Lab2 Solutions

Sorted Linked List

ใช้ linked list เพื่อเก็บข้อมูล ในการเพิ่มข้อมูล จะแทรกก่อน node แรกที่มีเลขมากกว่าเลขที่ต้องการเพิ่ม ทำได้โดยการ loop หาว่าค่าของ node ต่อไปมีค่ามากกว่าหรือไม่ แล้วเปลี่ยนแปลง pointer ที่เก็บที่อยู่ของ node ต่อไปของ node ปัจจุบัน และ node ใหม่ที่ต้องการเพิ่ม

Example Code

```
#include <bits/stdc++.h>
using namespace std;

typedef struct _node{
    int num;
    struct _node *next;
} Node;

Node* newNode(int num)
{
    Node* newnode = (Node*)malloc(sizeof(Node));
    newnode->num = num;
    newnode->next = NULL;
    return newnode;
}

Node* head = NULL;
void insertNode(int num)
{
    Node* newnode = newNode(num);
    if(head == NULL) head = newNode(num);
    else if(num <= head->num)
    {
        newnode->next = head;
        head = newnode;
    }
    else
    {
        Node* temp = head;
        while(temp->next != NULL && temp->next->num <= num) temp = temp->next;
        newnode->next = temp->next;
        temp->next = newnode;
    }
}
```

```
int main()
{
    while(true)
    {
        int x;
        cin >> x;
        if(x==-1) break;
        else insertNode(x);
    }

    Node* temp = head;
    while(temp!=NULL)
    {
        cout << temp -> num << endl;
        temp = temp->next;
    }
}
```

$time\ complexity : O(n)$

Sansaiyai

สร้าง struct ของนักเรียน นักเรียนแต่ละคนมี pointer แสดงนักเรียนทางซ้ายและนักเรียนทางขวา ในการหาจำนวนนักเรียนที่จับมือกันอยู่แต่ละกลุ่ม ให้เริ่มหาจากนักเรียนที่ไม่มีคนจับมือทางซ้ายแล้ว *loop* ไปทางขวาจนเจอนักเรียนที่ไม่มีใครจับมือทางขวา จะได้จำนวนนักเรียนในกลุ่มนั้น(เริ่มจากนักเรียนที่ไม่มีคนจับมือทางขวาแล้วไล่ไปทางซ้ายก็ได้เหมือนกัน) คำตอบคือกลุ่มที่มีนักเรียนมากที่สุด

Example Code

```cpp
#include <bits/stdc++.h>
using namespace std;

typedef struct Student{
    struct Student* left = NULL;
    struct Student* right = NULL;
} student;

int main()
{
    int n,maxstudent=0;
    cin >> n;

    student* arr[n+1];
    for(int i=0;i<=n;i++) arr[i] = (student*)malloc(sizeof(student));

    for(int i=1;i<=n;i++)
    {
        int l,r;
        cin >> l >> r;
        arr[i]->left = arr[l];
        arr[i]->right = arr[r];
    }
```

```
    for(int i=1;i<=n;i++)
    {
        if(arr[i]->left != arr[0]) continue;
        int count = 1;
        student* temp = arr[i];
        while(temp->right != arr[0])
        {
            count++;
            temp = temp->right;
        }
        maxstudent = max(maxstudent,count);
    }
    cout << maxstudent;
}
```

$time\ complexity : O(n)$

Line Up

ในข้อนี้จะใช้ linked list สองประเภทคือ

- linked list เก็บข้อมูลนักเรียนในแต่ละแถว
- linked list เก็บข้อมูลแถว แต่ละ node เก็บข้อมูลเป็น linked list ประเภทแรก
  (linked list ซ้อน linked list แหละเอาง่ายๆ)

การเพิ่มนักเรียนเข้าไปในแถวเดิมให้เพิ่ม node ไปยัง linked list ประเภทแรกที่เก็บไว้ใน node สุดท้ายของ linked list ประเภทที่สอง ส่วนการขึ้นแถวใหม่ให้เพิ่ม node ของ linked list ประเภทที่สอง

Example Code

```
#include <bits/stdc++.h>
using namespace std;

typedef struct node{
    int num;
    struct node* prev;
    struct node* next;
} Node;

typedef struct line{
    Node* head;
    Node* tail;
    struct line* prev;
    struct line* next;
} Line;

struct node* newNode(int num)
{
    Node* temp = (Node*)malloc(sizeof(Node));
    temp->num = num;
    temp->prev = NULL;
    temp->next = NULL;
    return temp;
}
```

```
Line* newLine()
{
    Line* temp = (Line*)malloc(sizeof(Line));
    temp->head = NULL;
    temp->tail = NULL;
    temp->prev = NULL;
    temp->next = NULL;
    return temp;
}

Line* front;
Line* back;
void insertBack(int n)
{
    if(back->tail == NULL)
    {
        back->tail = newNode(n);
        back->head = back->tail;
    }
    else
    {
        back->tail->next = newNode(n);
        (back->tail->next)->prev = back->tail;
        back->tail = back->tail->next;
    }
}

void insertFront(int n)
{
    if(back->head == NULL)
    {
        back->head = newNode(n);
        back->tail = back->head;
    }
    else
    {
        back->head->prev = newNode(n);
        (back->head->prev)->next = back->head;
        back->head = back->head->prev;
    }
}
```

```
int main()
{
    front = newLine(), back = front; // init Linked list

    int n;
    cin >> n;
    for(int i=0;i<n;i++)
    {
        int x;
        cin >> x;
        if(i==0) insertBack(x); // นักเรียนคนแรก
        else if(x > back->tail->num) insertBack(x); // ต่อท้าย
        else if(x < back->head->num) insertFront(x); // แทรกหน้า
        else // ขึ้นแถวใหม่
        {
            back->next = newLine();
            back = back->next;
            insertBack(x);
        }
    }

    Line* linetemp = front;
    while(linetemp!=NULL)
    {
        Node* studenttemp = linetemp->head;
        while(studenttemp!=NULL)
        {
            cout << studenttemp->num << " ";
            studenttemp = studenttemp->next;
        }
        cout << endl;
        linetemp = linetemp->next;
    }
}
```

$time\ complexity : O(n)$