บทที่ 10
ตัวชี้และโครงสร้างข้อมูลแบบพลวัต

วัตถุประสงค์

1. เพื่อให้นักศึกษาสามารถประกาศ กำหนด และจัดการกับโครงสร้างโปรแกรมที่มีการใช้โครงสร้างข้อมูลแบบพลวัตได้
2. เพื่อให้นักศึกษาสามารถเขียนโปรแกรมประยุกต์ใช้งานกับโครงสร้างโปรแกรมที่มีการใช้โครงสร้างข้อมูลแบบพลวัตได้

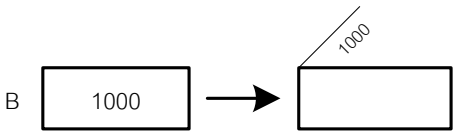
9.1 ตัวชี้ (Pointer) เป็นชนิดของข้อมูลหนึ่งที่เก็บเลขที่อยู่ของข้อมูล

รูปแบบ type * Variable;

ตัวอย่าง float *B;

การจองเนื้อที่ให้กับตัวแปรตัวชี้ new type;

ดังนั้นถ้ากำหนดคำสั่งดังนี้ B = new float;



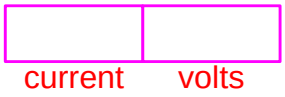
การนำข้อมูลที่จัดเก็บในเลขที่อยู่ที่ต้องการมาใช้งาน ใช้สัญลักษณ์ * หรือ Asterisk หรือที่เรียกว่า Indirection operator ใช้อ้างถึงค่าของข้อมูลที่ตัวแปรตัวชี้ ชี้อยู่จาก

```
float   *B;
 B  =  new float;
*B  =  15.5;
```

เป็นการนำค่าเลขทศนิยม 15.5 ไปจัดเก็บเลขที่อยู่ที่ตัวแปร B ชี้อยู่

การประกาศโครงสร้างระเบียนข้อมูลชื่อ electric ซึ่งประกอบด้วยเขตข้อมูลชื่อ current และเขตข้อมูลชื่อ volts ดังนี้

```
struct electric
{
    string current;
    int  volts;
};
```

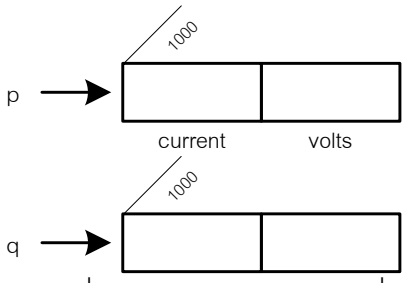


กำหนดตัวแปรชนิดตัวชี้ชื่อ p และ q ให้เก็บเลขที่อยู่ของโครงสร้างข้อมูลชนิด electric ได้ดังนี้ electric *p, *q;

การจัดสรรเนื้อที่สำหรับเก็บข้อมูลนั้นต้องใช้คำสั่ง new ดังนี้

```
p  =  new electric;
q  =  new electric;
```

การทำงานของคำสั่งจะจัดสรรเนื้อที่ว่างเป็นโครงสร้างของ electric โดยให้ตัวแปร p และ q โดยในที่นี้สมมุติว่า p ชี้ที่เลขที่อยู่ 1000 ส่วน q ชี้ที่เลขที่อยู่ 10000



การนำข้อมูลเก็บในโครงสร้างที่จัดสรรไว้ สามารถใช้คำสั่งดังนี้

```
(*p).current = "AC";
(*p).volts = 115;
```

หรือ

```
p -> current = "AC";
p -> volts = 115;
```

การลบโหนดข้อมูลเลขที่อยู่ที่ตัวแปรตัวชี้ชี้อยู่ใช้คำสั่ง delete มีรูปแบบดังนี้

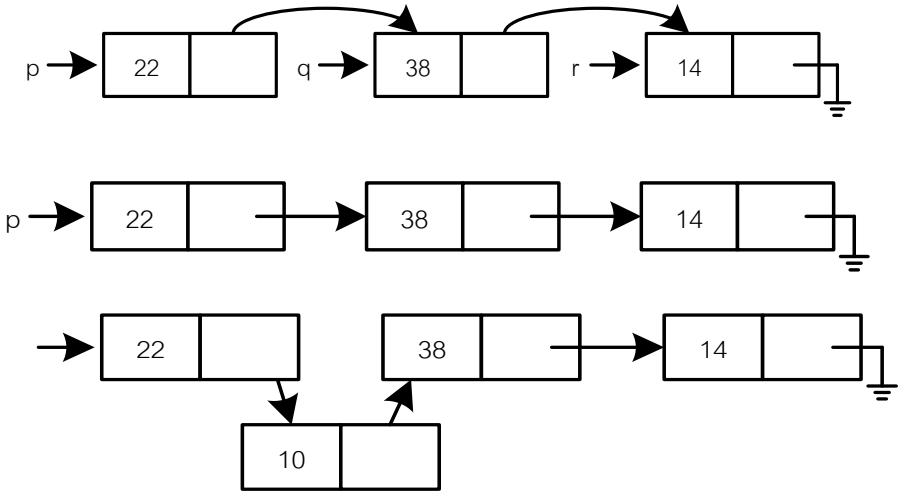
รูปแบบ **delete Variable;**

ตัวอย่าง delete q;

9.2 **รายการโยง(Linked Lists)** เป็นการรวมกลุ่มของข้อมูลที่มีการจัดสรรโครงสร้าง

ข้อมูลแบบพลวัตเป็นโครงสร้างข้อมูลที่ยืดหยุ่น โดยนำตัวแปรตัวชี้มาเก็บเลขที่อยู่ของข้อมูลตัวถัดไป เหมาะกับการแก้ปัญหาที่มีการแทรก หรือลบข้อมูล เพราะโปรแกรมเมอร์สามารถเขียนข้อความสั่งเพียงแต่เปลี่ยนเลขที่อยู่ของข้อมูลให้เก็บข้อมูลตัวถัดไปให้ถูกต้องเท่านั้น การปฏิบัติงานไม่มีการเคลื่อนย้ายข้อมูลทำให้ไม่เสียเวลาการเพิ่มหรือลบข้อมูลทำได้ง่าย ดังรูป

ดังรูป



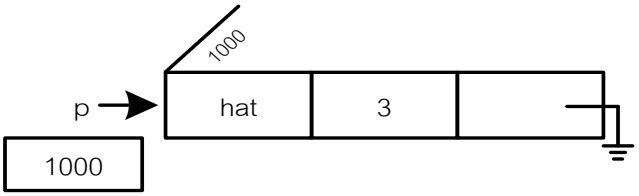
การจัดเก็บข้อมูลรายการโยงต้องมีการเก็บเลขที่อยู่ของข้อมูลตัวถัดไปพร้อมกับข้อมูลที่ต้องการจัดเก็บจริงๆ เรียกว่าโหนด(Node)ซึ่งเป็นโครงสร้างข้อมูลชนิด struct ที่ประกอบด้วยเขตข้อมูลแบ่งเป็น 2 กลุ่มคือ

1. Information field เป็นเขตข้อมูลที่จัดเก็บข้อมูล
2. Link field เป็นเขตข้อมูลที่เก็บเลขที่อยู่ของโหนดตัวถัดไป ในกรณีที่ไม่มีข้อมูลตัวถัดไป ค่าของรายการข้อมูลนี้มีค่าเท่ากับ NULL

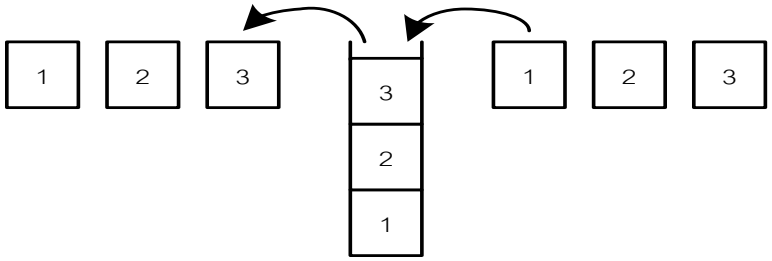
รูปแบบของโครงสร้างข้อมูลชนิด struct ได้ดังนี้

```
struct node  {
        string word;
        int count;
        node *link;
};
```

โปรแกรมเมอร์สามารถกำหนดตัวแปรชนิดตัวชี้เพื่อปฏิบัติการกับโหนดได้ดังนี้ node *p;
สามารถจองเนื้อที่ว่าง 1 โหนดเพื่อใช้งานได้ดังนี้ p = new node;
การนำข้อมูลใส่ในโครงสร้างที่จัดสรรเนื้อที่ไว้สามารถกระทำได้ดังนี้
p -> word = "hat"; p -> count = 3; p -> link = NULL;



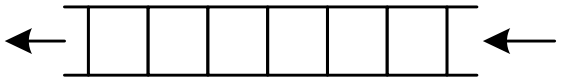
9.3 กองซ้อนที่เชื่อมโยงกัน กองซ้อน(Stack) เป็นโครงสร้างข้อมูลที่เป็นลักษณะ LI-FO (Last In - First Out) โดยข้อมูลนำเข้าตัวสุดท้ายจะได้รับการบริการก่อน ข้อมูนำเข้าตัวแรกจะได้รับการบริการเป็นลำดับสุดท้าย ซึ่งการนำเข้าและนำออกข้อมูลกระทำได้ที่ปลายด้านใดด้านหนึ่งเท่านั้น ดังรูป



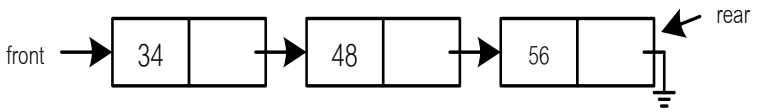
รายการโยงที่ทำงานลักษณะกองซ้อน โปรแกรมเมอร์ต้องกำหนดตัวแปรชื่อ Top ให้ชี้ที่โหนดข้อมูลแรก ซึ่งในกรณีที่รายการโยงว่างหรือไม่มีข้อมูล Top มีค่าเท่ากับ NULL การนำสมาชิกเข้าในรายการโยงจะกระทำที่ปลายด้านใดด้านหนึ่ง เรียกการปฏิบัติการนี้ว่า push และการนำสมาชิกออกจากรายการโยงจะกระทำที่ปลายด้านเดียวกันเรียกการปฏิบัติการนี้ว่า pop โดยตัวแปร Top จะชี้ที่ตำแหน่งโหนดข้อมูลที่ปฏิบัติการเสมอ

9.4 **แถวคอยที่เชื่อมโยงกัน** แถวคอย(Queued) เป็นโครงสร้างข้อมูลที่เป็นลักษณะ

FI-FO (First In - First Out) โดยการปฏิบัติงานกับข้อมูลในโครงสร้างนี้จะกระทำที่ปลาย 2 ด้าน โดยการนำสมาชิกเข้าไปเก็บในแถวคอยเรียกว่าการ insert โดยกระทำที่ปลายด้านหนึ่ง และการนำข้อมูลออกจากแถวคอยจะกระทำอีกปลายด้านหนึ่งเรียกว่าการ remove โปรแกรมเมอร์สามารถกำหนดรายการโยงที่มีการทำงานในลักษณะแถวคอยโดยกำหนดตัวแปรชื่อ rear ชี้อยู่ที่ข้อมูลโหนดตัวสุดท้ายของรายการโยงและตัวแปร front ชี้ที่ข้อมูลโหนดแรกของรายการโยง

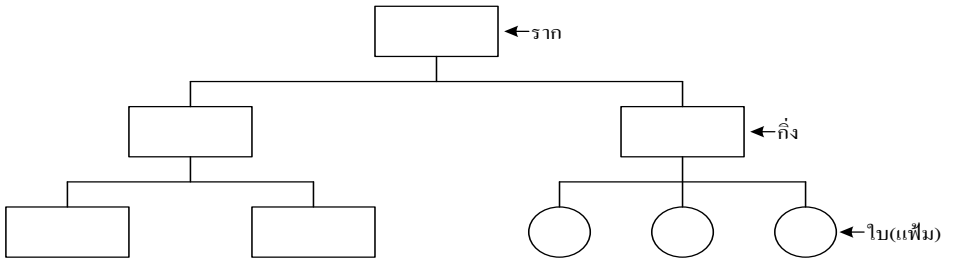


การเก็บข้อมูลในแถวคอยที่เชื่อมโยงกันนั้น จะคล้ายกับกองซ้อนที่เชื่อมโยงกันแตกต่างกันที่การนำเข้าและการนำออกข้อมูล โดยตัวแปร front เก็บเลขที่อยู่ของข้อมูลที่โหนดแรก และตัวแปร rear เก็บเลขที่อยู่ของข้อมูลโหนดสุดท้ายของรายการโยง โดยโหนดสุดท้ายของรายการโยงต้องชี้ไปที่ NULL



9.5 **โครงสร้างรูปต้นไม้** เป็นการนำตัวแปรชนิดตัวชี้มาใช้งานในอีกลักษณะหนึ่งโดยเก็บ

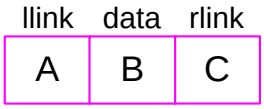
โหนดของข้อมูลไม่เป็นเชิงเส้น(Non-linear list) แต่จัดเก็บเป็นลำดับชั้น (Hierarchical) โดยลำดับชั้นบนสุดจะทำหน้าที่คล้ายรากของต้นไม้ และลำดับชั้นถัดๆไปเปรียบเสมือนกิ่งของต้นไม้ แต่ละกิ่งของต้นไม้สามารถแตกเป็นกิ่งย่อยๆได้ในลำดับชั้นถัดๆไปจนกระทั่งเป็นใบ



โครงสร้างรูปต้นไม้ทั่วไป ประกอบด้วย เซตของโหนดข้อมูล โดยโหนดข้อมูลจะมีลักษณะดังนี้

1. โหนดพิเศษ เรียกว่าราก(Root) เป็นโหนดบนสุดของลำดับชั้น
2. โหนดอื่นๆ ถูกแบ่งเป็นเซตย่อยๆ จะเรียกว่าต้นไม้ย่อย(Subtree)

การแทนความหมายของต้นไม้ในหน่วยความจำแบบ Link allocation เป็นการจัดเก็บข้อมูลโดยใช้รายการโยง โดยข้อมูลแต่ละโหนดประกอบด้วยตัวชี้ 2 ตัว โดยตัวชี้ตัวแรกเก็บเลขที่อยู่ของโหนดที่เป็นต้นไม้ย่อยทางซ้ายและตัวชี้อีกตัวหนึ่งจะเชื่อมโยงโดยเก็บเลขที่อยู่ของโหนดที่เป็นต้นไม้ย่อยทางขวา



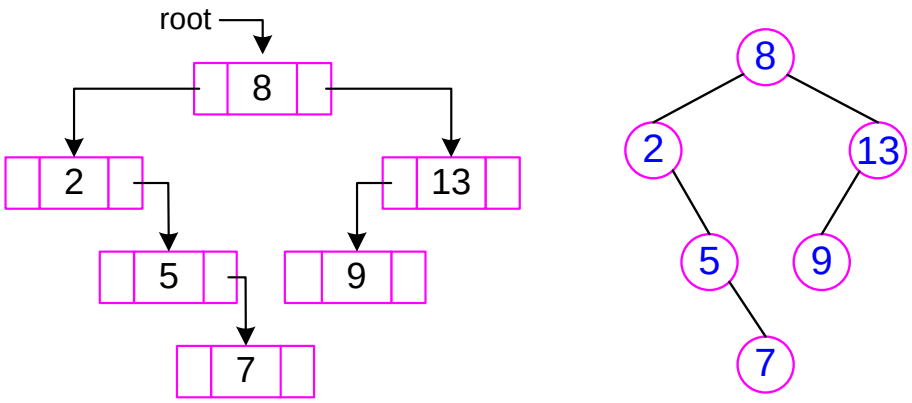
โดย llink เป็นตัวชี้ที่ ชี้ไปยังต้นไม้ย่อยทางซ้าย(Left sub tree)

data เป็นตัวแปรที่ใช้สำหรับ เก็บข้อมูล

rlink เป็นตัวชี้ ชี้ไปยังต้นไม้ย่อยทางขวา(Right sub tree)

ต้นไม้ค้นหาแบบทวิภาค(Binary Search Tree) เป็นต้นไม้แบบทวิภาคชนิดหนึ่ง มีกฎเกณฑ์ในการสร้างดังนี้

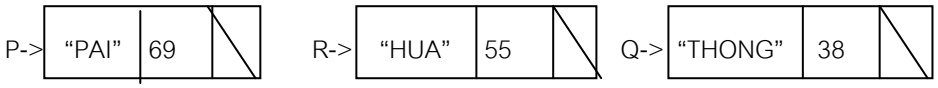
1. ค่าที่เก็บที่ราก(Root) ต้องมีค่ามากกว่าข้อมูลต้นไม้ย่อยทางซ้ายของทุกโหนด
2. ค่าที่เก็บในต้นไม้ย่อยทางขวา ต้องมากกว่าหรือเท่ากับราก(Root) ทุกๆ โหนด
3. แต่ละต้นไม้ย่อยต้องมีคุณสมบัติตามข้อที่ 1 และ 2



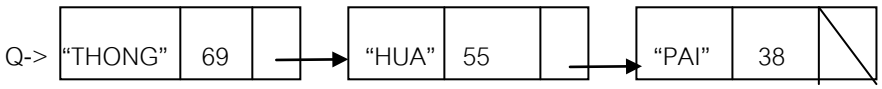
ปฏิบัติการครั้งที่ 12

ข้อ 1. พิจารณาโครงสร้างข้อมูลของนักศึกษาและรูปภาพต่อไปนี้แล้วตอบคำถาม

```
struct  STU
{
    string    name;
    double  score;
    STU     *link;
};
STU  *P, *Q, *R ,*Head;
```



1.1 จงเขียนส่วนของคำสั่งในการเชื่อมต่อโหนดทั้งสามเข้าด้วยกันให้ผลลัพธ์สุดท้ายเป็นดังนี้ตามภาพ



.....

.....

.....

.....

.....

.....

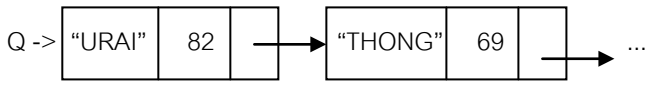
.....

.....

1.2 จงเขียนคำสั่งในการสร้างโหนดใหม่โดยเก็บ ข้อมูล “URAI” และ 82 ในเขตข้อมูลต่างๆตามลำดับ

……..
……..
……..
……..
……..

1.3 จงเขียนคำสั่งแทรกโหนดใหม่ที่สร้างในคำถามข้อ 1.2 ที่ต้นรายการโยง โดยให้ตัวแปร Q ชี้ ดังภาพ



……..
……..
……..
……..

1.4 จงเขียนคำสั่งในการลบโหนดแรกของรายการโยงโดยให้ Q ชี้ที่โหนดแรก

……..
……..
……..

1.5. จงเขียนฟังก์ชันชื่อ NumberoNode เพื่อนับจำนวนของโหนดทั้งหมด ที่อยู่ในรายการโยงที่ Q ชี้อยู่ที่โหนดแรก

……..
……..
……..
……..
……..
……..
……..
……..

ข้อที่ 2 โปรแกรมต่อไปนี้เป็นโปรแกรมในการวนรอบรับคะแนนของนักศึกษาจนกระทั่งผู้ใช้ป้อน ‘N’ จึงหยุดรับข้อมูลคะแนน การทำงานของโปรแกรมจะนำคะแนนเก็บในรายการโยงเดี่ยวโดยให้ตัวแปรพอยเตอร์ first ชี้ที่โหนดแรก โดยข้อมูลคะแนนที่ป้อนจะถูกเก็บต่อเนื่องเป็นลำดับ ผลของการทำงานจะพิมพ์ข้อมูลที่เก็บในรายการโยงนี้ออกทางจอภาพ จงเติมคำสั่งโปรแกรมให้สมบูรณ์

```
#include<iostream>
using namespace std;
struct node     {                                   // 1
                int score;
                node* link;
        };                                          .
void print(……..) {     // 2
        while (first != ……..)
        {
                cout<<first->score<<endl;
                first=first->link;
        }
}
int main() {
        char ch;
        node *p ;   node *first;   node *q;         // 3
        first = NULL;                               // 4
        do {
                p = new node;                       // 5
                cout<<"score="; cin>>p->score;
                p->link = NULL;
                if (first==NULL)                                // 6
                        first=……;
```

```
            else    {
                    q = first;
                    while (q->link != NULL)
                            q = ……;
                    q->link = ……;
            }
            cout<<"run again y/n?";cin>>ch;
    }while (ch=='y');
    print(first);
    return 0;
}
```

จงอธิบายการทำงานของคำสั่งที่ระบุ

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

ข้อ 3. โปรแกรมต่อไปนี้เป็นการ Intersaction รายการโยง 2 รายการโยง โดยโปรแกรมให้ผู้ใช้ป้อนข้อมูลเพื่อเก็บในรายการโยงแรกที่ตัวแปร first ชี้อยู่ และป้อนข้อมูลเพื่อเก็บในรายการโยงที่สองที่ตัวแปร second ชี้อยู่ข้อมูลที่จัดเก็บเป็นคำในภาษาอังกฤษซึ่งคำที่ป้อนหลังสุดจะถูกเก็บต่อท้ายเสมอ ผลของการทำงานจะหาคำที่ซ้ำๆกันในรายการโยงทั้งสองเพื่อเก็บในรายการโยงที่สามซึ่งตัวแปร third ชี้อยู่และพิมพ์ออกทางจอภาพ จงเติมคำสั่งให้สมบูรณ์

```
#include<iostream>
#include<string>
using namespace std;
struct node {
        string word;
        node *link;
};
void input(……..)
{
        node *p; string w;
        cout<<"word=?";
        cin>>w;
        while(……)
        {
                p = new node;
                p->word=w;
                p->link=NULL;
                if (first==NULL)
                        ……;
                else{
                        node *q;
                        q=……;
```

```
                    while(q->link!=NULL)
                        q=……;
                    q->link=……..;
                }
                cout<<"word=?"; ……;
        }
}
void print(……..) {
        if (first!=NULL) {
                cout<<first->word<<" ";
                ……;
        }
        else
                cout<<endl;
}
void Intersect(..……)
{
        node *p , *q , *r;
        while(first!=……)
        {
                r = second;
                while(r!=……..)
                {
                        if(r->word==first->word){
                                p=new node;
                                p->word=first->word;
                                p->link = NULL;
```

```
                        if (third==NULL)
                                third=……;
                        else
                        {
                                q = ……;
                                while(q->link != NULL)
                                        q=……..;
                                q->link = ……;
                        }
                }
                r=……;
        }
        first=……;
    }
}
int main(){
    node *first, *second , *third;
    first=NULL ; second=NULL; third=NULL;
     input(first); cout<<"A= ";
    print(first);
    input(second);cout<<"B= ";
    print(second);
    Intersect(first,second,third);
    cout<<"A intersect B = ";
    print(third);
    return 0;
}
```

ถ้าต้องการเขียนโปรแกรมเพื่อหา การ Union ของ รายการโยงที่ first และ second ชี้อยู่ที่โหนดแรก โดยผลของการ Union เก็บไว้ที่ third ดังคำสั่งต่อไปนี้

```
Union(first,second,third);
cout<<"A union B = ";
print(third);
```

จงสร้างฟังก์ชันชื่อ Union เพื่อทำงานได้ตามวัตถุประสงค์ข้างต้น

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

ปฏิบัติการครั้งที่ 13

ข้อ 1. จงอธิบายความแตกต่างของโครงสร้างข้อมูลแบบกองซ้อน และโครงสร้างข้อมูลแบบแถวคอยของการนำเข้าข้อมูลและนำออกข้อมูลจากโครงสร้างทั้งสองชนิด กำหนดให้ข้อมูลที่ต้องการประมวลผลเป็นเลขจำนวนเต็มดังนี้ 52 62 78 12 23 15 89 56

ข้อ 2. โปรแกรมต่อไปนี้เป็นโปรแกรมรับชื่อและคะแนนสอบของนักศึกษาเก็บใน Linked Stack และนำข้อมูลมาคิดเกรดของนักศึกษาตามกฎเกณฑ์ของมหาวิทยาลัยรามคำแหง สรุปจำนวนของนักศึกษาที่ได้เกรด G,Pและ F ออกทางจอภาพ จงเติมคำให้สมบูรณ์

```
#include<iostream>
#include<string>
using namespace std;
struct student
{
        string   name;
        int      score;
        ..... next;
};
void print(student* x)
{
        int f=0,p=0,g=0;
        while (…….) {
                cout<<x->name<<" "<<x->score<<endl;
                if (x->score<60)
                        ……..……..;
                else if (x->score<80)
                        ……..;
                else
                        ……;
                x=……;
        }
        cout<<"G= "<<g<<" P= "<<p<<" F= "<<f<<endl;
}
```

```
void push(……..…….. top)
{
        …… p;
        p = ……;
        cout<<"name=";  cin>>p->name;
        cout<<"score="; cin>>p->score;
        p->next = NULL;
        if (top==……)
                top=……;
        else
        {
                ……..……..;
                top = ……;
        }
}
int main(){
        char ch;
        student* top ;
        top = ……;
        do{
                push(top);
                cout<<"run again y/n?";cin>>ch;
        }while (ch != 'n');
        print(top);

        return 0;
}
```

จงเขียนฟังก์ชัน print ใหม่ให้เป็นฟังก์ชันเรียกตัวเอง

```
void print(student* x)
{
……
……
……
……
……
……
……
……
……
……
……
……
……
……
}
```

ข้อ 3. โปรแกรมต่อไปนี้เป็นโปรแกรมในการป้อนชื่อหนังสือ ผู้แต่งหนังสือ เก็บใน Linked Queue โดยให้ตัวแปรพอยเตอร์ front ชี้ที่โหนดแรกและตัวแปรพอยเตอร์ rear ชี้ที่โหนดสุดท้ายของรายการโยง ซึ่งการทำงานของโปรแกรมจะมีการเรียกใช้ฟังก์ชัน DelData เพื่อลบโหนดของข้อมูลที่มีผู้แต่งหนังสือคนเดียวกันทั้งหมดออกจากรายการโยงและพิมพ์ออกทางจอภาพ จงเติมคำสั่งโปรแกรมนี้ให้สมบูรณ์

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

```
struct book_node{
        string name_book;
        string author_book;
        book_node *link;
};
void EnQueue(……..……..)
{
        ……. *p;
        string name;
        cout<<"book name=?"; cin>>name;
        while(name != "####") {
                p = ……..……;
                p->name_book = ……;
                cout<<"author book=?";
                cin>>……;
                p->link = ……;
                if (rear==……) {
                        front = p;
                        rear = p;
                }
                else {
                        rear->link = ……;
                        rear=……..;
                }
                cout<<"book name=?"; cin>>……..;
        }
}
```

```
void Print(……) {
	if (front!=NULL)	{
		cout<<front->name_book<<" "<<front->author_book<<endl;
		Print(……..);
	}
}
bool DelData(book_node *&front, string name_del){
	book_node *prev,*current ,*p;
	prev = ……..;
	current = ……..;
	bool flag=false;
	while(current != ……..……..){
		if(current->author_book==name_del)
		{
			p=……;
			prev->link = ……;
			current=……;
			delete(p);
			flag=true;
		}
		else{
			prev=……;
			current=……;
		}
	}
	return flag;
}
```

```
int main(){
        book_node *front, *rear;
        string name_del;
        front=NULL; rear=NULL;
        EnQueue(front,rear);
        Print(front);
        cout<<"name delete=?"; cin>>name_del;
        if (DelData(front,name_del))
                Print(front);
        else
                cout<<"no data delete";
        return 0;
}
```

ข้อ 4. จงเขียนฟังก์ชัน pop เพื่อนำข้อมูลที่เก็บใน Linked Stack ในข้อที่ 2 พิมพ์ออกทางจอภาพ โดยที่ main function มีการเรียกใช้คำสั่งดังนี้

```
        bool empty=false;
        string name1;
        int score1;
        while (!empty)
        {
                empty=pop(name1,score1,top);
                if (!empty)
                        cout<<name1<<" "<<score1<<" "<<endl;
        }
```

จงเขียนฟังก์ชันชื่อ pop เพื่อทำงานได้ดังคำสั่งข้างต้น

bool pop(string& name1,int& score1,student*& p) {

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

}

ข้อ 5 จงเขียนโปรแกรมเพื่อรับกลุ่มของอักขระกำหนดอักขระตัวสุดท้ายของกลุ่มลงท้ายด้วย $ ภายในกลุ่มของอักขระต้องมีอักษร C นำอักขระกลุ่มนี้ตรวจสอบ ระหว่างอักขระก่อนอักขระ C และหลังอักขระ C ว่าเป็นส่วนผันกลับ(Reverse) หรือไม่ กรณีที่เป็นส่วนผันกลับให้พิมพ์ว่า “legal string” กรณีอื่นๆให้พิมพ์ว่า “illegal string”

ถ้าผู้ใช้ป้อน ACA$ หรือ 12ACA21$ ผลคือ legal string

11ACA12$ หรือ abcdd$ ผลคือ not legal หรือ illegal string

```
#include<iostream>
#include<string>
using namespace std;
struct node{
        char ch;
        node *next;
};
```

……..

……..

……..

……..

……..

……..

……..

……..

……..

……..

……..

……..

……..

……..

……..

……..

……..

……..

……..

……..

……

……

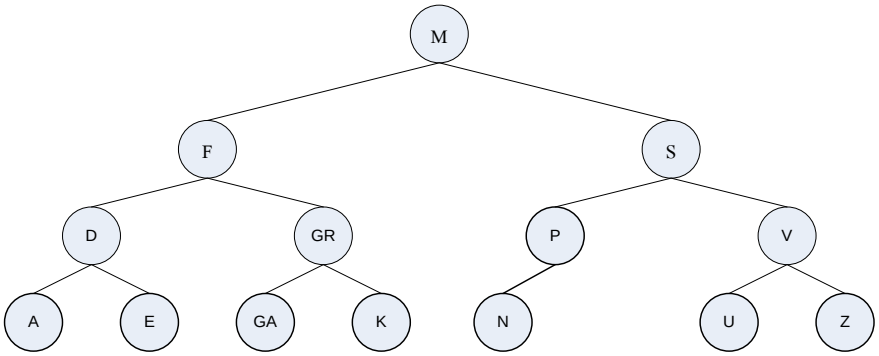
……

ปฏิบัติการครั้งที่ 14

ข้อ 1 พิจารณาข้อมูลต่อไปนี้ 45 85 95 23 42 52 84 12 35 24

จงวาดรูปต้นไม้ค้นหาแบบทวิภาค(Binary search tree) และเขียนฟังก์ชันในการแวะผ่านต้นไม้เพื่อพิมพ์ข้อมูลเฉพาะที่มากกว่า 50 ออกมาทางจอภาพ

ข้อ 2. พิจารณาโครงสร้างรูปต้นไม้ต่อไปนี้



2.1 จงประกาศโครงสร้างข้อมูลของต้นไม้แบบทวิภาคโดยวิธี link allocation

.....

.....

.....

.....

.....

.....

.....

.....

.....

2.2 จงท่องต้นไม้แบบ Inorder , Preorder และ Postorder

.....

.....

.....

.....

.....

ข้อ 3 โปรแกรมต่อไปนี้เป็นโปรแกรมในการนำข้อมูลคะแนนของนักศึกษาเก็บในโครงสร้าง Binary Search tree โดยให้ตัวแปรพอยเตอร์ n ชี้ที่ root node ผู้ใช้สามารถป้อนคะแนนวนซ้ำจนกระทั่งป้อน 0 จึงหยุด การทำงานของโปรแกรมจะทำการท่องต้นไม้แบบ inorder พิมพ์ออกทางจอภาพ นอกจากนี้โปรแกรมสามารถลบโหนดข้อมูลในโครงสร้างต้นไม้โดยให้ผู้ใช้ป้อนคะแนนใดๆที่ต้องการลบออกจากโครงสร้างต้นไม้ และพิมพ์ข้อมูลหลังจากการทำงานออกมาแบบ inorder จงเติมคำสั่งโปรแกรมให้สมบูรณ์

```
#include <iostream>
using namespace std;
struct node
{
      node *llink;
      int  data;
      node *rlink;};
void create(node*& p,int score)
{
    if(p == .....)
          {
                p = .....;
                p->data = score;
                p->llink = NULL;
                p->rlink = NULL;
          }
    else if(p->data > score)
          create(.....);
    else
          create(.....);
}
void inorder(node* p)
{
    if(……..……)
          {
                inorder(p->llink);
                cout<<p->data<<" ";
```

```
                inorder(p->rlink);
        }
}
…….. findMin(node* p)
{
        while(p->llink != NULL)
                p = ……..;
        return p;
}
void del(……)
{
    node *q;
    if(t == ……..)
        cout<<"Can not delete."<<endl;
    else
        {
                 if(x < t->data)
                        del(t->llink,x);
                else if(x > t->data)
                        del(t->rlink,x);

                else {
                        if(t->llink == NULL) {
                                q = t;
                                t = t->rlink;
                                delete(q);
                        }
```

```
                else if(t->rlink == NULL) {
                        q = t;
                        t = ……;
                        delete(q);
                }
                else {
                        q = findMin(……);
                        t->data = q->data;
                        del(t->rlink,t->data);
                }
        }
    }
}
int main()  {
    node *n;
    n = NULL;
    int a;
    cout<<"Binary sort and delete node"<<endl;
    cout<<"Input positive integer. Enter 0 for end of data."<<endl<<endl;
    do{
            cout<<"Data : "; cin>>a;
            if(a != 0)
                    create(n,a);
        } while(a != 0);
    inorder(n);
    cout<<"Number to delete : "; cin>>a;
    del(n,a);
```

```
    inorder(n);
    return 0;
}
```

จงเขียนฟังก์ชันในการหาค่าเฉลี่ยของคะแนนที่เก็บในโครงสร้างต้นไม้

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....