#include <iostream>

using namespace std;

struct Node { // Структура Node

 Node\* next; // Показчик на наступний елемент

 int val; // Значення де зберігаються коефіцієнти многочлена

 Node(int \_val) : val(\_val), next(nullptr) {} // Конструктор, який записує Студента в структуру node, для next вказівника встановлює nullptr

};

struct List { // Структура List

 Node\* first; // Вказівник first, перший елемент списку

 Node\* last; // Вказівник last, останній елемент списку

 List() : first(nullptr), last(nullptr) {} // Вказівники робляться nullptr, не указують ні на що

 bool is\_empty() { // Перевірка на те, що список пустий

 return first == nullptr; // Якщо first не вказує ні на що, то true, інакше false

 }

 void push\_back(int \_val) { // Функція для добавлення елементу в кінець списку

 Node\* p = new Node(\_val); // Створюємо новий список, з нашими данними про студента

 if (is\_empty()) { // Якщо наш список пустий, то

 first = p; // Перший елемент вказуємо на список p

 last = p; // Останній елемент також вказуємо на список p

 return; // Завершуємо функцію

 }

 last->next = p; // Якщо список був не пустий, то ми по вказівнику на наступий елемент призначаем вказівник на лист p

 last = p; // Запишемо інформацію листа p, в кінець

 }

 void print() { // Виводимо список на екран

 if (is\_empty()) return; // Якщо список пустий, то завершуємо функцію

 Node\* p = first; // Створюємо вказівник на перший елемет списку

 while (p) { // Поки вказівник не дійде до кінця списку, не буде nullptr

 cout << p->val << " ";

 p = p->next; // Рухаємось далі на наступний елемент списку

 }

 cout << endl;

 }

 int size(){

 int s = 0;

 if (is\_empty()) return 0; // Якщо список пустий, то завершуємо функцію

 Node\* p = first; // Створюємо вказівник на перший елемет списку

 while (p) { // Поки вказівник не дійде до кінця списку, не буде nullptr

 s++;

 p = p->next; // Рухаємось далі на наступний елемент списку

 }

 return s;

 }

 void newnode(Node\* p){

 if(p->next != nullptr){

 Node\* temp = p->next;

 Node\* t = new Node(0);

 t->next = temp;

 p->next = t;

 }

 else{

 Node\* temp = new Node(0);

 p->next = temp;

 }

 }

 void NotPaired(){

 if (is\_empty()) return; // Якщо список пустий, то завершуємо функцію

 Node\* p = first; // Створюємо вказівник на перший елемет списку

 while (p) { // Поки вказівник не дійде до кінця списку, не буде nullptr

 if(p->val % 2 == 1)

 newnode(p);

 p = p->next; // Рухаємось далі на наступний елемент списку

 }

 }

};

int main() {

 List list;

 list.push\_back(41);

 list.push\_back(43);

 list.print();

 list.NotPaired();

 list.print();

}