Class Design Principles

Exercise 1

Define the responsibilities of the following classes!

a) **The Modem Class**

Listing 1: **The Modem Class**

```
1  class Modem
2  {
3    public:
4      void dial(String phoneNumber);
5      void hangup();
6      void send(char aCharacter);
7      char receive();
8  }
```

b) **The TStyle Class**

Listing 2: **The TStyle Class**

```
class TStyle : public TNamed,
  public TAttLine,
  public TAttFill,
  public TAttMarker,
  public TAttText {
  //...
   virtual Int_t     DistancetoPrimitive(Int_t px,
                                         Int_t py);
   Int_t             GetNdivisions(Option_t *) const;
   TAttText         *GetAttDate() ;
   Color_t           GetAxisColor(Option_t *) const;
   Color_t           GetLabelColor(Option_t *) const;
   Style_t           GetLabelFont(Option_t *) const;
   Float_t           GetLabelOffset(Option_t *) const;
   Float_t           GetLabelSize(Option_t *) const;
   Color_t           GetTitleColor(Option_t *) const;
   Style_t           GetTitleFont(Option_t *) const;
   Float_t           GetTitleOffset(Option_t *) const;
   Float_t           GetTitleSize(Option_t *) const;
   Float_t           GetTickLength(Option_t *) const;

   Float_t           GetBarOffset() const ;
   Float_t           GetBarWidth() const ;
   Int_t             GetDrawBorder() const ;
   Float_t           GetEndErrorSize() const ;
   Float_t           GetErrorX() const ;
   Bool_t            GetCanvasPreferGL() const ;
   Color_t           GetCanvasColor() const ;
   Width_t           GetCanvasBorderSize() const ;
   Int_t             GetCanvasBorderMode() const ;
   Int_t             GetCanvasDefH() const       ;
   Int_t             GetCanvasDefW() const       ;
   Int_t             GetCanvasDefX() const       ;
   Int_t             GetCanvasDefY() const       ;
   Int_t             GetColorPalette(Int_t i) const;
   Int_t             GetColorModelPS() const     ;
   Float_t           GetDateX()  const           ;
   Float_t           GetDateY() const            ;
   // ... 200 more functions including
   //     the setters for the above getters ...
   void              SaveSource(const char *filename,
                                Option_t *option=0);
   //...
};
```

Exercise 2

Imagine we are writing a small piece of software to draw various geometric objects.

Listing 3: **The Square/Circle Problem**

```
1  //--shape.hh-----------------------
2  enum ShapeType {circle, square};
3
4  struct Shape
5  {
6     ShapeType itsType;
7  };
8
9  //--circle.hh----------------------
10 struct Circle
11 {
12    ShapeType itsType;
13    double itsRadius;
14    Point itsCenter;
15 };
16
17 void DrawCircle(Circle*);
18
19 //--square.hh----------------------
20 struct Square
21 {
22    ShapeType itsType;
23    double itsSide;
24    Point itsTopLeft;
25 };
26
27 void DrawSquare(Square*);
28
29
30 //--DrawAllShapes.cc---------------
31 void DrawAllShapes(Shape* list[], int n)
32 {
33    int i;
34    for (i=0; i<n; i++)
35    {
36      Shape* s = list[i];
37      switch (s->itsType)
38      {
39      case square:
40        DrawSquare((struct Square*)s);
41      break;
42      case circle:
43        DrawCircle((struct Circle*)s);
44      break;
45      }
46    }
47 }
```

a) How many responsibilities has `DrawAllShapes` in Listing 3?

b) We are adding a new class `Triangle` and we want it to be drawn as well. How does Listing 3 adapt to this?

Exercise 3

Assume 2 classes representing 2 related geometric entities.

Listing 4: **The Square/Rectangle Problem**

```
1  //--Rectangle.hh-----------------------------------------------
2  enum GeoType { Rectangle, Square };
3
4  class Rectangle
5  {
6    public:
7      virtual void SetWidth(double w)  {itsWidth=w;}
8      virtual void SetHeight(double h) {itsHeight=h;}
9      double       GetHeight() const   {return itsHeight;}
10     double       GetWidth() const    {return itsWidth;}
11     GeoType itsType;
12   private:
13     double itsHeight;
14     double itsWidth;
15
16 };
17
18 //--Square.hh-------------------------------------------------
19 class Square : public Rectangle
20 {
21   public:
22   virtual void SetWidth(double w);
23   virtual void SetHeight(double h);
24 };
25
26 void Square::SetWidth(double w)
27 {
28   Rectangle::SetWidth(w);
29   Rectangle::SetHeight(w);
30 }
31
32 void Square::SetHeight(double h)
33 {
34   Rectangle::SetHeight(h);
35   Rectangle::SetWidth(h);
36 }
```

Consider the following use of Rectangle and Square:

Listing 5: Using Square and Rectangle

```
1    void g(Rectangle& r)
2    {
3      r.SetWidth(5);
4      r.SetHeight(4);
5      assert(r.GetWidth() * r.GetHeight()) == 20);
6    }
```

a) What will happen if Listing 5 is called with a `Square` or a `Rectangle` object?

b) Given the design in Listing 4, what counter-measures are necessary to make Listing 5 work?

Exercise 4

Given the following `Lamp` class definition:

Listing 6: A Lamp class

```
1 class Lamp
2 {
3   public:
4     void TurnOn();
5     void TurnOff();
6 };
```

a) Write or sketch a `Button` class that turns `Lamp` on and off!

Disclaimer

All Source code snippets were adapted from the book:

author	Martin, Robert C. and Newkirk, James W. and Koss, Robert S.
title	Agile Software Development
publisher	Prentice Hall
year	2003
note	http://www.objectmentor.com/resources/publishedArticles.html