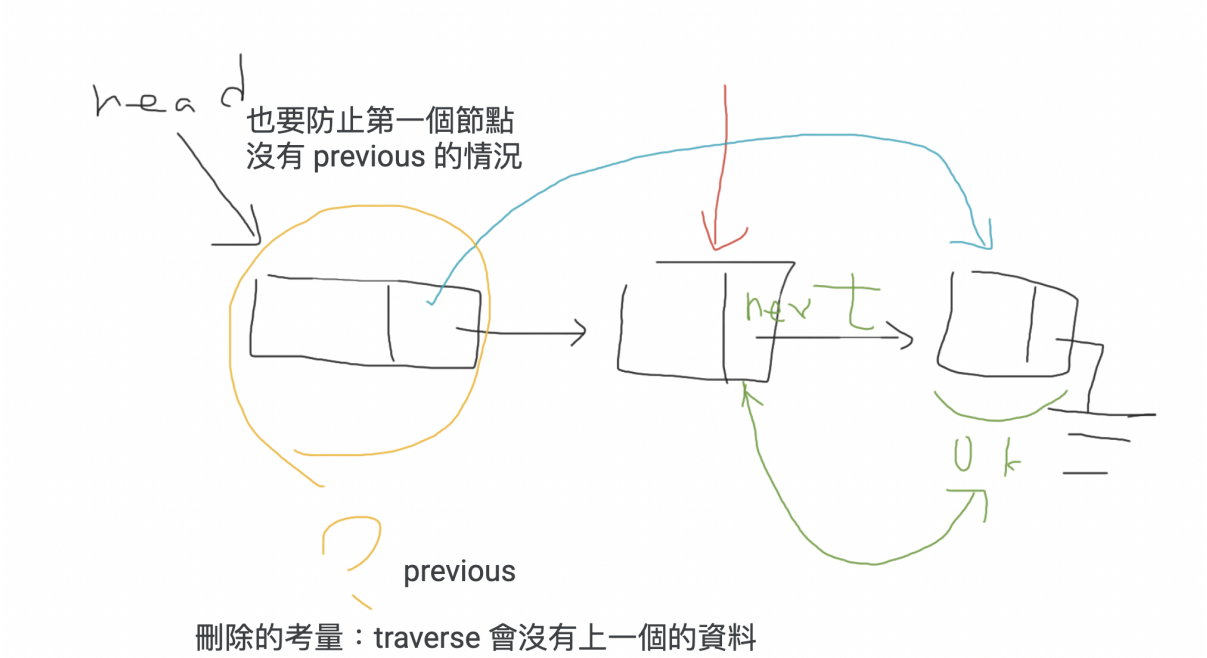
Phill-DS-0913

```
#include <iostream>
using namespace std;

struct Node{
  int data;
  Node* next;
};


class LinkedList{
  private:
    Node* head;

  public:
    LinkedList(){
      head=nullptr;
    }

    void add(int data){
      Node* newNode = new Node{data, nullptr};
      if(head==nullptr){ //串列是空的
        head=newNode;
      }
      else{ //有 node inside
        Node* current=head; //產生新的魁儡變數 current
        while(current->next!=nullptr){ //還沒到底
          current = current -> next; //連下一個
        }
        current -> next = newNode;//把自己接到尾巴後面
      }
    }

    void print(){
      Node* current=head;
      while(current != nullptr){
          cout << current->data << "->";
          current = current -> next;
      }
      cout << "nullptr" << endl;
    }

    Node* search(int data){
      Node* current = head;
      while(current !=nullptr){
        if(current->data==data){ // hit data
          return current; //把節點傳回去
        }
        current = current ->next; //繼續找
      }
      return nullptr; //什麼都沒找到, 傳 null
    }
```

```
    void update(int oldData, int newData){
      Node* node = search(oldData); //接住找到的節點
      if(node != nullptr){ // 檢查清楚 防止沒找到的情況 !!
        node-> data = newData; // in lvalue -> 賦值
      }
    } // return bool 當作業

    void remove(int data){
      Node* current = head;
      Node* previous = nullptr;

      while(current != nullptr){
        if(current -> data == data){ //找到了
          if( previous== nullptr){ //代表現在還沒 bookeeping, 是頭節點
              head = current-> next; //直接把頭指到我的下一個 因我要被刪掉
          }
          else{ //不是頭節點的狀況 做連接手術
              previous->next = current -> next; //刪除 接續 把上一個直接接到下一個
          }
          delete current; //處理自己空間的釋放
          return; //返回
        }
        previous = current; //把自己記起來 因為我是下一個人的 previous, bookeeping
        current = current -> next; //放心繼續找下去
      }
    } // bool version -> 當作業

    ~LinkedList(){
      Node* current =head;
      while(current!=nullptr){
          Node* temp = current; //先紀錄我的位址
          current = current -> next; //先把魁儡變數移到下一個人 避免我被release 掉消失
          delete temp; // 釋放自己
      }
    }
};

int main()
{
    LinkedList list;
    list.add(10);
    list.add(20);
    list.add(30);

    list.print();//加成功了

    list.update(20, 25);
    list.print();

    list.remove(10); //試特殊的頭
    list.print();
    return 0;
}
```



練習 怎麼做 ok → 怎麼用

Linked list 來做工程師考績 (HR)

- 姓名
- 工號 (key) → 避免兩條鏈
- add, update, remove, print