CP 111 Computer Programming

2/2554 Section B01/B02

Lab 12

หมายเหตุ การเขียนโปรแกรมที่คำนวณกับตัวเลขที่มีจุดทศนิยม ให้ประกาศตัวแปรที่เกี่ยวข้องเป็นแบบ `double` (Double-precision floating point)

1. Problem name: `Linked_List_Add_Student`

จงเขียนโปรแกรม (ทำโปรแกรมข้างล่างให้สมบูรณ์) เพื่ออ่านข้อมูลของนักเรียนที่ลงทะเบียนเรียนในวิชาหนึ่งๆ มาเก็บไว้ใน Linked list โดยใช้ฟังก์ชัน `Add_Student` โดยการเพิ่ม node ใหม่ลงใน linked list ให้นำไปเพิ่มที่ต้น linked list (เพิ่มเป็น node แรก) หลังจากนั้นให้แสดงข้อมูลของนักเรียนที่เก็บอยู่ใน linked list เรียงจาก node แรก (head node) ไปยัง node สุดท้าย (tail node)

กำหนดให้ข้อมูลนำเข้าในแต่ละบรรทัด ประกอบด้วย รหัสนิสิต ชื่อไม่เกิน 20 ตัวอักษร คะแนนสอบ (เลขจำนวนเต็ม) เช่น 10101 John 60 เป็นต้นโดยในกรณีที่ข้อมูลบรรทัดสุดท้ายคือ -1 -1 -1 จะเป็นบรรทัดที่ระบุจุดสิ้นสุดของข้อมูล

ตัวอย่างการทำงานของโปรแกรม

ข้อมูลนำเข้า (Input)	ข้อมูลส่งออก (Output)
10101 John 60 10009 Sarah 50 10001 Abby 100 10084 Onny 90 11101 Jane 15 12009 Soe 80 12001 Tab 80 13084 Anny 50 13184 Peter 40 13284 Bob 40 -1 -1 -1	13284 Bob 40 13184 Peter 40 13084 Anny 50 12001 Tab 80 12009 Soe 80 11101 Jane 15 10084 Onny 90 10001 Abby 100 10009 Sarah 50 10101 John 60

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

typedef struct s_node_tag S_NODE;
struct  s_node_tag {
  int ID;
  char name[21];
  int score;
  __________________
};
int Add_Student(S_NODE *head,int ID,char *name,int score);
int main()
{
S_NODE dummy_head,*ptr;
int s_id,s_score;
char s_name[21];

       dummy_head.next=NULL;
       while(1)
       {
              scanf("%d %s %d",&s_id,s_name,&s_score);
              if(s_id==-1)
                     break;
              Add_Student(&dummy_head,s_id,s_name,s_score);
       }
       ptr=_________________
       while(1)
       {

              if(_________________)
              {
                     break;
              }
              printf("%d %s %d\n",ptr->ID,ptr->name,ptr->score);
              ptr=_________________
       }
       return 0;
}
int AddStudent(S_NODE *head,int id,char *name)
{
S_NODE *new_s;
     // จองเนื้อที่สำหรับ node ใหม่

     ..............................................................................
     if(new_s==NULL)
        return -1;

     // กำหนดข้อมูลในฟิลด์ย่อย ID และ name

     ..............................................................................
     ..............................................................................
// กำหนดข้อมูลในฟิลด์ย่อย next เพื่อใส่ node นี้ไว้เป็น node แรกของ Linked list

     ..............................................................................
     ..............................................................................
     return 0;
}
```

2. Problem name: `Linked_List_Find_Student`

จงเขียนโปรแกรม ต่อจากโปรแกรมในข้อที่แล้ว (ไม่เอาส่วนของการแสดงข้อมูลใน Linked list) โดยทำการค้นหาข้อมูลของนักเรียนที่มี ID เท่ากับ ค่าที่รับเข้ามาทางคีย์บอร์ด โดยในกรณีที่เจอให้แสดงข้อมูลของนิสิตคนที่มี ID ดังกล่าว แต่ถ้าไม่เจอให้แสดงค่า -1

กำหนดให้เขียนและเรียกใช้ฟังก์ชัน `Find_Student` ซึ่งจะทำการค้นหา node ที่เก็บข้อมูลของนิสิตที่มีรหัสตรงกับค่าอาร์กิวเมนต์ q_id ถ้าเจอ node ดังกล่าวให้ฟังก์ชันคืนค่า address หรือ pointer ที่ไปยัง node นั้น แต่ถ้าไม่เจอให้คืนค่า NULL

ตัวอย่างการทำงานของโปรแกรม

ข้อมูลนำเข้า (Input)	ข้อมูลส่งออก (Output)
10101 John 60 10009 Sarah 50 10001 Abby 100 10084 Onny 90 11101 Jane 15 12009 Soe 80 12001 Tab 80 13084 Anny 50 13184 Peter 40 13284 Bob 40 -1 -1 -1 10001	10001 Abby 100
10101 John 60 10009 Sarah 50 10001 Abby 100 10084 Onny 90 11101 Jane 15 12009 Soe 80 12001 Tab 80 13084 Anny 50 13184 Peter 40 13284 Bob 40 -1 -1 -1 11111	-1

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

typedef struct s_node_tag S_NODE;
struct  s_node_tag {
  int ID;
  char name[21];
  int score;
  S_NODE *next;
};
int Add_Student(S_NODE *head,int ID,char *name,int score);
S_NODE *FindStudent(S_NODE *head,int q_id);
int main()
{
S_NODE dummy_head,*ptr;
int s_id,s_score,query_id;
char s_name[21];

       dummy_head.next=NULL;
       while(1)
       {
              scanf("%d %s %d",&s_id,s_name,&s_score);
              if(s_id==-1)
                     break;
              Add_Student(&dummy_head,s_id,s_name,s_score);
       }
       scanf("%d",&query_id);
       ptr=Find_Student(&dummy_head,query_id);
       if(     )
              printf("%d %s %d\n",             );
       else
              printf("    ");

       return 0;
}

int Add_Student(S_NODE *head,int ID,char *name,int score)
// { reuse the code from the previous proble, }

S_NODE *FindStudent(S_NODE *head,int q_id)
{
S_NODE *ptr;
     // กำหนดให้ ptr ชี้ไปที่ node แรกใน linked list

     ..............................................................................

     //  วนกำหนด ptr ให้ชี้ไปแต่ละโหนดใน linked list จนกว่าจะเจอ tail

     while(.............................................)

     {

       // ตรวจว่าข้อมูลในโหนดที่ชี้โดย ptr มีค่าเท่ากับ q_id ถ้าใช่คือ เจอ

        if(.....................................................)

              ..........................................................

       // เปลี่ยนค่า ptr ให้ชี้ไปที่โหนดถัดไป

          ..............................................................................

     }

     ..............................................................................

}
```

3. Problem name: `Linked_List_Delete_Student`

จงเขียนโปรแกรม ต่อจากโปรแกรมในข้อ 1 แล้ว ให้ทำการลบข้อมูลของนักเรียนที่มี ID เท่ากับค่าที่รับเข้ามาทางคีย์บอร์ด หลังจากนั้นให้แสดงข้อมูลปัจจุบันที่เก็บอยู่ใน linked list เรียงจาก node แรก (head node) ไปยัง node สุดท้าย (tail node) ซึ่งจะต้องไม่มีข้อมูลของนักเรียนที่ถูกลบไปก่อนหน้านี้

กำหนดให้เขียนและเรียกใช้ฟังก์ชัน `Delete_Student` ซึ่งจะทำลบ node ที่เก็บข้อมูลของนิสิตที่มีรหัสตรงกับค่าอาร์กิวเมนต์ del_id ในกรณีที่ไม่เจอ node ดังกล่าวให้ฟังก์ชันคืนค่า -1 โดยที่ข้อมูลใน Linked list จะไม่มีการเปลี่ยนแปลง ในกรณีที่เจอ node ดังกล่าวให้ฟังก์ชันทำการลบ node นั้นและคืนค่า 0

ตัวอย่างการทำงานของโปรแกรม

ข้อมูลนำเข้า (Input)	ข้อมูลส่งออก (Output)
10101 John 60 10009 Sarah 50 10001 Abby 100 10084 Onny 90 11101 Jane 15 12009 Soe 80 12001 Tab 80 13084 Anny 50 13184 Peter 40 13284 Bob 40 -1 -1 -1 10001	13284 Bob 40 13184 Peter 40 13084 Anny 50 12001 Tab 80 12009 Soe 80 11101 Jane 15 10084 Onny 90 10009 Sarah 50 10101 John 60
10101 John 60 10009 Sarah 50 10001 Abby 100 10084 Onny 90 11101 Jane 15 12009 Soe 80 12001 Tab 80 13084 Anny 50 13184 Peter 40 13284 Bob 40 -1 -1 -1 11111	13284 Bob 40 13184 Peter 40 13084 Anny 50 12001 Tab 80 12009 Soe 80 11101 Jane 15 10084 Onny 90 10001 Abby 100 10009 Sarah 50 10101 John 60

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

typedef struct s_node_tag S_NODE;
struct  s_node_tag {
  int ID;
  char name[21];
  int score;
  S_NODE *next;
};

int Add_Student(S_NODE *head,int ID,char *name,int score);
S_NODE *Find_Student(S_NODE *head,int q_id);
int Delete_Student(S_NODE *head,int del_id);

int main()
{
S_NODE dummy_head,*ptr;
int s_id,s_score,d_id,retval;
char s_name[21];

       dummy_head.next=NULL;
       while(1)
       {
              scanf("%d %s %d",&s_id,s_name,&s_score);
              if(s_id==-1)
                     break;
              Add_Student(&dummy_head,s_id,s_name,s_score);
       }
       scanf("%d",&d_id);

       retval=Delete_Student(                          );

       ptr=dummy_head.next;
       while(1)
       {
              if(ptr==NULL)
                     break;
              printf("%d %s %d\n",ptr->ID,ptr->name,ptr->score);
              ptr=ptr->next;
       }

       return 0;
}

S_NODE *Find_Student(S_NODE *head,int q_id)
// { reuse the code from the previous problem}
```

```
int Delete_Student(S_NODE *head,int del_id)
{
S_NODE *prev,*curr;
    // กำหนดให้ curr ชี้ไปที่ node แรก และ prev ชี้ไปที่ node ก่อนหน้า curr
    ..............................................................................
    ..............................................................................
    //  วนกำหนด curr ให้ชี้ไปแต่ละโหนดใน linked list จนกว่าจะเจอ tail
    while(.................................................)
    { // ตรวจว่าข้อมูลในโหนดที่ชี้โดย curr มีค่าเท่ากับ del_id ถ้าใช่คือ เจอ
      if(.....................................................)
      {  // ปรับให้ prev ชี้ไปที่ node ถัดจาก curr
           ...................................................
         // คืนหน่วยความจำที่จองไว้สำหรับโหนด curr
           ...................................................
      }
         // เปลี่ยนค่า prev และ curr ให้ชี้ไปที่โหนดถัดไป
       ..................................................................
       ..................................................................
    }
     ..................................................................
}
```

Partial Codes for Problems 1 to 3

```
int Add_Student(S_NODE *head,int ID,char *name,int score)
{
S_NODE *new_s;

      new_s=(S_NODE *)malloc(sizeof(S_NODE));
      if(new_s==NULL)
            return -1;

      new_s->ID=ID;
      strcpy(new_s->name,name);
      new_s->score=score;

      new_s->next=head->next;
      head->next=new_s;
      return 0;
}
```

```
S_NODE *Find_Student(S_NODE *head,int q_id)
{
S_NODE *ptr;
      ptr=head->next;
      while(ptr!=NULL)
      {
            if(ptr->ID==q_id)
                  return(ptr);
            ptr=ptr->next;
      }
      return(NULL);

}
```

```
int Delete_Student(S_NODE *head,int del_id)
{
S_NODE *prev,*curr;

            prev=head;
            curr=head->next;
            while(curr!=NULL)
            {
                  if(curr->ID==del_id)
                  {
                        prev->next=curr->next;
                        free(curr);
                        return 0;
                  }
                  prev=curr;
                  curr=curr->next;
            }
            return -1;
}
```

4. Problem name: `Max_occurrence_2_Linked_List`

กำหนดให้ข้อมูลนำเข้าเป็นจำนวนเต็มที่มีค่ามากกว่าหรือเท่ากับ 0 เข้ามาบรรทัดละตัวและสิ้นสุดด้วยจำนวนเต็มลบ จงเขียนโปรแกรมเพื่อนับว่าค่าใด ของข้อมูลนำเข้าถูกป้อนเข้ามามากที่สุด โดยให้แสดงค่าตัวเลข และจำนวนครั้งที่เจอออกสู่หน้าจอที่ละบรรทัด (ข้อนี้ขอบเขตของข้อมูลนำเข้าคือ $0-(2^{31}-1)$) และจำนวนข้อมูลนำเข้าอาจจะมี 1 ตัว ถึง 1,000,000 ตัว <u>กำหนดให้การเขียนโปรแกรม</u>ใช้โครงสร้างข้อมูลแบบ linked list โดยที่แต่ละ Node นิยามโดย self-referenced structure `DATA_NODE` ดังโค้ดข้างล่าง

ข้อมูลนำเข้า (Input)	ข้อมูลส่งออก (Output)
41 12 31 12 31 12 31 12 12 41 31 -1	12 5
99 11 12 13 14 15 16 17 18 19 0 99 -1	99 2

```
#include <stdio.h>
#include <stdlib.h>
typedef struct data_node_tag DATA_NODE;
struct  data_node_tag {
  int data;
  int count;
  DATA_NODE *next;
};
int Update_Count(DATA_NODE *head, int data);
int main()
{
DATA_NODE dummy_head,*ptr,*del_node;
int retval,max_count,max_data,x;
    dummy_head.next=NULL;
    while(1)
    {
        scanf("%d",&x);
        if(x<0)
            break;
        // เพิ่มค่า count ของ x (ถ้า x ไม่มีอยู่ใน Linked list เพิ่ม node ใหม่แล้ว)
        Update_Count(&dummy_head,x);
    }
    max_count=0;
    max_data=-1;
    ptr=
    while(ptr!=NULL)
    {
        if(________________________)
        {
            max_count=_____________
            max_data=______________
        }
        del_node=ptr;
        ptr=ptr->next;
        free(del_node);
    }
    printf("%d\n%d\n",max_data,max_count);
    return 0;
}
int Update_Count(DATA_NODE *head, int data)
{
DATA_NODE *ptr,*new_node;
    // วนลูปหา node ที่ฟิลด์ย่อย data ตรงกับ data แล้วเพิ่ม count ของ node นั้น ถ้าไม่เจอ node ดังกล่าว เพิ่ม node ใหม่
        ptr=________________________
        while(_____________________)
        {
               if(ptr->data==___________________)
               {
                      ptr->count=_______________
                      return 0;
               }
               ptr=______________________
        }
        new_node=(DATA_NODE *)malloc(sizeof(DATA_NODE));
        if(new_node==NULL)
               return -1;
        new_node->data=___________
        new_node->count=___________
        new_node->next=___________
        ____________________
        return 0;
}
```