1. The data structure needs a new field to store the current minimum value of the Binary Search Tree. In the program, we’ll call this variable ’min’.
2. The following changes are needed to return the minimum value in the Binary Search Tree:
3. **Algorithm:** BSTDict.min()
4. *node* = root
5. **while** *node* has a left child **do**
6. *node* = node.left
7. **end**
8. *min* = *node*
9. return *min*
10. **Algorithm:** BSTDict.insert(new)
11. *node* = root
12. Boolean *bool* = false;
13. **while** *node* isn't NIL do
14. **if** *node*.value ≤ *new* **then**
15. **if** *node*.left = NIL **then**
16. Add *new* as left child of *node*
17. *node* = *node*.left
18. **end**
19. *node* = *node*.left
20. **else**
21. *bool* = true;
22. **if** *node*.right = NIL **then**
23. Add *new* as right child of *node*
24. *node* = *node*.right
25. **end**
26. *node* = *node*.right
27. **end**
28. **if** *bool* is false **then**
29. Set *min* to be *new*;
30. **end**
31. **end**
32. **Algorithm:** BSTDict.delete(*node*)
33. **if** *node* has two children **then**
34. *swapnode* = right
35. **while** *swapnode* has a left child **do**
36. *swapnode* = *swapnode*.left
37. **end**
38. Swap *node*'s parent and children links with *swapnode*
39. **if** *node* is the BST root **then**
40. Set root to be *swapnode*
41. **end**
42. **end**
43. **if** *node* has no children **then**
44. **if** *node* is the *min* **then**
45. Set *min* to be *node*.parent
46. **if** *node* is the root **then**
47. Set root to be NIL
48. **else**
49. Set *node*.parent's child to be NIL
50. **end**
51. **else**
52. /\* *node* must have one child \*/
53. **if** *node* is the *min* **then**
54. Set *min* to be *node*.right
55. **while** *node*.right has a left child
56. *min* = *node*.left
57. **end**
58. **end**
59. **if** *node* is the root **then**
60. Set root to be *node*'s child
61. **else**
62. Set *node*.parent's child to be *node*'s child
63. **end**
64. Set *node*'s child's parent to be *node*.parent
65. **end**