Задачи от 07.10.2020

1.Напишете функция, която получава началото на едносвързан списък с цели числа и премахва всички **поредни** повторения.

```
void task1(Node* first){
//1.обхождане на списъка while(first)
//2.внимателни проверки за следващ елемент while(first->next …)
//3.премахване на елемента от дясно ако се повтаря
Node* n = first->next;
first->next = n->next;
delete n;
//4. Преместване на пойнтера напред first = first->next;
}
```

Примерен вход:

5 -> 6 -> 6 -> 7 -> 2 -> 2 -> 2 -> 1

Изход:

5 -> 6 -> 7 -> 2 -> 1

2.Напишете функция, която получава цяло число **key** и началото на едносвързан списък с цели числа. Функцията трябва да премахне всички елементи, които са равни на **key**.

```
void removeAll(int key, Node* & first){
//Защо взимаме първия елемент по референция(случая, когато трябва да махнем число, което стои в началото на списъка и трябва да преместим началото напред)?
// Edge case, когато имаме горната ситуация while(first) while(first->next && …)
```

…first = first->next;

// Итерираме останалите елементи Node* curr = first; while(curr)…..

// Изтриваме вдясно

}

Примерен вход:

2 -> 6 -> 3 -> 7 -> 2 -> 2 -> 2 -> 1

Изход:

6 -> 7 -> 1

Задачи от 14.10.2020

1.Напишете функция, която получава началото на двусвързан списък с цели числа и премахва елемент, ако сборът на съседите му е равен на стойността на елемента.

void process(Node* & head){

//Защо взимаме първия елемент по референция(случая, когато имаме само нули в началото на списъка и трябва да преместим началото напред)?

while(check(head)) { remove(head);} функцията check проверява сумата.

Функцията remove премахва съответния елемент и премества указателя напред.

//Проверка дали има още елементи

// Итерираме останалите елементи и правим същата операция като в първата стъпка Node* curr = head;

// местим пойнтера напред като внимаваме той вече да не е nullptr

curr = curr ? curr->next : nullptr;

}

//отбелязвам, че функцията за премахване е void. На упражнението я написахме, че връща Node*. Съжалявам за объркването.

```
void remove(Node* & n){

Node* next = n->next;

if(n->prev) n->prev->next = n->next;

if(n->next) n->next->prev = n->prev;

delete n;

n = next;

}
```

2.Напишете функция, която получава началото на двусвързан списък с цели числа и вмъква елемент, който съдържа броят **поредни** повторения на елемента след него и изтрива всички негови повторения.

```
void func(Node* & head){

// итерираме списъка  Node* curr = head; while(curr)

//създаваме нов Node*(кутийка), който съдържа броят на повтарящите се елементи Node* counter = new Node(1);

//закачаме новата кутийка пред текущия елемент

//внимаваме със закачането(ако има елемент преди него) Помогнете си с лист хартия и преместване на стрелки

// продължава итерацията, докато имаме еднакви елементи. Инкрементираме data-та в новата кутия и премахваме елемента текущия или елемента вдясно(внимаваме със закачането на указателите)

while(curr->next && …) counter->data++; remove(curr/curr->next);

// връщаме head най-отпред, тъй като във всеки случай(освен празния списък) ще добавяме елемент отпред и ще трябва да преместим head напред.
```

while(head->prev) head = head->prev;

}

Примерен вход:

5 -> 3 -> 3 -> 2 -> 2 -> 2 -> 2 -> 1

Изход:

1 -> 5 -> **2** -> 3 -> **4** -> 2 -> **1** -> 1

Задача за домашно/подготовка за следващия път:

Напишете функция, която приема несортиран масив от цели числа и връща начало към свързан списък, който е сортиран. Нямате право да сортирате масива предварително!

Примерен вход:

5, 3, 9, 1, 2, 13, 0, 12, 3

Изход:

0 -> 1 -> 2 -> 3 -> 3 -> 5 -> 9 -> 12 -> 13