1.124 Quiz 1

Thursday October 8, 1998

Time: 1 hour 15 minutes
Answer all questions. All questions carry equal marks.

Question 1. The following code is to be built and run as follows:

Compile as	g++ -c Point.C
	g++ -c myprog.C
Link as	g++ -o myprog myprog.o Point.o
Run as	myprog

Would you expect to see
(a) a compile-time error?
(b) a link-time error?
(c) a run-time error?
(d) none of the above errors.

Explain briefly.

Answer:

Point.h

```
class Point {
  private:
 int x, y;
  public:
 Point() {}
 void set_coords(int x, int y);
};
```

Point.C

```
#include "Point.h"

void Point::set_coords(int x, int y) {
 // Assume that this sets the private data.
}
```

myprog.C

```
#include "Point.h"
extern Point a;

int main() {
 a.set_coords(2,3);
 return 0;
}
```

Question 2. Fill in the body of the member function, *set_coords*, so that it properly sets the private member data in class *Point*.

Answer:

```
void Point::set_coords(int x, int y) {




    }
```

Question 3. Write a member function, *access_x*, that can be used **either to set or to get** the value of the private member, *x*. Your function should work with the following code:

```
class Point {
    private:
  int x, y;

    public:
  Point() {}
```

Answer:

```
};
```

```
int main() {
  Point a;
  int i;

  a.access_x() = 5;
  i = a.access_x();
  return 0;
}
```

Question 4. Is the following class declaration valid? Explain briefly.

```
class Point {
   private:
    int x, y;
    Point a;

    public:
    Point() {}
};
```

Answer:

Question 5. Examine the following code carefully and explain the **exact** sequence of constructor calls.

```
class Point {
   private:
    int x, y;

    public:
    Point() { x = y = 0; }                        // Constructor #1
    Point(int ix, int iy) { x = ix; y = iy; }     // Constructor #2
    Point(const Point& p) { x = p.x; y = p.y; }   // Constructor #3
    ~Point() {}
};

Point foo(Point p) {
  static Point c(p);
  return c;
}

int main() {
  Point a(2,3);
  Point b;

  b = foo(a);
  a = foo(b);
  return 0;
}
```

Answer:

Constructor #	Reason

Question 6. Examine the following code carefully and draw a diagram to illustrate the data structures that it creates. Be sure to indicate all data types on your diagram.

```
class Point {
   private:
  int x, y;

   public:
  Point() { x = y = 0; }
  Point(int ix, int iy) { x = ix; y = iy; }
};

int main() {
 Point **p;
 p  = new Point *;
 *p = new Point[3];
 for (int i = 0; i < 3; i++) {
   (*p)[i] = Point(i, 0);
 }
 return 0;
}
```

Answer:

Question 7. In the following code, circle the statements that will produce compilation errors. Explain your reasoning.

```
class Point {
   private:
  int x, y;
  static int i;

   public:
  Point() {}
  static void set_data(int a, int b, int c) {
   x = a;  y = b;  i = c;
  }
};

int Point::i = 0;

int main() {
 Point::i++;
 Point *p = new Point;
 p->set_data(2,3,1);
 delete p;
 return 0;
}
```

Answer:

Question 8. Identify and explain the errors, if any, in the following code.

#include <iostream.h>

```
class Point {
  private:
 const int x, y;

   public:
 Point(int ix = 0, int iy = 0) : x(ix), y(iy) { }
 void print() const { cout << "(" << x << ", " << y << ")" << endl; }
};

int main() {
 const int i = 0;
 const Point a;
 Point * const p = new Point(2,3);
 const Point& b = a;

 b.print();
 p->print();

 delete p;
 return 0;
}
```

Answer:

Part II
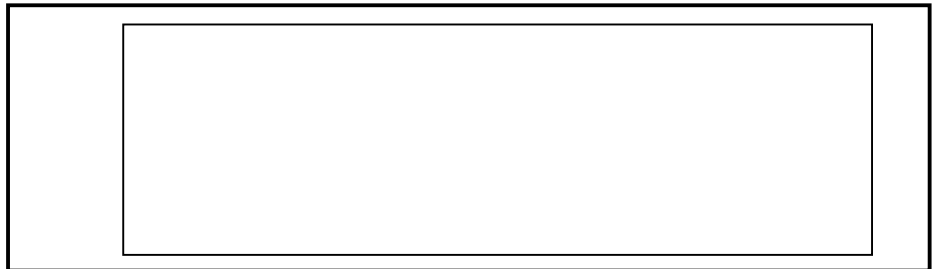
In the code below answer the following questions:
#include <iostream.h>

```
class Point {                          // Line 1
 private:
   int x, y;
 public:
   Point(){x=0;y=0;}
   Point(int a, int b){x=a;y=b;}
   ~Point(){};
   ??? operator+(Point&);
   ??? operator<<(???);
   ??? operator[](???);
};
```
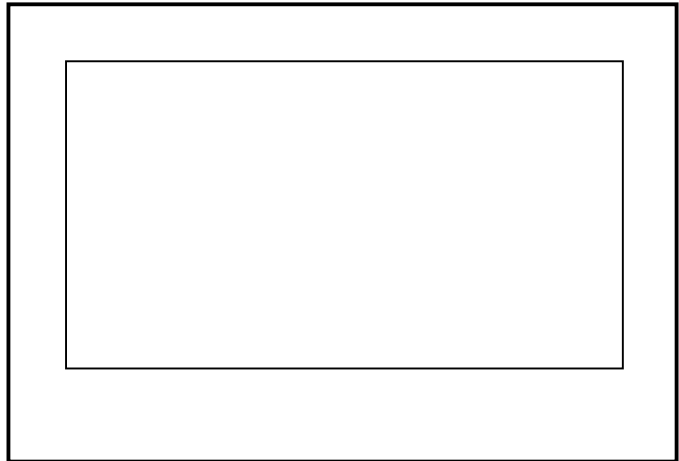
Question 9. Give the body of the code i.e. the definition, for *operator+* including the return type so that the code adds together two Point objects so that the following code will work.

```
main(){
   Point a(2,3),b(3,3);
   Point c = a + b;
}
```
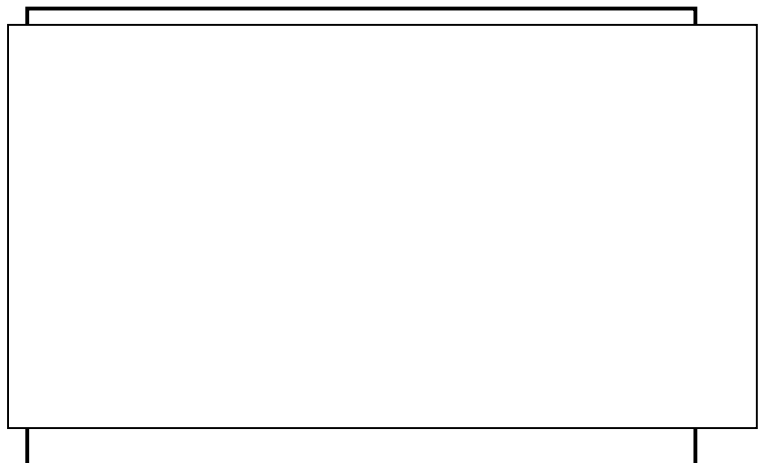
Question 10 What changes would you make to make the following code work. Make any assumptions you need to about how the code should interpret the meaning of the statements below.

```
main(){
   Point a(2,3),b(3,3);
   Point c =  b + 7;
}
```

Question 11   Give the body of the code to overload operator[] which should allow the following code to work

```
main(){
   Point a(2,3);
   a[0] = 4;
   a[1] = 2*a[0];
}
```

Question 12 Give the body of
the code to overload the
output operator<< for the
Point object so that the
following code will work.

```
#include <iostream.h>
main(){
  Point a(2,3), b(3,3);
  cout << a << b;
}
```

Answer:

For the code below answer the following questions:

```
class Shape{
   private:
  Point center;
   public:
  Shape(const Point&);
  virtual ~Shape();
  void set_center(const Point&);
  Point get_center(){return center;}
  virtual void print(){cout << "Center at  "<<center[0]<<","<<center[1]<< endl;};
};

class Circle:public Shape
{
   private:
       int radius;
   public:
       Circle();
    Circle(int,int, int)?????
    Circle(Point&,int)?????
   Circle(Circle&) ????
    ~Circle();
    void print(){};
};
```

Question 13 How many bytes of data are
needed for the variables in a Circle
object?

Question 14 Are there any inline functions in class Shape? If so name them.

Question 15 Write down all the functions in class Circle that can access **center** in the private part of Shape.

Question 16 Write the definition of the copy constructor for class Circle.

Question 17 How would we make the class Shape an abstract class? What does an abstract class mean?

Question 18 Will the following code compile without errors?

```
main(){
  int  x=5.0,y=6.0;
  int radius = 2;
  Shape* a;
  a = new Circle(x, y, radius);
  Circle b(*a);
  a = &b;
  a->print();
  delete a;
};
```

Question 19 What would the code above print out if print() was **NOT** a virtual function in class Shape?

Question 20 How many bytes of memory would be released on line 10 if the destructors were NOT virtual.

```
main(){
  int  x=5.0,y=6.0;
  int radius = 2;
  Shape* a;
  a = new Circle(x, y, radius);
  Circle b(*a);
 // a=&b;  this line taken out
  a->print();
  delete a;
};
```

File point.h

```cpp
#include <iostream.h>
class Point {                    // Line 1
 private:
  int x, y;
 public:
  Point(){x=0;y=0;}
  Point(int a, int b=0):x(a),y(b){};
  Point(const Point&);
  Point operator+(const Point&)const;
  int& operator[](int);
  ~Point(){};
  friend ostream& operator<<(ostream&, Point&);
};
```

file shape.h

```cpp
#include "point.h"
#include <iostream.h>
class Shape{
   private:
 Point center;
   public:
               Shape(){center[0]=0;center[1]=0;}
 Shape(const Point&);
 virtual ~Shape(){};
 void set_center(const Point&);
 Point get_center(){return center;}
 virtual void print(){cout <<"center at "<<center[0] <<","<< center[1];};
};
```

file circle.h

```cpp
#include "shape.h"
class Circle:public Shape
{
   private:
        int radius;
   public:
        Circle():Shape(),radius(0){};
   Circle(int a,int b, int r):radius(r){set_center(Point(a,b));}
   Circle(Point& p,int r):Shape(p),radius(r){}
   Circle(Circle& c){radius = c.radius;set_center(c.get_center());}
   virtual ~Circle();
        void print(){Shape::print(); cout<<" with radius "<< radius<<endl;};
```

```
};
```

```
Point::Point(const Point& p)
{
        x = p.x;
        y = p.y;

}
Point Point::operator +(const Point& p)const{
        return Point(x+p.x,y+p.y);
}

int& Point::operator [](int i){
        if(i == 0) return x;
        else if(i == 1) return y;
        else {
                cout << "error in index for Point object"<< endl;
        }
}

ostream& operator<<(ostream& o, Point& p){
                o << "x = "<< p[0] <<", y = "<< p[1] << endl;
                return o;
}
```

```
#include "shape.h"

void Shape::set_center(const Point& p){
        center = p;
}
```

```
#include "circle.h"

Circle::~Circle(){}
```

File shape_test.C

```cpp
#include "point.h"
#include "shape.h"
#include "circle.h"
int main(int argc, char* argv[])
{
  Point a(2,3), b(3,3);
  Point c =  b + 7;
  c = a + b;

  cout << a << b;
  Circle d(4,5,6);
  d.print();
  Circle e(d);
  e.print();
  Shape* sp;
  sp = &e;
  sp->print();
        return 0;
}
```