Bài 10: Linked List

# Khái niệm

Linked list là một cấu trúc dữ liệu trong lập trình máy tính, được sử dụng để tổ chức và lưu trữ dữ liệu. Một linked list bao gồm một chuỗi các "nút" (nodes), mỗi nút chứa một giá trị dữ liệu và một con trỏ (pointer) đến nút tiếp theo trong chuỗi.

Có hai loại linked list chính:

* Singly Linked List (Danh sách liên kết đơn): Mỗi nút chỉ chứa một con trỏ đến nút tiếp theo trong chuỗi.
* Doubly Linked List (Danh sách liên kết đôi): Mỗi nút chứa hai con trỏ, một trỏ đến nút tiếp theo và một trỏ đến nút trước đó.

Một linked list cung cấp một cách linh hoạt để thêm, xóa và chèn các phần tử mà không cần phải di chuyển toàn bộ dãy số như mảng. Tuy nhiên, nó cũng có một số nhược điểm, như việc cần thêm một con trỏ cho mỗi nút, tăng độ phức tạp của bộ nhớ và có thể dẫn đến hiệu suất kém hơn trong một số trường hợp so với mảng.

Dưới đây là định nghĩa 1 node trong singly linked list:

| **struct** **Node** {  **int** data; // Dữ liệu của nút  Node\* next; // Con trỏ đến nút tiếp theo trong danh sách }; |
| --- |

Một số thao tác trong list:

Khởi tạo một node:

| node \***createNode**(**int** value) {  node \*ptr = (node\*)malloc(**sizeof**(node));  ptr->value = value;  ptr->next = NULL;  **return** ptr; } |
| --- |

Chèn một node vào cuối list:

| **void** **push\_back**(node\*\* array, **int** value) {  node \*temp, \*p;  temp = createNode(value);    p = \*array; // use p instead of array because we are using pointer, use array will change the structure of linkedlist   **if** (\*array == NULL) // if array doesn't have any node yet  {    \*array = temp;  }  **else** // if array has some node  {   **while** (p->next != NULL) // which mean the current node is not the last node  {  p = p->next; // check next node until it a last node    }    p->next = temp; // change it next member point to address of new node have just create  } } |
| --- |

Xóa node cuối cùng của list:

| **void** **pop\_back**(node \*\*array) {  node \*p, \*temp;  p = \*array;  **int** i = 0; // to    **while** (p->next != NULL) // free the last node in the list  {  p = p->next;  i++;  }  free(p);   temp = \*array;  **int** j;  **for** ( j = 0; j < i - 1 ; j++) // change the next member of the node at the (last - 1) position to NULL  {  temp = temp->next;  }    temp->next = NULL; } |
| --- |