Thời gian còn lại 47:56:41

Top of Form

**Câu hỏi 1**

Chính xác

Điểm 1,00 của 1,00

Cờ câu hỏi

**Nội dung câu hỏi**

Cho chương trình khởi tạo, trong đó:  
- struct node: đại diện cho một node của linked list  
- Hàm print: nhận vào con trỏ head của linked list và in ra từng phần tử của linked list  
- Hàm createLinkedList:  
    + Nhận vào số phần tử (>0) của link list  
    + Xây dựng một linked list với dữ liệu của các node được nhập từ đầu vào chuẩn (stdin), node mới được thêm vào **CUỐI**linked list  
    + Trả về con trỏ đến node đầu tiên của linked list.  
- Hàm main đọc vào số phần tử của linked list, gọi hàm createLinkedList để khởi tạo linked list, sau đó gọi hàm print để in ra các phần tử của linked list.  
Bạn hãy hoàn thành hàm createLinkedList  
**Đầu vào:**Số n là số lượng phần tử trong linked list (0 < n < 5000)  
n số tiếp theo là giá trị của mỗi node trong linked list, giá trị là một số nguyên có giá trị trong khoảng (-5000; 5000)  
**Đầu ra:**Thỏa yêu cầu bài toán  
Write a program where:  
- struct node: represents a node in linkedlists  
- functions createLinkedList:  
    + Receive the size of a linked list (>0)  
    + create a linked list with value from standard input (stdin) a new node is add to the  **END** of the linked list  
    + Return a pointer which points to the first node of the linked list.  
- Function main reads the size of the linked list, calls function createLinkedList to Initialize the linked list, then call function print to print the linked list.  
Complete function createLinkedList  
**Input:**  
size n of the linked list(0 < n < 5000)  
following n numbers are values of each node in the linked list each number is an integers in (-5000; 5000)  
**Input:**  
Satisfy the requirements  
**Template**:

Bottom of Form

| **Test** | **Input** | **Result** |
| --- | --- | --- |
| 1 | 5  1 3 5 7 9 | 1  3  5  7  9 |

#include <iostream>

using namespace std;

struct node

{

int data;

node \*next;

};

node \*createLinkedList(int n)

{

if (n <= 0)

{

return nullptr; // Không có phần tử nào, trả về nullptr

}

node \*head = nullptr; // Con trỏ đến đầu danh sách liên kết

node \*tail = nullptr; // Con trỏ đến phần tử cuối cùng của danh sách liên kết

for (int i = 0; i < n; i++)

{

int value;

cin >> value; // Nhập giá trị cho node mới

node \*newNode = new node; // Tạo node mới

newNode->data = value; // Gán giá trị cho node mới

newNode->next = nullptr; // Thiết lập con trỏ next của node mới thành nullptr

if (head == nullptr) // Nếu danh sách liên kết đang rỗng

{

head = newNode; // Node mới sẽ là đầu danh sách liên kết

tail = newNode; // Node mới cũng là phần tử cuối cùng

}

else

{

tail->next = newNode; // Gán con trỏ next của phần tử cuối cùng hiện tại tới node mới

tail = newNode; // Cập nhật phần tử cuối cùng mới là node mới

}

}

return head; // Trả về con trỏ đến đầu danh sách liên kết

}

void print(node \*head)

{

while (head != nullptr)

{

cout << head->data << endl;

head = head->next;

}

}

int main()

{

int n = 0;

cin >> n;

if (n > 0)

{

node \*head = createLinkedList(n);

print(head);

}

else

{

cout << "Invalid n" << endl;

}

return 0;

}

Cho chương trình khởi tạo, trong đó:  
- struct node: đại diện cho một node của linked list  
- Hàm print: nhận vào con trỏ head của linked list và in ra từng phần tử của linked list  
- Hàm createLinkedList:  
    + Nhận vào số phần tử (>0) của link list  
    + Xây dựng một linked list với dữ liệu của các node được nhập từ đầu vào chuẩn (stdin), node mới được thêm vào **CUỐI**linked list  
    + Trả về con trỏ đến node đầu tiên của linked list.  
- Hàm insertNode  
    + Nhận vào con trỏ head của linked list, con trỏ của một node mới, và vị trí position  
    + Hàm sẽ chèn node mới này vào vị trí position (lấy vị trí của node head là 1). Nếu position <=0, hàm không làm gì cả. Nếu position lớn hơn số phần tử hiện tại của linked list thì node mới được chèn vào cuối của linked list.  
- Hàm main đọc vào số phần tử của linked list, gọi hàm createLinkedList để khởi tạo linked list, sau đó gọi hàm print để in ra các phần tử của linked list.  
Bạn hãy hoàn thành hàm **insertNode**  
**Đầu vào**:   
Các giá trị nhập vào từ đầu vào chuẩn (stdin) đều có giá trị trong khoảng (0; 5000)  
**English version**Write a program where:  
- struct node: represents a node in linkedlists  
- functions createLinkedList:  
    + Receive the size of a linked list (>0)  
    + create a linked list with value from standard input (stdin) a new node is add to the  **END** of the linked list  
    + Return a pointer which points to the first node of the linked list.  
- Function insertNode  
    + receive the linked list's head pointers, a pointer of a new nodei, the position  
    + The function will insert the new node to the input position(head's position is 1). If position <=0, do nothing. If position is greater than the size of the linked list, insert to the end of the linked list.  
- function main reads the size of the linked list, calls function createLinkedList to initialize the linked list, then call function print to print the linked list.  
Complete functions **insertNode**  
**Input**:   
Value from standards input (stdin) with value in (0; 5000)

**For example:**

| **Test** | **Input** | **Result** |
| --- | --- | --- |
| 1 | 5  1 3 5 7 9  2 4 | 1  3  5  2  7  9 |

#include <iostream>

using namespace std;

struct node

{

int data;

node \*next;

};

node \*createLinkedList(int n); // The implementation is provided implicitly

node \*insertNode(node \*head, node \*newNode, int position)

{

if (position <= 0)

{

// Nếu vị trí <= 0, không làm gì cả và trả về node đầu danh sách liên kết ban đầu

return head;

}

if (head == nullptr)

{

// Nếu danh sách liên kết rỗng, node mới sẽ trở thành đầu danh sách liên kết

return newNode;

}

if (position == 1)

{

// Nếu vị trí là 1, node mới sẽ trở thành node đầu danh sách liên kết

newNode->next = head;

return newNode;

}

// Tìm vị trí node trước vị trí cần chèn (số nút trước khi chèn)

node \*prevNode = head;

int count = 1;

while (count < position - 1 && prevNode->next != nullptr)

{

prevNode = prevNode->next;

count++;

}

// Chèn node mới vào sau prevNode

newNode->next = prevNode->next;

prevNode->next = newNode;

return head;

}

void print(node \*head)

{

while (head != nullptr)

{

cout << head->data << endl;

head = head->next;

}

}

int main()

{

int n = 0;

cin >> n;

node \*head = createLinkedList(n);

node \*newNode = new node();

cin >> newNode->data;

int position = 0;

cin >> position;

head = insertNode(head, newNode, position);

print(head);

return 0;

}

Cho chương trình khởi tạo, trong đó:  
- struct node: đại diện cho một node của linked list  
- Hàm print: nhận vào con trỏ head của linked list và in ra từng phần tử của linked list  
- Hàm createLinkedList:  
    + Nhận vào số phần tử (>0) của link list  
    + Xây dựng một linked list với dữ liệu của các node được nhập từ đầu vào chuẩn (stdin), node mới được thêm vào **CUỐI**linked list  
    + Trả về con trỏ đến node đầu tiên của linked list.  
- Hàm replace:  
    + Nhận vào con trỏ head của linked list  
    + Vị trí position muốn thay thế (tính từ index 0)  
    + Giá trị value  
    + Hàm replace sẽ thay thế giá trị tại vị trí position bằng giá trị value. Nếu position lớn hơn hoặc bằng số lượng node hoặc position âm thì hàm replace không làm gì cả.  
- Hàm main đọc vào số phần tử của linked list, gọi hàm createLinkedList để khởi tạo linked list, sau đó gọi hàm print để in ra các phần tử của linked list.  
Bạn hãy hoàn thành hàm **replace**  
**Đầu vào:**Tất cả giá trị từ đầu vào chuẩn (stdin) đều nằm trong khoảng (0, 5000). Riêng giá trị của position có thể âm  
**Đầu ra:**Thỏa yêu cầu bài toán  
**English version**Write a program where:  
- struct node: represents a node in linkedlists  
- functions createLinkedList:  
    + Receive the size of a linked list (>0)  
    + create a linked list with value from standard input (stdin) a new node is add to the  **END** of the linked list  
    + Return a pointer which points to the first node of the linked list.  
- Function replace:  
    + receive the head pointer of the linked list  
    + position that needed to be replaced (count from 0)  
    + new Value  
    + function replace will replace the value at input position to the new Value. If position is equal or greater than the size of the linked list or negative, do nothing.  
- Function main reads the linked list's size, calls function createLinkedList to initialize the linked list, then calls function print to print the linked list.  
Complete functions **replace**  
**Input:**All the input from standard input (stdin) with value in (0, 5000). Except position can be negative  
**Output:**Satisfy the requirements

**For example:**

| **Test** | **Input** | **Result** |
| --- | --- | --- |
| 1 | 5  1 3 5 7 9  0 10 | 1 3 5 7 9  10 3 5 7 9 |

#include <iostream>

using namespace std;

struct node

{

int data;

node \*next;

};

node \*createLinkedList(int n); // The implementation is provided implicitly

void replace(node\* head, int position, int value)

{

if (position < 0)

{

// Nếu vị trí âm, không làm gì cả

return;

}

node\* currentNode = head;

int count = 0;

// Tìm node tại vị trí cần thay thế

while (count < position && currentNode != nullptr)

{

currentNode = currentNode->next;

count++;

}

if (currentNode != nullptr)

{

// Thay thế giá trị của node tại vị trí cần thay thế

currentNode->data = value;

}

}

void print(node \*head)

{

while (head != nullptr)

{

cout << head->data << " ";

head = head->next;

}

cout<<endl;

}

int main()

{

int n = 0;

cin >> n;

node \*head = createLinkedList(n);

print(head);

int pos, val;

cin>>pos>>val;

replace(head, pos, val);

print(head);

return 0;

}

Cho chương trình khởi tạo, trong đó:  
- struct node: đại diện cho một node của linked list  
- Hàm print: nhận vào con trỏ head của linked list và in ra từng phần tử của linked list  
- Hàm createLinkedList:  
    + Nhận vào số phần tử (>0) của link list  
    + Xây dựng một linked list với dữ liệu của các node được nhập từ đầu vào chuẩn (stdin), node mới được thêm vào **CUỐI**linked list  
    + Trả về con trỏ đến node đầu tiên của linked list.  
- Hàm searchLinkedList:   
    + Nhận vào con trỏ head của linked list  
    + Giá trị cần tìm  
    + Nếu tìm thấy thì trả về vị trí đầu tiên của nó trong Linked List (index từ 0), không thì trả về -1.  
- hàm main đọc vào số phần tử của linked list, gọi hàm createLinkedList để khởi tạo linked list, sau đó gọi hàm print để in ra các phần tử của linked list.  
Bạn hãy hoàn thành hàm **searchLinkedList**  
**Đầu vào:**Các giá trị từ đầu vào chuẩn đều có giá trị trong khoảng từ (0; 5000)  
**English version**Write a program where:  
- struct node: represents a node in linkedlists  
- functions createLinkedList:  
    + Receive the size of a linked list (>0)  
    + create a linked list with value from standard input (stdin) a new node is add to the  **END** of the linked list  
    + Return a pointer which points to the first node of the linked list.  
- function searchLinkedList:   
    + receive the linked list's head pointer  
    + The searching Value  
    + If found, return the first position the value appeared in the Linked List (index start from 0), otherwise -1.  
- function main read the linked list's size, calls function createLinkedList to initialize the linked list, then calls function print to print the linked list.  
Complete functions **searchLinkedList**  
**Input:**All the input from standard input (stdin) with value in (0; 5000)

**For example:**

| **Test** | **Input** | **Result** |
| --- | --- | --- |
| 1 | 5  1 3 5 7 9  3 | 1  3  5  7  9  1 |

#include <iostream>

using namespace std;

struct node

{

int data;

node \*next;

};

node \*createLinkedList(int n); // The implementation is provided implicitly

int searchLinkedList(node\* head, int key)

{

int position = 0;

node\* currentNode = head;

while (currentNode != nullptr)

{

if (currentNode->data == key)

{

// Nếu tìm thấy giá trị trong danh sách liên kết, trả về vị trí

return position;

}

currentNode = currentNode->next;

position++;

}

// Nếu không tìm thấy giá trị, trả về -1

return -1;

}

void print(node \*head)

{

while (head != nullptr)

{

cout << head->data << endl;

head = head->next;

}

}

int main()

{

int n = 0;

cin >> n;

node \*head = createLinkedList(n);

print(head);

int m;

cin>>m;

cout<<searchLinkedList(head,m);

return 0;

}