Write a function/method called **evenRev()** that takes the head of a singly linked list in its parameter. The method creates a new LinkedList consisting of only the even elements from the original linked list but in reverse order. Finally, the method returns the head of the newly created linked list. If there are no even elements in the Linked List then return None/null.

**Note:** You can assume that a Node class is already given. No need to write it

| **Sample Given Linked List** | **Sample returned Linked List** |
| --- | --- |
| **8 -> 1 -> 50 -> 45 -> 10 -> 13 -> 3 -> 2** | **2 -> 10 -> 50 -> 8** |
| **11 -> 25 -> 33 -> 41** | **None (python) / null (java)** |
| **Python Driver Code** | **Java Driver Code** |
| newH = oddRev( head )  printLL( newH )  # assumer printLL() is function that prints  # a Linked List. You don’t need to implement it | Node newH = oddRev( head );  printLL( newH );  // assumer printLL() is a static method that  // prints a Linked List. You don’t need to  // implement it |