**Assignment 1: Implement Various Operations of Singly Linked Lists**

**Data Structures & Algorithms**

**Section:** BSCS-10-A&B **Session:** Fall-2021

**Instructor:** Dr. Yasir Faheem **Mapped to**: CLO-1

**start**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 |  |  | 2 |  |  | 3 |  |  | 4 |  |  | 5 |  |  | 6 |  |  | 7 | / |

1. Implement a function that prints all nodes of a linked list in the reverse order. A call to the above function should print 7, 6, 5, 4, 3, 2, 1. Note that you are not allowed to copy the contents of the list in any other structure.
2. Implement a function which reverses the order of a linked list. If the original list is 1-> 2->3->4->5 -> 6, the updated list should be 6 -> 5 -> 4 -> 3 -> 2 ->1. Note that this function should only change the pointer field in all nodes of a list; you are not allowed to modify data field of any node.
3. Implement a function that deletes all those nodes from a linked list which have **odd numbered** **value** in their data part. For instance, if the original list is 1-> 2->3->5->6 -> 7, then, the updated list should be 2 -> 6.

**Note:** Include all checks for various input cases like: a) all elements are odd numbers 1->3->5, b) only the last element is odd numbered 2->4->5, c) the list has only value which is odd e.g. 1, d) both first and last values are odd 1->2->4->5 etc.

1. Write code of a function which rearranges the linked list by separately grouping nodes having **even numbered** and **odd numbered value** in their data part. If the input list is 1-> 2->3->4->5 -> 6, the updated list may look like 1-> 3 -> 5->2-> 4-> 6 or 2-> 4-> 6 -> 1-> 3 -> 5.
2. Implement a function that rearranges a linked list by separating the **odd positioned** nodes and even **positioned nodes**. Moreover, it should also reverse the sub-list of the even positioned nodes and then append it at the end of odd positioned nodes sub-list. If the original list is 2->4->5->7->8->9 , the updated list should be 2->5->8 ->9->7->4.

**Note:** Include checks for all possible cases; the length of a list may be any number n >=0.

1. Implement a function which takes two values as input from the user and searches them in the list. If both the values are found, your task is to swap both the nodes in which these values are found. Note, that you are not supposed to swap values. You are supposed to change next pointer fields in the concerned nodes. Your function should handle all possible cases:
   * The two nodes are not adjacent and none of them is the first or the last node.
   * The two nodes are not adjacent to each other and one of them is the first/last node.
   * Both nodes are adjacent and none of them is the first or the last one.
   * Both nodes are adjacent and one of them is the first node.
   * Both are adjacent, one is first and the other is last. This is the case in which the length of a list is two.

**Deliverables:**

* You should include screenshots of the outputs of all the implemented functions in this file.
* Moreover, you should upload your source code in a separate .cpp file.