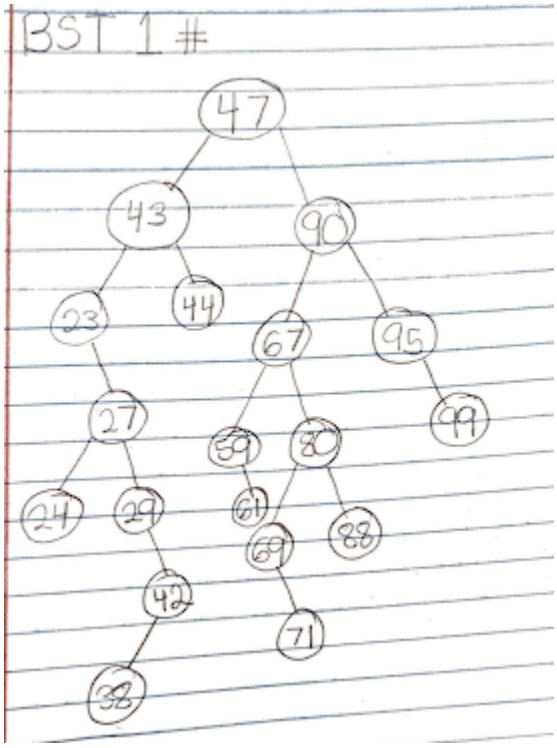
OUTPUT OF CODE:

Inorder Traversal of Binary Search Tree #1:
23 24 27 29 38 42 43 44 47 59 61 67 69 71 80 88 90 95 99
After Deletion:
23 24 29 42 43 47 61 67 69 71 80 90 99

Inorder Traversal of Binary Search Tree #2:
10 15 17 20 22 34 46 49 52 57 59 69 76 86 89 97 99
After Deletion:
17 20 34 46 49 52 57 59 76 86 89 97

SHOWING WORK WITH DRAWINGS AND EXPLANATIONS

BINARY SEARCH TREE #1



OUTPUT AFTER INSERTION:
INORDER TRAVERSAL
23 24 27 29 38 42 43 44 47 59 61 67 69 71 80 88 90 95 99

Explanation of Code

- **Insert(47)**

  Since this is the first insertion 47 is the root node

- **Insert(43)**

  Since 43 < 47 we check if it has any left child and insert it making it the left child of 47

- **Insert(23)**

  Since 23 < 47 we move to the left child, and
  since 23 < 43 we check if it has a left child and since it doesn't it becomes the left child of 43

- **Insert(90)**

  Since 90 > 47 we check if it has a right child and make it the the new right child

- **Insert(95)**

  Since 95 > 47 we check the right child, and since 95 > 90 we make it the new right child

- **Insert(27)**

  Since 27 < 47 we check the left child
  since 27 < 43
  since 27 > 23 we make it the right child of 23.

- **Insert(67)**

  Since 67 > 47 we check the right child, and since 67 < 90 we make it the 67 the left child of 90

- **Insert(80)**

  Since 80 > 47, we move to the right child (90).
  Since 80 < 90, we move to the left child (67).
  Since 80 > 67, we check if 67 has a right child. It doesn't, so 80 becomes the right child of 67.

- **Insert(88)**

  Since 88 > 47, we move to the right child (90).
  Since 88 < 90, we move to the left child (67).
  Since 88 > 67, we move to the right child (80).
  Since 88 > 80, we check if 80 has a right child. It doesn't, so 88 becomes the right child of 80.

- **Insert(29)**

  Since 29 < 47, we move to the left child (43).

Since 29 < 43, we move to the left child (23).
Since 29 > 23, we move to the right child (27).
Since 29 > 27, we check if 27 has a right child. It doesn't, so 29 becomes the right child of 27.

- **Insert(59)**

Since 59 > 47, we move to the right child (90).
Since 59 < 90, we move to the left child (67).
Since 59 < 67, we check if 67 has a left child. It doesn't, so 59 becomes the left child of 67.

- **Insert(24)**

Since 24 < 47, we move to the left child (43).
Since 24 < 43, we move to the left child (23).
Since 24 > 23, we move to the right child (27).
Since 24 < 27, we check if 27 has a left child. It doesn't, so 24 becomes the left child of 27.

- **Insert(67)**

Since 69 > 47, we move to the right child (90).
Since 69 < 90, we move to the left child (67).
Since 69 > 67, we move to the right child (80).
Since 69 < 80, we check if 80 has a left child. It doesn't, so 69 becomes the left child of 80.

- **Insert(44)**

Since 44 < 47, we move to the left child (43).
Since 44 > 43, we check if 43 has a right child. It doesn't, so 44 becomes the right child of 43.

- **Insert(71)**

Since 71 > 47, we move to the right child (90).
Since 71 < 90, we move to the left child (67).
Since 71 > 67, we move to the right child (80).
Since 71 < 80, we move to the left child (69).
Since 71 > 69, we check if 69 has a right child. It doesn't, so 71 becomes the right child of 69.

- **Insert(61)**

Since 61 > 47, we move to the right child (90).
Since 61 < 90, we move to the left child (67).
Since 61 < 67, we move to the left child (59).
Since 61 > 59, we check if 59 has a right child. It doesn't, so 61 becomes the right child of 59.

- **Insert(99)**

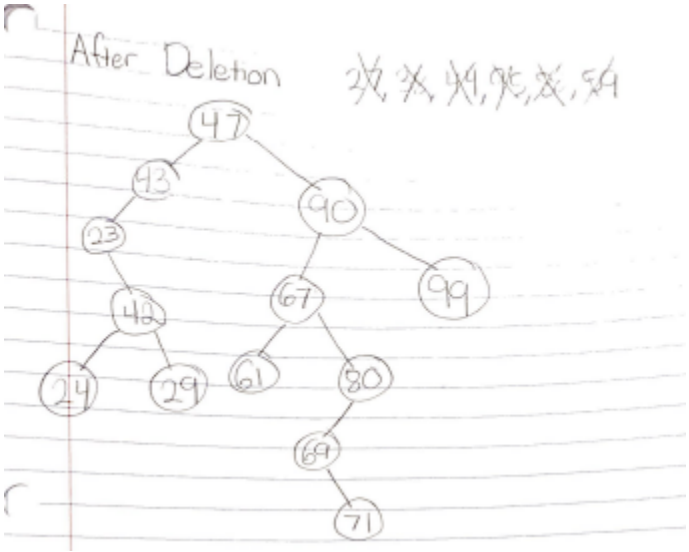
Since 99 > 47, we move to the right child (90).
Since 99 > 90, we move to the right child (95).
Since 99 > 95, we check if 95 has a right child. It doesn't, so 99 becomes the right child of 95.

- **Insert(42)**

Since 42 < 47, we move to the left child (43).
Since 42 < 43, we move to the left child (23).
Since 42 > 23, we move to the right child (27).
Since 42 > 27, we move to the right child (29).
Since 42 > 29, we check if 29 has a right child. It doesn't, so 42 becomes the right child of 29.

- **Insert(38)**

Since 38 < 47, we move to the left child (43).
Since 38 < 43, we move to the left child (23).
Since 38 > 23, we move to the right child (27).
Since 38 > 27, we move to the right child (29).
Since 38 < 42, we check if 29 has a left child. It doesn't, so 38 becomes the left child of 29.



OUTPUT AFTER DELETION:
INORDER TRAVERSAL
23 24 29 42 43 47 61 67 69 71 80 90 99

Explanation of Code

- **Delete(27)**
  Since 27 has child nodes, we must take its greatest successor by going down its right subtree to get 38 and replace it with 27 then delete the 27 node

- **Delete(38)**
  Since 38 is a leaf node we can just delete it

- **Delete(44)**
  Since 44 is a leaf node we can just delete it

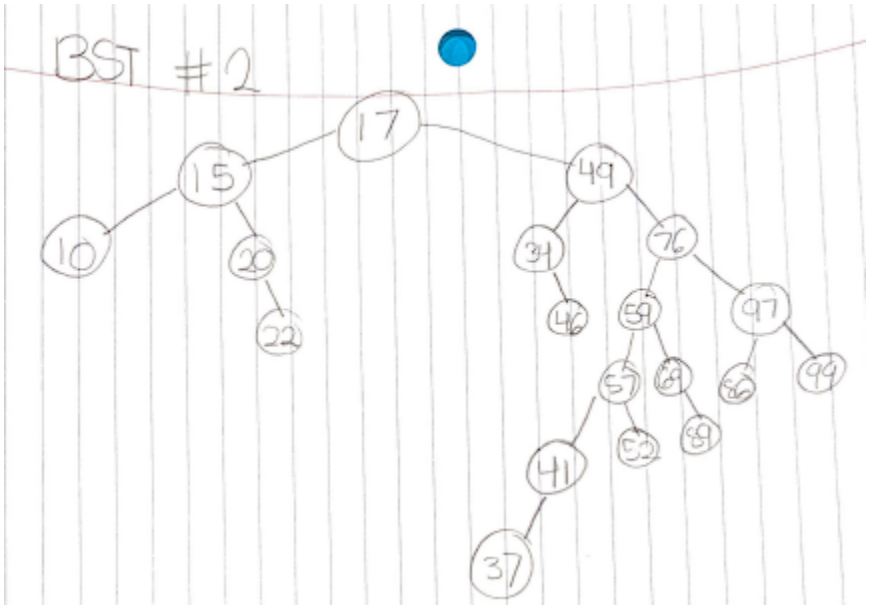
- **Delete(95)**
  Since 95 has a child we just replace it with its childs node 99 and
  delete 95

- **Delete(88)**
  Since 88 is a leaf node we can just delete it

- **Delete(59)**
  Since 59 has a child we can replace it with its child node 61 and delete 59 node

---

BINARY SEARCH TREE #2



OUTPUT AFTER INSERTION:
INORDER TRAVERSAL
10 15 17 20 22 34 46 49 52 57 59 69 76 86 89 97 99

Explanation of Code

- **Insert(17)**
  Since this is the first insertion, 17 becomes the root node.

- **Insert(15)**
  Since 15 < 17, we check if 17 has a left child. It doesn't, so 15 becomes the left child of 17.

- **Insert(49)**
  Since 49 > 17, we check if 17 has a right child. It doesn't, so 49 becomes the right child of 17.

- **Insert(34)**
  Since 34 > 17, we move to the right child (49).
  Since 34 < 49, we check if 49 has a left child. It doesn't, so 34 becomes the left child of 49.

- **Insert(76)**
  Since 76 > 17, we move to the right child (49).
  Since 76 > 49, we check if 49 has a right child. It doesn't, so 76 becomes the right child of 49.

- **Insert(59)**
  Since 59 > 17, we move to the right child (49).
  Since 59 > 49, we move to the right child (76).
  Since 59 < 76, we check if 76 has a left child. It doesn't, so 59 becomes the left child of 76.

- **Insert(97)**
  Since 97 > 17, we move to the right child (49).
  Since 97 > 49, we move to the right child (76).
  Since 97 > 76, we check if 76 has a right child. It doesn't, so 97 becomes the right child of 76.

- **Insert(69)**
  Since 69 > 17, we move to the right child (49).
  Since 69 > 49, we move to the right child (76).
  Since 69 < 76, we move to the left child (59).
  Since 69 > 59, we check if 59 has a right child. It doesn't, so 69 becomes the right child of 59.

- **Insert(46)**
  Since 46 > 17, we move to the right child (49).
  Since 46 < 49, we move to the left child (34).
  Since 46 > 34, we check if 34 has a right child. It doesn't, so 46 becomes the right child of 34.

- **Insert(86)**
  Since 86 > 17, we move to the right child (49).
  Since 86 > 49, we move to the right child (76).
  Since 86 > 76, we move to the right child (97).
  Since 86 < 97, we check if 97 has a left child. It doesn't, so 86 becomes the left child of 97.

- **Insert(20)**
  Since 20 > 17, we move to the right child (49).
  Since 20 < 49, we move to the left child (34).
  Since 20 < 34, we check if 34 has a left child. It doesn't, so 20 becomes the left child of 34.

- **Insert(99)**
  Since 99 > 17, we move to the right child (49).
  Since 99 > 49, we move to the right child (76).
  Since 99 > 76, we move to the right child (97).
  Since 99 > 97, we check if 97 has a right child. It doesn't, so 99 becomes the right child of 97.

- **Insert(22)**
  Since 22 > 17, we move to the right child (49).
  Since 22 < 49, we move to the left child (34).
  Since 22 < 34, we move to the left child (20).
  Since 22 > 20, we check if 20 has a right child. It doesn't, so 22 becomes the right child of 20.

- **Insert(52)**
  Since 52 > 17, we move to the right child (49).
  Since 52 > 49, we move to the right child (76).
  Since 52 < 76, we move to the left child (59).
  Since 52 < 59, we check if 59 has a left child. It doesn't, so 52 becomes the left child of 59.

- **Insert(89)**
  Since 89 > 17, we move to the right child (49).
  Since 89 > 49, we move to the right child (76).
  Since 89 > 76, we move to the right child (97).
  Since 89 < 97, we move to the left child (86).
  Since 89 > 86, we check if 86 has a right child. It doesn't, so 89 becomes the right child of 86.

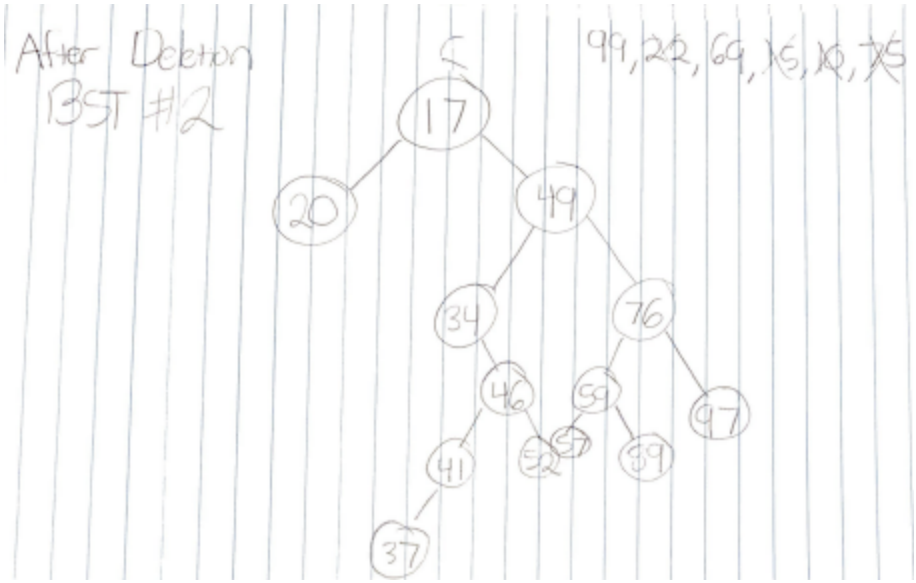
- **Insert(57)**
  Since 57 > 17, we move to the right child (49).
  Since 57 > 49, we move to the right child (76).
  Since 57 < 76, we move to the left child (59).
  Since 57 < 59, we move to the left child (52).
  Since 57 > 52, we check if 52 has a right child. It doesn't, so 57 becomes the right child of 52.

- **Insert(10)**
  Since 10 < 17, we check if 17 has a left child. It does, so we move to the left child (15).
  Since 10 < 15, we check if 15 has a left child. It doesn't, so 10 becomes the left child of 15.

- **Insert(41)**
  Since 41 > 17, we move to the right child (49).
  Since 41 < 49, we move to the left child (34).
  Since 41 > 34, we move to the right child (46).
  Since 41 < 46, we check if 46 has a left child. It doesn't, so 41 becomes the left child of 46.

- **Insert(75)**
  Since 75 > 17, we move to the right child (49).
  Since 75 > 49, we move to the right child (76).
  Since 75 < 76, we check if 76 has a left child. It doesn't, so 75 becomes the left child of 76.

- **Insert(37)**
  Since 37 > 17, we move to the right child (49).
  Since 37 < 49, we move to the left child (34).
  Since 37 > 34, we check if 34 has a right child. It doesn't, so 37 becomes the right child of 34.



OUTPUT AFTER DELETION:
INORDER TRAVERSAL
17 20 34 46 49 52 57 59 76 86 89 97

Explanation of Code

- **Delete(99)**
  Since 99 is a leaf node we can just delete that node

- **Delete(22)**
  Since 22 is a leaf node we can just delete the node

- **Delete(69)**
  Since 69 has a child we replace the value of 69 with its child 89, then we delete the 69 node.

- **Delete(15)**
  Since 15 has a child we must replace it with its greatest successor which is 20

- **Delete(10)**
  Since 10 is a leaf node we can just delete that node

- **Delete(75)**
  Since 75 is a leaf node we can just delete it