

Problem 1:

Approach:

- Took user input for the size of linked list, and added the node values entered by the user.
- Declared two pointers, initially pointing to the head of the node.
- Used two while loops to compare the node values and remove the duplicate ones.
- Displayed the updated linked list with no duplicate values.

Why I used mutex in the addNode function?

If two or more users want to add a new node, then the first user to add a node will set a lock, i.e. it will not allow any other user to add a node, till the first user finishes adding the nodes. After which it will unlock the mutex.

Solution:

Open "Solution/Problem_1/bin/Debug/LinkedList.exe" to see the executable for Problem 1.

Code files: main.cpp, LinkedList.cpp, LinkedList.h

Output:

```
Enter the size of the linked list: 10
Enter the value of node 1: 3
Enter the value of node 2: 5
Enter the value of node 3: 6
Enter the value of node 4: 3
Enter the value of node 5: 5
Enter the value of node 6: 5
Enter the value of node 7: 5
Enter the value of node 8: 3
Enter the value of node 9: 3
Enter the value of node 10: 6

Node 1->3
Node 2->5
Node 3->6
Node 4->3
Node 5->5
Node 6->5
Node 7->5
Node 8->3
Node 9->3
Node 10->6

After removing duplicates:
Node 1->3
Node 2->5
Node 3->6

Enter '1' to continue '0' to exit:
```

Problem 2:

Approach:

- Took user input in the form of character array. (Assumption: Space is not considered as a character in this solution)
- Added an option where the user can select the setting to even or odd parity bit.
- Converted each character to its ASCII value.
- Counted the number of ones' present in the binary value of each of them and simultaneously added the parity bit as required, which makes it 8-bits.
- Divided the 8-bit value into two equal parts, and found the hexadecimal value for each part by converting them to integers.
- If the integer is between 0-9, added 48 to them, as the ASCII value is 48-57 respectively, and if the integer is between 10-15, added 55 to them, as the ASCII value is 65-70 respectively.
- Displayed the result in the form of characters of their respective ASCII value.

Solution:

Open "Solution/Problem_2/bin/Debug/two.exe" to see the executable for Problem 2.

Code files: main.cpp

Output:

For odd parity:

```
Enter the stream of characters: ABcde123
Number of characters: 8
Enter '0' for even parity or '1' for odd parity: 1

A | Integer Value:65 | Number of ones: 2 | Binary Value after setting parity: 11000001
B | Integer Value:66 | Number of ones: 2 | Binary Value after setting parity: 11000010
c | Integer Value:99 | Number of ones: 4 | Binary Value after setting parity: 11100011
d | Integer Value:100 | Number of ones: 3 | Binary Value after setting parity: 01100100
e | Integer Value:101 | Number of ones: 4 | Binary Value after setting parity: 11100101
1 | Integer Value:49 | Number of ones: 3 | Binary Value after setting parity: 00110001
2 | Integer Value:50 | Number of ones: 3 | Binary Value after setting parity: 00110010
3 | Integer Value:51 | Number of ones: 4 | Binary Value after setting parity: 10110011

After setting parity:
Hex value of A: C1
Hex value of B: C2
Hex value of c: E3
Hex value of d: 64
Hex value of e: E5
Hex value of 1: 31
Hex value of 2: 32
Hex value of 3: B3
Enter '1' to continue or '0' to exit: _
```

For even parity:

```
Enter the stream of characters: ABCde123
Number of characters: 8
Enter '0' for even parity or '1' for odd parity: 0

A | Integer Value:65 | Number of ones: 2 | Binary Value after setting parity: 01000001
B | Integer Value:66 | Number of ones: 2 | Binary Value after setting parity: 01000010
C | Integer Value:99 | Number of ones: 4 | Binary Value after setting parity: 01100011
d | Integer Value:100 | Number of ones: 3 | Binary Value after setting parity: 11100100
e | Integer Value:101 | Number of ones: 4 | Binary Value after setting parity: 01100101
1 | Integer Value:49 | Number of ones: 3 | Binary Value after setting parity: 10110001
2 | Integer Value:50 | Number of ones: 3 | Binary Value after setting parity: 10110010
3 | Integer Value:51 | Number of ones: 4 | Binary Value after setting parity: 00110011

After setting parity:
Hex value of A: 41
Hex value of B: 42
Hex value of C: 63
Hex value of d: E4
Hex value of e: 65
Hex value of 1: B1
Hex value of 2: B2
Hex value of 3: 33
Enter '1' to continue or '0' to exit: _
```