

Our Solution(s)

Run Code

Your Solutions

Run Code

Solution 1

Solution 2

```
1 // Copyright © 2020 AlgoExpert, LLC. All rights reserved.
2
3 class Program {
4     public static class LinkedList {
5         int value;
6         LinkedList next;
7
8         LinkedList(int value) {
9             this.value = value;
10            this.next = null;
11        }
12    }
13
14    // O(n + m) time | O(1) space - where n is the number of nodes in the first linked list and m is the number of nodes in the second linked list
15    // Linked List and m is the number of nodes in the second linked list
16    public static LinkedList mergeLinkedLists(LinkedList headOne, LinkedList headTwo) {
17        LinkedList p1 = headOne;
18        LinkedList p1Prev = null;
19        LinkedList p2 = headTwo;
20        while (p1 != null && p2 != null) {
21            if (p1.value < p2.value) {
22                p1Prev = p1;
23                p1 = p1.next;
24            } else {
25                if (p1Prev != null) p1Prev.next = p2;
26                p1Prev = p2;
27                p2 = p2.next;
28                p1Prev.next = p1;
29            }
30        }
31        if (p1 == null) p1Prev.next = p2;
32        return headOne.value < headTwo.value ? headOne : headTwo;
33    }
}
```

Solution 1

Solution 2

Solution 3

```
1 import java.util.*;
2
3 class Program {
4     // This is an input class. Do not edit.
5     public static class LinkedList {
6         int value;
7         LinkedList next;
8
9         LinkedList(int value) {
10            this.value = value;
11            this.next = null;
12        }
13    }
14
15    public static LinkedList mergeLinkedLists(LinkedList headOne, LinkedList headTwo) {
16        // Write your code here.
17        return null;
18    }
19 }
20
```

Our Tests

Custom Output

Submit Code

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27

28

29

30

31

32

33

34

35

36

37

38

39

40

41

42

43

44

45

46

47

48

49

50

51

52

53

54

55

56

57

58

59

60

61

62

63

64

65

66

67

68

69

70

71

72

73

74

75

76

77

78

79

80

81

82

83

84

85

86

87

88

89

90

91

92

93

94

95

96

97

98

99

100

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27

28

29

30

31

32

33

34

35

36

37

38

39

40

41

42

43

44

45

46

47

48

49

50

51

52

53

54

55

56

57

58

59

60

61

62

63

64

65

66

67

68

69

70

71

72

73

74

75

76

77

78

79

80

81

82

83

84

85

86

87

88

89

90

91

92

93

94

95

96

97

98

99

100

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27

28

29

30

31

32

33

34

35

36

37

38

39

40

41

42

43

44

45

46

47

48

49

50

51

52

53

54

55

56

57

58

59

60

61

62

63

64

65

66

67

68

69

70

71

72

73

74

75

76

77

78

79

80

81

82

83

84

85

86

87

88

89

90

91

92

93

94

95

96

97

98

99

100

```

10         correct = correct + 1
11     }
12     for int index = 0; index < correct; index++
13     {
14         correct = correct + 1;
15     }
16     }
17     }
18     }
19     }
20     }
21     }
22     }
23     }
24     }
25     }
26     }
27     }
28     }
29     }
30     }
31     }
32     }
33     }
34     }
35     }
36     }
37     }
38     }
39     }
40     }
41     }
42     }
43     }
44     }
45     }
46     }
47     }
48     }
49     }
50     }
51     }
52     }
53     }
54     }
55     }
56     }
57     }
58     }
59     }
60     }
61     }
62     }
63     }
64     }
65     }
66     }
67     }
68     }
69     }
70     }
71     }
72     }
73     }
74     }
75     }
76     }
77     }
78     }
79     }
80     }
81     }
82     }
83     }
84     }
85     }
86     }
87     }
88     }
89     }
90     }
91     }
92     }
93     }
94     }
95     }
96     }
97     }
98     }
99     }
100    }

```

Run or submit code when you're ready.