B\_TREE\_DELETE(*T, x, k*)

1. **if** *x.leaf*//cases1
2. **for** *i* = 1 to *x.n*
3. **if** *x.keyi* == *k*
4. Delete key *k* from *x*
5. *x.n* = *x.n* -1
6. DISK\_WRITE(*x*)
7. **return**
8. *print“the tree has no target element”*
9. **return**
10. *i* = 1
11. **while** *i* ≤ *x.n* && *x.keyi* < *k*
12. *i* = *i* + 1
13. **if** *x.keyi* == *k*//cases2
14. DISK\_READ(*x.ci*)
15. **if** *x.ci.n* ≥ *t*//cases2a
16. *k`* = B\_TREE\_PRED(T, *x.ci*, *k*)
17. *x.keyi* = *k`*
18. DISK\_WRITE(*x*)
19. B\_TREE\_DELETE(*T*, *x.ci*, *k`*)
20. **return**
21. DISK\_READ(*x.ci*+1)
22. **if** *x.ci+*1*.n* ≥ *t*//case2b
23. *k`* = B\_TREE\_SUCC(*T*, *x.ci*+1, *k*)
24. *x.keyi* = *k`*
25. DISK\_WRITE(*x*)
26. B\_TREE\_DELETE(*T*, *x.ci+1 ,k`*)
27. **return**
28. B\_TREE\_MERGE(*T, x, i, x.ci, x.ci+*1)//
29. B\_TREE\_DELETE(*x.ci, k*)
30. **if** *x.n* == 0
31. *free*(*x*)
32. *root* = *x.ci*
33. **return**
34. DISK\_READ(*x.ci*)
35. **if** *x.ci.n* ≥ *t*
36. B\_TREE\_DELETE(*x.ci*, *k*)
37. **return**
38. *set y and z are new Node point to NULL*
39. **if** *i* ≤ *x.n*
40. DISK\_READ(*x.ci*+1)
41. *z* = *x.ci*+1
42. **if** *z*.*n* ≥ *t*
43. B\_TREE\_LSHIFT(*T, x, i, x.ci, z*)
44. B\_TREE\_DELETE(*x.ci*, *k*)
45. **return**
46. **if** *i* > 1
47. DISK\_READ(*x.ci*-1)
48. *y* = *x.ci*-1
49. **if** *y*.*n* ≥ *t*
50. B\_TREE\_RSHIFT(*T, x, i*-1*, y, x.ci*)
51. B\_TREE\_DELETE(*x.ci*, *k*)
52. **return**
53. **if** *i* == 1
54. B\_TREE\_MERGE(*T, x, i, x.ci, z*)
55. **elseif** *i* == *x.n*+1
56. B\_TREE\_MERGE(*T, x, i*-1*, y, x.ci*)
57. *i* = *i* - 1
58. **else**
59. B\_TREE\_MERGE(*T, x, i, x.ci, z*)
60. B\_TREE\_DELETE(*x.ci*, *k*)
61. **if** *x.n* == 0
62. *free*(*x*)
63. *root* = *x.ci*
64. **return**

**----------------------------------------------------------------**

B\_TREE\_MERGE(*T, x, i, y, z*)

1. *y.keyy.n*+1 = *x.keyi*
2. **for** *j* = 1 to *y.n*
3. *y.keyj*+*t* = *z.keyj*
4. **for** *j* = 1 to *y.n* + 1
5. *y.cj*+*t* = *z.cj*
6. *y.n* = *y.n* + *z.n* + 1
7. **for** *j* = *i +* 1**to** *x.n*
8. *x.cj* = *x.cj*+1
9. *x.n* = *x.n* - 1
10. *free*(*z*)
11. DISK\_WRITE(*x*)
12. DISK\_WRITE(*y*)
13. DISK\_WRITE(*z*)

----------------------------------------------------------------

B\_TREE\_LSHIFT(*T, x, i, l, r*)

1. *l.n* = *l.n* + 1
2. *l.keyl.n* = *x.keyi*
3. *x.keyi* = *r*.*key*1
4. *l.cl.n* +1 = *r*.c1
5. **for** *j* = 1 to *r*.*n -* 1
6. *r*.*keyj* = *r*.*keyj*+1
7. **for** *j* = 1 to *r*.*n*
8. *r*.*cj* = *r*.*cj*+1
9. *r*.*n* = *r*.*n* - 1
10. DISK\_WRITE(*x*)
11. DISK\_WRITE(*l*)
12. DISK\_WRITE(*r*)

----------------------------------------------------------------

B\_TREE\_RSHIFT(*T, x, i, l, r*)

1. *r.n* = *r.n* + 1
2. **for** *j* = *r*.*n* **downto** 2
3. *r*.*keyj* = *r*.*keyj*-1
4. **for** *j* = *r*.*n*+1 **downto** 2
5. *r*.*cj* = *r*.*cj*-1
6. *r.key*1 = *x.keyi*
7. *x.keyx.n*= *l*.*keyl.n*
8. *r*.*c*1 = *l*.*cl.n*+1
9. *l*.*n* = *l*.*n* - 1
10. DISK\_WRITE(*x*)
11. DISK\_WRITE(*l*)
12. DISK\_WRITE(*r*)----------------------------------------------------------------
13. B\_TREE\_PRED(*T, x, k*)
14. **if** *x.leaf*
15. **return** *x.keyx.n*
16. **else**
17. DISK\_READ(*x.cx.n*+1)
18. **return** B\_TREE\_PRED(*T, x.cx.n*+1*, k*)

----------------------------------------------------------------

B\_TREE\_SUCC(*T, x, k*)

1. **if** *x.leaf*
2. **return** *x.key*1
3. **else**
4. DISK\_READ(*x.cx.*1)
5. **return** B\_TREE\_PRED(*T, x.c*1*, k*)