```c
#include <stdio.h>
#define MAX 100
struct Node {
    char data;
    int next;
};
struct Node nodes[MAX];
int nodeCount = 0;
int head = -1;
int createNode(char data) {
    nodes[nodeCount].data = data;
    nodes[nodeCount].next = -1;
    return nodeCount++;
}
void append(char data) {
    int newNode = createNode(data);
    if (head == -1) {
        head = newNode;
    } else {
        int temp = head;
        while (nodes[temp].next != -1)
            temp = nodes[temp].next;
        nodes[temp].next = newNode;
```

```
    }
}
int reverse(int start) {
    int prev = -1, curr = start, next;
    while (curr != -1) {
        next = nodes[curr].next;
        nodes[curr].next = prev;
        prev = curr;
        curr = next;
    }
    return prev;
}
int compare(int h1, int h2) {
    while (h1 != -1 && h2 != -1) {
        if (nodes[h1].data != nodes[h2].data)
            return 0;
        h1 = nodes[h1].next;
        h2 = nodes[h2].next;
    }
    return 1;
}
```

```
45 int isPalindrome() {
46     int slow = head, fast = head, prev = -1;
47     while (fast != -1 && nodes[fast].next != -1) {
48         fast = nodes[nodes[fast].next].next;
49         prev = slow;
50         slow = nodes[slow].next;
51     }
52     int secondHalf = (fast == -1) ? slow : nodes[slow].next;
53     nodes[prev].next = -1;
54     secondHalf = reverse(secondHalf);
55 
56     return compare(head, secondHalf);
57 }
```

```
int main() {
    append('R');
    append('A');
    append('D');
    append('A');
    append('R');
    if (isPalindrome())
        printf("Yes\n");
    else
        printf("No\n");
    return 0;
}

```

Yes