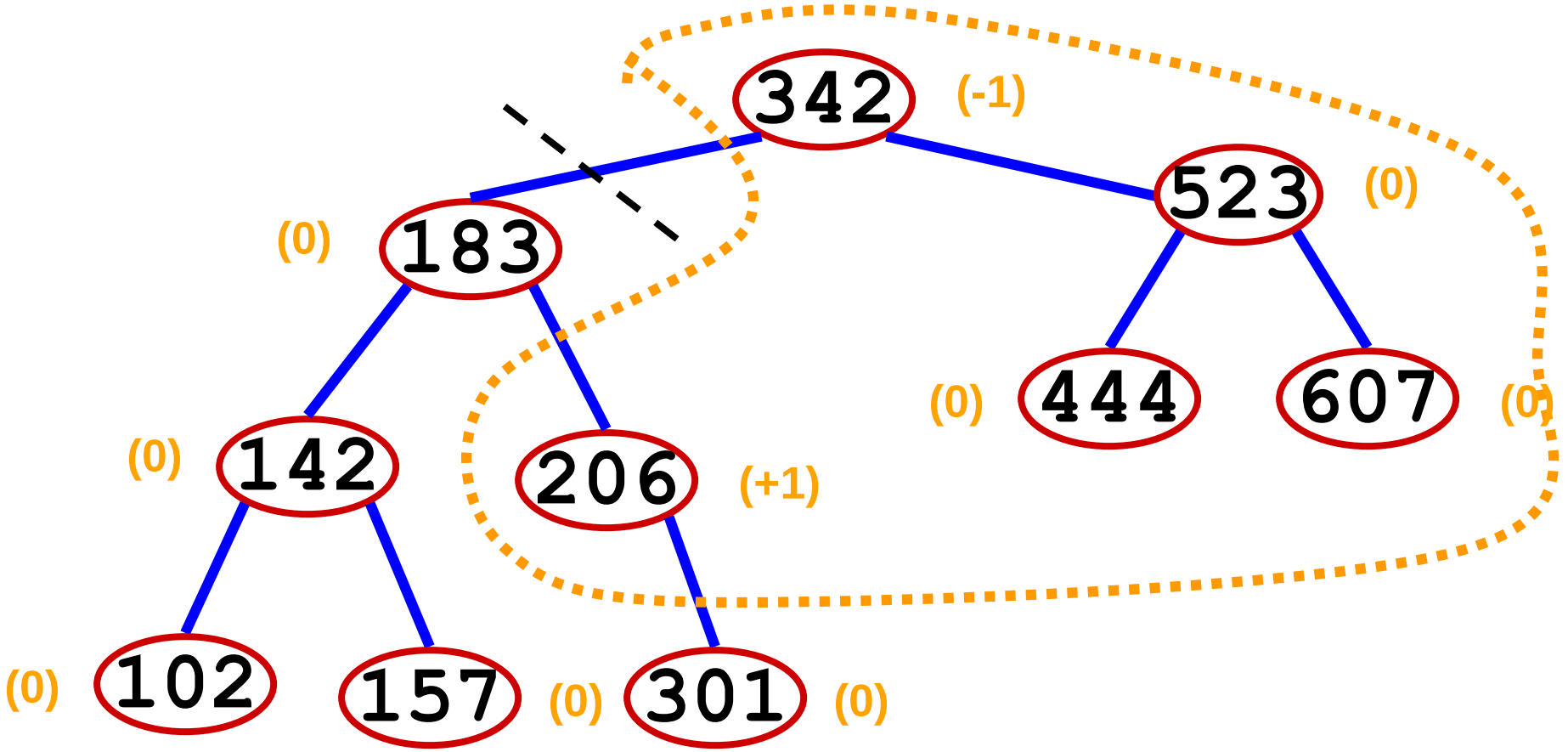
UNIVERSITY OF CALGARY



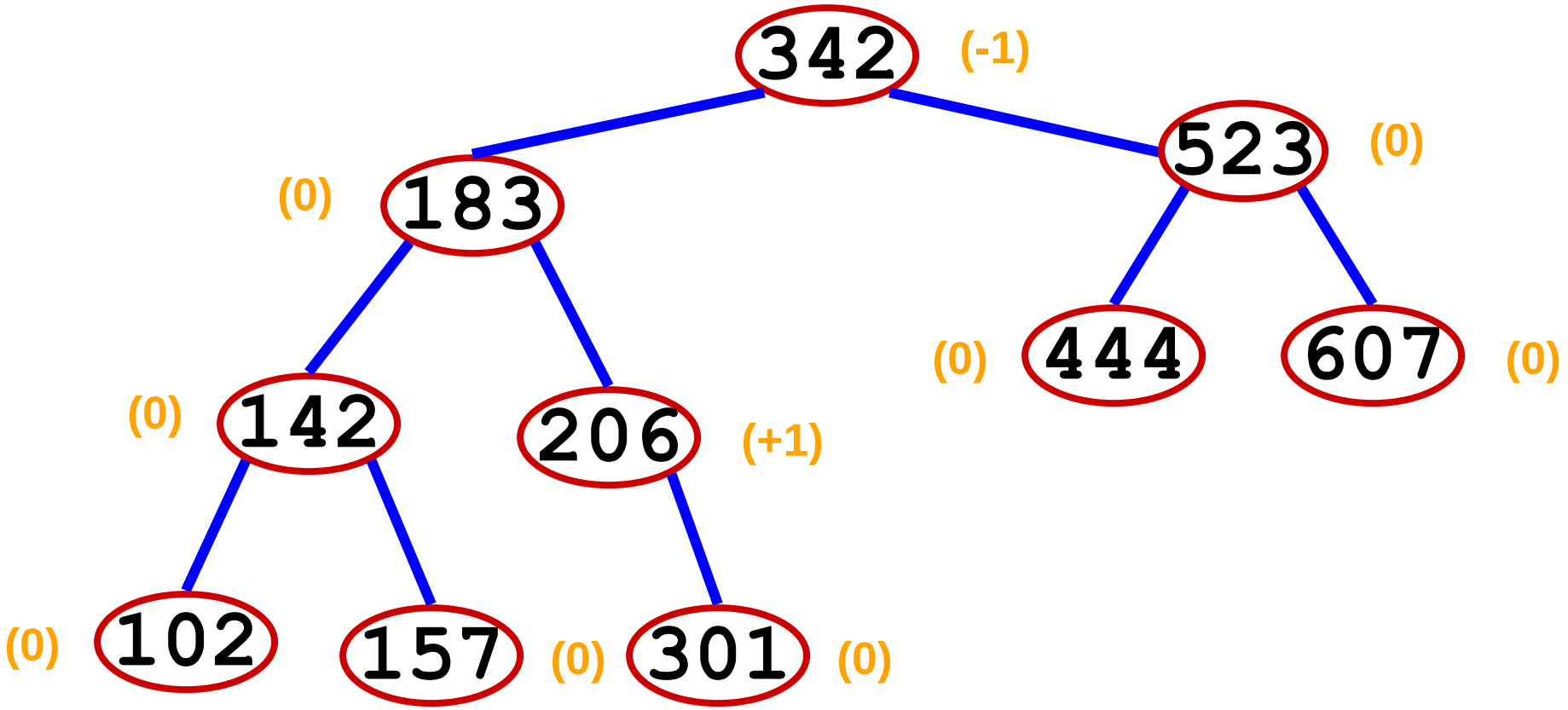
UNIVERSITY OF CALGARY



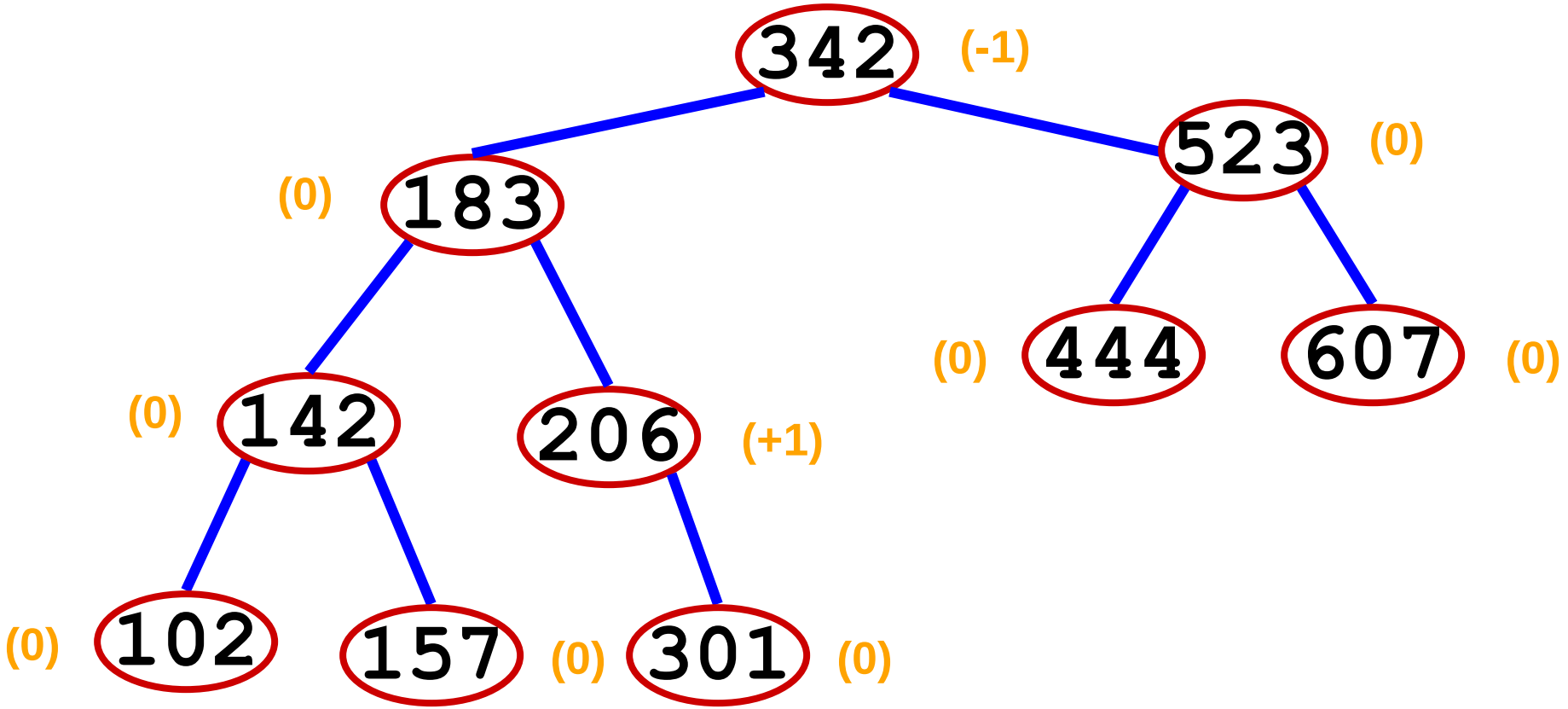
Back to the rest of the tree…



New factors …



Left hand subtree is now almost perfectly balanced


342
(-1)
523
(0)
(0)
183
(0)
444
607
(0)
(0)
142
206
(+1)
(0)
102
157
(0)
301
(0)