**Bài tập về danh sách liên kết**

# Bài tập Danh sách liên kết trong C

[Trang trước](https://vietjack.com/bai-tap-c/bai-tap-struct-trong-c.jsp)

[Trang sau](https://vietjack.com/bai-tap-c/bai-tap-doc-ghi-file-trong-c.jsp)

## Danh sách liên kết (Linked List) là gì ?

Một Danh sách liên kết (Linked List) là một dãy các cấu trúc dữ liệu được kết nối với nhau thông qua các liên kết (link). Hiểu một cách đơn giản thì Danh sách liên kết là một cấu trúc dữ liệu bao gồm một nhóm các nút (node) tạo thành một chuỗi. Mỗi nút gồm dữ liệu ở nút đó và tham chiếu đến nút kế tiếp trong chuỗi.

Danh sách liên kết là cấu trúc dữ liệu được sử dụng phổ biến thứ hai sau mảng. Dưới đây là các khái niệm cơ bản liên quan tới Danh sách liên kết:

* **Link (liên kết)**: mỗi link của một Danh sách liên kết có thể lưu giữ một dữ liệu được gọi là một phần tử.
* **Next**: Mỗi liên kết của một Danh sách liên kết chứa một link tới next link được gọi là Next.
* **First**: một Danh sách liên kết bao gồm các link kết nối tới first link được gọi là First.

## Các loại Danh sách liên kết (Linked List)

Dưới đây là các loại Danh sách liên kết (Linked List) đa dạng:

* **Danh sách liên kết đơn (Simple Linked List)**: chỉ duyệt các phần tử theo chiều về trước.
* **Danh sách liên kết đôi (Doubly Linked List)**: các phần tử có thể được duyệt theo chiều về trước hoặc về sau.
* **Danh sách liên kết vòng (Circular Linked List)**: phần tử cuối cùng chứa link của phần tử đầu tiên như là next và phần tử đầu tiên có link tới phần tử cuối cùng như là prev.

Để tìm hiểu thêm về các khái niệm của cấu trúc dữ liệu Danh sách liên kết, mời bạn tham khảo ở chương [**Cấu trúc dữ liệu Danh sách liên kết**](https://vietjack.com/cau-truc-du-lieu-va-giai-thuat/cau-truc-du-lieu-danh-sach-lien-ket.jsp).

Phần dưới đây là danh sách các bài tập C về Danh sách liên kết:

## Bài tập C về Danh sách liên kết đơn

Dưới đây là các bài tập về Danh sách liên kết đơn trong C:

* [**Tạo danh sách liên kết** trong C](https://vietjack.com/bai-tap-c/tao-danh-sach-lien-ket-don-trong-c.jsp)
* [Chương trình C để **hiển thị danh sách liên kết** theo chiều đảo ngược](https://vietjack.com/bai-tap-c/hien-thi-danh-sach-lien-ket-theo-chieu-dao-nguoc-trong-c.jsp)
* [**Tìm kích cỡ** của danh sách liên kết trong C](https://vietjack.com/bai-tap-c/tim-kich-co-cua-danh-sach-lien-ket-don-trong-c.jsp)
* [Chương trình C để **tìm kiếm một phần tử** trong danh sách liên kết](https://vietjack.com/bai-tap-c/tim-kiem-phan-tu-trong-danh-sach-lien-ket-don-trong-c.jsp)
* [**Cập nhật một phần tử** của danh sách liên kết trong C](https://vietjack.com/bai-tap-c/cap-nhat-mot-phan-tu-cua-danh-sach-lien-ket-don-trong-c.jsp)
* [Chương trình C để **xóa một phần tử** từ danh sách liên kết](https://vietjack.com/bai-tap-c/xoa-mot-phan-tu-tu-danh-sach-lien-ket-don-trong-c.jsp)
* [**Nối hai danh sách liên kết** trong C](https://vietjack.com/bai-tap-c/noi-hai-danh-sach-lien-ket-don-trong-c.jsp)
* [Chương trình C để **chia một danh sách liên kết thành hai**](https://vietjack.com/bai-tap-c/cach-chia-mot-danh-sach-lien-ket-don-trong-c.jsp)

## Bài tập C về Danh sách liên kết vòng

Để tìm hiểu thêm về các khái niệm của cấu trúc dữ liệu Danh sách liên kết vòng, mời bạn tham khảo ở chương [**Cấu trúc dữ liệu Danh sách liên kết vòng**](https://vietjack.com/cau-truc-du-lieu-va-giai-thuat/cau-truc-du-lieu-danh-sach-lien-ket-vong.jsp).

Dưới đây là các bài tập về Danh sách liên kết vòng trong C:

* [**Tạo danh sách liên kết vòng** trong C](https://vietjack.com/bai-tap-c/tao-danh-sach-lien-ket-vong-trong-c.jsp)
* [Chương trình C để **hiển thị danh sách liên kết vòng** theo chiều đảo ngược](https://vietjack.com/bai-tap-c/hien-thi-danh-sach-lien-ket-vong-theo-chieu-dao-nguoc-trong-c.jsp)
* [**Tìm kích cỡ** của danh sách liên kết vòng trong C](https://vietjack.com/bai-tap-c/tim-kich-co-cua-danh-sach-lien-ket-vong-trong-c.jsp)
* [Chương trình C để **tìm kiếm một phần tử** trong danh sách liên kết vòng](https://vietjack.com/bai-tap-c/tim-kiem-phan-tu-trong-danh-sach-lien-ket-vong-trong-c.jsp)
* [**Cập nhật một phần tử** của danh sách liên kết vòng trong C](https://vietjack.com/bai-tap-c/cap-nhat-mot-phan-tu-cua-danh-sach-lien-ket-vong-trong-c.jsp)
* [Chương trình C để **xóa một phần tử** từ danh sách liên kết vòng](https://vietjack.com/bai-tap-c/xoa-mot-phan-tu-tu-danh-sach-lien-ket-vong-trong-c.jsp)
* [**Nối hai danh sách liên kết vòng** trong C](https://vietjack.com/bai-tap-c/noi-hai-danh-sach-lien-ket-vong-trong-c.jsp)
* [Chương trình C để **chia một danh sách liên kết vòng thành hai**](https://vietjack.com/bai-tap-c/cach-chia-mot-danh-sach-lien-ket-vong-trong-c.jsp)

## Bài tập C về Danh sách liên kết đôi

Để tìm hiểu thêm về các khái niệm của cấu trúc dữ liệu Danh sách liên kết đôi, mời bạn tham khảo ở chương [**Cấu trúc dữ liệu Danh sách liên kết đôi**](https://vietjack.com/cau-truc-du-lieu-va-giai-thuat/cau-truc-du-lieu-danh-sach-lien-ket-doi.jsp).

Dưới đây là các bài tập về Danh sách liên kết đôi trong C:

* [**Tạo danh sách liên kết đôi** trong C](https://vietjack.com/bai-tap-c/tao-danh-sach-lien-ket-doi-trong-c.jsp)
* [Chương trình C để **hiển thị danh sách liên kết đôi** theo chiều đảo ngược](https://vietjack.com/bai-tap-c/hien-thi-danh-sach-lien-ket-doi-theo-chieu-dao-nguoc-trong-c.jsp)
* [**Tìm kích cỡ** của danh sách liên kết đôi trong C](https://vietjack.com/bai-tap-c/tim-kich-co-cua-danh-sach-lien-ket-doi-trong-c.jsp)
* [Chương trình C để **tìm kiếm một phần tử** trong danh sách liên kết đôi](https://vietjack.com/bai-tap-c/tim-kiem-phan-tu-trong-danh-sach-lien-ket-doi-trong-c.jsp)
* [**Cập nhật một phần tử** của danh sách liên kết đôi trong C](https://vietjack.com/bai-tap-c/cap-nhat-mot-phan-tu-cua-danh-sach-lien-ket-doi-trong-c.jsp)
* [Chương trình C để **xóa một phần tử** từ danh sách liên kết đôi](https://vietjack.com/bai-tap-c/xoa-mot-phan-tu-tu-danh-sach-lien-ket-doi-trong-c.jsp)
* [**Nối hai danh sách liên kết đôi** trong C](https://vietjack.com/bai-tap-c/noi-hai-danh-sach-lien-ket-doi-trong-c.jsp)
* [Chương trình C để **chia một danh sách liên kết đôi thành hai**](https://vietjack.com/bai-tap-c/cach-chia-mot-danh-sach-lien-ket-doi-trong-c.jsp)

**DSLK Đơn**

**Bài 1:**

Tạo danh sách liên kết đơn trong C

[Trang trước](https://vietjack.com/bai-tap-c/danh-sach-lien-ket-trong-c.jsp)

[Trang sau](https://vietjack.com/bai-tap-c/danh-sach-lien-ket-trong-c.jsp)

Bài tập C: Tạo danh sách liên kết đơn

Bài tập C này giúp bạn làm quen dần với cách tạo danh sách liên kết đơn trong C. Để giải bài tập này, mình sử dụng cấu trúc **struct** trong C.

Chương trình C

Dưới đây là chương trình C để giải bài tập tạo danh sách liên kết đơn trong C:

#include <stdio.h>

#include <stdlib.h>

struct node {

int data;

struct node \*next;

};

struct node \*head = NULL;

struct node \*current = NULL;

//hien thi list

void printList() {

struct node \*ptr = head;

printf("\n[head] =>");

//bat dau tu phan dau cua list

while(ptr != NULL) {

printf(" %d =>",ptr->data);

ptr = ptr->next;

}

printf(" [null]\n");

}

//chen link tai vi tri dau tien

void insert(int data) {

//tao mot link

struct node \*link = (struct node\*) malloc(sizeof(struct node));

//link->key = key;

link->data = data;

//tro link do toi first node cu

link->next = head;

//tro first toi first node moi

head = link;

}

int main() {

insert(10);

insert(20);

insert(30);

insert(1);

insert(40);

insert(56);

printList();

return 0;

}

Biên dịch chương trình C trên sẽ cho kết quả:

Tạo danh sách liên kết đơn trong C

**Bài 2:**

# Hiển thị danh sách liên kết theo chiều đảo ngược trong C

[Trang trước](https://vietjack.com/bai-tap-c/danh-sach-lien-ket-trong-c.jsp)

[Trang sau](https://vietjack.com/bai-tap-c/danh-sach-lien-ket-trong-c.jsp)

## Bài tập C: Hiển thị danh sách liên kết theo chiều đảo ngược

Bài tập C này giúp bạn làm quen dần với cách tạo danh sách liên kết đơn và cách hiển thị danh sách liên kết đơn trong C. Để giải bài tập này, mình sử dụng cấu trúc **struct** trong C.

## Chương trình C

Dưới đây là chương trình C để giải bài tập hiển thị danh sách liên kết theo chiều đảo ngược trong C:

#include <stdio.h>

#include <stdlib.h>

struct node {

int data;

struct node \*next;

};

struct node \*head = NULL;

struct node \*current = NULL;

//hien thi list

void reverse\_print(struct node \*list) {

if(list == NULL) {

printf("[null] => ");

return;

}

reverse\_print(list->next);

printf(" %d =>",list->data);

}

//tao danh sach lien ket

void insert(int data) {

// cap phat bo nho cho node moi;

struct node \*link = (struct node\*) malloc(sizeof(struct node));

link->data = data;

link->next = NULL;

// neu head la trong, tao list moi

if(head==NULL) {

head = link;

return;

}

current = head;

// di chuyen toi phan cuoi list

while(current->next!=NULL)

current = current->next;

// chen link vao phan cuoi cua list

current->next = link;

}

int main() {

insert(10);

insert(20);

insert(30);

insert(1);

insert(40);

insert(56);

reverse\_print(head);

return 0;

}

Biên dịch chương trình C trên sẽ cho kết quả:

Hiển thị danh sách liên kết đơn trong C

**Bài 3:**

# Tìm kích cỡ của danh sách liên kết đơn trong C

[Trang trước](https://vietjack.com/bai-tap-c/danh-sach-lien-ket-trong-c.jsp)

[Trang sau](https://vietjack.com/bai-tap-c/danh-sach-lien-ket-trong-c.jsp)

## Bài tập C: tìm kích cỡ của danh sách liên kết đơn

Bài tập C này giúp bạn làm quen dần với cách tạo danh sách liên kết đơn và cách tìm kích cỡ của danh sách liên kết đơn trong C. Để giải bài tập này, mình sử dụng cấu trúc **struct** trong C.

## Chương trình C

Dưới đây là chương trình C để giải bài tập tìm kích cỡ của danh sách liên kết đơn trong C:

#include <stdio.h>

#include <stdlib.h>

struct node {

int data;

struct node \*next;

};

struct node \*head = NULL;

struct node \*current = NULL;

//tao danh sach lien ket

void insert(int data) {

// cap phat bo nho cho node moi;

struct node \*link = (struct node\*) malloc(sizeof(struct node));

link->data = data;

link->next = NULL;

// neu head la trong, tao list moi

if(head==NULL) {

head = link;

return;

}

current = head;

// di chuyen toi phan cuoi list

while(current->next!=NULL)

current = current->next;

// chen link vao phan cuoi cua list

current->next = link;

}

void size\_of\_list() {

int size = 0;

if(head==NULL) {

printf("Kich co cua Danh sach lien ket: %d ", size);

return;

}

current = head;

size = 1;

while(current->next!=NULL) {

current = current->next;

size++;

}

printf("Kich co cua Danh sach lien ket: %d ", size);

}

int main() {

insert(10);

insert(20);

insert(30);

insert(1);

insert(40);

insert(56);

size\_of\_list();

return 0;

}

Biên dịch chương trình C trên sẽ cho kết quả:

Tìm kích cỡ của danh sách liên kết đơn trong C

**Bài 4:**

# Tìm kiếm phần tử trong danh sách liên kết đơn trong C

[Trang trước](https://vietjack.com/bai-tap-c/danh-sach-lien-ket-trong-c.jsp)

[Trang sau](https://vietjack.com/bai-tap-c/danh-sach-lien-ket-trong-c.jsp)

## Bài tập C: tìm kiếm một phần tử trong danh sách liên kết đơn

Bài tập C này giúp bạn làm quen dần với cách tạo danh sách liên kết đơn và cách tìm kiếm một phần tử trong danh sách liên kết đơn trong C. Để giải bài tập này, mình sử dụng cấu trúc **struct** trong C.

## Chương trình C

Dưới đây là chương trình C để giải bài tập tìm kiếm một phần tử trong danh sách liên kết đơn trong C:

#include <stdio.h>

#include <stdlib.h>

struct node {

int data;

struct node \*next;

};

struct node \*head = NULL;

struct node \*current = NULL;

//tao danh sach lien ket

void insert(int data) {

// cap phat bo nho cho node moi;

struct node \*link = (struct node\*) malloc(sizeof(struct node));

link->data = data;

link->next = NULL;

// neu head la trong, tao list moi

if(head==NULL) {

head = link;

return;

}

current = head;

// di chuyen toi phan cuoi list

while(current->next!=NULL)

current = current->next;

// chen link vao phan cuoi cua list

current->next = link;

}

void find\_data(int item) {

int pos = 0;

if(head==NULL) {

printf("Danh sach lien ket chua duoc khoi tao");

return;

}

current = head;

while(current->next!=NULL) {

if(current->data == item) {

printf("Tim thay %d tai vi tri %d\n", item, pos);

return;

}

current = current->next;

pos++;

}

printf("%d khong ton tai trong list", item);

}

int main() {

insert(10);

insert(20);

insert(30);

insert(1);

insert(40);

insert(56);

find\_data(40);

find\_data(44);

return 0;

}

Biên dịch chương trình C trên sẽ cho kết quả:

Tìm kiếm phần tử trong danh sách liên kết trong C

**Bài 5:**

# Cập nhật một phần tử của danh sách liên kết đơn trong C

[Trang trước](https://vietjack.com/bai-tap-c/danh-sach-lien-ket-trong-c.jsp)

[Trang sau](https://vietjack.com/bai-tap-c/danh-sach-lien-ket-trong-c.jsp)

## Bài tập C: Cập nhật một phần tử của danh sách liên kết đơn

Bài tập C này giúp bạn làm quen dần với cách tạo danh sách liên kết đơn và cách cập nhật một phần tử của danh sách liên kết đơn trong C. Để giải bài tập này, mình sử dụng cấu trúc **struct** trong C.

## Chương trình C

Dưới đây là chương trình C để giải bài tập cập nhật một phần tử của danh sách liên kết đơn trong C:

#include <stdio.h>

#include <stdlib.h>

struct node {

int data;

struct node \*next;

};

struct node \*head = NULL;

struct node \*current = NULL;

//tao danh sach lien ket

void insert(int data) {

// cap phat bo nho cho node moi;

struct node \*link = (struct node\*) malloc(sizeof(struct node));

link->data = data;

link->next = NULL;

// neu head la trong, tao list moi

if(head==NULL) {

head = link;

return;

}

current = head;

// di chuyen toi phan cuoi list

while(current->next!=NULL)

current = current->next;

// chen link vao phan cuoi cua list

current->next = link;

}

void display() {

struct node \*ptr = head;

printf("\n[head] =>");

//bat dau tu phan dau cua list

while(ptr != NULL) {

printf(" %d =>",ptr->data);

ptr = ptr->next;

}

printf(" [null]\n");

}

void update\_data(int old, int new1) {

int pos = 0;

if(head==NULL) {

printf("Danh sach lien ket chua duoc khoi tao");

return;

}

current = head;

while(current->next!=NULL) {

if(current->data == old) {

current->data = new1;

printf("\nTim thay %d tai vi tri %d, duoc thay the voi %d\n", old, pos, new1);

return;

}

current = current->next;

pos++;

}

printf("%d khong ton tai trong list\n", old);

}

int main() {

insert(10);

insert(20);

insert(30);

insert(1);

insert(40);

insert(56);

display();

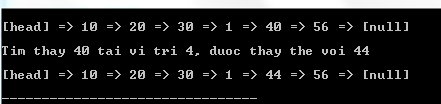
update\_data(40, 44);

display();

return 0;

}

Biên dịch chương trình C trên sẽ cho kết quả:



**Bài 6:**

# Xóa một phần tử từ danh sách liên kết đơn trong C

[Trang trước](https://vietjack.com/bai-tap-c/danh-sach-lien-ket-trong-c.jsp)

[Trang sau](https://vietjack.com/bai-tap-c/danh-sach-lien-ket-trong-c.jsp)

## Bài tập C: Xóa một phần tử từ danh sách liên kết

Bài tập C này giúp bạn làm quen dần với cách tạo danh sách liên kết đơn và cách xóa một phần tử từ danh sách liên kết đơn trong C. Để giải bài tập này, mình sử dụng cấu trúc **struct** trong C.

## Chương trình C

Dưới đây là chương trình C để giải bài tập xóa một phần tử từ danh sách liên kết đơn trong C:

#include <stdio.h>

#include <stdlib.h>

struct node {

int data;

struct node \*next;

};

struct node \*head = NULL;

struct node \*current = NULL;

struct node \*prev = NULL;

//tao danh sach lien ket

void insert(int data) {

// cap phat bo nho cho node moi;

struct node \*link = (struct node\*) malloc(sizeof(struct node));

link->data = data;

link->next = NULL;

// neu head la trong, tao list moi

if(head==NULL) {

head = link;

return;

}

current = head;

// di chuyen toi phan cuoi list

while(current->next!=NULL)

current = current->next;

// chen link vao phan cuoi cua list

current->next = link;

}

void display() {

struct node \*ptr = head;

printf("head] =>");

//bat dau tu phan dau cua list

while(ptr != NULL) {

printf(" %d =>",ptr->data);

ptr = ptr->next;

}

printf(" [null]\n");

}

void remove\_data(int data) {

int pos = 0;

if(head==NULL) {

printf("Danh sach lien ket chua duoc khoi tao");

return;

}

if(head->data == data) {

if(head->next != NULL) {

head = head->next;

return;

}else {

head = NULL;

printf("Bay gio List la trong");

return;

}

}else if(head->data != data && head->next == NULL) {

printf("Khong tim thay %d trong list\n", data);

return;

}

// prev = head;

current = head;

while(current->next != NULL && current->data != data) {

prev = current;

current = current->next;

}

if(current->data == data) {

prev->next = prev->next->next;

free(current);

}else

printf("Khong tim thay %d trong list.", data);

}

int main() {

insert(10);

insert(20);

insert(30);

insert(1);

insert(40);

insert(56);

printf("Truoc khi xoa: ");

display();

remove\_data(30);

printf("Sau khi xoa: ");

display();

return 0;

}

Biên dịch chương trình C trên sẽ cho kết quả:

Xóa một phần tử từ danh sách liên kết đơn trong C

**Bài 7:**

# Nối hai danh sách liên kết đơn trong C

[Trang trước](https://vietjack.com/bai-tap-c/danh-sach-lien-ket-trong-c.jsp)

[Trang sau](https://vietjack.com/bai-tap-c/danh-sach-lien-ket-trong-c.jsp)

## Bài tập C: Nối hai danh sách liên kết đơn

Bài tập C này giúp bạn làm quen dần với cách tạo danh sách liên kết đơn và cách nối hai danh sách liên kết đơn trong C. Để giải bài tập này, mình sử dụng cấu trúc **struct** trong C.

## Chương trình C

Dưới đây là chương trình C để giải bài tập nối hai danh sách liên kết đơn trong C:

#include <stdio.h>

#include <stdlib.h>

struct node {

int data;

struct node \*next;

};

struct node \*even = NULL;

struct node \*odd = NULL;

struct node \*list = NULL;

//tao danh sach lien ket

void insert(int data) {

// cap phat bo nho cho node moi;

struct node \*link = (struct node\*) malloc(sizeof(struct node));

struct node \*current;

link->data = data;

link->next = NULL;

if(data%2 == 0) {

if(even == NULL) {

even = link;

return;

}else {

current = even;

while(current->next != NULL)

current = current->next;

// chen link vao phan cuoi cua list

current->next = link;

}

}else {

if(odd == NULL) {

odd = link;

return;

}else {

current = odd;

while(current->next!=NULL)

current = current->next;

// chen link vao phan cuoi cua list

current->next = link;

}

}

}

void display(struct node \*head) {

struct node \*ptr = head;

printf("[head] =>");

while(ptr != NULL) {

printf(" %d =>",ptr->data);

ptr = ptr->next;

}

printf(" [null]\n");

}

void combine() {

struct node \*link;

list = even;

link = list;

while(link->next!= NULL) {

link = link->next;

}

link->next = odd;

}

int main() {

int i;

for(i=1; i<=10; i++)

insert(i);

printf("Danh sach chan: ");

display(even);

printf("Danh sach le: ");

display(odd);

combine();

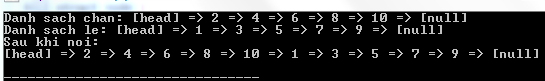
printf("Sau khi noi: \n");

display(list);

return 0;

}

Biên dịch chương trình C trên sẽ cho kết quả:



**Bài 8:**

# Cách chia một danh sách liên kết đơn trong C

[Trang trước](https://vietjack.com/bai-tap-c/danh-sach-lien-ket-trong-c.jsp)

[Trang sau](https://vietjack.com/bai-tap-c/danh-sach-lien-ket-trong-c.jsp)

## Bài tập C: chia một danh sách liên kết đơn

Bài tập C này giúp bạn làm quen dần với cách tạo danh sách liên kết đơn và cách chia một danh sách liên kết đơn trong C. Để giải bài tập này, mình sử dụng cấu trúc **struct** trong C.

## Chương trình C

Dưới đây là chương trình C để giải bài tập chia một danh sách liên kết đơn trong C:

#include <stdio.h>

#include <stdlib.h>

struct node {

int data;

struct node \*next;

};

struct node \*even = NULL;

struct node \*odd = NULL;

struct node \*list = NULL;

//tao danh sach lien ket

void insert(int data) {

// cap phat bo nho cho node moi;

struct node \*link = (struct node\*) malloc(sizeof(struct node));

struct node \*current;

link->data = data;

link->next = NULL;

if(list == NULL) {

list = link;

return;

}

current = list;

while(current->next!=NULL)

current = current->next;

// chen link vao phan cuoi cua list

current->next = link;

}

void display(struct node \*head) {

struct node \*ptr = head;

printf("[head] =>");

//bat dau tu phan dau cua list

while(ptr != NULL) {

printf(" %d =>",ptr->data);

ptr = ptr->next;

}

printf(" [null]\n");

}

void split\_list() {

// cap phat bo nho cho node moi;

struct node \*link;

struct node \*current;

while(list != NULL) {

struct node \*link = (struct node\*) malloc(sizeof(struct node));

link->data = list->data;

link->next = NULL;

if(list->data%2 == 0) {

if(even == NULL) {

even = link;

list = list->next;

continue;

}else {

current = even;

while(current->next != NULL)

current = current->next;

// chen link vao phan cuoi cua list

current->next = link;

}

list = list->next;

}else {

if(odd == NULL) {

odd = link;

list = list->next;

continue;

}else {

current = odd;

while(current->next!=NULL)

current = current->next;

// chen link vao phan cuoi cua list

current->next = link;

}

list = list->next;

}

}

}

int main() {

int i;

for(i=1; i<=10; i++)

insert(i);

printf("Danh sach ban dau: \n");

display(list);

split\_list();

printf("\nDanh sach le: ");

display(odd);

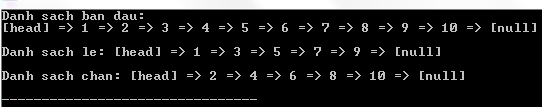
printf("\nDanh sach chan: ");

display(even);

return 0;

}

Biên dịch chương trình C trên sẽ cho kết quả:



**DSLK Vòng**

**Bài 1:**

# Tạo danh sách liên kết vòng trong C

[Trang trước](https://vietjack.com/bai-tap-c/danh-sach-lien-ket-trong-c.jsp)

[Trang sau](https://vietjack.com/bai-tap-c/danh-sach-lien-ket-trong-c.jsp)

## Bài tập C: Tạo danh sách liên kết vòng

Bài tập C này giúp bạn làm quen dần với cách tạo danh sách liên kết vòng trong C. Để giải bài tập này, mình sử dụng cấu trúc **struct** trong C.

## Chương trình C

Dưới đây là chương trình C để giải bài tập tạo danh sách liên kết vòng trong C:

#include <stdio.h>

#include <stdlib.h>

struct node {

int data;

struct node \*next;

};

struct node \*head = NULL;

struct node \*current = NULL;

//chen link tai vi tri dau tien

void insert(int data) {

// cap phat bo nho cho node moi;

struct node \*link = (struct node\*) malloc(sizeof(struct node));

link->data = data;

link->next = NULL;

// neu head la trong, tao list moi

if(head==NULL) {

head = link;

head->next = link;

return;

}

current = head;

// di chuyen toi phan cuoi list

while(current->next != head)

current = current->next;

// chen link vao phan cuoi cua list

current->next = link;

// lien ket last node voi head

link->next = head;

}

//hien thi list

void printList() {

struct node \*ptr = head;

printf("\n[head] =>");

//bat dau tu phan dau cua list

while(ptr->next != head) {

printf(" %d =>",ptr->data);

ptr = ptr->next;

}

printf(" %d =>",ptr->data);

printf(" [head]\n");

}

int main() {

insert(10);

insert(20);

insert(30);

insert(1);

insert(40);

insert(56);

printList();

return 0;

}

Biên dịch chương trình C trên sẽ cho kết quả:

Tạo danh sách liên kết vòng trong C

**Bài 2:**

# Hiển thị danh sách liên kết vòng theo chiều đảo ngược trong C

[Trang trước](https://vietjack.com/bai-tap-c/danh-sach-lien-ket-trong-c.jsp)

[Trang sau](https://vietjack.com/bai-tap-c/danh-sach-lien-ket-trong-c.jsp)

## Bài tập C: Hiển thị danh sách liên kết vòng theo chiều đảo ngược

Bài tập C này giúp bạn làm quen dần với cách tạo danh sách liên kết vòng và cách hiển thị danh sách liên kết vòng trong C. Để giải bài tập này, mình sử dụng cấu trúc **struct** trong C.

## Chương trình C

Dưới đây là chương trình C để giải bài tập hiển thị danh sách liên kết vòng theo chiều đảo ngược trong C:

#include <stdio.h>

#include <stdlib.h>

struct node {

int data;

struct node \*next;

};

struct node \*head = NULL;

struct node \*current = NULL;

//chen link tai vi tri dau tien

void insert(int data) {

// cap phat bo nho cho node moi;

struct node \*link = (struct node\*) malloc(sizeof(struct node));

link->data = data;

link->next = NULL;

// neu head la trong, tao list moi

if(head==NULL) {

head = link;

head->next = link;

return;

}

current = head;

// di chuyen toi phan cuoi list

while(current->next != head)

current = current->next;

// chen link vao phan cuoi cua list

current->next = link;

// lien ket last node voi head

link->next = head;

}

//hien thi list

void reverse\_print(struct node \*list) {

if(list->next == head) {

printf(" %d =>",list->data);

return;

}

reverse\_print(list->next);

printf(" %d =>",list->data);

}

int main() {

insert(10);

insert(20);

insert(30);

insert(1);

insert(40);

insert(56);

reverse\_print(head);

printf(" [head]\n");

return 0;

}

Biên dịch chương trình C trên sẽ cho kết quả:

Hiển thị danh sách liên kết vòng trong C

**Bài 3:**

# Tìm kích cỡ của danh sách liên kết vòng trong C

[Trang trước](https://vietjack.com/bai-tap-c/danh-sach-lien-ket-trong-c.jsp)

[Trang sau](https://vietjack.com/bai-tap-c/danh-sach-lien-ket-trong-c.jsp)

## Bài tập C: Tìm kích cỡ của danh sách liên kết vòng

Bài tập C này giúp bạn làm quen dần với cách tạo danh sách liên kết vòng và cách tìm kích cỡ của danh sách liên kết vòng trong C. Để giải bài tập này, mình sử dụng cấu trúc **struct** trong C.

## Chương trình C

Dưới đây là chương trình C để giải bài tập tìm kích cỡ của danh sách liên kết vòng trong C:

#include <stdio.h>

#include <stdlib.h>

struct node {

int data;

struct node \*next;

};

struct node \*head = NULL;

struct node \*current = NULL;

//chen link tai vi tri dau tien

void insert(int data) {

// cap phat bo nho cho node moi;

struct node \*link = (struct node\*) malloc(sizeof(struct node));

link->data = data;

link->next = NULL;

// neu head la trong, tao list moi

if(head==NULL) {

head = link;

head->next = link;

return;

}

current = head;

// di chuyen toi phan cuoi list

while(current->next != head)

current = current->next;

// chen link vao phan cuoi cua list

current->next = link;

// lien ket last node voi head

link->next = head;

}

//hien thi list

void list\_size(struct node \*list) {

int size = 0;

if(head==NULL) {

printf("Kich co list: %d ", size);

return;

}

current = head;

size = 1;

while(current->next != head) {

current = current->next;

size++;

}

printf("Kich co list: %d ", size);

}

int main() {

insert(10);

insert(20);

insert(30);

insert(1);

insert(40);

insert(56);

list\_size(head);

return 0;

}

Biên dịch chương trình C trên sẽ cho kết quả:

Tìm kích cỡ của danh sách liên kết vòng trong C

**Bài 4:**

# Tìm kiếm phần tử trong danh sách liên kết vòng trong C

[Trang trước](https://vietjack.com/bai-tap-c/danh-sach-lien-ket-trong-c.jsp)

[Trang sau](https://vietjack.com/bai-tap-c/danh-sach-lien-ket-trong-c.jsp)

## Bài tập C: Tìm kiếm phần tử trong danh sách liên kết vòng

Bài tập C này giúp bạn làm quen dần với cách tạo danh sách liên kết vòng và cách tìm kiếm phần tử trong danh sách liên kết vòng trong C. Để giải bài tập này, mình sử dụng cấu trúc **struct** trong C.

## Chương trình C

Dưới đây là chương trình C để giải bài tập tìm kiếm phần tử trong danh sách liên kết vòng trong C:

#include <stdio.h>

#include <stdlib.h>

struct node {

int data;

struct node \*next;

};

struct node \*head = NULL;

struct node \*current = NULL;

//chen link tai vi tri dau tien

void insert(int data) {

// cap phat bo nho cho node moi;

struct node \*link = (struct node\*) malloc(sizeof(struct node));

link->data = data;

link->next = NULL;

// neu head la trong, tao list moi

if(head==NULL) {

head = link;

head->next = link;

return;

}

current = head;

// di chuyen toi phan cuoi list

while(current->next != head)

current = current->next;

// chen link vao phan cuoi cua list

current->next = link;

// lien ket last node voi head

link->next = head;

}

//hien thi list

void find\_data(int item) {

int pos = 0;

if(head == NULL) {

printf("Danh sach lien ket chua duoc khoi tao");

return;

}

current = head;

while(current->next != head) {

if(current->data == item) {

printf("Tim thay %d tai vi tri %d\n", item, pos);

return;

}

current = current->next;

pos++;

}

if(current->data == item) {

printf("Tim thay %d tai vi tri %d\n", item, pos);

return;

}

printf("%d khong ton tai trong list", item);

}

int main() {

insert(10);

insert(20);

insert(30);

insert(1);

insert(40);

insert(56);

find\_data(56);

return 0;

}

Biên dịch chương trình C trên sẽ cho kết quả:

Tìm kiếm phần tử trong danh sách liên kết vòng trong C

**Bài 5:**

# Cập nhật một phần tử của danh sách liên kết vòng trong C

[Trang trước](https://vietjack.com/bai-tap-c/danh-sach-lien-ket-trong-c.jsp)

[Trang sau](https://vietjack.com/bai-tap-c/danh-sach-lien-ket-trong-c.jsp)

## Bài tập C: Cập nhật một phần tử của danh sách liên kết vòng

Bài tập C này giúp bạn làm quen dần với cách tạo danh sách liên kết vòng và cách cập nhật một phần tử của danh sách liên kết vòng trong C. Để giải bài tập này, mình sử dụng cấu trúc **struct** trong C.

## Chương trình C

Dưới đây là chương trình C để giải bài tập cập nhật một phần tử của danh sách liên kết vòng trong C:

#include <stdio.h>

#include <stdlib.h>

struct node {

int data;

struct node \*next;

};

struct node \*head = NULL;

struct node \*current = NULL;

//chen link tai vi tri dau tien

void insert(int data) {

// cap phat bo nho cho node moi;

struct node \*link = (struct node\*) malloc(sizeof(struct node));

link->data = data;

link->next = NULL;

// neu head la trong, tao list moi

if(head==NULL) {

head = link;

head->next = link;

return;

}

current = head;

// di chuyen toi phan cuoi list

while(current->next != head)

current = current->next;

// chen link vao phan cuoi cua list

current->next = link;

// lien ket last node voi head

link->next = head;

}

void update\_data(int old, int new1) {

int pos = 0;

if(head==NULL) {

printf("Danh sach lien ket chua duoc khoi tao");

return;

}

current = head;

while(current->next!=head) {

if(current->data == old) {

current->data = new1;

printf("\nTim thay %d tai vi tri %d, duoc thay the voi %d\n", old, pos, new1);

return;

}

current = current->next;

pos++;

}

if(current->data == old) {

current->data = new1;

printf("\nTim thay %d tai vi tri %d, duoc thay the voi %d\n", old, pos, new1);

return;

}

printf("%d khong ton tai trong list\n", old);

}

//hien thi list

void printList() {

struct node \*ptr = head;

printf("\n[head] =>");

//bat dau tu phan dau cua list

while(ptr->next != head) {

printf(" %d =>",ptr->data);

ptr = ptr->next;

}

printf(" %d =>",ptr->data);

printf(" [head]\n");

}

int main() {

insert(10);

insert(20);

insert(30);

insert(1);

insert(40);

insert(56);

printList();

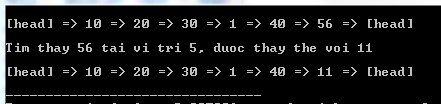
update\_data(56,11);

printList();

return 0;

}

Biên dịch chương trình C trên sẽ cho kết quả:



**Bài 6:**

# Xóa một phần tử từ danh sách liên kết vòng trong C

[Trang trước](https://vietjack.com/bai-tap-c/danh-sach-lien-ket-trong-c.jsp)

[Trang sau](https://vietjack.com/bai-tap-c/danh-sach-lien-ket-trong-c.jsp)

## Bài tập C: Xóa một phần tử từ danh sách liên kết vòng

Bài tập C này giúp bạn làm quen dần với cách tạo danh sách liên kết vòng và cách xóa một phần tử từ danh sách liên kết vòng trong C. Để giải bài tập này, mình sử dụng cấu trúc **struct** trong C.

## Chương trình C

Dưới đây là chương trình C để giải bài tập xóa một phần tử từ danh sách liên kết vòng trong C:

#include <stdio.h>

#include <stdlib.h>

struct node {

int data;

struct node \*next;

};

struct node \*head = NULL;

struct node \*current = NULL;

struct node \*prev = NULL;

//chen link tai vi tri dau tien

void insert(int data) {

// cap phat bo nho cho node moi;

struct node \*link = (struct node\*) malloc(sizeof(struct node));

link->data = data;

link->next = NULL;

// neu head la trong, tao list moi

if(head==NULL) {

head = link;

head->next = link;

return;

}

current = head;

// di chuyen toi phan cuoi list

while(current->next != head)

current = current->next;

// chen link vao phan cuoi cua list

current->next = link;

// lien ket last node voi head

link->next = head;

}

void remove\_data(int data) {

int pos = 0;

if(head==NULL) {

printf("Danh sach lien ket chua duoc khoi tao");

return;

}

if(head->data == data) {

if(head->next != head) {

current = head;

while(current->next!=head) {

current = current->next;

}

current->next = head->next;

head = head->next;

return;

}else {

head = NULL;

printf("List la trong\n");

exit(0);

}

}else if(head->data != data && head->next == NULL) {

printf("Khong tim thay %d trong list\n", data);

return;

}

current = head;

while(current->next != head && current->data != data) {

prev = current;

current = current->next;

}

if(current->data == data) {

prev->next = prev->next->next;

free(current);

}else

printf("Khong tim thay %d trong list.", data);

}

//hien thi list

void printList() {

struct node \*ptr = head;

printf("\n[head] =>");

//bat dau tu phan dau cua list

while(ptr->next != head) {

printf(" %d =>",ptr->data);

ptr = ptr->next;

}

printf(" %d =>",ptr->data);

printf(" [head]\n");

}

int main() {

insert(10);

insert(20);

insert(30);

insert(1);

insert(40);

insert(56);

printf("List ban dau: ");

printList();

remove\_data(1);

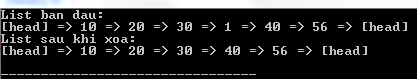
printf("List sau khi xoa: ");

printList();

return 0;

}

Biên dịch chương trình C trên sẽ cho kết quả:



**Bài 7:**

# Cách nối hai danh sách liên kết vòng trong C

[Trang trước](https://vietjack.com/bai-tap-c/danh-sach-lien-ket-trong-c.jsp)

[Trang sau](https://vietjack.com/bai-tap-c/danh-sach-lien-ket-trong-c.jsp)

## Bài tập C: nối hai danh sách liên kết vòng

Bài tập C này giúp bạn làm quen dần với cách tạo danh sách liên kết vòng và cách nối hai danh sách liên kết vòng trong C. Để giải bài tập này, mình sử dụng cấu trúc **struct** trong C.

## Chương trình C

Dưới đây là chương trình C để giải bài tập nối hai danh sách liên kết vòng trong C:

#include <stdio.h>

#include <stdlib.h>

struct node {

int data;

struct node \*next;

};

struct node \*even = NULL;

struct node \*odd = NULL;

struct node \*list = NULL;

//tao danh sach lien ket

void insert(int data) {

// cap phat bo nho cho node moi;

struct node \*link = (struct node\*) malloc(sizeof(struct node));

struct node \*current;

link->data = data;

link->next = NULL;

if(data%2 == 0) {

if(even == NULL) {

even = link;

even->next = link;

return;

}else {

current = even;

while(current->next != even)

current = current->next;

// chen link vao phan cuoi cua list

current->next = link;

link->next = even;

}

}else {

if(odd == NULL) {

odd = link;

odd->next = link;

return;

}else {

current = odd;

while(current->next != odd) {

current = current->next;

}

// chen link vao phan cuoi cua list

current->next = link;

link->next = odd;

}

}

}

void display(struct node \*head) {

int temp;

struct node \*ptr = head;

if(head == NULL) {

printf("Danh sach lien ket chua duoc khoi tao");

return;

}else if(head->next == head) {

printf("%d", head->data);

return;

}

printf("[head] =>");

while(ptr->next != head) {

printf(" %d =>",ptr->data);

ptr = ptr->next;

}

printf(" %d =>",ptr->data);

printf(" [head]\n");

}

void combine() {

struct node \*elink, \*olink;

list = even;

elink = list;

while(elink->next!= even) {

elink = elink->next;

}

elink->next = odd;

olink = odd;

while(olink->next!= odd) {

olink = olink->next;

}

olink->next = even;

}

int main() {

int i;

for(i=1; i<=10; i++)

insert(i);

printf("Danh sach chan: ");

display(even);

printf("Danh sach le: ");

display(odd);

combine();

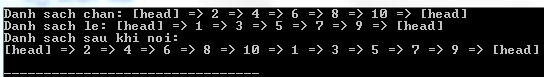
printf("Danh sach sau khi noi:\n");

display(list);

return 0;

}

Biên dịch chương trình C trên sẽ cho kết quả:



**Bài 8:**

# Cách chia một danh sách liên kết vòng trong C

[Trang trước](https://vietjack.com/bai-tap-c/danh-sach-lien-ket-trong-c.jsp)

[Trang sau](https://vietjack.com/bai-tap-c/danh-sach-lien-ket-trong-c.jsp)

## Bài tập C: chia một danh sách liên kết vòng

Bài tập C này giúp bạn làm quen dần với cách tạo danh sách liên kết vòng và cách chia một danh sách liên kết vòng trong C. Để giải bài tập này, mình sử dụng cấu trúc **struct** trong C.

## Chương trình C

Dưới đây là chương trình C để giải bài tập chia một danh sách liên kết vòng trong C:

#include <stdio.h>

#include <stdlib.h>

struct node {

int data;

struct node \*next;

};

struct node \*even = NULL;

struct node \*odd = NULL;

struct node \*list = NULL;

//tao danh sach lien ket

void insert(int data) {

// cap phat bo nho cho node moi;

struct node \*link = (struct node\*) malloc(sizeof(struct node));

struct node \*current;

link->data = data;

link->next = NULL;

if(list == NULL) {

list = link;

list->next = link;

return;

}

current = list;

while(current->next != list)

current = current->next;

// chen link vao phan cuoi cua list

current->next = link;

link->next = list;

}

void display(struct node \*head) {

struct node \*ptr = head;

printf("[head] =>");

//bat dau tu phan dau cua list

while(ptr->next != head) {

printf(" %d =>",ptr->data);

ptr = ptr->next;

}

printf(" %d =>",ptr->data);

printf(" [head]\n");

}

void split\_list() {

int count = 0;

// cap phat bo nho cho node moi;

struct node \*list1;

struct node \*link;

struct node \*current;

list1 = list;

while(list1->next != list) {

struct node \*link = (struct node\*) malloc(sizeof(struct node));

link->data = list1->data;

link->next = NULL;

if(list1->data%2 == 0) {

if(even == NULL) {

even = link;

even->next = link;

list1 = list1->next;

continue;

}else {

current = even;

while(current->next != even) {

current = current->next;

}

// chen link vao phan cuoi cua list

current->next = link;

link->next = even;

}

list1 = list1->next;

}else {

if(odd == NULL) {

odd = link;

odd->next = link;

list1 = list1->next;

continue;

}else {

current = odd;

while(current->next!= odd) {

current = current->next;

}

// chen link vao phan cuoi cua list

current->next = link;

link->next = odd;

}

list1 = list1->next;

}

}

// xu ly last node

link = (struct node\*) malloc(sizeof(struct node));

link->data = list1->data;

link->next = NULL;

if(list1->data%2 == 0) {

current = even;

while(current->next != even) {

current = current->next;

}

// chen link vao phan cuoi cua list

current->next = link;

link->next = even;

}else {

current = odd;

while(current->next!= odd) {

current = current->next;

}

// chen link vao phan cuoi cua list

current->next = link;

link->next = odd;

}

}

int main() {

int i;

for(i=1; i<=10; i++)

insert(i);

printf("Danh sach ban dau: \n");

display(list);

split\_list();

printf("\nDanh sach le: ");

display(odd);

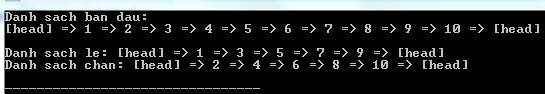
printf("Danh sach chan: ");

display(even);

return 0;

}

Biên dịch chương trình C trên sẽ cho kết quả:



**DSLK Đôi**

**Bài 1:**

# Tạo danh sách liên kết đôi trong C

[Trang trước](https://vietjack.com/bai-tap-c/danh-sach-lien-ket-trong-c.jsp)

[Trang sau](https://vietjack.com/bai-tap-c/danh-sach-lien-ket-trong-c.jsp)

## Bài tập C: Tạo danh sách liên kết đôi

Bài tập C này giúp bạn làm quen dần với cách tạo danh sách liên kết đôi trong C. Để giải bài tập này, mình sử dụng cấu trúc **struct** trong C.

## Chương trình C

Dưới đây là chương trình C để giải bài tập tạo danh sách liên kết đôi trong C:

#include <stdio.h>

#include <stdlib.h>

struct node {

int data;

struct node \*prev;

struct node \*next;

};

struct node \*head = NULL;

struct node \*last = NULL;

struct node \*current = NULL;

//tao danh sach lien ket

void insert(int data) {

// cap phat bo nho cho node moi;

struct node \*link = (struct node\*) malloc(sizeof(struct node));

link->data = data;

link->prev = NULL;

link->next = NULL;

// neu head la trong, tao list moi

if(head==NULL) {

head = link;

return;

}

current = head;

// di chuyen toi phan cuoi list

while(current->next!=NULL)

current = current->next;

// chen link vao phan cuoi cua list

current->next = link;

last = link;

link->prev = current;

}

//hien thi list

void printList() {

struct node \*ptr = head;

printf("\n[head] <=>");

//bat dau tu phan dau cua list

while(ptr->next != NULL) {

printf(" %d <=>",ptr->data);

ptr = ptr->next;

}

printf(" %d <=>",ptr->data);

printf(" [head]\n");

}

int main() {

insert(10);

insert(20);

insert(30);

insert(1);

insert(40);

insert(56);

printList();

return 0;

}

Biên dịch chương trình C trên sẽ cho kết quả:

Tạo danh sách liên kết đôi trong C

**Bài 2:**

# Hiển thị danh sách liên kết đôi theo chiều đảo ngược trong C

[Trang trước](https://vietjack.com/bai-tap-c/danh-sach-lien-ket-trong-c.jsp)

[Trang sau](https://vietjack.com/bai-tap-c/danh-sach-lien-ket-trong-c.jsp)

## Bài tập C: Hiển thị danh sách liên kết đôi theo chiều đảo ngược

Bài tập C này giúp bạn làm quen dần với cách tạo danh sách liên kết đôi và cách hiển thị danh sách liên kết đôi trong C. Để giải bài tập này, mình sử dụng cấu trúc **struct** trong C.

## Chương trình C

Dưới đây là chương trình C để giải bài tập hiển thị danh sách liên kết đôi theo chiều đảo ngược trong C:

#include <stdio.h>

#include <stdlib.h>

struct node {

int data;

struct node \*prev;

struct node \*next;

};

struct node \*head = NULL;

struct node \*last = NULL;

struct node \*current = NULL;

//hien thi list

void printList() {

struct node \*ptr = head;

printf("\n[head] <=>");

//bat dau tu phan dau cua list

while(ptr != NULL) {

printf(" %d <=>",ptr->data);

ptr = ptr->next;

}

printf(" [last]\n");

}

//hien thi list

void print\_backward() {

struct node \*ptr = last;

printf("\n[head] <=>");

//bat dau tu phan dau cua list

while(ptr != NULL) {

printf(" %d <=>",ptr->data);

ptr = ptr->prev;

}

printf(" [last]\n");

}

//tao danh sach lien ket

void insert(int data) {

// cap phat bo nho cho node moi;

struct node \*link = (struct node\*) malloc(sizeof(struct node));

link->data = data;

link->prev = NULL;

link->next = NULL;

// neu head la trong, tao list moi

if(head==NULL) {

head = link;

return;

}

current = head;

// di chuyen toi phan cuoi list

while(current->next!=NULL)

current = current->next;

// chen link vao phan cuoi cua list

current->next = link;

last = link;

link->prev = current;

}

int main() {

insert(10);

insert(20);

insert(30);

insert(1);

insert(40);

insert(56);

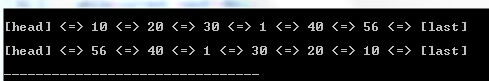
printList();

print\_backward();

return 0;

}

Biên dịch chương trình C trên sẽ cho kết quả:



**Bài 3:**

# Tìm kích cỡ của danh sách liên kết đôi trong C

[Trang trước](https://vietjack.com/bai-tap-c/danh-sach-lien-ket-trong-c.jsp)

[Trang sau](https://vietjack.com/bai-tap-c/danh-sach-lien-ket-trong-c.jsp)

## Bài tập C: Tìm kích cỡ của danh sách liên kết đôi

Bài tập C này giúp bạn làm quen dần với cách tạo danh sách liên kết đôi và cách tìm kích cỡ của danh sách liên kết đôi trong C. Để giải bài tập này, mình sử dụng cấu trúc **struct** trong C.

## Chương trình C

Dưới đây là chương trình C để giải bài tập tìm kích cỡ của danh sách liên kết đôi trong C:

#include <stdio.h>

#include <stdlib.h>

struct node {

int data;

struct node \*prev;

struct node \*next;

};

struct node \*head = NULL;

struct node \*last = NULL;

struct node \*current = NULL;

//hien thi list

void printList() {

struct node \*ptr = head;

printf("\n[head] <=>");

//bat dau tu phan dau cua list

while(ptr != NULL) {

printf(" %d <=>",ptr->data);

ptr = ptr->next;

}

printf(" [last]\n");

}

//tao danh sach lien ket

void insert(int data) {

// cap phat bo nho cho node moi;

struct node \*link = (struct node\*) malloc(sizeof(struct node));

link->data = data;

link->prev = NULL;

link->next = NULL;

// neu head la trong, tao list moi

if(head==NULL) {

head = link;

return;

}

current = head;

// di chuyen toi phan cuoi list

while(current->next!=NULL)

current = current->next;

// chen link vao phan cuoi cua list

current->next = link;

last = link;

link->prev = current;

}

**void sizeof\_list() {**

**int size = 0;**

**struct node \*link;**

**if(head==NULL) {**

**printf("\nKich co cua danh sach: %d\n", size);**

**return;**

**}**

**link = head;**

**while(link != NULL) {**

**link = link->next;**

**size++;**

**}**

**printf("\nKich co cua danh sach: %d\n", size);**

**}**

int main() {

insert(10);

insert(20);

insert(30);

insert(1);

insert(40);

insert(56);

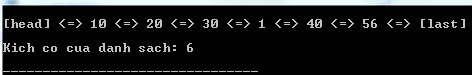
printList();

sizeof\_list();

return 0;

}

Biên dịch chương trình C trên sẽ cho kết quả:



**Bài 4:**

# Tìm kiếm phần tử trong danh sách liên kết đôi trong C

[Trang trước](https://vietjack.com/bai-tap-c/danh-sach-lien-ket-trong-c.jsp)

[Trang sau](https://vietjack.com/bai-tap-c/danh-sach-lien-ket-trong-c.jsp)

## Bài tập C: Tìm kiếm phần tử trong danh sách liên kết đôi

Bài tập C này giúp bạn làm quen dần với cách tạo danh sách liên kết đôi và cách tìm kiếm phần tử trong danh sách liên kết đôi trong C. Để giải bài tập này, mình sử dụng cấu trúc **struct** trong C.

## Chương trình C

Dưới đây là chương trình C để giải bài tập tìm kiếm phần tử trong danh sách liên kết đôi trong C:

#include <stdio.h>

#include <stdlib.h>

struct node {

int data;

struct node \*prev;

struct node \*next;

};

struct node \*head = NULL;

struct node \*last = NULL;

struct node \*current = NULL;

//hien thi list

void printList() {

struct node \*ptr = head;

printf("\n[head] <=>");

//bat dau tu phan dau cua list

while(ptr != NULL) {

printf(" %d <=>",ptr->data);

ptr = ptr->next;

}

printf(" [last]\n");

}

//tao danh sach lien ket

void insert(int data) {

// cap phat bo nho cho node moi;

struct node \*link = (struct node\*) malloc(sizeof(struct node));

link->data = data;

link->prev = NULL;

link->next = NULL;

// neu head la trong, tao list moi

if(head==NULL) {

head = link;

return;

}

current = head;

// di chuyen toi phan cuoi list

while(current->next!=NULL)

current = current->next;

// chen link vao phan cuoi cua list

current->next = link;

last = link;

link->prev = current;

}

**void search(int data) {**

**int pos = 0;**

**if(head==NULL) {**

**printf("Danh sach lien ket chua duoc khoi tao");**

**return;**

**}**

**current = head;**

**while(current!=NULL) {**

**pos++;**

**if(current->data == data) {**

**printf("Tim thay %d tai vi tri %d\n", data, pos);**

**return;**

**}**

**if(current->next != NULL)**

**current = current->next;**

**else**

**break;**

**}**

**printf("%d khong ton tai trong list\n", data);**

**}**

int main() {

insert(10);

insert(20);

insert(30);

insert(1);

insert(40);

insert(56);

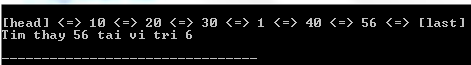
printList();

search(56);

return 0;

}

Biên dịch chương trình C trên sẽ cho kết quả:



**Bài 5:**

# Cập nhật một phần tử của danh sách liên kết đôi trong C

[Trang trước](https://vietjack.com/bai-tap-c/danh-sach-lien-ket-trong-c.jsp)

[Trang sau](https://vietjack.com/bai-tap-c/danh-sach-lien-ket-trong-c.jsp)

## Bài tập C: Cập nhật một phần tử của danh sách liên kết đôi

Bài tập C này giúp bạn làm quen dần với cách tạo danh sách liên kết đôi và cách cập nhật một phần tử của danh sách liên kết đôi trong C. Để giải bài tập này, mình sử dụng cấu trúc **struct** trong C.

## Chương trình C

Dưới đây là chương trình C để giải bài tập cập nhật một phần tử của danh sách liên kết đôi trong C:

#include <stdio.h>

#include <stdlib.h>

struct node {

int data;

struct node \*prev;

struct node \*next;

};

struct node \*head = NULL;

struct node \*last = NULL;

struct node \*current = NULL;

//hien thi list

void printList() {

struct node \*ptr = head;

printf("\n[head] <=>");

//bat dau tu phan dau cua list

while(ptr != NULL) {

printf(" %d <=>",ptr->data);

ptr = ptr->next;

}

printf(" [last]\n");

}

//tao danh sach lien ket

void insert(int data) {

// cap phat bo nho cho node moi;

struct node \*link = (struct node\*) malloc(sizeof(struct node));

link->data = data;

link->prev = NULL;

link->next = NULL;

// neu head la trong, tao list moi

if(head==NULL) {

head = link;

return;

}

current = head;

// di chuyen toi phan cuoi list

while(current->next!=NULL)

current = current->next;

// chen link vao phan cuoi cua list

current->next = link;

last = link;

link->prev = current;

}

**void update\_data(int old, int new1) {**

**int pos = 0;**

**if(head == NULL) {**

**printf("Danh sach lien ket chua duoc khoi tao");**

**return;**

**}**

**current = head;**

**while(current!=NULL) {**

**pos++;**

**if(current->data == old) {**

**current->data = new1;**

**printf("\nTim thay %d tai vi tri %d, duoc thay the voi %d\n", old, pos, new1);**

**return;**

**}**

**if(current->next != NULL)**

**current = current->next;**

**else**

**break;**

**}**

**printf("%d khong ton tai trong list\n", old);**

**}**

int main() {

insert(10);

insert(20);

insert(30);

insert(1);

insert(40);

insert(56);

printList();

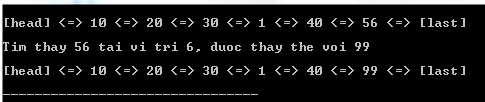
update\_data(56, 99);

printList();

return 0;

}

Biên dịch chương trình C trên sẽ cho kết quả:



**Bài 6:**

# Xóa một phần tử từ danh sách liên kết đôi trong C

[Trang trước](https://vietjack.com/bai-tap-c/danh-sach-lien-ket-trong-c.jsp)

[Trang sau](https://vietjack.com/bai-tap-c/danh-sach-lien-ket-trong-c.jsp)

## Bài tập C: Xóa một phần tử từ danh sách liên kết đôi

Bài tập C này giúp bạn làm quen dần với cách tạo danh sách liên kết đôi và cách xóa một phần tử từ danh sách liên kết đôi trong C. Để giải bài tập này, mình sử dụng cấu trúc **struct** trong C.

## Chương trình C

Dưới đây là chương trình C để giải bài tập xóa một phần tử từ danh sách liên kết đôi trong C:

#include <stdio.h>

#include <stdlib.h>

struct node {

int data;

struct node \*prev;

struct node \*next;

};

struct node \*head = NULL;

struct node \*last = NULL;

struct node \*current = NULL;

//hien thi list

void printList() {

struct node \*ptr = head;

printf("\n[head] <=>");

//bat dau tu phan dau cua list

while(ptr != NULL) {

printf(" %d <=>",ptr->data);

ptr = ptr->next;

}

printf(" [last]\n");

}

void print\_backward() {

struct node \*ptr = last;

printf("\n[last] <=>");

//bat dau tu phan dau cua list

while(ptr != NULL) {

printf(" %d <=>",ptr->data);

ptr = ptr->prev;

}

printf(" [head]\n");

}

//tao danh sach lien ket

void insert(int data) {

// cap phat bo nho cho node moi;

struct node \*link = (struct node\*) malloc(sizeof(struct node));

link->data = data;

link->prev = NULL;

link->next = NULL;

// neu head la trong, tao list moi

if(head==NULL) {

head = link;

return;

}

current = head;

// di chuyen toi phan cuoi list

while(current->next!=NULL)

current = current->next;

// chen link vao phan cuoi cua list

current->next = link;

last = link;

link->prev = current;

}

**void remove\_data(int data) {**

**int pos = 0;**

**struct node \*pre\_node;**

**if(head==NULL) {**

**printf("Danh sach lien ket chua duoc khoi tao");**

**return;**

**}**

**if(head->data == data) {**

**if(head->next != NULL) {**

**head->next->prev = NULL;**

**head = head->next;**

**return;**

**}else {**

**head = NULL;**

**printf("Bay gio List la trong");**

**return;**

**}**

**}else if(head->data != data && head->next == NULL) {**

**printf("Khong tim thay %d trong list\n", data);**

**return;**

**}**

**current = head;**

**while(current->next != NULL && current->data != data) {**

**pre\_node = current;**

**current = current->next;**

**}**

**if(current->data == data) {**

**pre\_node->next = pre\_node->next->next;**

**if(pre\_node->next != NULL) {**

**pre\_node->next->prev = pre\_node;**

**}else**

**last = pre\_node;**

**free(current);**

**}else**

**printf("Khong tim thay %d trong list.", data);**

**}**

int main() {

insert(10);

insert(20);

insert(30);

insert(1);

insert(40);

insert(56);

printList();

remove\_data(20);

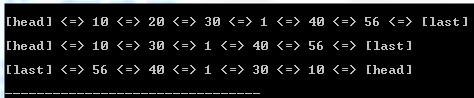
printList();

print\_backward();

return 0;

}

Biên dịch chương trình C trên sẽ cho kết quả:



**Bài 7:**

# Cách nối hai danh sách liên kết đôi trong C

[Trang trước](https://vietjack.com/bai-tap-c/danh-sach-lien-ket-trong-c.jsp)

[Trang sau](https://vietjack.com/bai-tap-c/danh-sach-lien-ket-trong-c.jsp)

## Bài tập C: nối hai danh sách liên kết đôi

Bài tập C này giúp bạn làm quen dần với cách tạo danh sách liên kết đôi và cách nối hai danh sách liên kết đôi trong C. Để giải bài tập này, mình sử dụng cấu trúc **struct** trong C.

## Chương trình C

Dưới đây là chương trình C để giải bài tập nối hai danh sách liên kết đôi trong C:

#include <stdio.h>

#include <stdlib.h>

struct node {

int data;

struct node \*prev;

struct node \*next;

};

struct node \*list = NULL;

struct node \*list\_last = NULL;

struct node \*even = NULL;

struct node \*even\_last = NULL;

struct node \*odd = NULL;

struct node \*odd\_last = NULL;

struct node \*current = NULL;

//tao danh sach lien ket

void insert(int data) {

// cap phat bo nho cho node moi;

struct node \*link = (struct node\*) malloc(sizeof(struct node));

link->data = data;

link->prev = NULL;

link->next = NULL;

// neu head la trong, tao list moi

if(data%2 == 0) {

if(even == NULL) {

even = link;

return;

}else {

current = even;

while(current->next != NULL)

current = current->next;

// chen link vao phan cuoi cua list

current->next = link;

even\_last = link;

link->prev = current;

}

}else {

if(odd == NULL) {

odd = link;

return;

}else {

current = odd;

while(current->next!=NULL)

current = current->next;

// chen link vao phan cuoi cua list

current->next = link;

odd\_last = link;

link->prev = current;

}

}

}

//hien thi list

void print\_backward(struct node \*head) {

struct node \*ptr = head;

printf("\n[last] <=>");

//bat dau tu phan dau cua list

while(ptr != NULL) {

printf(" %d <=>",ptr->data);

ptr = ptr->prev;

}

printf(" [head]\n");

}

//hien thi list

void printList(struct node \*head) {

struct node \*ptr = head;

printf("\n[head] <=>");

//bat dau tu phan dau cua list

while(ptr != NULL) {

printf(" %d <=>",ptr->data);

ptr = ptr->next;

}

printf(" [last]\n");

}

void combine() {

struct node \*link;

list = even;

link = list;

while(link->next!= NULL) {

link = link->next;

}

link->next = odd;

odd->prev = link;

// gan link\_last toi last node cua list moi

while(link->next!= NULL) {

link = link->next;

}

list\_last = link;

}

int main() {

int i;

for(i=1; i<=10; i++)

insert(i);

printf("Danh sach chan: ");

printList(even);

printf("Chieu dao nguoc Danh sach chan: ");

print\_backward(even\_last);

printf("Danh sach le: ");

printList(odd);

printf("Chieu dao nguoc Danh sach le: ");

print\_backward(odd\_last);

combine();

printf("Danh sach sau khi noi: \n");

printList(list);

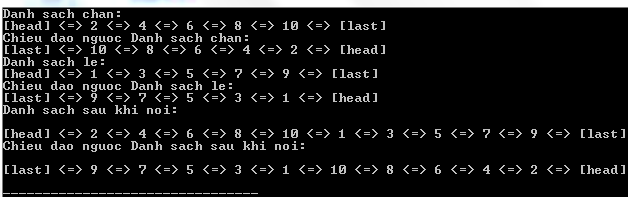
printf("Chieu dao nguoc Danh sach sau khi noi:\n");

print\_backward(list\_last);

return 0;

}

Biên dịch chương trình C trên sẽ cho kết quả:



**Bài 8:**

# Cách chia một danh sách liên kết đôi trong C

[Trang trước](https://vietjack.com/bai-tap-c/danh-sach-lien-ket-trong-c.jsp)

[Trang sau](https://vietjack.com/bai-tap-c/danh-sach-lien-ket-trong-c.jsp)

## Bài tập C: chia một danh sách liên kết đôi

Bài tập C này giúp bạn làm quen dần với cách tạo danh sách liên kết đôi và cách chia một danh sách liên kết đôi trong C. Để giải bài tập này, mình sử dụng cấu trúc **struct** trong C.

## Chương trình C

Dưới đây là chương trình C để giải bài tập chia một danh sách liên kết đôi trong C:

#include <stdio.h>

#include <stdlib.h>

#include <stdio.h>

#include <stdlib.h>

struct node {

int data;

struct node \*prev;

struct node \*next;

};

struct node \*list = NULL;

struct node \*list\_last = NULL;

struct node \*even = NULL;

struct node \*even\_last = NULL;

struct node \*odd = NULL;

struct node \*odd\_last = NULL;

struct node \*current = NULL;

//tao danh sach lien ket

void insert(int data) {

// cap phat bo nho cho node moi;

struct node \*link = (struct node\*) malloc(sizeof(struct node));

link->data = data;

link->prev = NULL;

link->next = NULL;

// neu head la trong, tao list moi

if(list == NULL) {

list = link;

return;

}

current = list;

// di chuyen toi phan cuoi list

while(current->next!=NULL)

current = current->next;

// chen link vao phan cuoi cua list

current->next = link;

list\_last = link;

link->prev = current;

}

//hien thi list

void print\_backward(struct node \*head) {

struct node \*ptr = head;

printf("\n[last] <=>");

//bat dau tu phan dau cua list

while(ptr != NULL) {

printf(" %d <=>",ptr->data);

ptr = ptr->prev;

}

printf(" [head]\n");

}

//hien thi list

void printList(struct node \*head) {

struct node \*ptr = head;

printf("\n[head] <=>");

//bat dau tu phan dau cua list

while(ptr != NULL) {

printf(" %d <=>",ptr->data);

ptr = ptr->next;

}

printf(" [last]\n");

}

void split\_list() {

// cap phat bo nho cho node moi;

struct node \*listp;

struct node \*link;

struct node \*current;

listp = list;

while(listp->next != NULL) {

struct node \*link = (struct node\*) malloc(sizeof(struct node));

link->data = listp->data;

link->prev = NULL;

link->next = NULL;

if(listp->data%2 == 0) {

if(even == NULL) {

even = link;

even\_last = link;

listp = listp->next;

continue;

}else {

current = even;

while(current->next != NULL) {

current = current->next;

}

// chen link vao phan cuoi cua list

current->next = link;

even\_last = link;

link->prev = current;

listp = listp->next;

}

}else {

if(odd == NULL) {

odd = link;

odd\_last = link;

listp = listp->next;

continue;

}else {

current = odd;

while(current->next!= NULL) {

current = current->next;

}

// chen link vao phan cuoi cua list

current->next = link;

odd\_last = link;

link->prev = current;

listp = listp->next;

}

}

}

// xu ly last node

if(listp!=NULL) {

link = (struct node\*) malloc(sizeof(struct node));

link->data = listp->data;

link->prev = NULL;

link->next = NULL;

if(listp->data%2 == 0) {

current = even;

while(current->next != NULL) {

current = current->next;

}

// chen link vao phan cuoi cua list

current->next = link;

even\_last = link;

link->prev = current;

}else {

current = odd;

while(current->next!= NULL) {

current = current->next;

}

// chen link vao phan cuoi cua list

current->next = link;

odd\_last = link;

link->prev = current;

}

}

}

int main() {

int i;

for(i=1; i<=10; i++)

insert(i);

printf("Danh sach ban dau: ");

printList(list);

printf("Danh sach voi chieu dao nguoc: ");

print\_backward(list\_last);

split\_list();

printf("\nSau khi chia danh sach: \n");

printf("Danh sach chan: ");

printList(even);

printf("Danh sach le: ");

printList(odd);

printf("\nHien thi danh sach vua chia theo chieu dao nguoc: \n");

printf("Danh sach chan: ");

print\_backward(even\_last);

printf("Danh sach le: ");

print\_backward(odd\_last);

return 0;

}

Biên dịch chương trình C trên sẽ cho kết quả:

