#include <stdio.h>

#include <stdlib.h>

// Define node structure

typedef struct node {

int info;

struct node \*next;

} node;

// Define list structure

typedef struct {

struct node \*top;

} list;

// Create a new list

list \*makelist(void) {

list \*l;

l = (list \*)malloc(sizeof(list));

l->top = NULL;

return l;

}

// Create and insert a node at the end

void createlist(list \*l, int v) {

node \*n, \*p;

n = (node \*)malloc(sizeof(node));

n->info = v;

if (l->top == NULL) {

n->next = NULL;

l->top = n;

} else {

p = l->top;

while (p->next != NULL) {

p = p->next;

}

p->next = n;

n->next = NULL;

}

}

// Push a node at the beginning (top of stack)

void push(list \*l, int v) {

node \*n;

n = (node \*)malloc(sizeof(node));

n->info = v;

if (l->top == NULL) {

n->next = NULL;

} else {

n->next = l->top;

}

l->top = n;

}

// Pop the top node (like stack pop)

void pop(list \*l) {

if (l->top == NULL) {

printf("List is empty.\n");

return;

}

node \*t;

t = l->top;

l->top = t->next;

free(t);

}

// Peep the top value of the stack

void peep(list \*l) {

node \*t;

t = l->top;

if (t != NULL) {

printf("%d\n", t->info);

} else {

printf("List is empty.\n");

}

}

// Print the full list

void printlist(list \*l) {

node \*t;

t = l->top;

while (t != NULL) {

printf("%d ", t->info);

t = t->next;

}

printf("\n");

}

// Main function

int main() {

list \*ls;

int a, i, v;

ls = makelist();

printf("ENTER VAL IN LIST:\n");

for (i = 0; i < 5; i++) {

scanf("%d", &a);

createlist(ls, a);

}

printlist(ls);

printf("\nENTER FIRST NODE IN LIST:\n");

scanf("%d", &v);

push(ls, v);

peep(ls);

printf("\nDelete first node:\n");

pop(ls);

printlist(ls);

return 0;

}