#include <bits/stdc++.h>

using namespace std;

struct Node {

int data;

struct Node\* next;

};

struct Node\* addToEmpty(struct Node\* last, int data)

{

if (last != NULL)

return last;

struct Node\* temp = (struct Node\*)malloc(sizeof(struct Node));

temp->data = data;

last = temp;

last->next = last;

return last;

}

struct Node\* addBegin(struct Node\* last, int data)

{

if (last == NULL)

return addToEmpty(last, data);

struct Node\* temp

= (struct Node\*)malloc(sizeof(struct Node));

temp->data = data;

temp->next = last->next;

last->next = temp;

return last;

}

struct Node\* addEnd(struct Node\* last, int data)

{

if (last == NULL)

return addToEmpty(last, data);

struct Node\* temp = (struct Node\*)malloc(sizeof(struct Node));

temp->data = data;

temp->next = last->next;

last->next = temp;

last = temp;

return last;

}

struct Node\* addAfter(struct Node\* last, int data, int item)

{

if (last == NULL)

return NULL;

struct Node \*temp, \*p;

p = last->next;

do {

if (p->data == item) {

temp = (struct Node\*)malloc(sizeof(struct Node));

temp->data = data;

temp->next = p->next;

p->next = temp;

if (p == last)

last = temp;

return last;

}

p = p->next;

} while (p != last->next);

cout << item << " not present in the list." << endl;

return last;

}

void traverse(struct Node\* last)

{

struct Node\* p;

if (last == NULL) {

cout << "List is empty." << endl;

return;

}

p = last->next;

do {

cout << p->data << " ";

p = p->next;

} while (p != last->next);

}

int main()

{

struct Node\* last = NULL;

last = addToEmpty(last, 6);

last = addBegin(last, 4);

last = addBegin(last, 2);

last = addEnd(last, 8);

last = addEnd(last, 12);

last = addAfter(last, 10, 8);

traverse(last);

return 0;

}