แบบฝึกหัดที่ 21

1. จงติดตามการทำงานของโปรแกรมว่าได้ผลลัพธ์อะไรบ้าง

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

class A {
      int a;
      static int b;
      public:
            A() { a=1; cout<<a<<endl; }
            A(int x) { a=x+1; }
            A(A &y) { b++; cout<<b<<endl;}
            A f(int k) { a+=k; cout<<a<<endl; return *this; }
};

int A::b;

class B: public A {
      int y;
      public:
            B() { y=2; };
            B(B &b) { y=b.y+1; cout<<y<<endl; }
            B f() { A::f(y); return *this; }
};

void main() {
    A x(2);
    x.f(3);
    B z;
    B k(z);
    k.f();
    A y=x;
}
```

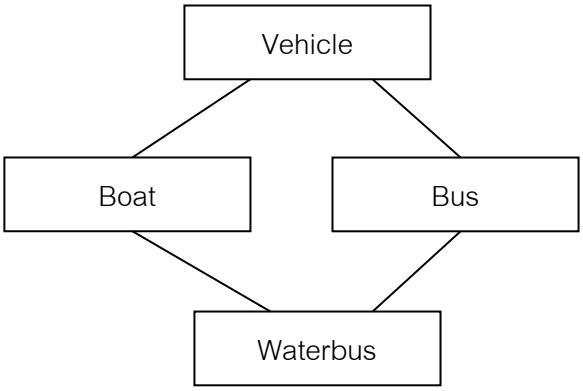
2. โปรแกรมข้างล่างนี้มีที่ผิดตรงไหนบ้าง

```
#include<iostream>
using namespace std;
class A
{
     int x;
     protected:
        int a;
     public:
        A() { x=2; a=3; }
        void f() { a=a+x; }
};
class B: protected A
{
     int x;
     protected:
        int b;
     public:
        B() { x=1; b=a; }
        void g() { x=a+b; }
};
class C: protected B
{
     int x;
     protected:
        int c;
     public:
        C() { x=a; c=b; }
        void h() { c=b+a; }
        void g() { B::g(); }
};
class D: private C
{
     int x;
     protected:
        int d;
     public:
        D() { x=b+c; d=0; }
        void k() { g(); }
        void m() { f(); }
};
class F: public D
{
     int x;
     public:
        F() { x=d; }
        void p() { x=c; }
        void q() { k(); }
        void r() { g(); }
};

void main() {
   F f;
   f.q();
   f.k();
   f.h();
   f.g();
   D d;
   d.k();
   d.h();
   d.g();
   C c;
   c.g();
   c.B::g();
   c.f();
   B b;
   b.g();
   b.f();
}
```

3. จงติดตามการทำงานของโปรแกรมว่าได้ผลลัพธ์อะไรบ้าง

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

class A {
   public:
   A() { cout<<"A"<<endl; }
   virtual ~A() {cout<<"a"<<endl;}
   void f() { cout<<"fA"<<endl; }
   virtual int g() { return 1; }
};

class B: public A {
   public:
   B() { cout<<"B"<<endl; }
   ~B() { cout<<"b"<<endl; }
   void f() { cout<<"fB"<<endl; }
   int g() { return 2; }
};

class C: virtual public B {
   static int c;
   public:
    C() { c=0; cout<<"C"<<endl; }
    ~C() { cout<<"c"<<endl; }
    void f() { cout<<"fC"<<endl; }
    int g() { return ++c; }
};
int C:: c;

class D: virtual public B {
   int d;
   public:
     D() {d=0; cout<<"D"<<endl; }
     ~D() { cout<<"d"<<endl; }
    void f() { cout<<"fD"<<endl; }
    int g() { return ++d; }
};

class E: public D, public C {
   public:
    E() { cout<<"E"<<endl; }
    ~E() { cout<<"e"<<endl; }
    void f() { cout<<"fE"<<endl; }
    int g() { return 5; }
};

class F: public D {
   public:
     F() { cout<<"F"<<endl; }
     ~F() { cout<<"f"<<endl; }
     void f() { cout<<"fF"<<endl; }
     int g() { return 6; }
};

void main() {
    A *a = new F;
    A *b = new E;

    if (a->g() == b->g())
      cout<<"Dog"<<endl;
    else
      cout<<"Cat"<<endl;

    a->f();
    C *c = new C[2];

    for (int i=0; i<2; i++) {
       cout<<c[i].g()<<endl;
       D d;
       cout<<d.g()<<endl;
    }

    delete[] c;
    b->f();
}
```

4. มี class อยู่ 4 class คือ Vehicle, Boat, Bus, Waterbus ทั้ง 4 คลาสมีความสัมพันธ์ดังรูป และมีรายละเอียดแต่ละคลาสดังนี้



class Vehicle	ประกอบด้วยข้อมูล	รหัสยานพาหนะ จำนวนที่นั่ง
class Bus สืบทอดมาจาก Vehicle	ประกอบด้วยข้อมูล	จำนวนล้อรถ ความเร็วสูงสุด
class Boat สืบทอดมาจาก Vehicle	ประกอบด้วยข้อมูล	ความเร็วสูงสุด

class Waterbus สืบทอดมาจาก Bus และ Boat

1. จงเขียนโปรแกรมภาษา C++ ในการออกแบบ class ทั้งสี่ตามความสัมพันธ์ดังรูป โดยให้ Vehicle เป็น Abstract class ส่วน class อื่นๆ ให้เป็น Concrete class และให้แต่ละคลาสให้สร้างข้อมูลไว้ในส่วนของ private โดยที่ รหัสยานพาหนะ จะต้องถูกเพิ่มค่าโดยอัตโนมัติเมื่อมีการสร้างอ็อบเจกต์ที่เป็น Vehicle การเรียกใช้ฟังก์ชันระหว่าง class จะต้องไม่มีความกำกวมเกิดขึ้น

2. ทุก class จงสร้าง
   2.1 constructor overloading ในการกำหนดค่าเริ่มต้นทุกค่า (รวมทั้งค่าของ based class)
   2.2 destructor ให้พิมพ์ข้อมูลทุกอย่างของ object ออกทางหน้าจอ
   2.3 printing operator overloading (<<) ในการพิมพ์ค่าทุกค่าออกทางจอภาพ โดยใช้หลักการ dynamic binding
   2.4 ฟังก์ชัน set ในการกำหนดค่าต่างๆโดยใช้หลักการ dynamic binding

3. class Waterbus จงสร้างฟังก์ชันเพิ่มเติมดังนี้
   3.1 copy constructor ในการก็อปปี้ค่าทุกค่า (รวมทั้งค่าของ based class)
   3.2 สร้างฟังก์ชันในการนับจำนวนอ็อบเจกต์ของ Waterbus อย่างอัตโนมัติ กรณีที่ไม่มีอ็อบเจกต์ของ Waterbus ในโปรแกรมเลย ก็ต้องสามารถบอกได้ด้วย
   3.3 function overloading ++ โดยค่าความเร็วสูงสุดของ Bus จะถูกบวกเพิ่มทีละ 2 และค่าความเร็วสูงสุดของ Boat จะถูกบวกเพิ่มทีละ 1 (ให้สร้างเป็น friend function)
   3.4 function overloading * จะได้ Waterbus คันใหม่ที่มีจำนวนที่นั่ง จำนวนล้อ และความเร็วสูงสุด เท่ากับค่าที่มากที่สุดของ Waterbus ทั้งสอง (ให้สร้างเป็นสมาชิกของคลาส)

3.5 assignment operator overloading ที่สามารถกำหนดค่าได้ถูกต้อง และให้พิมพ์คำว่า "copy ready" ออกทางจอภาพด้วย

3.6 ฟังก์ชัน setMaxSpeed ในการกำหนดค่าความเร็วสูงสุดของทั้งสองคลาสโดยใช้หลักการ function overloading

4. ใน main จงสร้าง

4.1 จงสร้าง a ให้เป็นอาร์เรย์ของพอยต์เตอร์ที่ชี้ไปยังคลาส Vehicle ให้มีขนาดเท่ากับ 4 และกำหนดให้ a[0] a[1] a[2] a[3] เป็นพอยเตอร์ที่ชี้ไปยังอ็อบเจกต์ของ Boat, Bus, Waterbus, Waterbus ตามลำดับ โดยมีการจองพื้นที่ให้กับอ็อบเจกต์เหล่านี้ด้วย

4.2 ให้กำหนดค่าให้กับสมาชิกทุกตัวใน a โดยใช้หลักการ dynamic binding

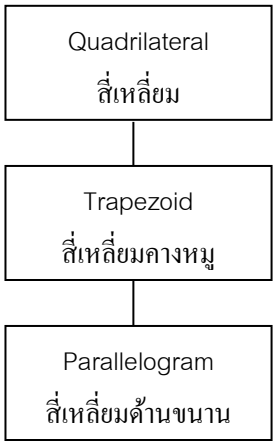
4.3 ให้พิมพ์ค่าของสมาชิกทุกตัวใน a โดยใช้หลักการ dynamic binding

4.4 ให้มีการเรียกใช้ ++ และ *

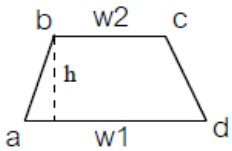
4.5 ให้จัดการคืนหน่วยความจำให้เรียบร้อยด้วย

(หมายเหตุ: ทุกๆ object จะต้องถูกสร้างและถูกทำลายได้อย่างถูกต้อง และน.ศ.สามารถสร้าง attributes และ function เพิ่มเติมได้ตามความเหมาะสม)

5. จงสร้างคลาส Point, Quadrilateral, Trapezoid, Parallelogram โดยคลาสสี่เหลี่ยมทั้ง 3 คลาสมีความสัมพันธ์กันดังรูป



Quadrilateral เป็นสี่เหลี่ยมใดๆ ที่มีด้าน 4 ด้าน มีมุม 4 มุม ทุกมุมรวมกันได้ 360 องศา



Trapezoid

ด้านตรงข้ามขนานกัน

Parallelogram

ด้านตรงข้ามขนานกันและยาวเท่ากัน

มุมตรงข้ามเท่ากัน

แต่ละคลาสมีข้อมูลดังนี้
(กำหนดให้สี่เหลี่ยมคางหมูและสี่เหลี่ยมด้านขนานนั้น ขนานกับแกน x เสมอ)

คลาส Point ประกอบด้วย - โคออร์ดิเนต x และ y

คลาส Quadrilateral ประกอบด้วย - จุด 4 จุด คือจุด a b c d (แต่ละจุดเป็นอ็อบเจกต์ของคลาส Point)
- ความยาวด้าน 4 ด้าน

คลาส Trapezoid สืบทอดมาจาก Quadrilateral มีข้อมูลเพิ่มเติมคือ
- ความสูง h

คลาส Parallelogram สืบทอดมาจาก Trapezoid

ข้อมูลทั้งหมดให้เก็บไว้ในส่วนของ private ส่วนฟังก์ชันให้เก็บไว้ที่ public

คลาส Point ให้สร้างฟังก์ชันต่างๆ เท่าที่จำเป็นเพื่อตอบคำถามข้ออื่นๆได้

คลาส Quadrilateral จงสร้าง
- ดีฟอลต์คอนสตรัคเตอร์ โดยกำหนดให้จุด a คือ (0,0) จุด b คือ (0,1) จุด c คือ (1,1) และจุด d คือ (1,0) และความยาวด้านทั้ง 4 เท่ากับ 1
- คอนสตรัคเตอร์ โดยรับค่าจุดทั้ง 4 จุด ส่วนความยาวด้านทั้ง 4 นั้นให้เรียกใช้ฟังก์ชัน setSide() เพื่อกำหนดค่าให้กับด้านทั้ง 4 โดยคำนวณจากจุดที่รับมา
- ฟังก์ชัน setSide() ในการกำหนดค่าความยาวของด้านทั้ง 4 ด้าน

คลาส Trapezoid จงสร้าง
- ดีฟอลต์คอนสตรัคเตอร์ โดยกำหนดค่าเช่นเดียวกับ Quadrilateral
- คอนสตรัคเตอร์ โดยรับค่าจุด 2 จุดคือจุด a และจุด b และรับค่าความยาวของด้าน w1 และ w2 จากนั้นให้คำนวณและเก็บค่าจุดทั้ง 4 และด้านทั้ง 4 ด้านให้ครบ รวมทั้งค่าความสูง h โดยการเรียกใช้ฟังก์ชัน setH()
- ฟังก์ชัน setH() ในการคำนวณและเก็บค่าความสูง h

คลาส Parallelogram จงสร้าง
- ดีฟอลต์คอนสตรัคเตอร์ โดยกำหนดค่าเช่นเดียวกับ Quadrilateral
- คอนสตรัคเตอร์ โดยรับค่าจุด 2 จุดคือจุด a และจุด b และรับค่าความยาวของด้าน w1 (ซึ่ง w1=w2) จากนั้นให้คำนวณและเก็บค่าจุดทั้ง 4 และด้านทั้ง 4 ด้านให้ครบ รวมทั้งค่าความสูง h
- โอเปอร์เรเตอร์โอเวอร์โหลดดิ้ง operator = ในการกำหนดค่าของ object ได้อย่างถูกต้อง และให้พิมพ์คำว่า “copy ready” ออกทางจอภาพด้วย

คลาสสี่เหลี่ยมทั้ง 3 คลาส
- จงสร้างก็อปปี้คอนสตรัคเตอร์โดยกำหนดค่าตามเงื่อนไขที่ระบุไว้ข้างต้น
- จงสร้างฟังก์ชัน set ในการกำหนดค่าให้กับข้อมูลทั้งหมดโดยใช้หลักการฟังก์ชันโอเวอร์ไรดิ้งและสามารถใช้หลักการไดนามิกไบดิ้งได้ด้วยโดยที่
    - Quadrilateral ให้รับข้อมูลจุดทั้ง 4 แล้วกำหนดค่าต่างๆให้ครบ
    - Trapezoid ให้รับจุด a และ b และความยาวด้าน w1 w2 แล้วกำหนดค่าต่างๆให้ครบ
    - Parallelogram ให้รับจุด a และ b และความยาวด้าน w1 แล้วกำหนดค่าต่างๆให้ครบ
- ฟังก์ชัน get ในการรีเทิร์นค่าของข้อมูลแต่ละตัว
- operator <<ในการแสดงค่าออกทางจอภาพ โดยใช้หลักการ dynamic binding

- ฟังก์ชันในการนับจำนวนอ็อบเจกต์ของคลาส Trapezoid และถึงแม้ว่าจะไม่มีอ็อบเจกต์ถูกสร้างขึ้นเลย ก็ต้องสามารถบอกได้ด้วยว่าไม่มีอ็อบเจกต์อยู่ในโปรแกรม

จงสร้างฟังก์ชันโอเวอร์โหลดดิ้งในการกำหนดค่าต่างๆ ให้กับสี่เหลี่ยมด้านขนาน

จงสร้างเฟรนด์ฟังก์ชันในการตรวจสอบว่าอ็อบเจกต์ของคลาส Trapezoid และอ็อบเจกต์ของคลาส Parallelogram นั้นมีจุด a เป็นจุดเดียวกันหรือไม่ ถ้าเป็นคนละจุดให้รีเทิร์นจุดกึ่งกลางระหว่างจุด a ทั้งสองออกจากฟังก์ชัน แต่ถ้าเป็นจุดเดียวกันให้รีเทิร์นจุด a นั้นออกจากฟังก์ชัน

จงสร้าง operator + ในการบวกสี่เหลี่ยมด้านขนาน 2 รูป โดยจะได้สี่เหลี่ยมด้านขนานใหม่ที่มีจุด a และ b มีค่าเท่ากับจุด a และ b ของสี่เหลี่ยมด้านขนานรูปแรก และมีความยาวด้านเท่ากับความยาวด้านของสี่เหลี่ยมด้านขนานทั้ง 2 บวกกัน

ใน main

- จงสร้างarray ของ object ของคลาส Parallelogram ให้มีขนาดเท่ากับ 3 และมีการกำหนดค่าเริ่มต้นโดยใช้ฟังก์ชัน set และให้แสดงผลลัพธ์ที่อยู่ใน array ทั้งหมดโดยใช้ฟังก์ชัน get
- จงสร้าง object ให้ไปเรียกใช้ฟังก์ชัน set โดยใช้หลักการ dynamic binding
- จงสร้าง array ของ object ของคลาส Trapezoid ให้มีขนาดเท่ากับ 4 และให้กำหนดค่าต่างๆของ object ใน array ด้วย และจงตรวจสอบว่า object ของคลาส Trapezoid ใน array ที่เพิ่งสร้างขึ้นมานั้น มี object ใดบ้างที่มีจุด a เป็นจุดเดียวกับจุด a ของสี่เหลี่ยมผืนผ้าซึ่งเป็น object ของคลาส Parallelogram ใน array ที่เพิ่งสร้างขึ้นมา โดยถ้ามีจุด a เป็นจุดเดียวกันให้แสดงผลลัพธ์เป็น index ของทั้งสอง array ส่วนกรณีที่ทั้งสอง object มีค่าจุด a ไม่เท่ากันให้ทำการตรวจสอบว่าผลรวมของความยาวด้านทั้งสี่ของ Trapezoid นั้นยาวกว่าผลรวมของความยาวด้านทั้งสี่ของ Parallelogram หรือไม่ ถ้ายาวกว่าให้พิมพ์ค่า index ของ array ที่เก็บ Trapezoid นั้น และพิมพ์คำว่า “Trapezoid” ออกทางจอภาพ แต่ถ้าไม่ยาวกว่าให้พิมพ์คำว่า “Hello Parallelogram”

(หมายเหตุ: ทุกๆ object จะต้องถูกสร้างและถูกทำลายได้อย่างถูกต้องและน.ศ.สามารถสร้าง attributeและ function เพิ่มเติมได้ตามความเหมาะสม)