

本节内容

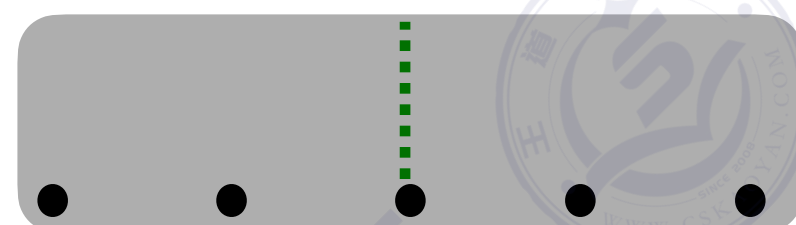
B树

插入和删除

B树的插入



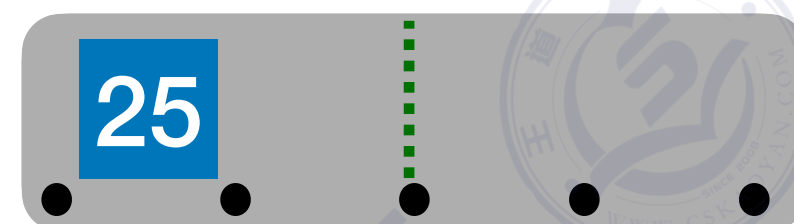
5阶B树——结点关键字个数 $\lceil m/2 \rceil - 1 \leq n \leq m-1$
即： $2 \leq n \leq 4$ （注：此处省略失败结点）



B树的插入



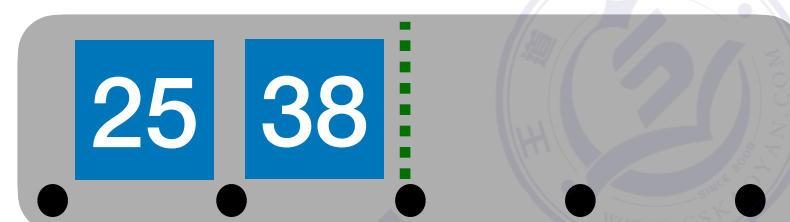
5阶B树——结点关键字个数 $\lceil m/2 \rceil - 1 \leq n \leq m-1$
即： $2 \leq n \leq 4$ （注：此处省略失败结点）



B树的插入



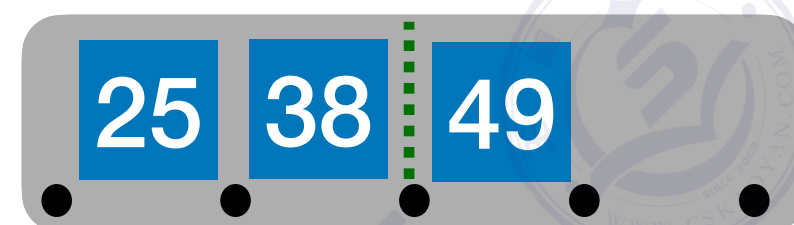
5阶B树——结点关键字个数 $\lceil m/2 \rceil - 1 \leq n \leq m-1$
即： $2 \leq n \leq 4$ （注：此处省略失败结点）



B树的插入



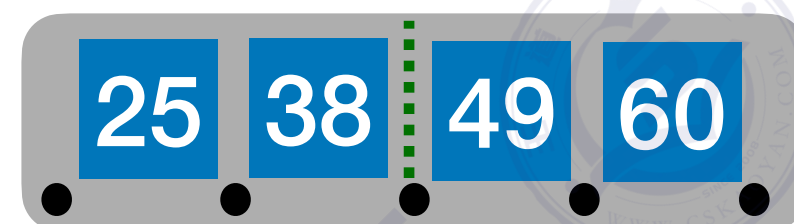
5阶B树——结点关键字个数 $\lceil m/2 \rceil - 1 \leq n \leq m-1$
即： $2 \leq n \leq 4$ （注：此处省略失败结点）



B树的插入



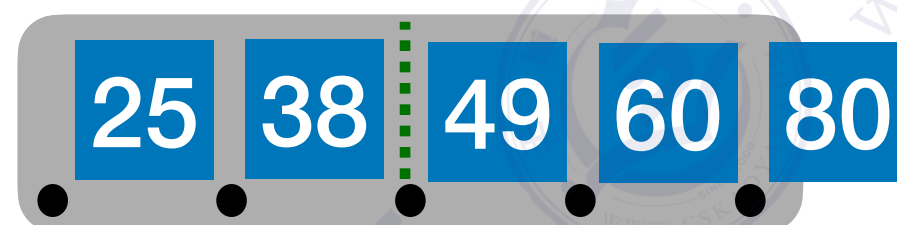
5阶B树——结点关键字个数 $\lceil m/2 \rceil - 1 \leq n \leq m-1$
即： $2 \leq n \leq 4$ （注：此处省略失败结点）



B树的插入



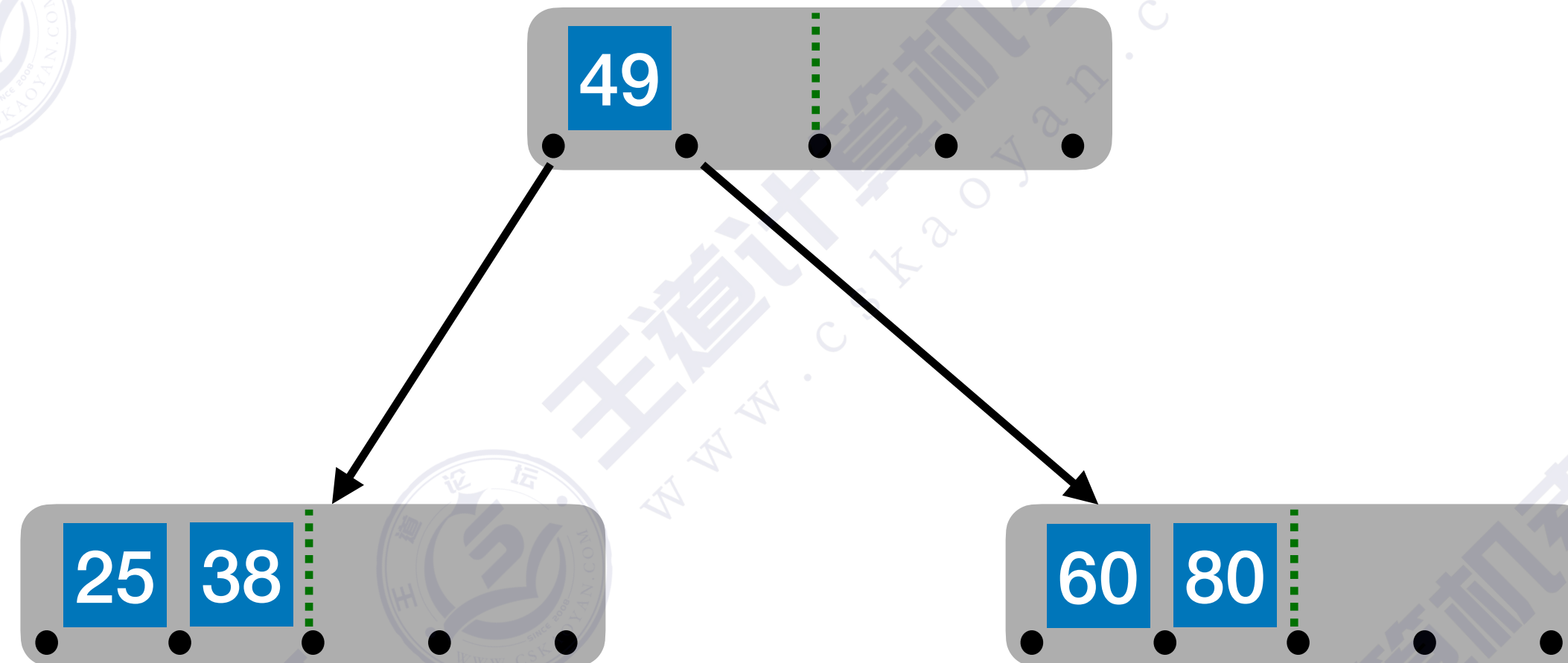
5阶B树——结点关键字个数 $\lceil m/2 \rceil - 1 \leq n \leq m-1$
即： $2 \leq n \leq 4$ （注：此处省略失败结点）



B树的插入



5阶B树——结点关键字个数 $\lceil m/2 \rceil - 1 \leq n \leq m-1$
即： $2 \leq n \leq 4$ （注：此处省略失败结点）

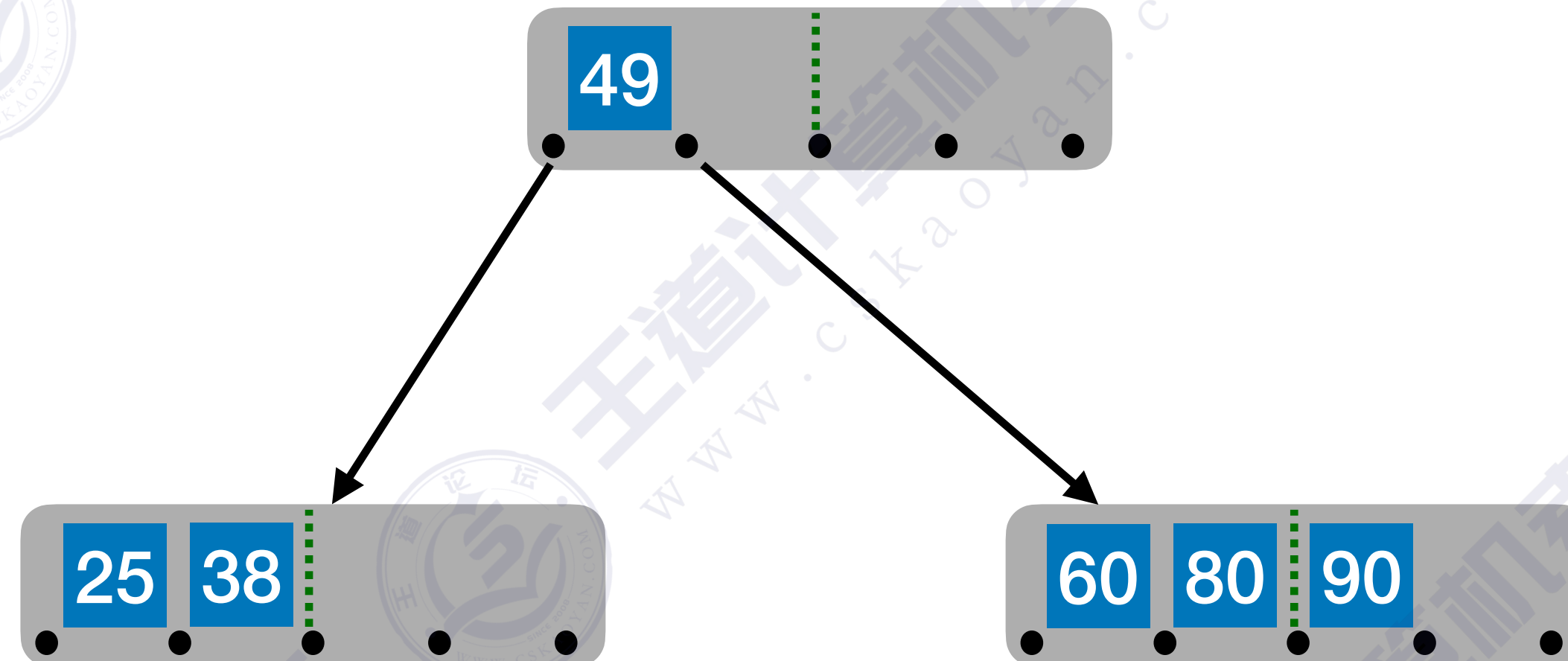


在插入key后，若导致原结点关键字数超过上限，则从中间位置（ $\lceil m/2 \rceil$ ）将其中的关键字分为两部分，左部分包含的关键字放在原结点中，右部分包含的关键字放到新结点中，中间位置（ $\lceil m/2 \rceil$ ）的结点插入原结点的父结点

B树的插入



5阶B树——结点关键字个数 $\lceil m/2 \rceil - 1 \leq n \leq m-1$
即： $2 \leq n \leq 4$ （注：此处省略失败结点）

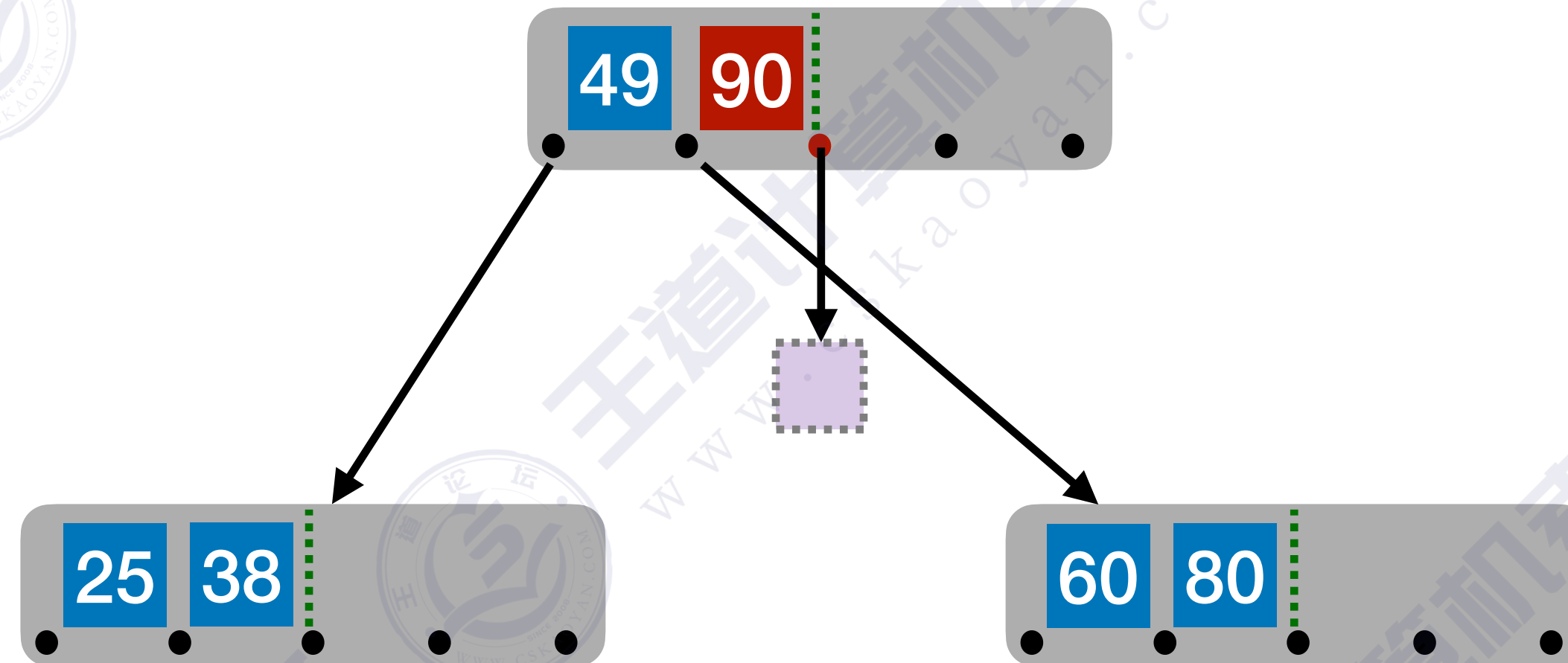


新元素一定是插入到最底层“终端节点”，用“查找”来确定插入位置

B树的插入



5阶B树——结点关键字个数 $\lceil m/2 \rceil - 1 \leq n \leq m-1$
即： $2 \leq n \leq 4$ （注：此处省略失败结点）



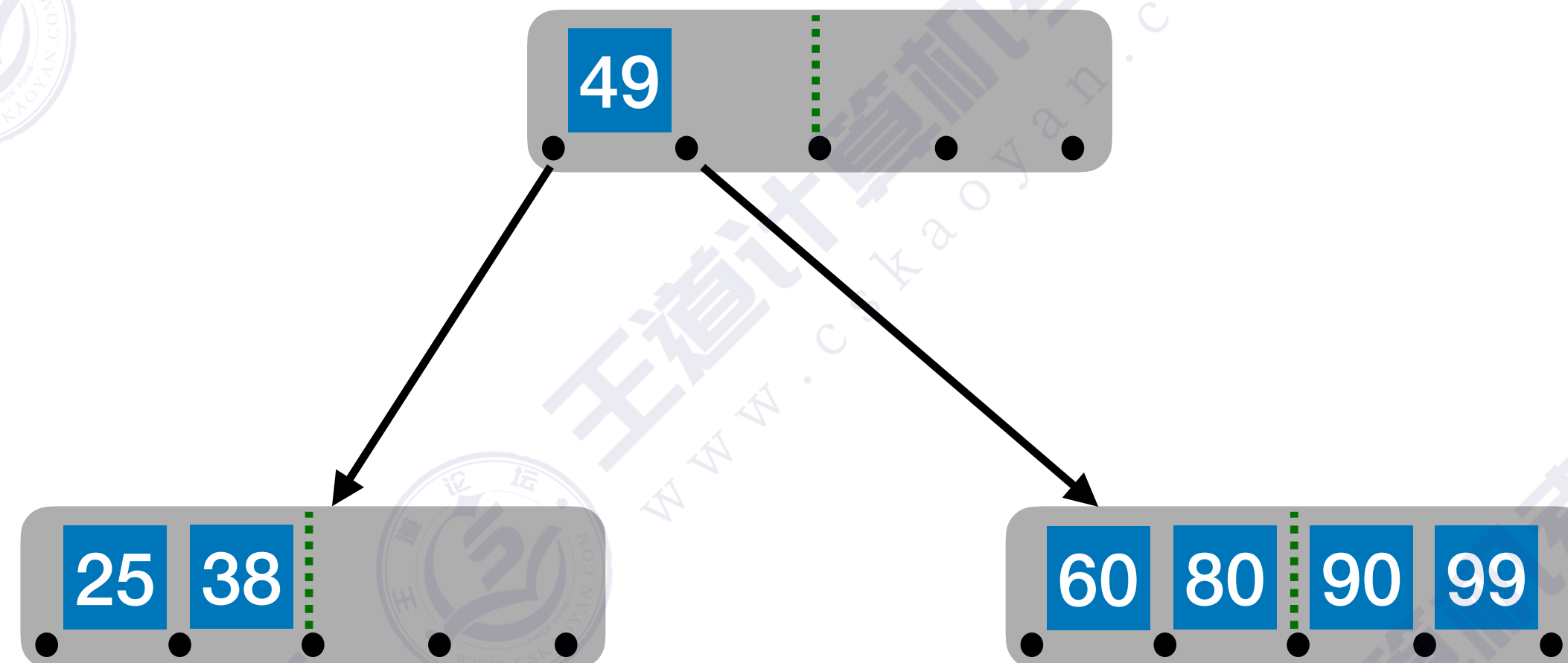
错误示范

注意：B树的失败结点只能出现在最下面一层

B树的插入



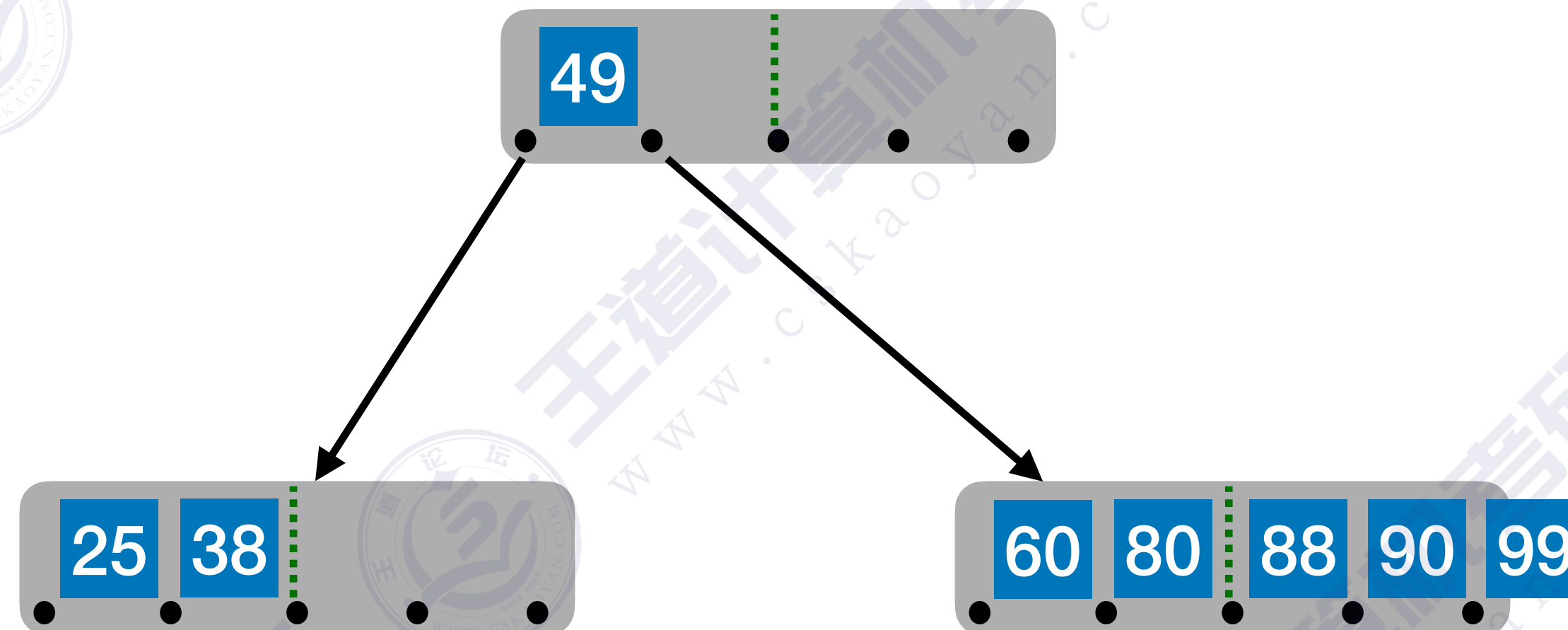
5阶B树——结点关键字个数 $\lceil m/2 \rceil - 1 \leq n \leq m-1$
即： $2 \leq n \leq 4$ （注：此处省略失败结点）



B树的插入



5阶B树——结点关键字个数 $\lceil m/2 \rceil - 1 \leq n \leq m-1$
即： $2 \leq n \leq 4$ （注：此处省略失败结点）

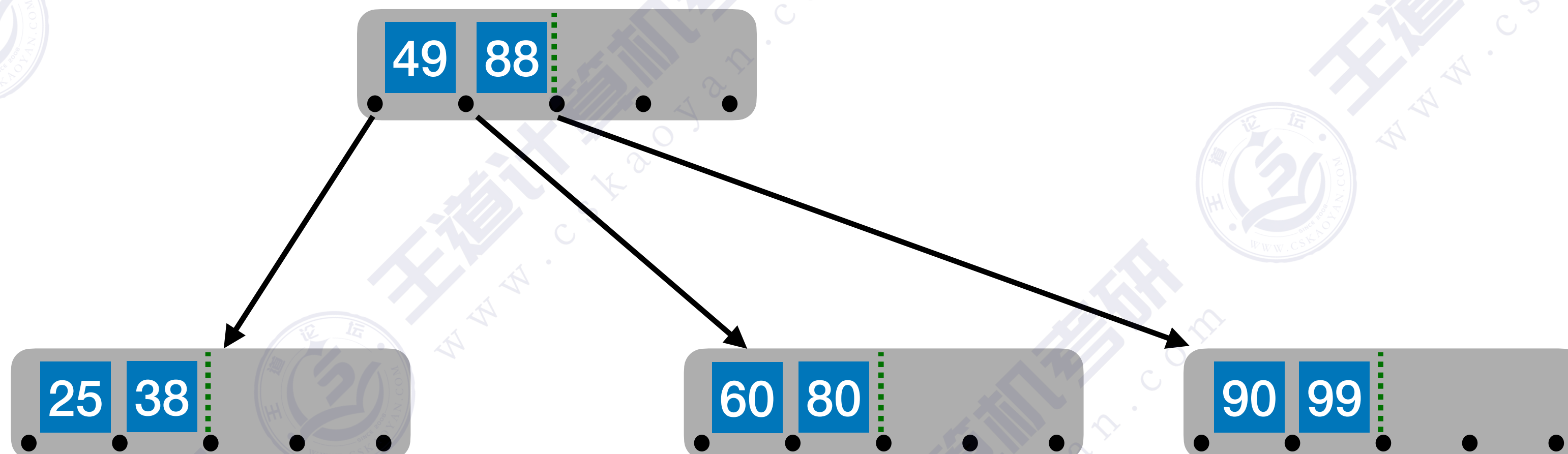


在插入key后，若导致原结点关键字数超过上限，则从中间位置（ $\lceil m/2 \rceil$ ）将其中的关键字分为两部分，左部分包含的关键字放在原结点中，右部分包含的关键字放到新结点中，中间位置（ $\lceil m/2 \rceil$ ）的结点插入原结点的父结点

B树的插入



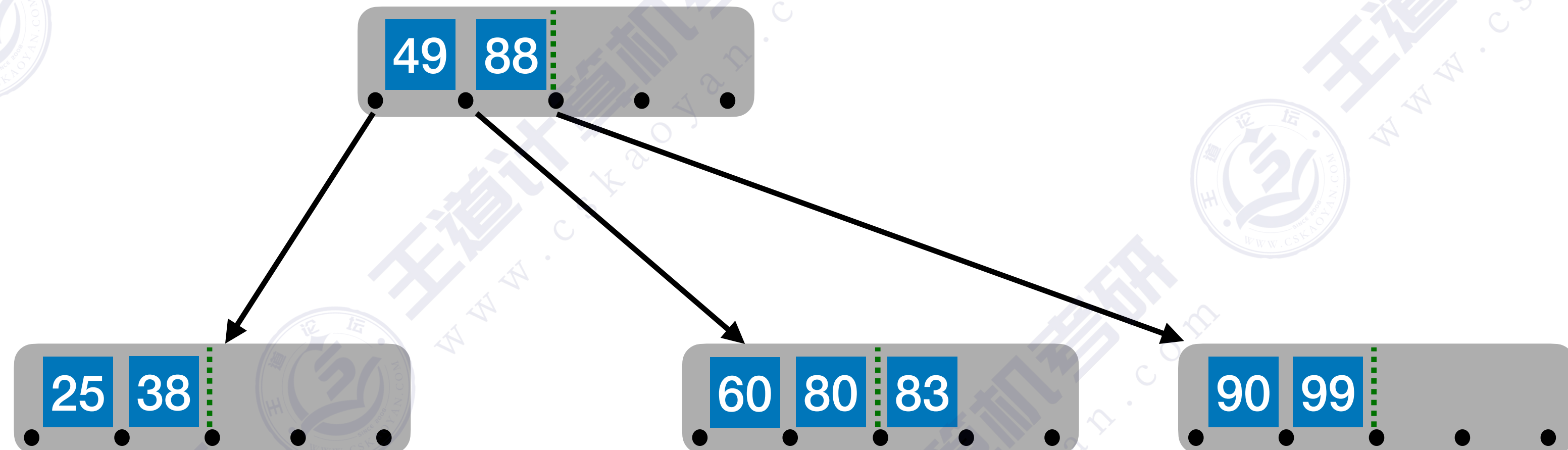
5阶B树——结点关键字个数 $\lceil m/2 \rceil - 1 \leq n \leq m-1$
即： $2 \leq n \leq 4$ （注：此处省略失败结点）



B树的插入



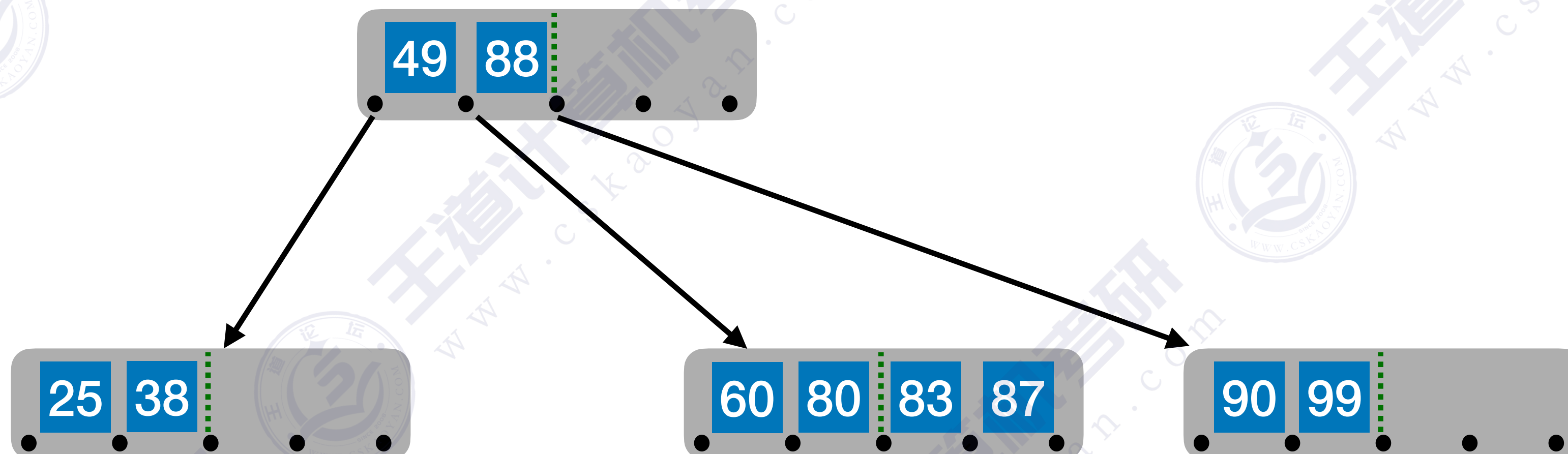
5阶B树——结点关键字个数 $\lceil m/2 \rceil - 1 \leq n \leq m-1$
即： $2 \leq n \leq 4$ （注：此处省略失败结点）



B树的插入



5阶B树——结点关键字个数 $\lceil m/2 \rceil - 1 \leq n \leq m-1$
即： $2 \leq n \leq 4$ （注：此处省略失败结点）

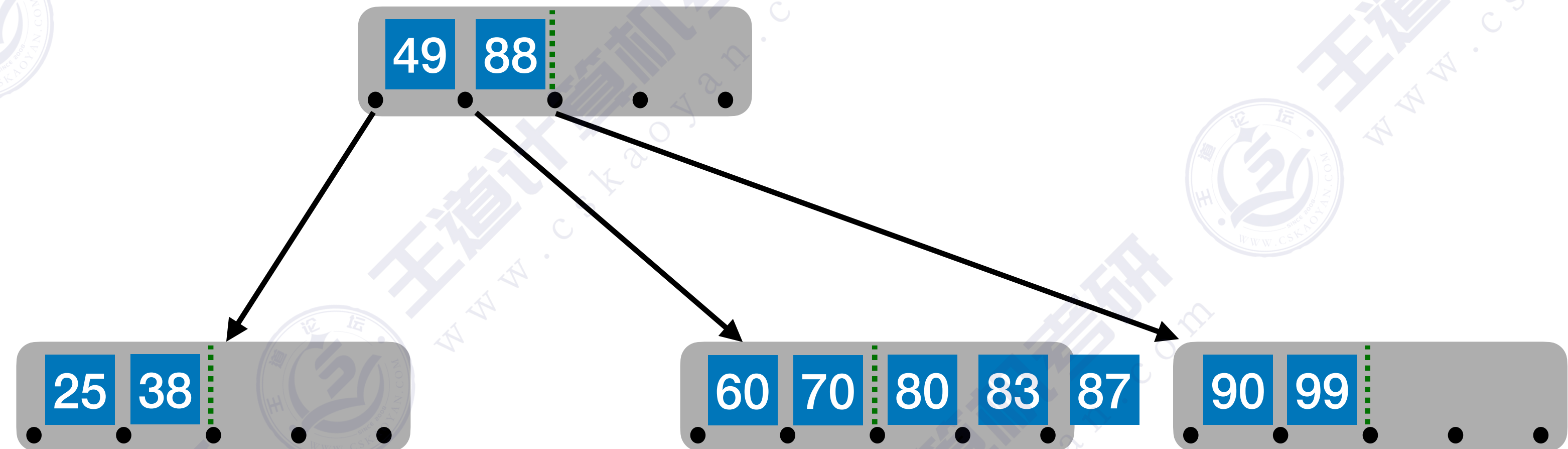


B树的插入



思考：80要放到父节点中，放在哪个位置合适？

5阶B树——结点关键字个数 $\lceil m/2 \rceil - 1 \leq n \leq m-1$
即： $2 \leq n \leq 4$ （注：此处省略失败结点）



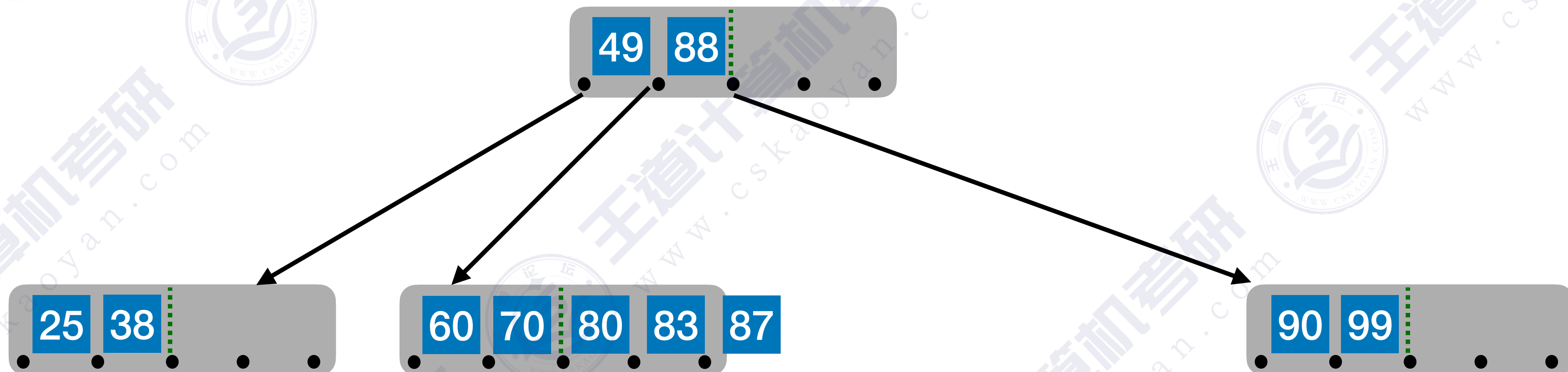
在插入key后，若导致原结点关键字数超过上限，则从中间位置（ $\lceil m/2 \rceil$ ）将其中的关键字分为两部分，左部分包含的关键字放在原结点中，右部分包含的关键字放到新结点中，中间位置（ $\lceil m/2 \rceil$ ）的结点插入原结点的父结点

B树的插入



思考：80要放到父节点中，放在哪个位置合适？

5阶B树——结点关键字个数 $\lceil m/2 \rceil - 1 \leq n \leq m-1$
即： $2 \leq n \leq 4$ （注：此处省略失败结点）

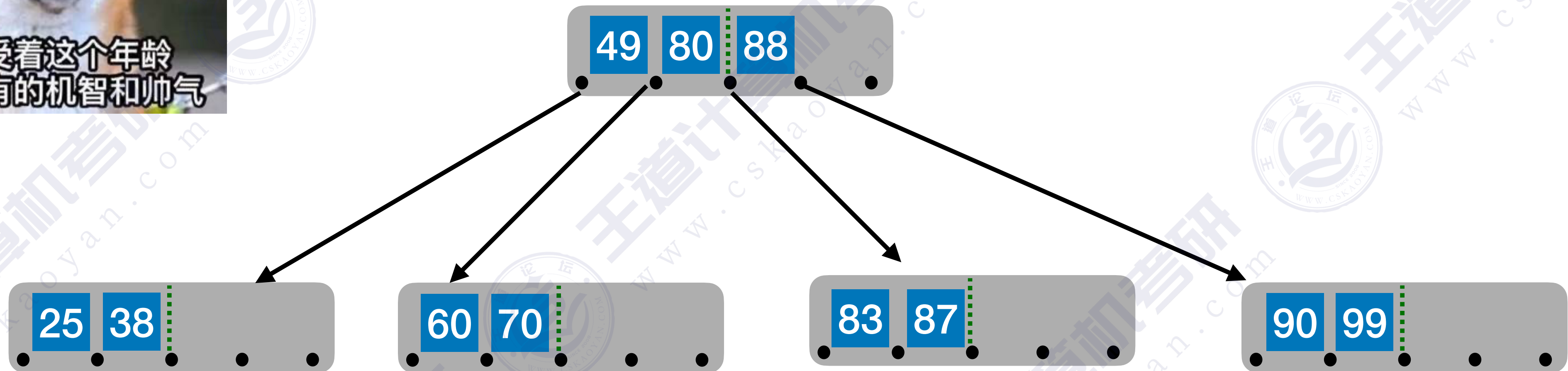


在插入key后，若导致原结点关键字数超过上限，则从中间位置（ $\lceil m/2 \rceil$ ）将其中的关键字分为两部分，左部分包含的关键字放在原结点中，右部分包含的关键字放到新结点中，中间位置（ $\lceil m/2 \rceil$ ）的结点插入原结点的父结点

B树的插入



5阶B树——结点关键字个数 $\lceil m/2 \rceil - 1 \leq n \leq m-1$
即： $2 \leq n \leq 4$ （注：此处省略失败结点）

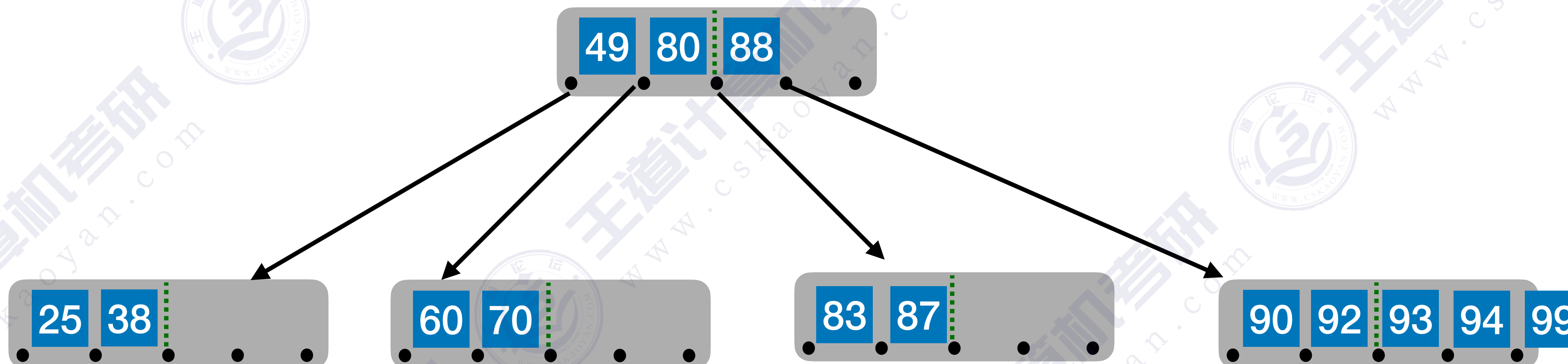


在插入key后，若导致原结点关键字数超过上限，则从中间位置（ $\lceil m/2 \rceil$ ）将其中的关键字分为两部分，左部分包含的关键字放在原结点中，右部分包含的关键字放到新结点中，中间位置（ $\lceil m/2 \rceil$ ）的结点插入原结点的父结点

B树的插入



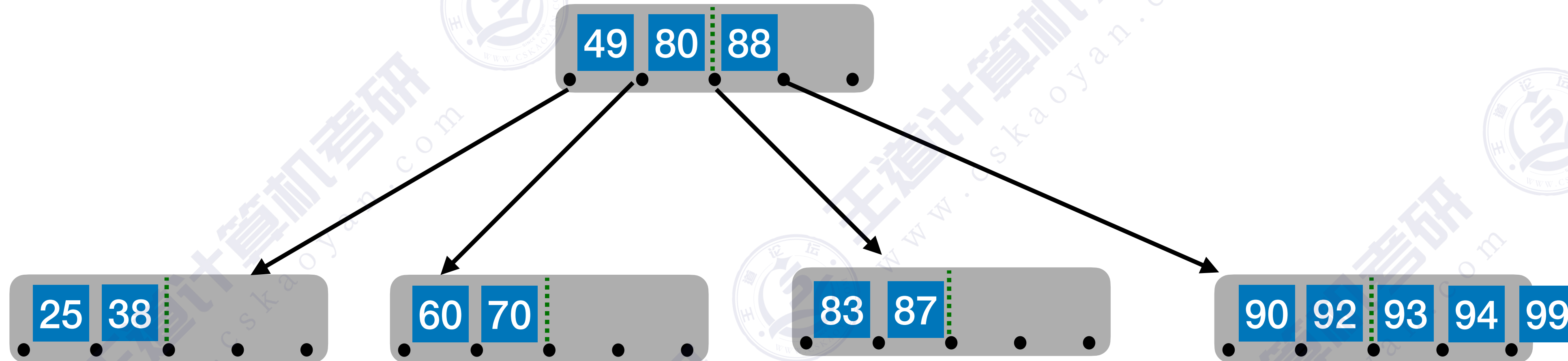
5阶B树——结点关键字个数 $\lceil m/2 \rceil - 1 \leq n \leq m-1$
即： $2 \leq n \leq 4$ （注：此处省略失败结点）



B树的插入



5阶B树——结点关键字个数 $\lceil m/2 \rceil - 1 \leq n \leq m-1$
即： $2 \leq n \leq 4$ （注：此处省略失败结点）

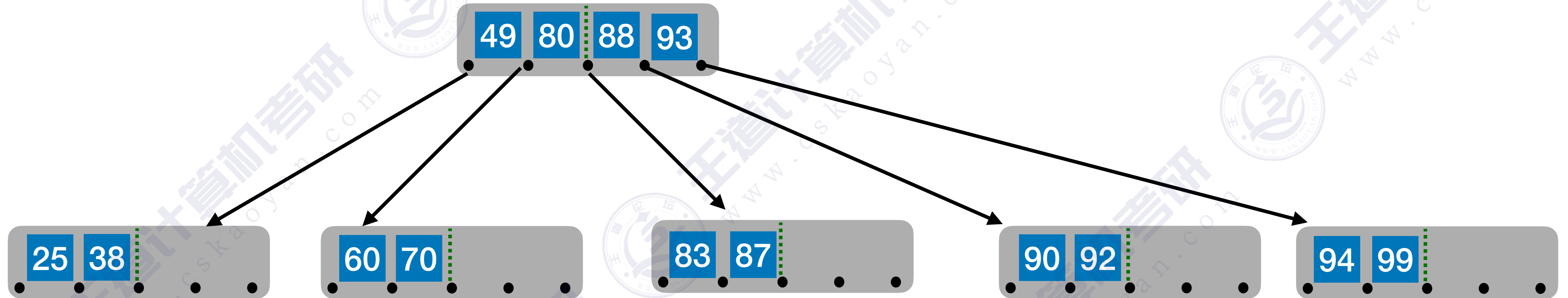


在插入key后，若导致原结点关键字数超过上限，则从中间位置（ $\lceil m/2 \rceil$ ）将其中的关键字分为两部分，左部分包含的关键字放在原结点中，右部分包含的关键字放到新结点中，中间位置（ $\lceil m/2 \rceil$ ）的结点插入原结点的父结点

B树的插入



5阶B树——结点关键字个数 $\lceil m/2 \rceil - 1 \leq n \leq m-1$
即： $2 \leq n \leq 4$ （注：此处省略失败结点）

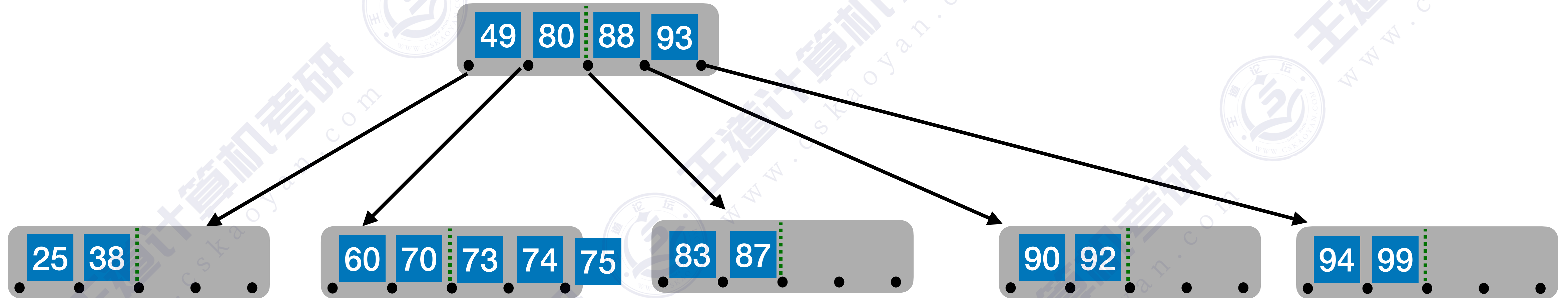


在插入key后，若导致原结点关键字数超过上限，则从中间位置（ $\lceil m/2 \rceil$ ）将其中的关键字分为两部分，左部分包含的关键字放在原结点中，右部分包含的关键字放到新结点中，中间位置（ $\lceil m/2 \rceil$ ）的结点插入原结点的父结点

B树的插入



5阶B树——结点关键字个数 $\lceil m/2 \rceil - 1 \leq n \leq m-1$
即： $2 \leq n \leq 4$ （注：此处省略失败结点）

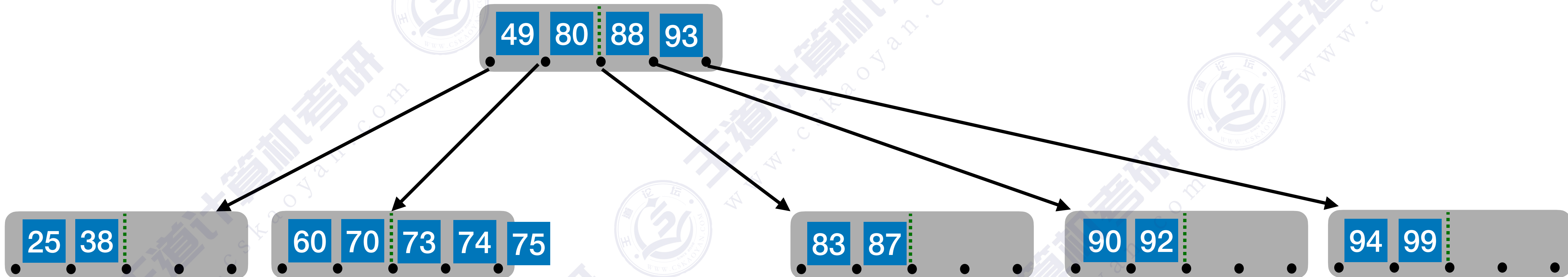


在插入key后，若导致原结点关键字数超过上限，则从中间位置（ $\lceil m/2 \rceil$ ）将其中的关键字分为两部分，左部分包含的关键字放在原结点中，右部分包含的关键字放到新结点中，中间位置（ $\lceil m/2 \rceil$ ）的结点插入原结点的父结点

B树的插入



5阶B树——结点关键字个数 $\lceil m/2 \rceil - 1 \leq n \leq m-1$
即： $2 \leq n \leq 4$ （注：此处省略失败结点）

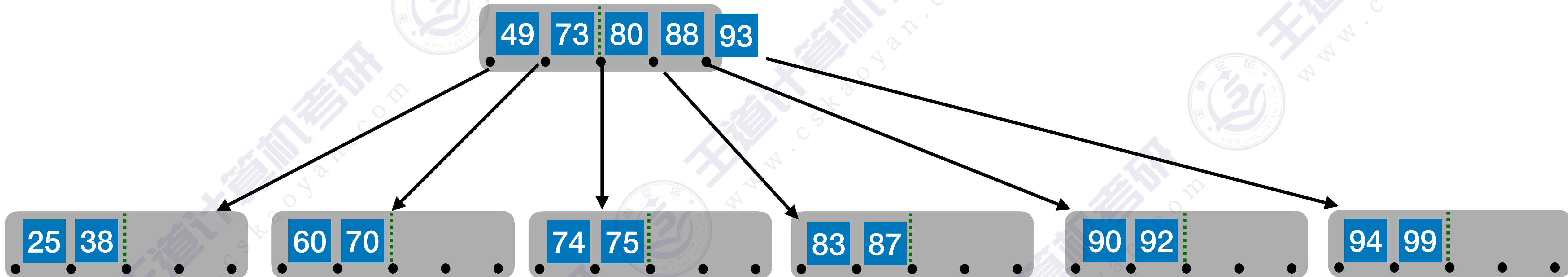


在插入key后，若导致原结点关键字数超过上限，则从中间位置（ $\lceil m/2 \rceil$ ）将其中的关键字分为两部分，左部分包含的关键字放在原结点中，右部分包含的关键字放到新结点中，中间位置（ $\lceil m/2 \rceil$ ）的结点插入原结点的父结点

B树的插入



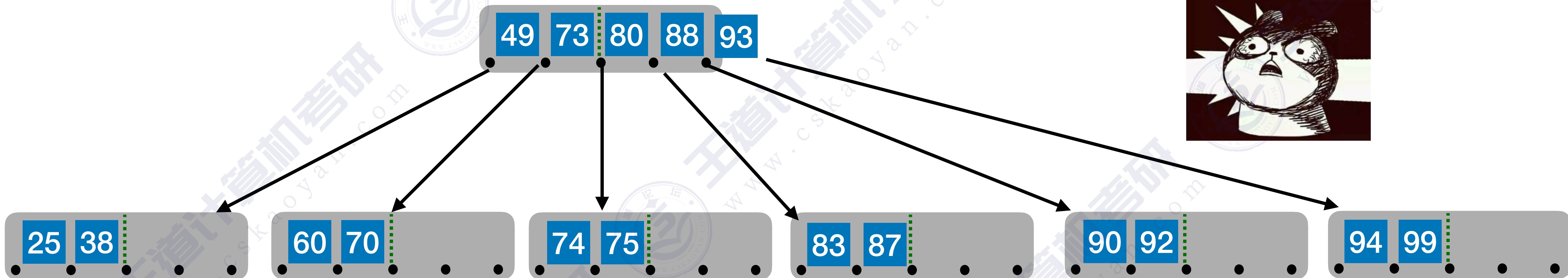
5阶B树——结点关键字个数 $\lceil m/2 \rceil - 1 \leq n \leq m-1$
即： $2 \leq n \leq 4$ （注：此处省略失败结点）



在插入key后，若导致原结点关键字数超过上限，则从中间位置（ $\lceil m/2 \rceil$ ）将其中的关键字分为两部分，左部分包含的关键字放在原结点中，右部分包含的关键字放到新结点中，中间位置（ $\lceil m/2 \rceil$ ）的结点插入原结点的父结点

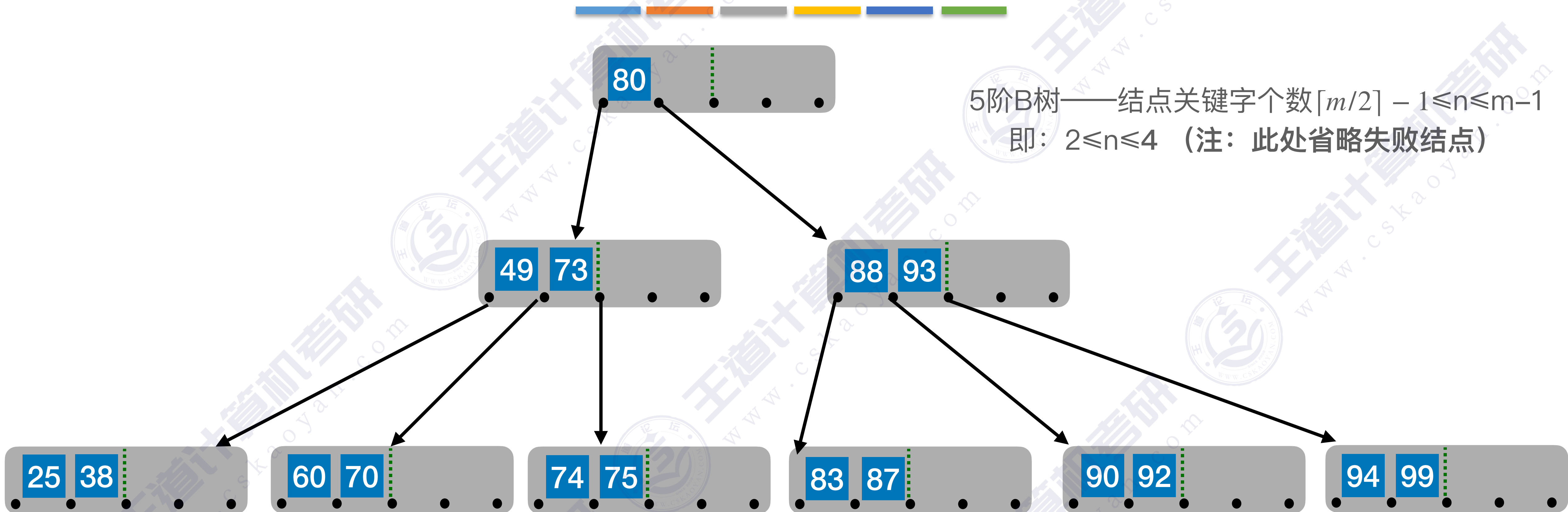
B树的插入

5阶B树——结点关键字个数 $\lceil m/2 \rceil - 1 \leq n \leq m-1$
即： $2 \leq n \leq 4$ （注：此处省略失败结点）



在插入key后，若导致原结点关键字数超过上限，则从中间位置 ($\lceil m/2 \rceil$) 将其中的关键字分为两部分，左部分包含的关键字放在原结点中，右部分包含的关键字放到新结点中，中间位置 ($\lceil m/2 \rceil$) 的结点插入原结点的父结点。若此时导致其父结点的关键字个数也超过了上限，则继续进行这种分裂操作，直至这个过程传到根结点为止，进而导致B树高度增1。

B树的插入



在插入key后，若导致原结点关键字数超过上限，则从中间位置（ $\lceil m/2 \rceil$ ）将其中的关键字分为两部分，左部分包含的关键字放在原结点中，右部分包含的关键字放到新结点中，中间位置（ $\lceil m/2 \rceil$ ）的结点插入原结点的父结点。若此时导致其父结点的关键字数也超过了上限，则继续进行这种分裂操作，直至这个过程传到根结点为止，进而导致B树高度增1。

B树的插入



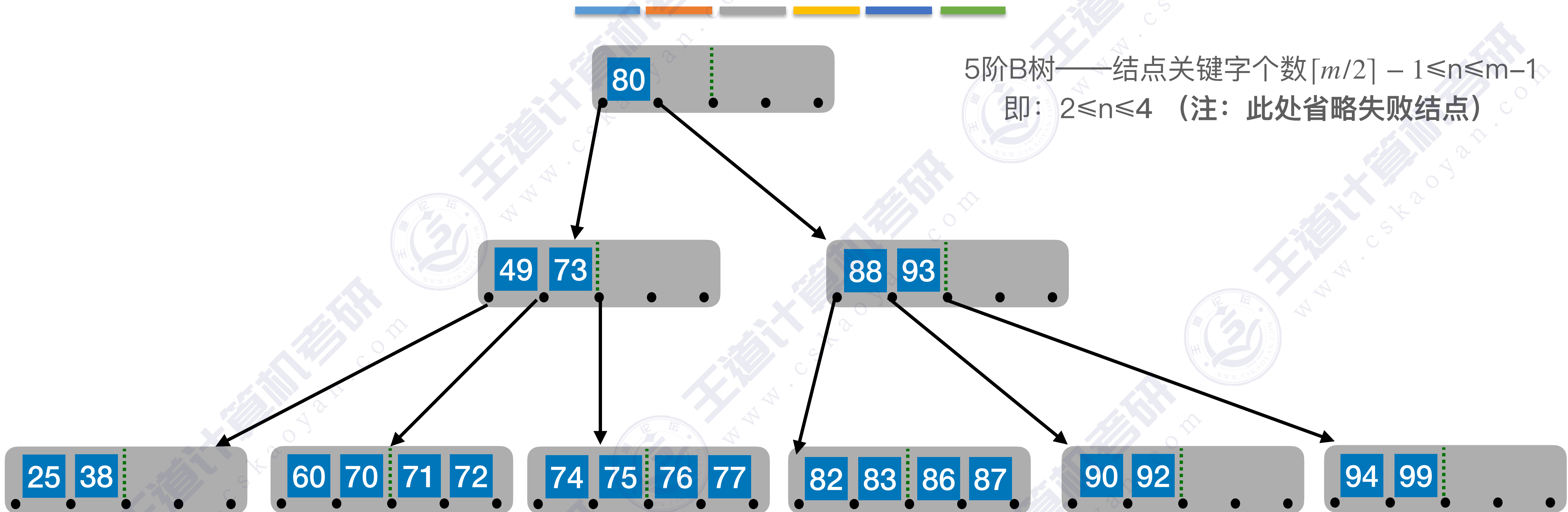
核心要求：

- ①对 m 阶B树——除根节点外，结点关键字个数 $\lceil m/2 \rceil - 1 \leq n \leq m-1$
- ②子树 $0 < \text{关键字} 1 < \text{子树} 1 < \text{关键字} 2 < \text{子树} 2 < \dots$

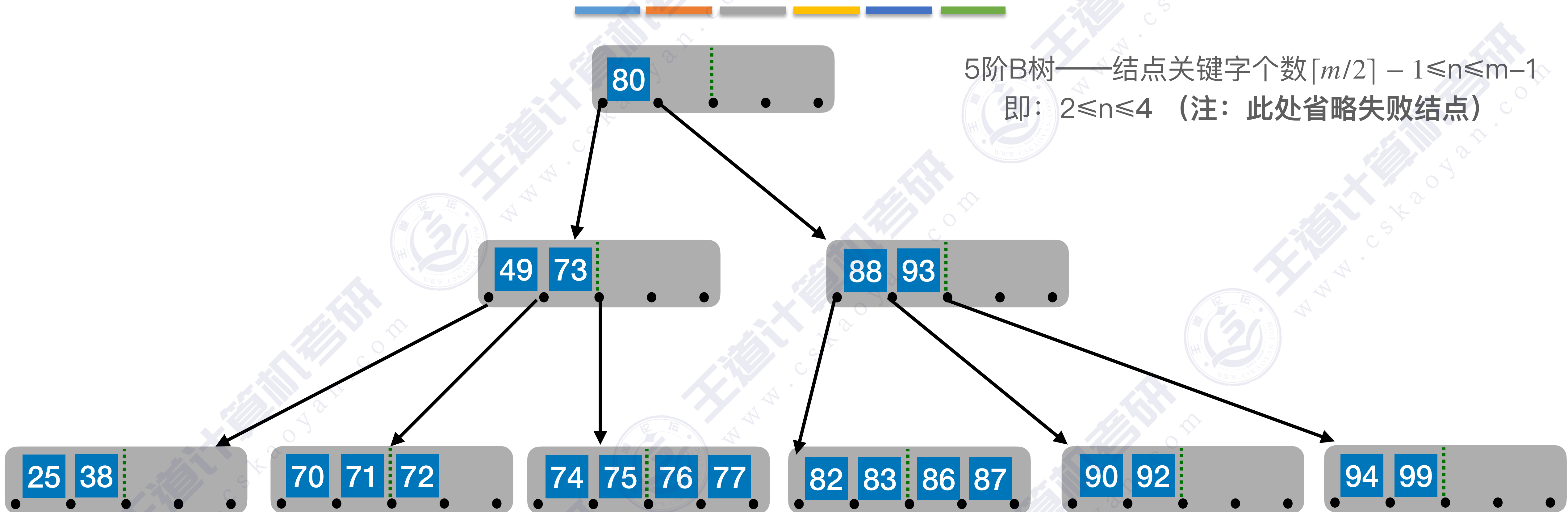
新元素一定是插入到最底层“终端节点”，用“查找”来确定插入位置

在插入key后，若导致原结点关键字数超过上限，则从中间位置（ $\lceil m/2 \rceil$ ）将其中的关键字分为两部分，左部分包含的关键字放在原结点中，右部分包含的关键字放到新结点中，中间位置（ $\lceil m/2 \rceil$ ）的结点插入原结点的父结点。若此时导致其父结点的关键字数也超过了上限，则继续进行这种分裂操作，直至这个过程传到根结点为止，进而导致B树高度增1。

B树的删除

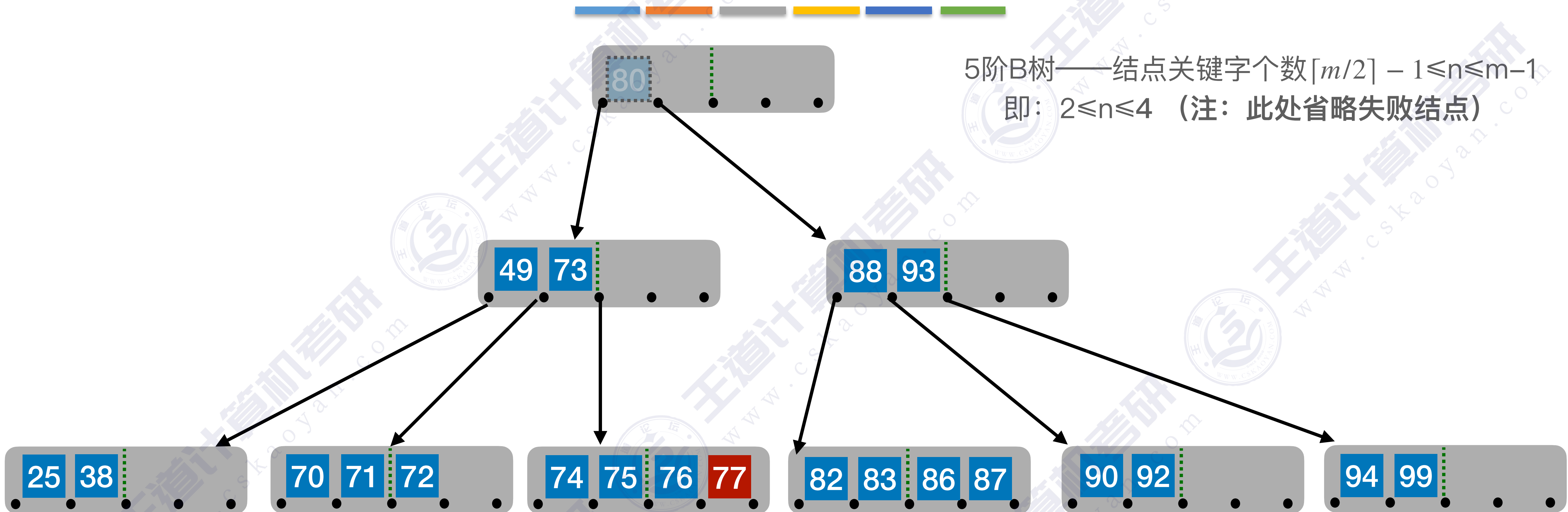


B树的删除



若被删除关键字在**终端节点**，则直接删除该关键字（要注意节点关键字个数是否低于下限 $\lceil m/2 \rceil - 1$ ）

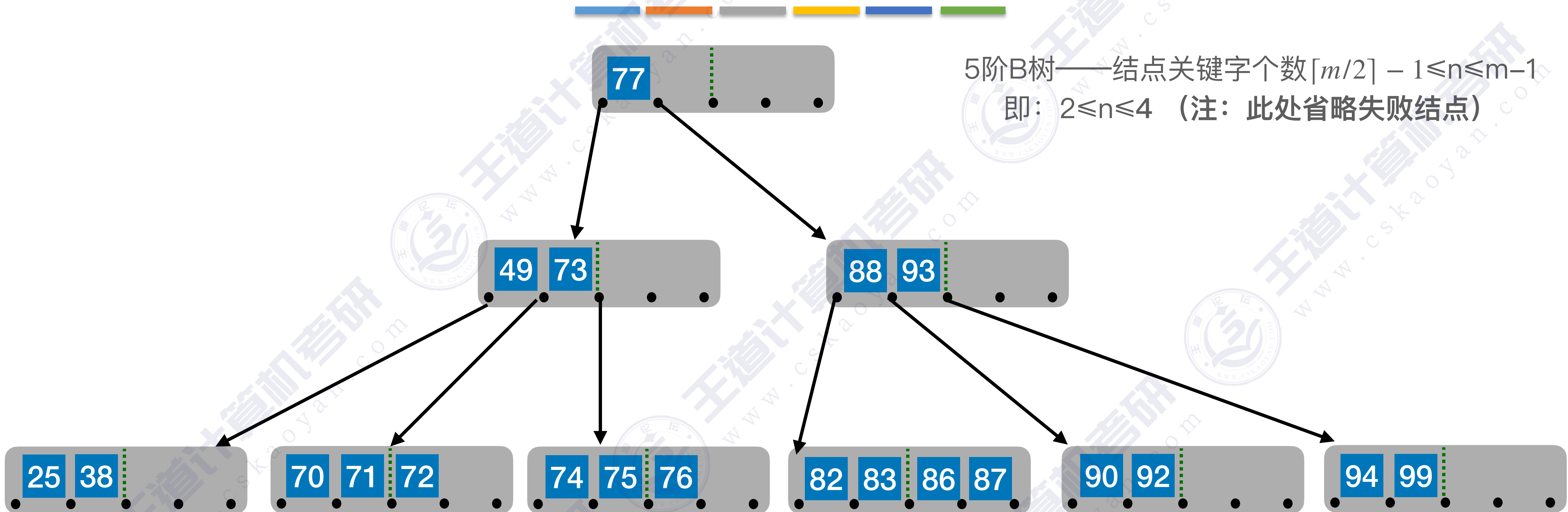
B树的删除



若被删除关键字在**非终端节点**，则用直接前驱或直接后继来替代被删除的关键字

直接前驱：当前关键字**左侧指针所指子树中“最右下”**的元素

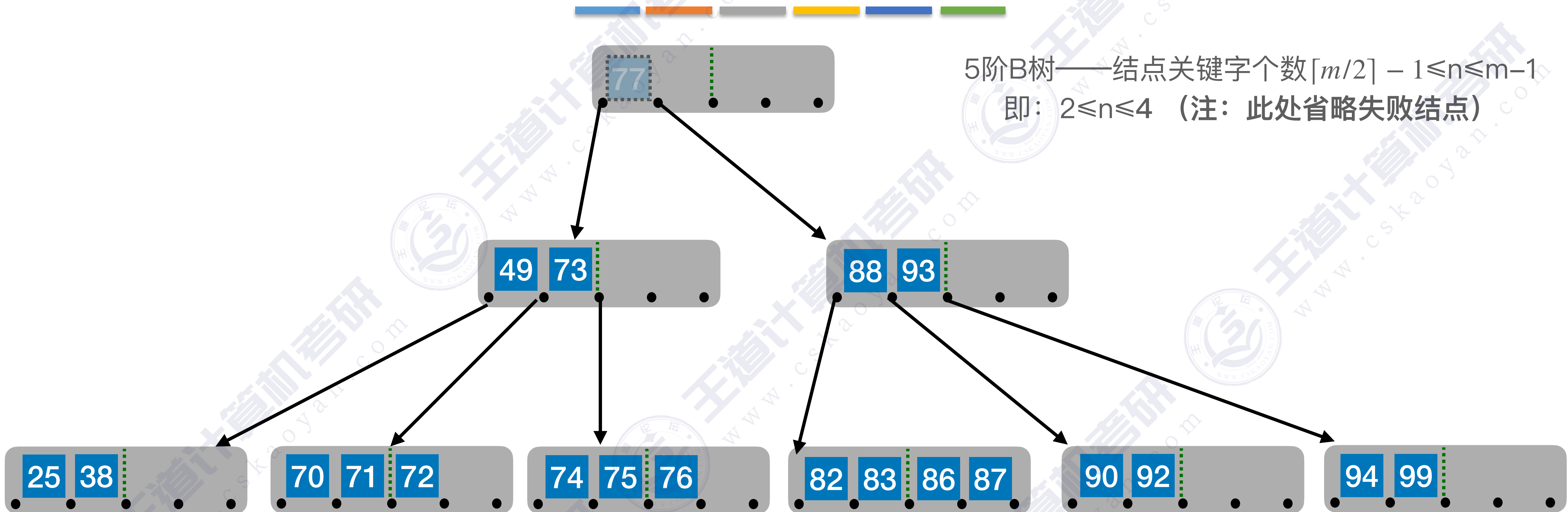
B树的删除



若被删除关键字在**非终端节点**，则用直接前驱或直接后继来替代被删除的关键字

直接前驱：当前关键字**左侧指针所指子树中“最右下”**的元素

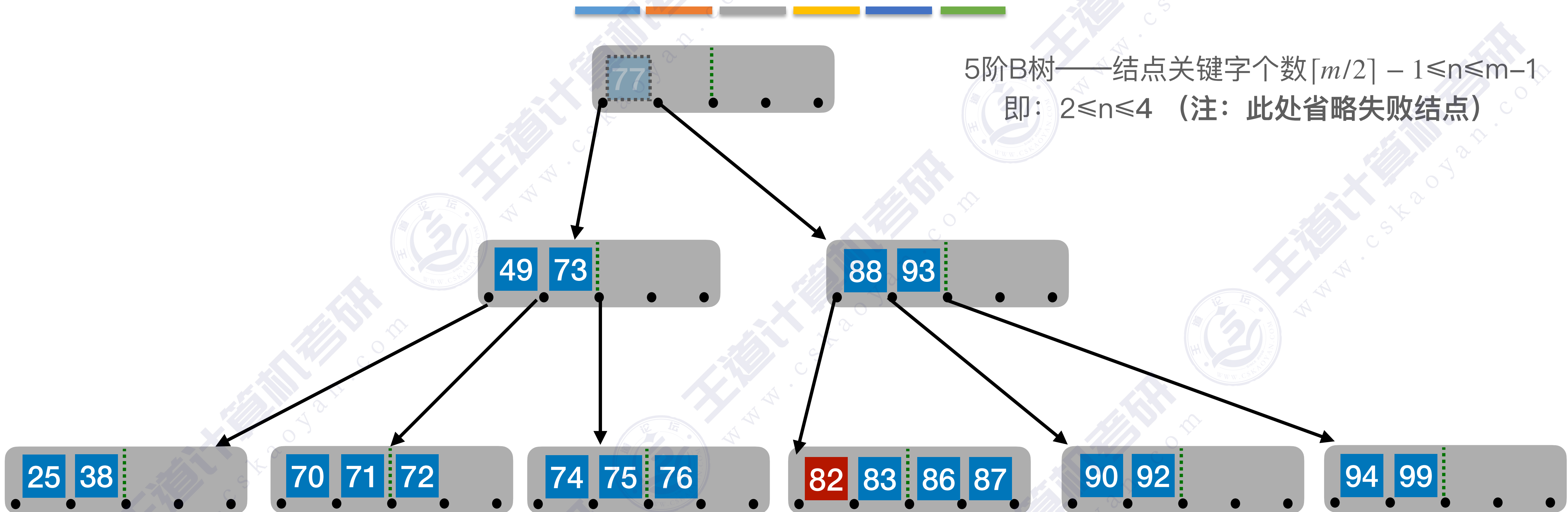
B树的删除



若被删除关键字在**非终端节点**，则用直接前驱或直接后继来替代被删除的关键字

直接前驱：当前关键字**左侧指针**所指子树中“**最右下**”的元素

B树的删除

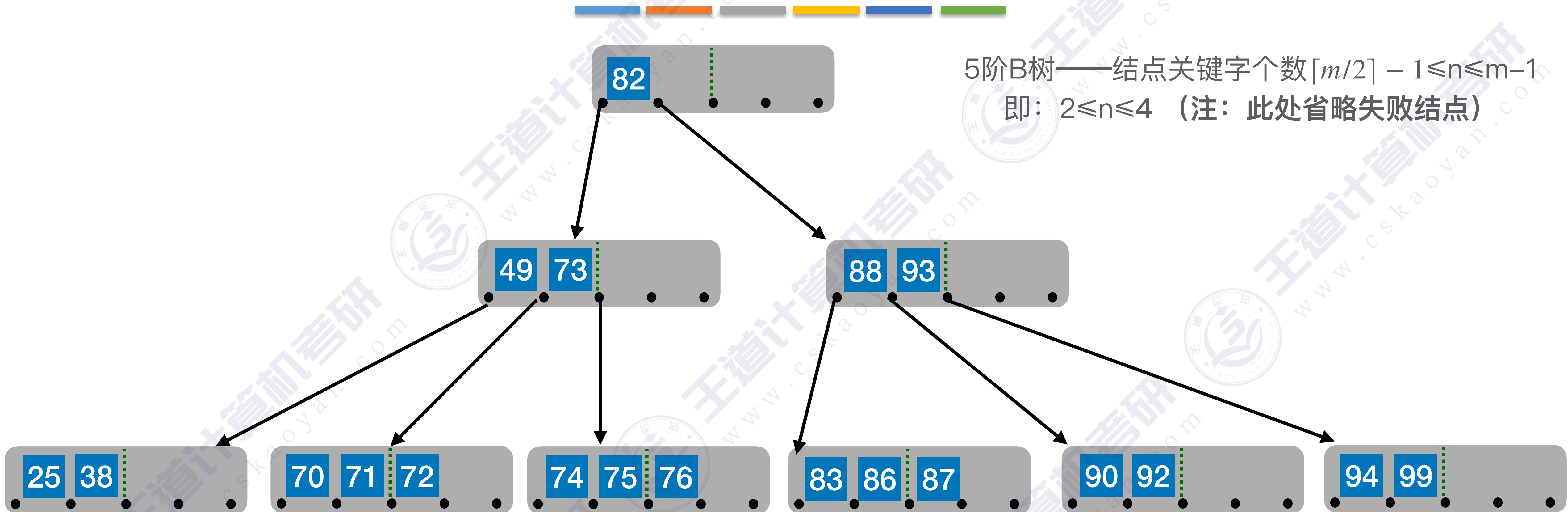


若被删除关键字在**非终端节点**，则用直接前驱或直接后继来替代被删除的关键字

直接前驱：当前关键字**左侧**指针所指子树中“**最右下**”的元素

直接后继：当前关键字**右侧**指针所指子树中“**最左下**”的元素

B树的删除



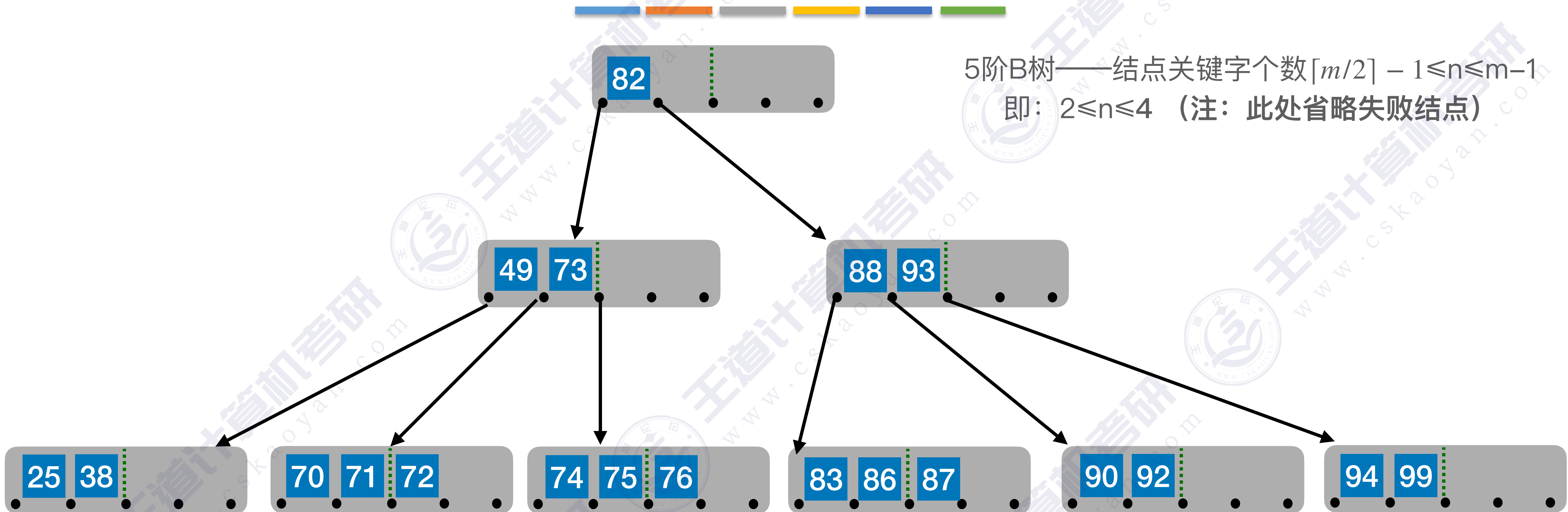
若被删除关键字在**非终端节点**，则用直接前驱或直接后继来替代被删除的关键字

直接前驱：当前关键字**左侧**指针所指子树中“**最右下**”的元素

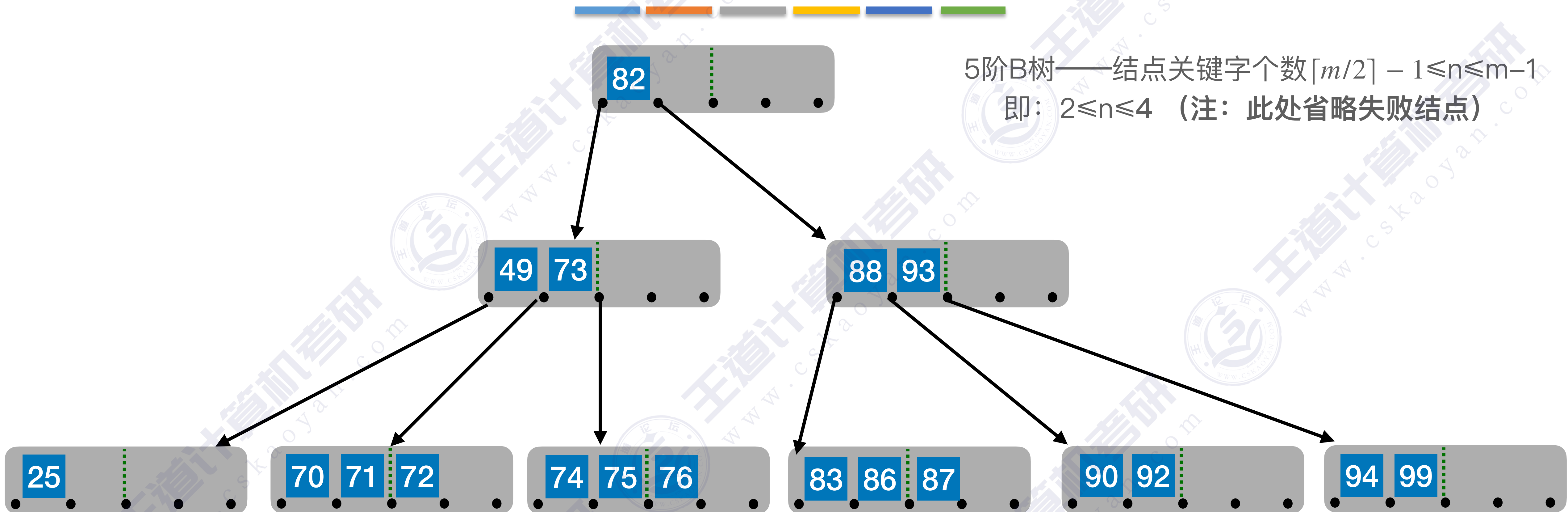
直接后继：当前关键字**右侧**指针所指子树中“**最左下**”的元素

对非终端结点关键字的删除，必然可以转化为对终端结点的删除操作

B树的删除

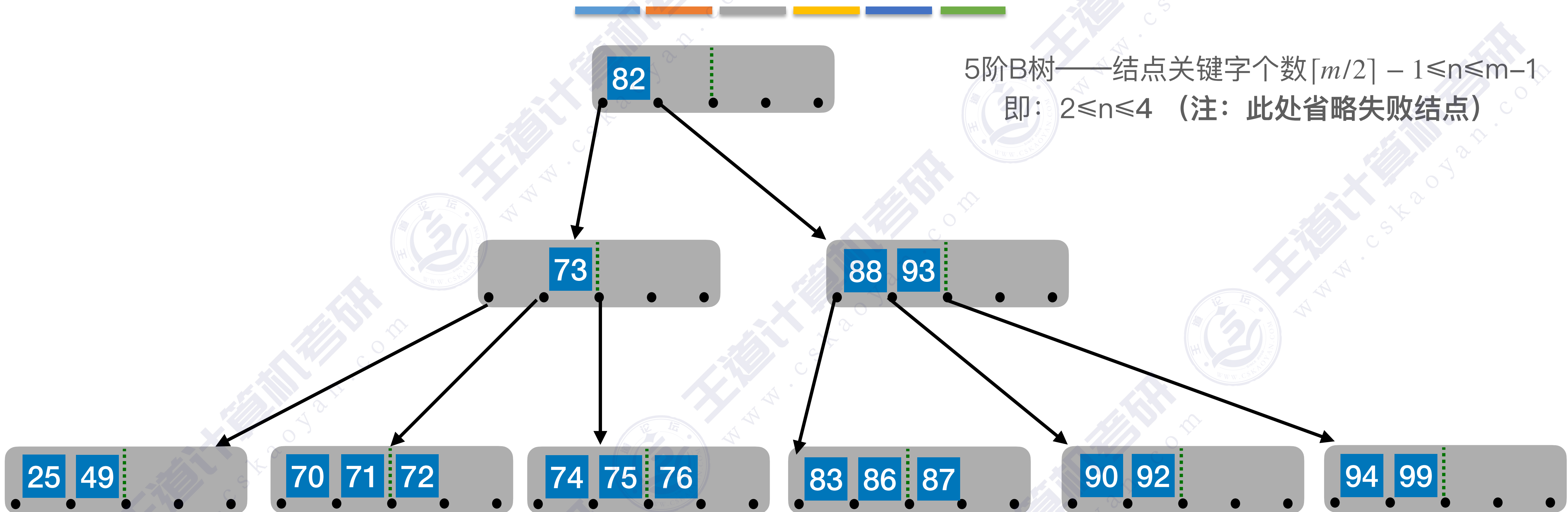


B树的删除



兄弟够借。若被删除关键字所在结点删除前的关键字个数低于下限，且与此结点右（或左）兄弟结点的关键字个数还很宽裕，则需要调整该结点、右（或左）兄弟结点及其双亲结点（父子换位法）

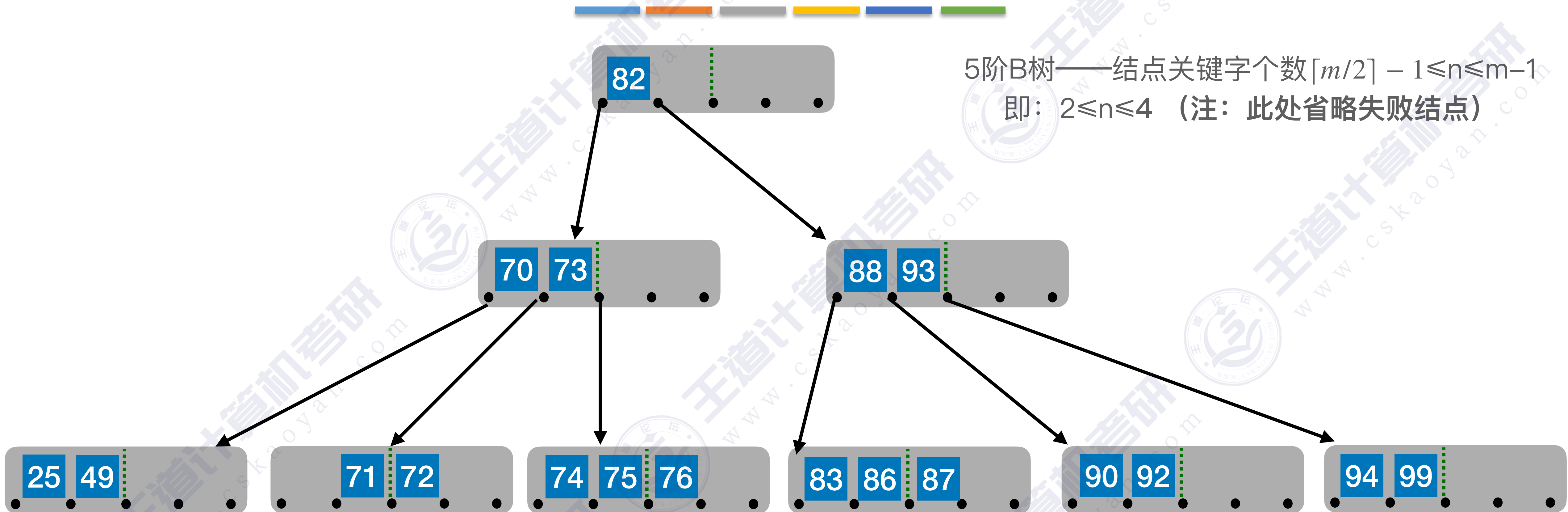
B树的删除



在？借点钱？

兄弟够借。若被删除关键字所在结点删除前的关键字个数低于下限，且与此结点右（或左）兄弟结点的关键字个数还很宽裕，则需要调整该结点、右（或左）兄弟结点及其双亲结点（父子换位法）

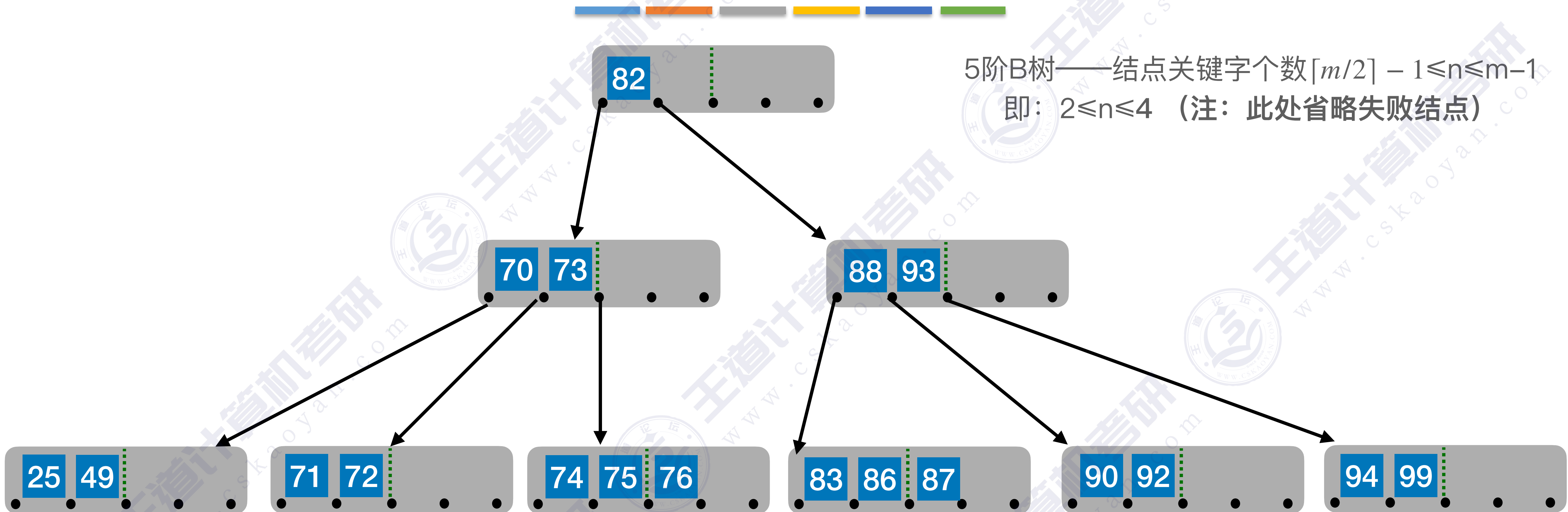
B树的删除



在？借点钱？

兄弟够借。若被删除关键字所在结点删除前的关键字个数低于下限，且与此结点右（或左）兄弟结点的关键字个数还很宽裕，则需要调整该结点、右（或左）兄弟结点及其双亲结点（父子换位法）

B树的删除

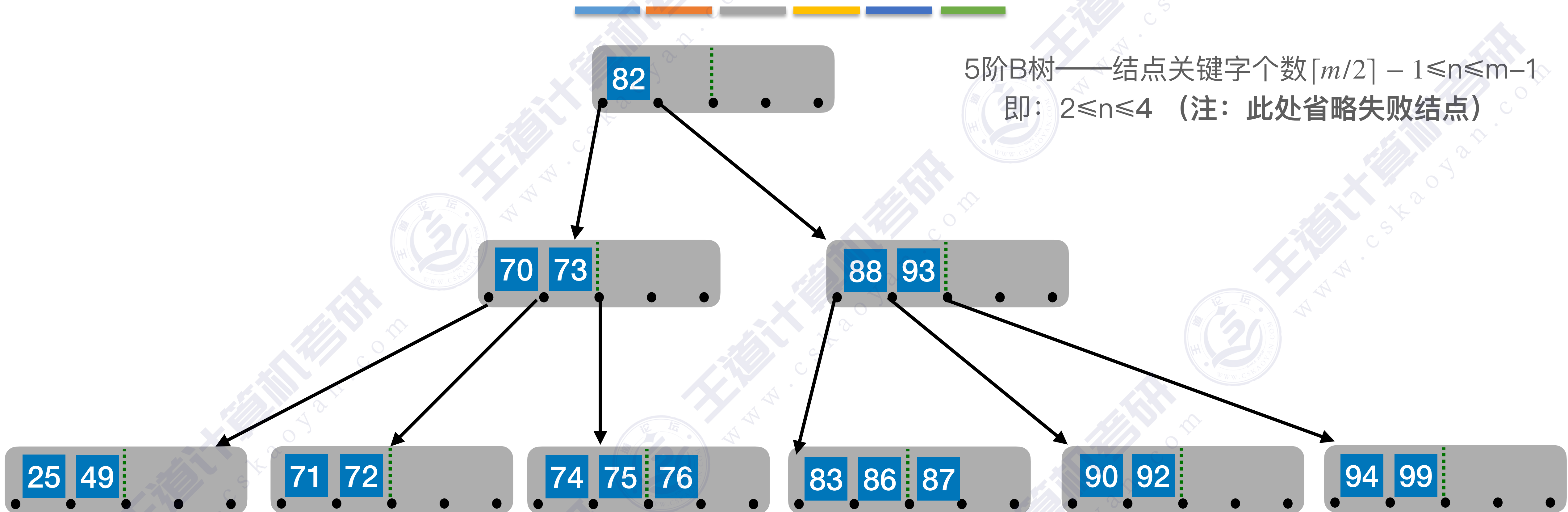


在？借点钱？

兄弟够借。若被删除关键字所在结点删除前的关键字个数低于下限，且与此结点右（或左）兄弟结点的关键字个数还很宽裕，则需要调整该结点、右（或左）兄弟结点及其双亲结点（父子换位法）

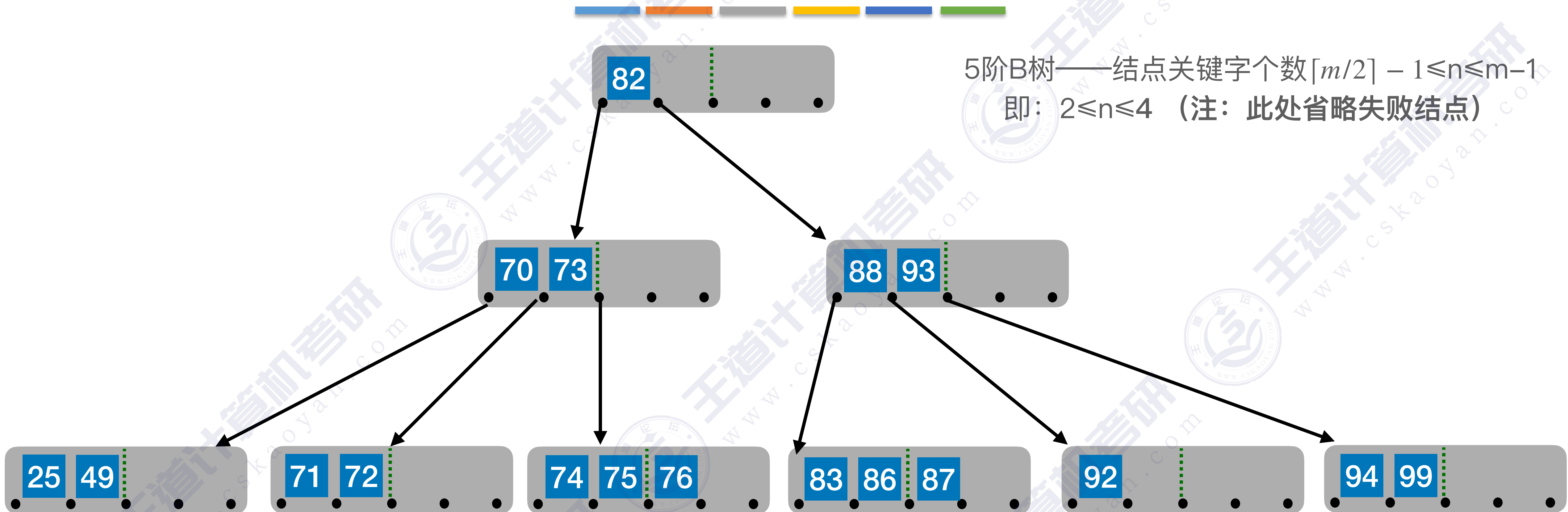
说白了，当**右兄弟很宽裕**时，用当前结点的**后继**、**后继的后继**来填补空缺

B树的删除



兄弟够借。若被删除关键字所在结点删除前的关键字个数低于下限，且与此结点右（或左）兄弟结点的关键字个数还很宽裕，则需要调整该结点、右（或左）兄弟结点及其双亲结点（父子换位法）

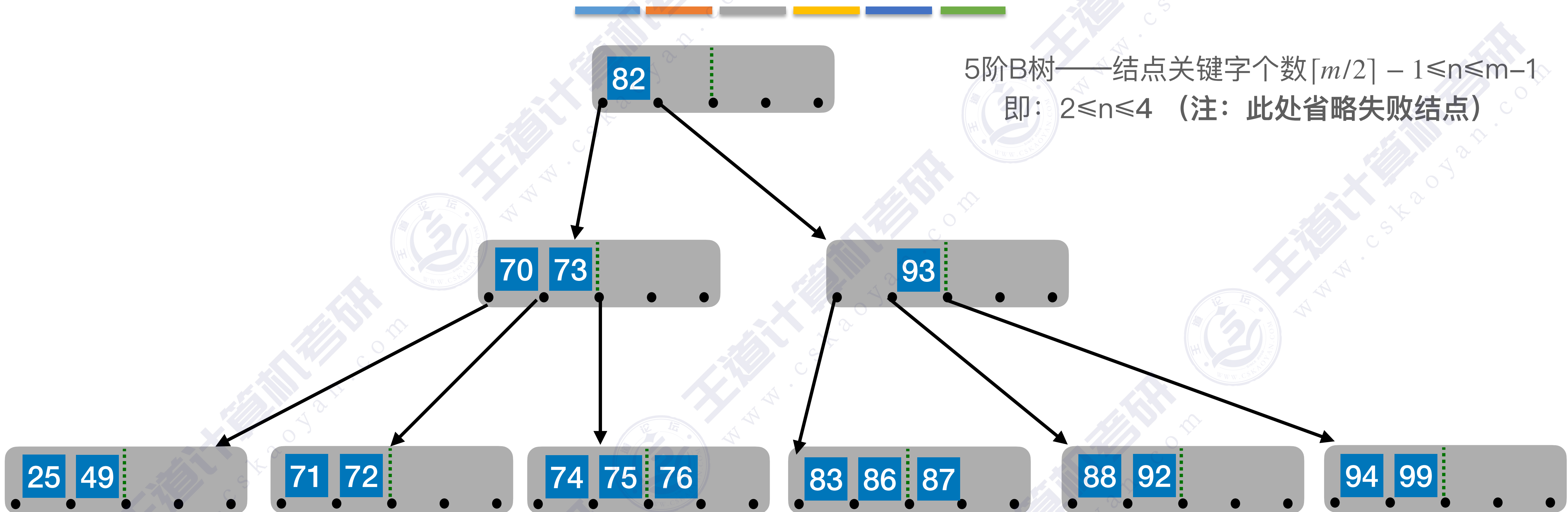
B树的删除



兄弟够借。若被删除关键字所在结点删除前的关键字个数低于下限，且与此结点右（或左）兄弟结点的关键字个数还很宽裕，则需要调整该结点、右（或左）兄弟结点及其双亲结点（父子换位法）

当左兄弟很宽裕时，用当前结点的前驱、前驱的前驱来填补空缺

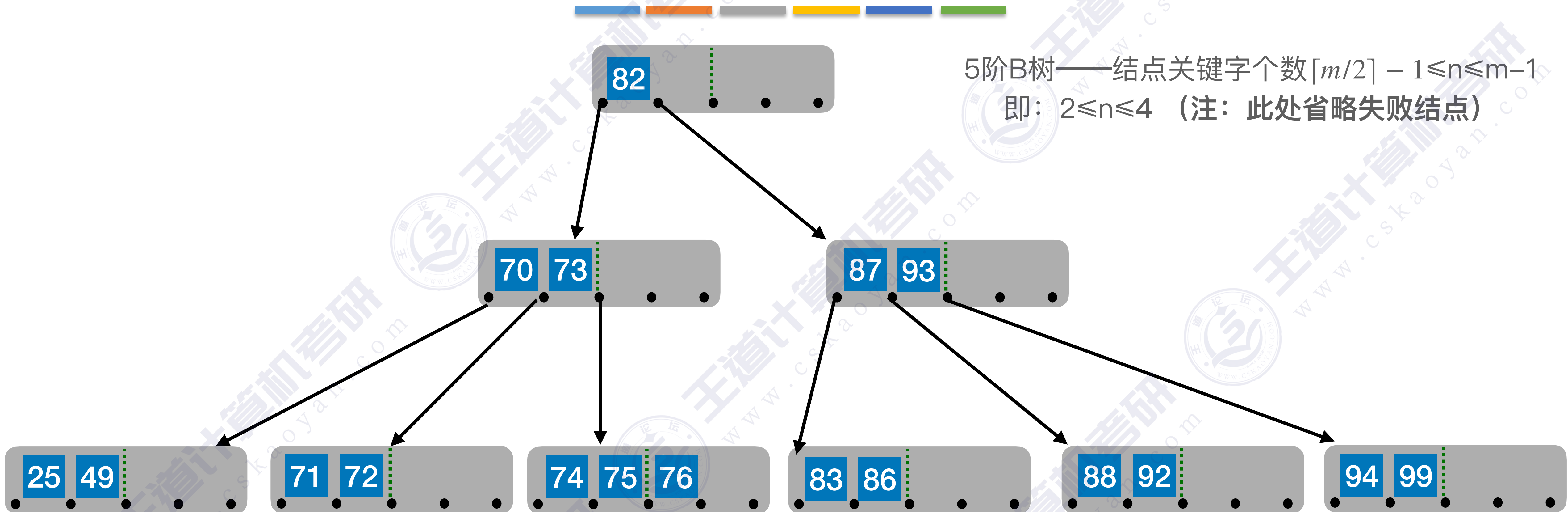
B树的删除



兄弟够借。若被删除关键字所在结点删除前的关键字个数低于下限，且与此结点右（或左）兄弟结点的关键字个数还很宽裕，则需要调整该结点、右（或左）兄弟结点及其双亲结点（父子换位法）

当左兄弟很宽裕时，用当前结点的前驱、前驱的前驱来填补空缺

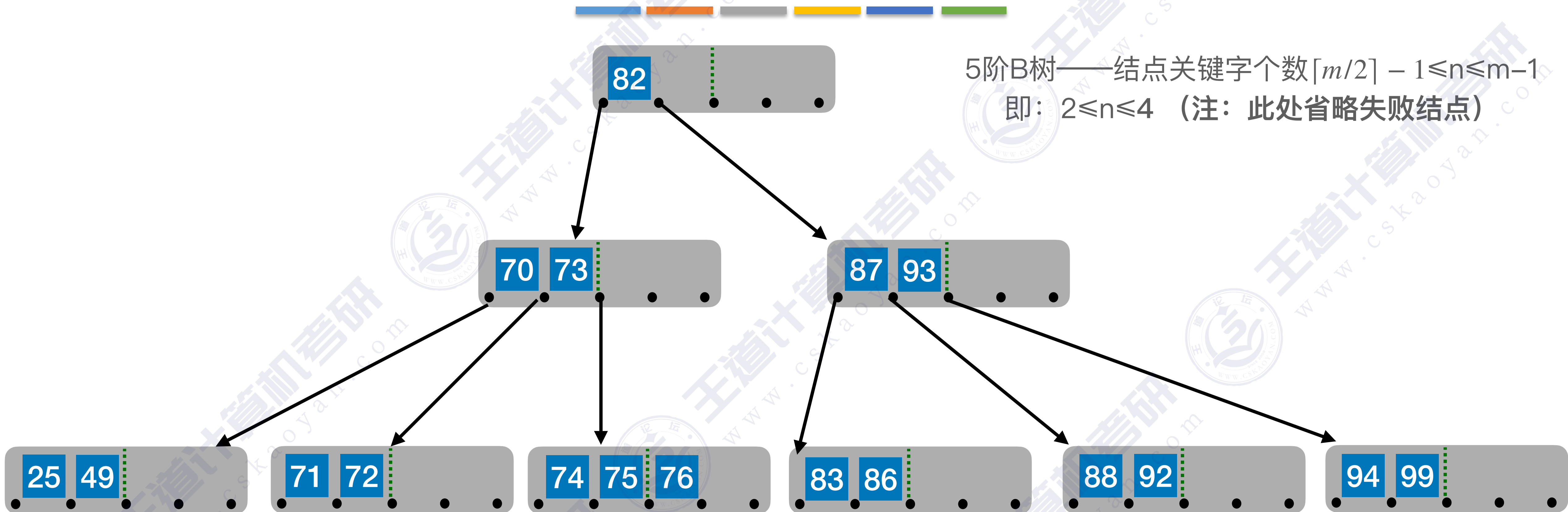
B树的删除



兄弟够借。若被删除关键字所在结点删除前的关键字个数低于下限，且与此结点右（或左）兄弟结点的关键字个数还很宽裕，则需要调整该结点、右（或左）兄弟结点及其双亲结点（父子换位法）

当左兄弟很宽裕时，用当前结点的前驱、前驱的前驱来填补空缺

B树的删除

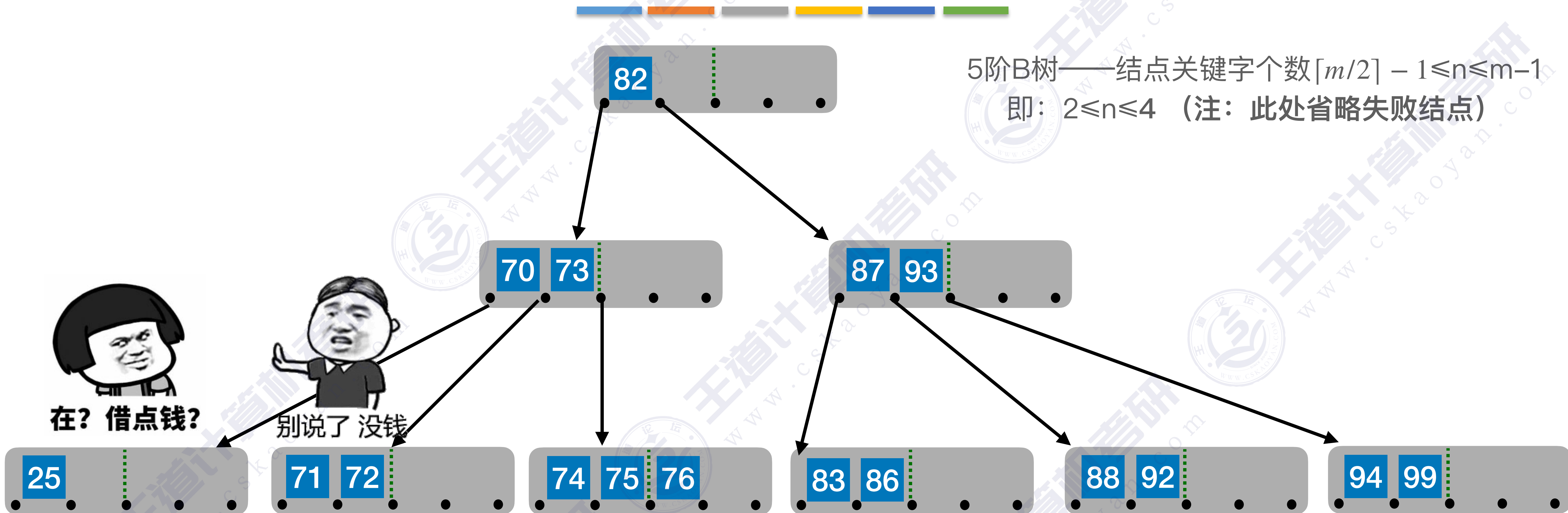


兄弟够借。若被删除关键字所在结点删除前的关键字个数低于下限，且与此结点右（或左）兄弟结点的关键字个数还很宽裕，则需要调整该结点、右（或左）兄弟结点及其双亲结点（父子换位法）

本质：要永远保证 子树0<关键字1<子树1<关键字2<子树2<....

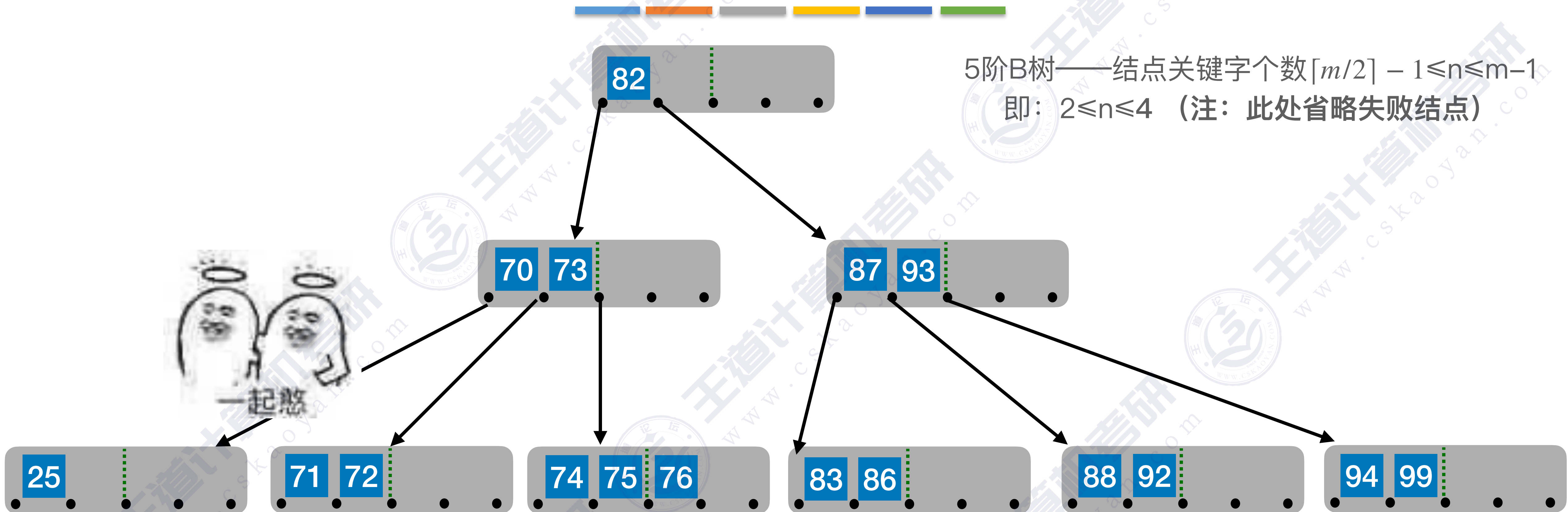
B树的删除

5阶B树——结点关键字个数 $\lceil m/2 \rceil - 1 \leq n \leq m-1$
即： $2 \leq n \leq 4$ （注：此处省略失败结点）



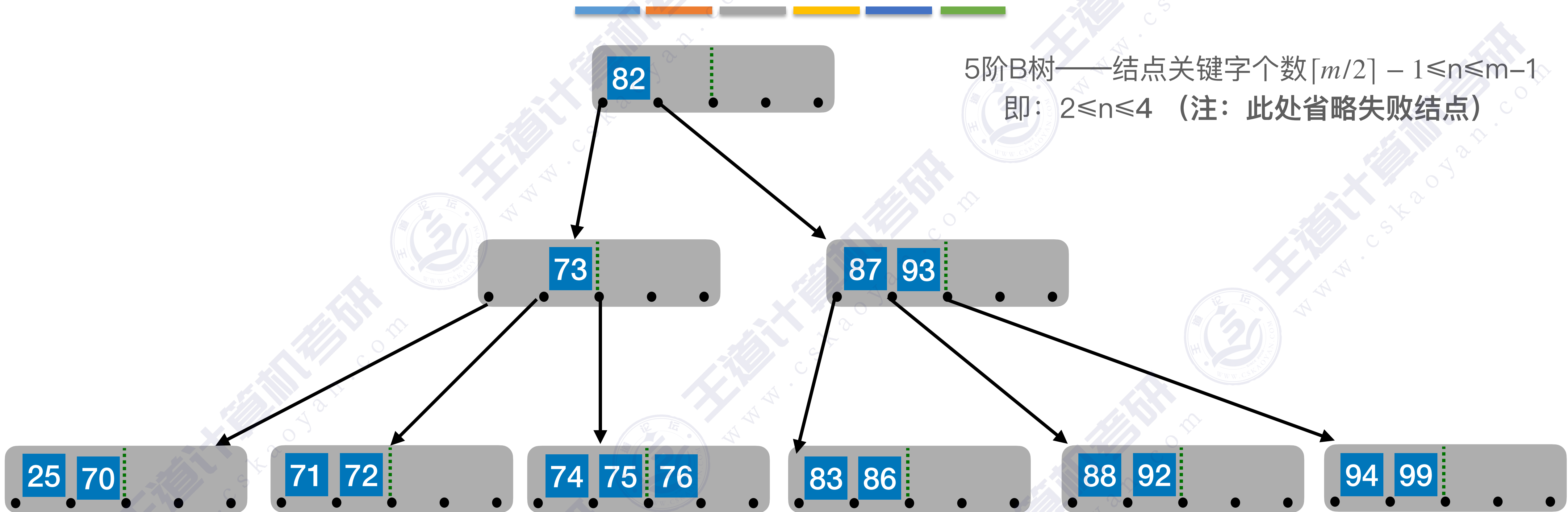
兄弟不够借。若被删除关键字所在结点删除前的关键字个数低于下限，且此时与该结点相邻的左、右兄弟结点的关键字个数均 $= \lceil m/2 \rceil - 1$ ，则将关键字删除后与左（或右）兄弟结点及双亲结点中的关键字进行合并

B树的删除



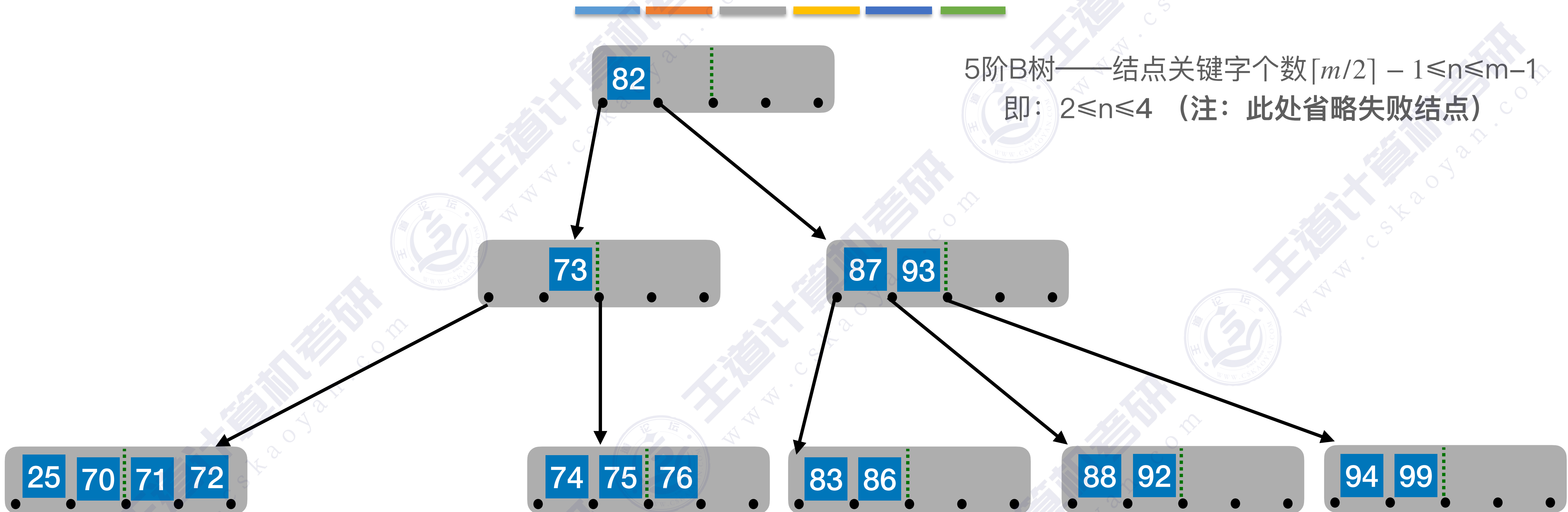
兄弟不够借。若被删除关键字所在结点删除前的关键字个数低于下限，且此时与该结点相邻的左、右兄弟结点的关键字个数均 $= \lceil m/2 \rceil - 1$ ，则将关键字删除后与左（或右）兄弟结点及双亲结点中的关键字进行合并

B树的删除



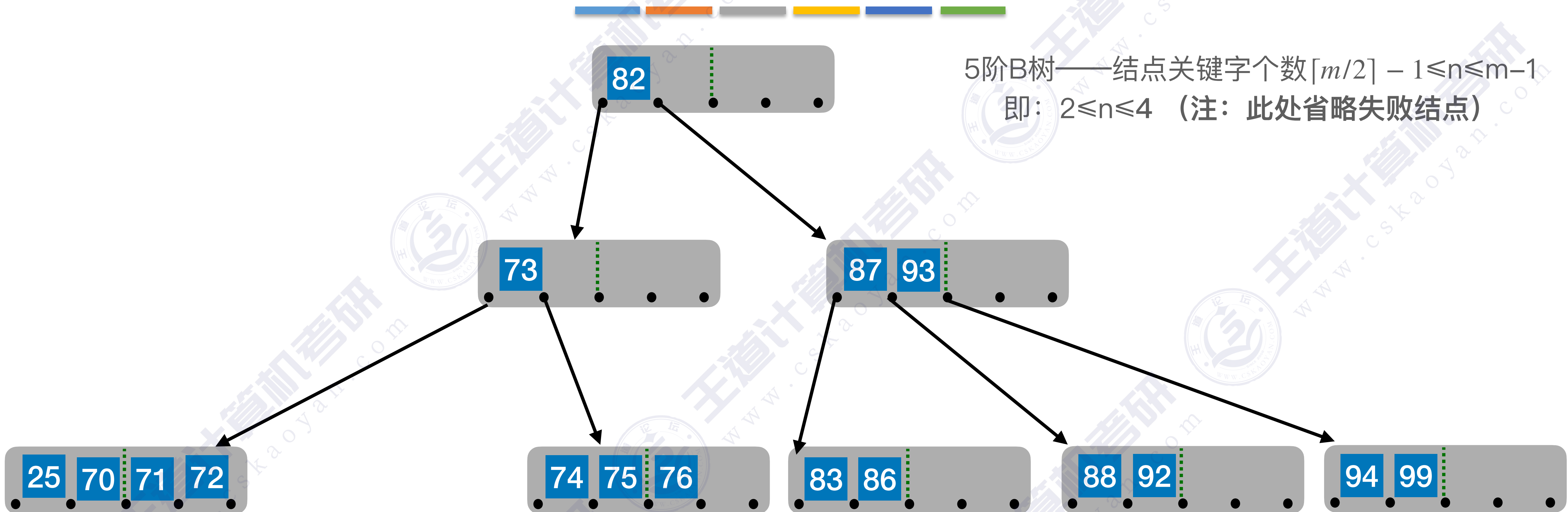
兄弟不够借。若被删除关键字所在结点删除前的关键字个数低于下限，且此时与该结点相邻的左、右兄弟结点的关键字个数均 $= \lceil m/2 \rceil - 1$ ，则将关键字删除后与左（或右）兄弟结点及双亲结点中的关键字进行合并

B树的删除



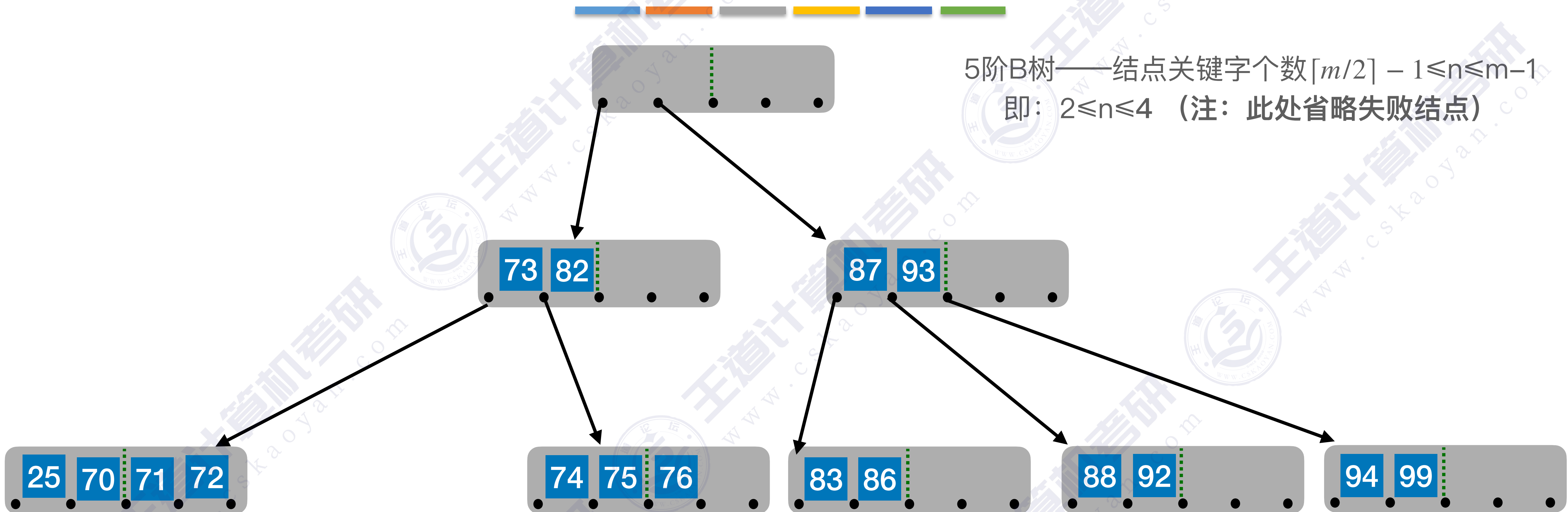
兄弟不够借。若被删除关键字所在结点删除前的关键字个数低于下限，且此时与该结点相邻的左、右兄弟结点的关键字个数均 $= \lceil m/2 \rceil - 1$ ，则将关键字删除后与左（或右）兄弟结点及双亲结点中的关键字进行合并

B树的删除



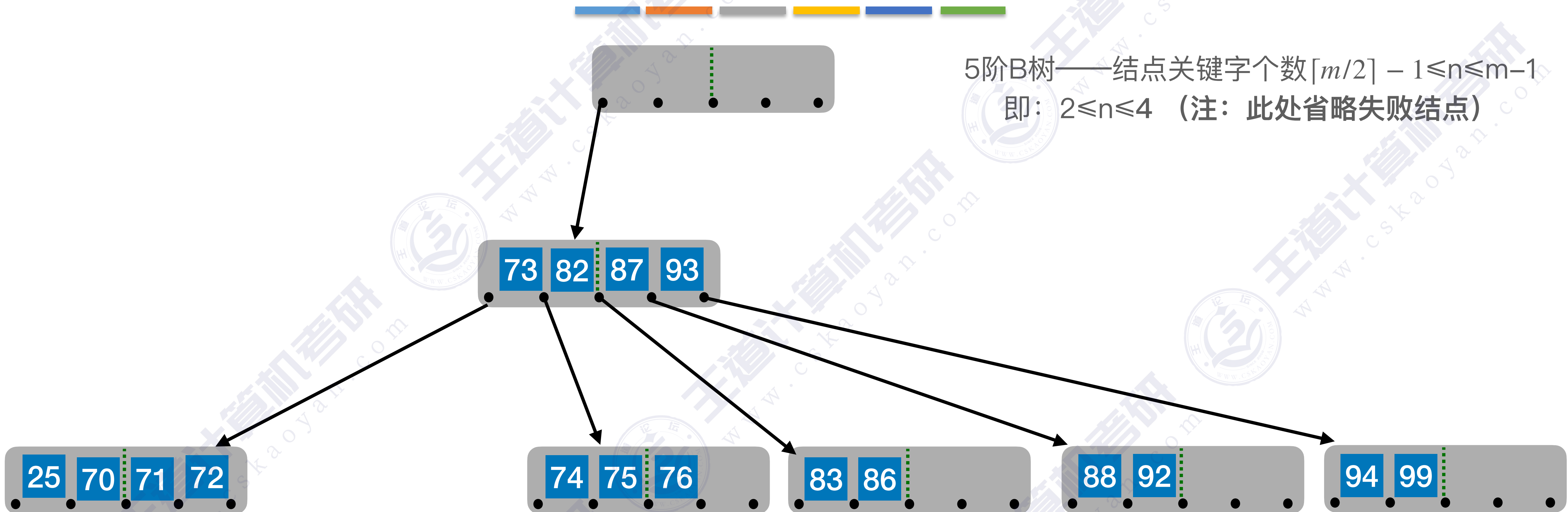
兄弟不够借。若被删除关键字所在结点删除前的关键字个数低于下限，且此时与该结点相邻的左、右兄弟结点的关键字个数均 $= \lceil m/2 \rceil - 1$ ，则将关键字删除后与左（或右）兄弟结点及双亲结点中的关键字进行合并

B树的删除



兄弟不够借。若被删除关键字所在结点删除前的关键字个数低于下限，且此时与该结点相邻的左、右兄弟结点的关键字个数均 $= \lceil m/2 \rceil - 1$ ，则将关键字删除后与左（或右）兄弟结点及双亲结点中的关键字进行合并

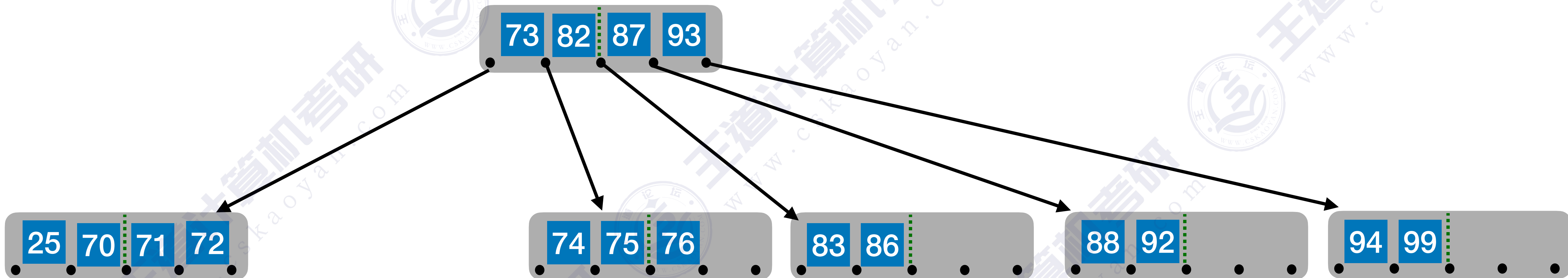
B树的删除



兄弟不够借。若被删除关键字所在结点删除前的关键字个数低于下限，且此时与该结点相邻的左、右兄弟结点的关键字个数均 $= \lceil m/2 \rceil - 1$ ，则将关键字删除后与左（或右）兄弟结点及双亲结点中的关键字进行合并

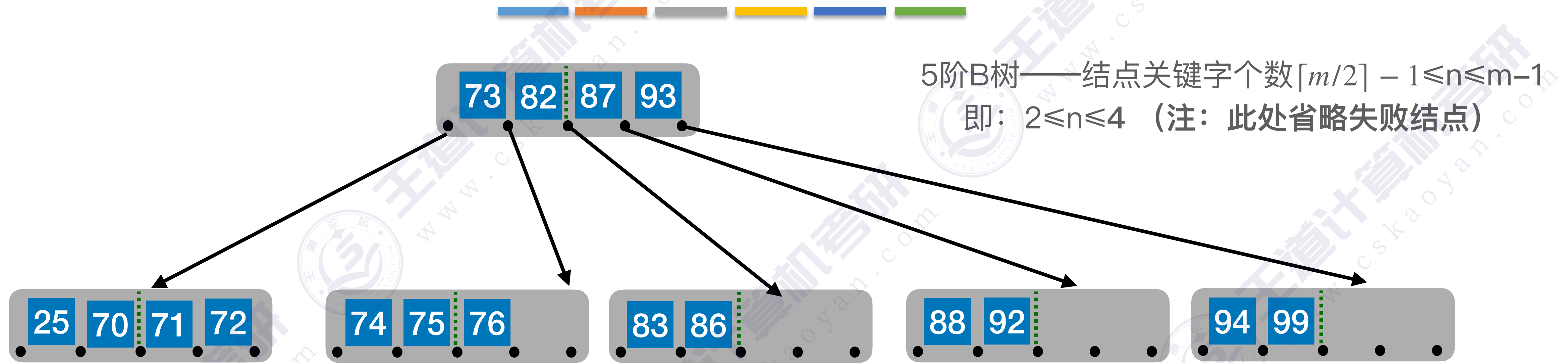
B树的删除

5阶B树——结点关键字个数 $\lceil m/2 \rceil - 1 \leq n \leq m-1$
即： $2 \leq n \leq 4$ （注：此处省略失败结点）



兄弟不够借。若被删除关键字所在结点删除前的关键字个数低于下限，且此时与该结点相邻的左、右兄弟结点的关键字个数均 $= \lceil m/2 \rceil - 1$ ，则将关键字删除后与左（或右）兄弟结点及双亲结点中的关键字进行合并

B树的删除



兄弟不够借。若被删除关键字所在结点删除前的关键字个数低于下限，且此时与该结点相邻的左、右兄弟结点的关键字个数均 $= \lceil m/2 \rceil - 1$ ，则将关键字删除后与左（或右）兄弟结点及双亲结点中的关键字进行合并

在合并过程中，双亲结点中的关键字个数会减1。若其双亲结点是根结点且关键字个数减少至0（根结点关键字个数为1时，有2棵子树），则直接将根结点删除，合并后的新结点成为根；若双亲结点不是根结点，且关键字个数减少到 $\lceil m/2 \rceil - 2$ ，则又要与它自己的兄弟结点进行调整或合并操作，并重复上述步骤，直至符合B树的要求为止。

知识回顾与重要考点

通过“查找”确定插入位置（一定是在终端结点）

若插入后结点关键字个数未超过上限，则无需做其他处理

若插入后关键字个数超过上限，则需要将当前结点的中间元素放到父节点中，当前结点分裂为两个部分；该操作会导致父节点关键字个数+1，若父节点关键字个数也超过了上限，则需要再向上分裂；根节点的分裂会导致B树高度+1。

用其直接前驱或直接后继替代其位置，转化为对“终端结点”的删除

直接前驱：当前关键字左边指针所指子树中“最右下”的元素

直接后继：当前关键字右边指针所指子树中“最左下”的元素

删除后结点关键字个数未低于下限，无需任何处理

右兄弟够借，则用当前结点的后继、后继的后继依次顶替空缺

左兄弟够借，则用当前结点的前驱、前驱的前驱依次顶替空缺

左(右)兄弟都不够借，则需要与父结点内的关键字、左(右)兄弟进行合并。合并后导致父节点关键字数量-1，可能需要继续合并。

核心要求：

①对m阶B树——除根节点外，结点关键字个数 $[m/2] - 1 \leq n \leq m-1$

②子树0<关键字1<子树1<关键字2<子树2<....