Danh sách liên kết

1. Danh sách liên kết đơn

Ảnh có chứa văn bản

Mô tả được tạo tự động

VD: struct Node{

int data;

Node\* next;

};

1. Danh sách liên kết đôi:

Ảnh có chứa văn bản

Mô tả được tạo tự động

VD: struct tagNode{  
int data;  
tagNode \*pNext, \*pPrew;  
}

1. Danh sách liên kết đơn vòng:

Ảnh có chứa văn bản

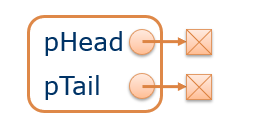
Mô tả được tạo tự động

1. Danh sách liên kết kép vòng:

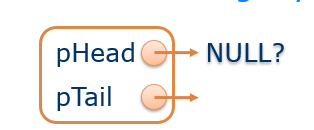
Ảnh có chứa văn bản

Mô tả được tạo tự động

1. Các khái niệm cơ bản:
2. Khởi tạo:



1. Kiểm tra danh sách có rỗng:



1. Tạo nút mới:

Ảnh có chứa văn bản, màn hình, ảnh chụp màn hình

Mô tả được tạo tự động

1. Thêm vào đầu:

Ảnh có chứa văn bản

Mô tả được tạo tự động

1. Thêm vào cuối:

Ảnh có chứa văn bản

Mô tả được tạo tự động

1. Chèn vào y vào sau vị trí x:

void addNextX(Node\* &head, int x, int y){ // chèn vào sau vị trí x

    Node\* p = newNode(y);

    Node\* i = head;

    while(i->next != NULL){

        if(x == 0){

            p->next = i->next;

            i->next = p;

            return;

        }

        x--;

        i = i->next;

    }

}

1. Chèn vào vị trí x:

void addX(Node\* &head, int x, int y){ // chèn vào vị trí x

    Node\* p = newNode(y);

    Node\* i = head;

    while(i->next != NULL){

        if(x == 0){

            if(i == head){

                p->next = head;

                head = p;

            }else{

                p->next = i->next;

                i->next = p;

            }

            return;

        }

        x--;

        i = i->next;

    }

}

void addX\_2(Node\* &head, int x, int y){

    Node\* q = NULL;

    Node\* p = newNode(y);

    Node\* i = head;

    while(i != NULL){

        if( x == 0){

            if(q == NULL){

                head = p;

            }else{

                p->next = q->next;

                q->next = p;

            }

            return;

        }

        q = i;

        i = i->next;

    }

}

1. Xóa phần tử vị trí x:

void deleteX(Node\* &head, int x){

    Node\* i = head;

    while(i != NULL){

        if(x == 0){

            if(i == head){

                head = head->next;

            }else

                i->next = i->next->next;

            return;

        }

        x--;

        i = i->next;

    }

}

1. Xoá phần tử ở sau vị trí x:

void deleteNextX(Node\* &head, int x){

    Node\* i = head;

    while(i->next != NULL){

        if(x == 0){

            i->next = i->next->next;

            return;

        }

        x--;

        i = i->next;

    }

}

1. Lấy giá trị tại vị trí x:

int value(Node\* head, int x){

    Node\* i = head;

    while(i->next != NULL){

        if(x == 0){

            return i->data;

        }

        x--;

        i = i->next;

    }

}

1. Xuất:

void Print(Node\* head){

    Node\* i = head;

    while(i != NULL){

        cout << i->data << " ";

        i = i->next;

    }

    cout << endl;

}

FULL CODE:

#include <iostream>

using namespace std;

struct Node{

    int data;

    Node \*next;

};

Node\* newNode(int x){ // tạo node

    Node\* p = new Node;

    p->data = x;

    p->next = NULL;

    return p;

}

Node\* addFirst(Node\* head, int x){ // Thêm vào đầu

    Node\* p = newNode(x);

    p->next = head;

    return p;

}

void addLast(Node\* &head, int x){ // thêm vào cuối

    Node\* p = newNode(x);

    if(head == NULL){

        head = p;

        return;

    }

    Node\* i = head;

    while(i->next != NULL)

        i = i->next;

    i->next = p;

}

void addNextX(Node\* &head, int x, int y){ // chèn vào sau vị trí x

    Node\* p = newNode(y);

    Node\* i = head;

    while(i->next != NULL){

        if(x == 0){

            p->next = i->next;

            i->next = p;

            return;

        }

        x--;

        i = i->next;

    }

}

void addX(Node\* &head, int x, int y){ // chèn vào vị trí x

    Node\* p = newNode(y);

    Node\* i = head;

    while(i->next != NULL){

        if(x == 0){

            if(i == head){

                p->next = head;

                head = p;

            }else{

                p->next = i->next;

                i->next = p;

            }

            return;

        }

        x--;

        i = i->next;

    }

}

void addX\_2(Node\* &head, int x, int y){

    Node\* q = NULL;

    Node\* p = newNode(y);

    Node\* i = head;

    while(i != NULL){

        if( x == 0){

            if(q == NULL){

                head = p;

            }else{

                p->next = q->next;

                q->next = p;

            }

            return;

        }

        q = i;

        i = i->next;

    }

}

void deleteNextX(Node\* &head, int x){

    Node\* i = head;

    while(i->next != NULL){

        if(x == 0){

            i->next = i->next->next;

            return;

        }

        x--;

        i = i->next;

    }

}

void deleteX(Node\* &head, int x){

    Node\* i = head;

    while(i != NULL){

        if(x == 0){

            if(i == head){

                head = head->next;

            }else

                i->next = i->next->next;

            return;

        }

        x--;

        i = i->next;

    }

}

int value(Node\* head, int x){

    Node\* i = head;

    while(i->next != NULL){

        if(x == 0){

            return i->data;

        }

        x--;

        i = i->next;

    }

}

void Print(Node\* head){

    Node\* i = head;

    while(i != NULL){

        cout << i->data << " ";

        i = i->next;

    }

    cout << endl;

}

int main(){

    Node\* head = NULL;

    head = addFirst(head, 5);

    Print(head);

    addLast(head, 6);

    addLast(head, 7);

    Print(head);

    addNextX(head, 1, 1);

    Print(head);

    addX(head, 0, 0);

    addX(head, 1, 2);

    Print(head);

    deleteNextX(head, 0);

    Print(head);

    deleteNextX(head, 2);

    Print(head);

    deleteX(head, 3);

    Print(head);

    cout << "gia tri tai vi tri 2 la: " << value(head, 1) << endl;

    return 0;

}

Thư viện: