Doubly Linked List Editing

locked

Given a doubly linked list, iterate through the list. On encountering a node whose info is a multiple of 10, increment the value of info of its previous node by 1(if it is not the first node) and delete its next node(if it is not the last node). Return the modified list from the edits function. Use the functions already given for assistance.

Input Format

Number_of_nodes
Elements of list(space separated)

Constraints

0

Output Format

Elements of modified list(space separated)

Sample Input 0

```
5
2 40 6 80 1
```

Sample Output 0

```
3 41 80
```

Explanation 0

On iterating through, 40 is the first multiple of 10. Previous node is incremented- 2 becomes 3 and the following node i.e 6 is deleted.80 is the next multiple of 10. Previous node is incremented- 40 becomes 41 and the following node i.e 1 is deleted.

Sample Input 1

```
3
50 20 30
```

Sample Output 1

```
51 30
```

Explanation 1

On iterating through, 50 is the first multiple of 10. There is no previous node and the following node i.e 20 is deleted.30 is the next multiple of 10. Previous node is incremented- 50 becomes 51 and there is no node after to be deleted.

Submissions: 65
Max Score: 10
Difficulty: Easy

Rate This Challenge:

More

C

```c
#include<stdio.h>
#include<stdlib.h>

typedef struct node
{
  struct node* prev;
  int info;
  struct node* next;
} NODE;

typedef struct dlist
{
  NODE *head;
} DLIST;

void initList(DLIST *pdl);
void insertLast(DLIST *pdl, int ele);
void display(DLIST *pdl);
void freeList(DLIST *pdl);
void deleteAtPos(DLIST *pdl, int *pe, int pos);
int countNodes(DLIST *pdl);
NODE* getNode(int ele);
void edits(DLIST *pdl);

int main()
{
  DLIST lobj;
  initList(&lobj);
  int n, ele;

  scanf("%d", &n);

  for (int i = 0; i < n; i++)
  {
    scanf("%d", &ele);
    insertLast(&lobj, ele);
  }
  // Apply edits function
  edits(&lobj);

  display(&lobj);
  freeList(&lobj);
  return 0;
}

// Edits function that modifies the list based on multiples of 10
void edits(DLIST *pdl)
{
  NODE *head = pdl->head;
  int nodecount = 0;

  while (head!=NULL) {

    if (head->info % 10 == 0) {
      if (head->prev!=NULL)
        head->prev->info++;

      if (head->next!=NULL)
        deleteAtPos(pdl, NULL, nodecount+1);
    }

    nodecount++;
    head = head->next;
  }

```

```
}

void initList(DLIST *pdl)
{
  pdl->head = NULL;
}

NODE *getNode(int ele)
{
  NODE *temp = (NODE *)malloc(sizeof(NODE));
  temp->prev = NULL;
  temp->info = ele;
  temp->next = NULL;
  return temp;
}

void insertLast(DLIST *pdl, int ele)
{
  NODE *temp = getNode(ele);
  if (pdl->head == NULL)
  {
    pdl->head = temp;
  }
  else
  {
    NODE *p = pdl->head;
    while (p->next != NULL)
    {
      p = p->next;
    }
    p->next = temp;
    temp->prev = p;
  }
}

void display(DLIST *pdl)
{
  NODE *p = pdl->head;
  if (p == NULL)
  {
    printf("Empty list\n");
  }
  else
  {
    while (p != NULL)
    {
      printf("%d ", p->info);
      p = p->next;
    }
    printf("\n");
  }
}
void freeList(DLIST *pdl)
{
  NODE *p = pdl->head;
  NODE *q = NULL;
  while (p != NULL)
  {
    q = p;
    p = p->next;
    free(q);
  }
  pdl->head = NULL;
}

void deleteAtPos(DLIST *pdl, int *pe, int pos) {

  int nodecount = 0;
  NODE *temp = pdl->head;

  while (nodecount != pos) {
    temp = temp->next;
    nodecount++;
```

```
139   }
140
141   if (temp->prev)
142     temp->prev->next = temp->next;
143
144
145   if (temp->next)
146     temp->next->prev = temp->prev;
147 }
```

Line: 147 Col: 2

Upload Code as File Test against custom input Run Code Submit Code

Testcase 0 ✔ Testcase 1 ✔

Congratulations, you passed the sample test case.

Click the **Submit Code** button to run your code against all the test cases.

Input (stdin)

```
5
2 40 6 80 1
```

Your Output (stdout)

```
3 41 80
```

Expected Output

```
3 41 80
```