Classes and objects II

Dr. Ahmed ElShafee

Agenda

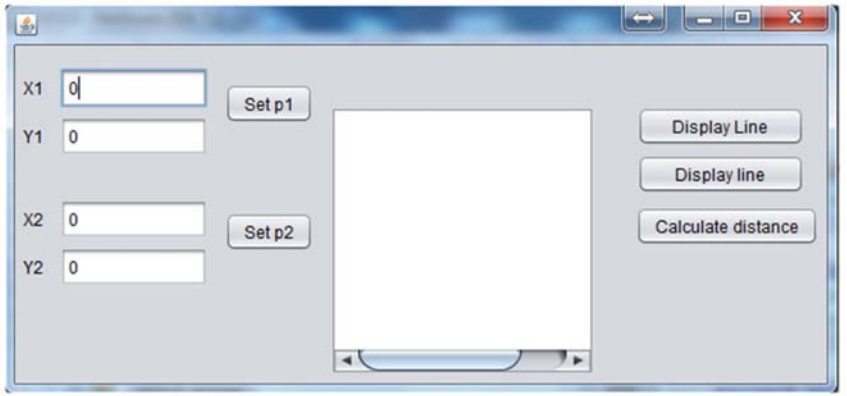
1. *
2. *
3. *

Method Overloading

- Java does not allow two variables to have the same name in the same method.
- Although two methods should have unique names in the same program, a class can have different methods with the same name if you follow some rules.
- The ability to have various methods with the same name in the same program is referred to as method overloading.
- To perform overloading, the methods must have different numbers or different type(s) of arguments.

OO13

Check lab manual



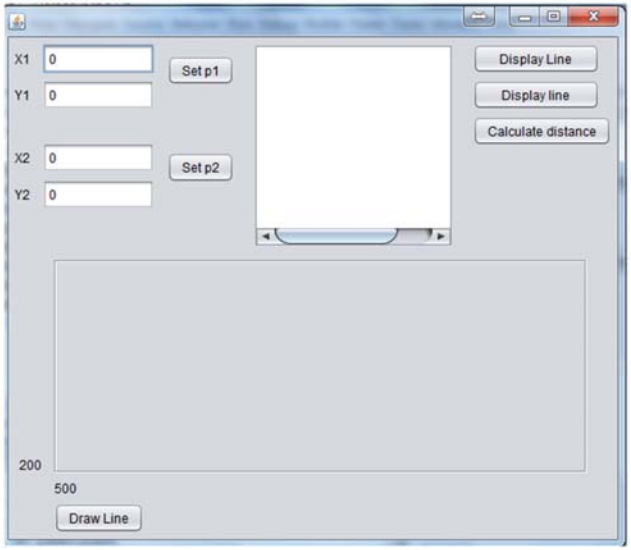
OO13

-

LineXY
int x1 = 0, x2 = 0, y1 = 0, y2 = 0;
void setP1(int x, int y) void setP2(int x, int y) String getP1() void getP1(int p[]) String getP2() void getP2(int p[]) double getLength()

OO14

Check lab manual



OO14

-

LineXY
int x1 = 0, x2 = 0, y1 = 0, y2 = 0;
void setP1(int x, int y) void setP2(int x, int y) int getPX(String P) int getPY(String P) String getP1() void getP1(int p[]) String getP2() void getP2(int p[]) double getLength()

Class Construction

- A constructor is a special method that is created when the object comes to life.
- This particular method holds the same name as the class and it initializes the object whenever that object is created.
- When you create a class, if you do not create a constructor, the compiler creates one for you; this is useful because it lets all other objects of the program know that the object exists.
- This compiler-created constructor is called the default constructor.
- If you want, you can create your own constructor.

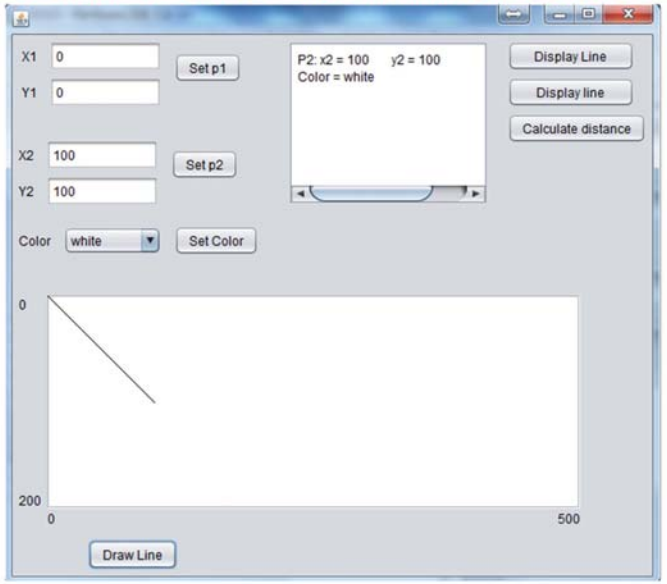
- To create a constructor, declare a method that holds the same name as the class. The method must not return any value.

```
class PointXY{
PointXY() {
}
}
```

- When you declare an instance of the class, whether you use that object or not, a constructor for the object is created.
- When an instance of a class has been declared, the default constructor is called, whether the object is used or not.

OO15

Check lab manual



OO15

-

LineXY
int x1, x2, y1, y2; String color;
LineXY() LineXY(int x1,int y1,int x2,int y2, String color) void setP1(int x, int y) void setP2(int x, int y) void setColor(String color) int getPX(String P) int getPY(String P) String getP1() void getP1(int p[]) String getP2() void getP2(int p[]) double getLength() String getColor()

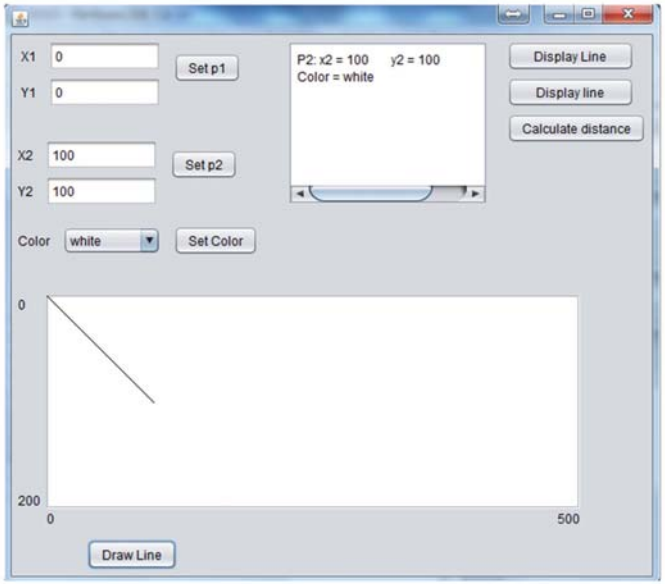
An Object as a Field in class

- class as a member variable of another class, simply declare its variable as you would proceed with any of the member variables we have declared so far.

```
public class LineXY {
   PointXY p1=new PointXY();
}
```

OO16

Check lab manual



OO16

-

PointXY

public String
coordinateType="cartesian ";
int X;
int Y;

void initPointXY()
int getX()
int getY()
String getCordinateType()
void setX(int X)
void setY(int Y)
void setCoordinateType(String st)

LineXY

PointXY p1=new PointXY();
PointXY p2=new PointXY();
Color color;

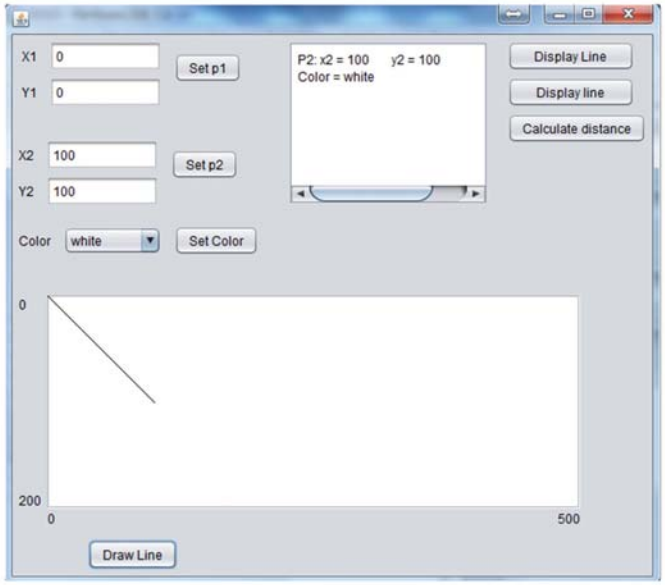
LineXY()
LineXY(int x1, int y1, int x2, int y2, String color)
void setP1(int x, int y)
void setP2(int x, int y)
void setColor(String color)
int getPX(String P)
int getPY(String P)
String getP1()
void getP1(int p[])
String getP2()
void getP2(int p[])
double getLength()
String getColor()

Passing an Object as Argument

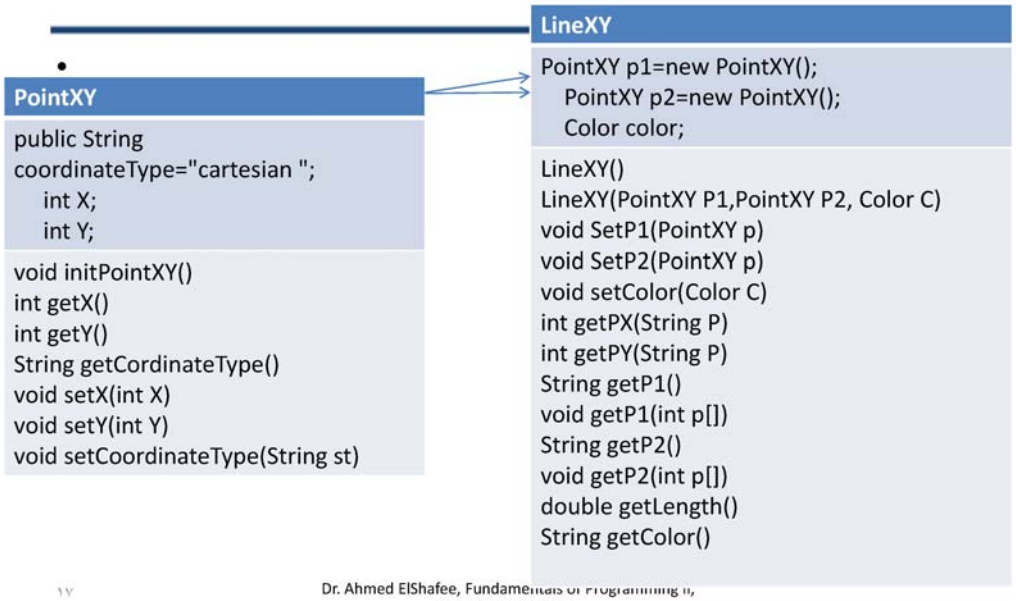
- Once a class has been created, it can be used like any other variable.
- For example, its variable can be passed as argument to a method of another class.
- When a class is passed as argument, its members are available to the method that uses it.

OO17

Check lab manual



OO17



Returning an Object From a Method

- Like a value from a regular type, you can return a class value from a method of a class.
- To do this, you can first declare the method and specify the class as the return type

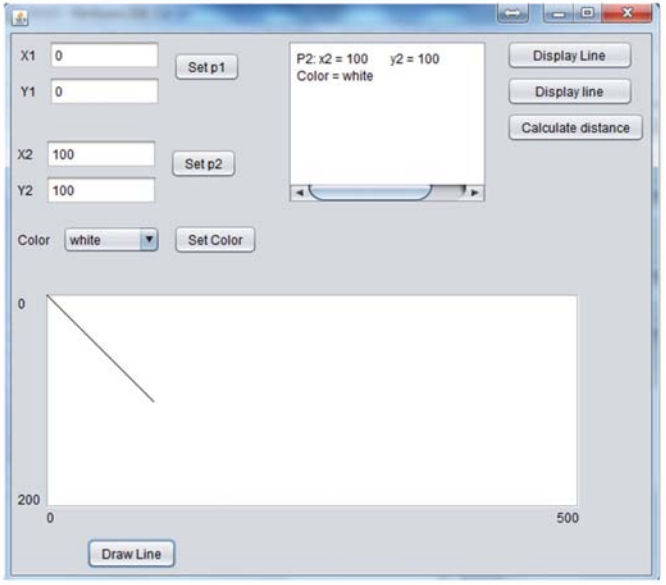
```
public class LineXY {
…
PointXY getP1() {
        return p1;
   }
```


- After implementing the method, you must return a value that is conform to the class, otherwise you would receive an error when compiling the application.

```
public class LineXY {
…
Color getColor() {
        return color;
   }
}
```


OO18

Check lab manual



OO18

•

PointXY

public String
coordinateType="cartesian ";
int X;
int Y;

void initPointXY()
int getX()
int getY()
String getCordinateType()
void setX(int X)
void setY(int Y)
void setCoordinateType(String st)

LineXY

PointXY p1=new PointXY();
PointXY p2=new PointXY();
Color color;

LineXY()
LineXY(PointXY P1,PointXY P2, Color C)
void SetP1(PointXY p)
void SetP2(PointXY p)
void setColor(Color C)
PointXY getP1()
PointXY getP2()
double getLength()
Color getColor()

Passing a Class as its Own Argument

- An instance of a class can be passed as an argument to one of its own methods
- To do this, you primarily pass the argument as if it were any class

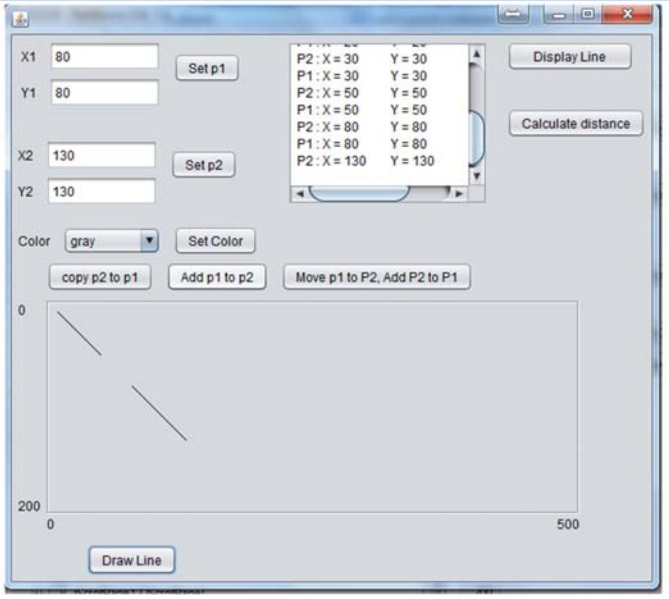
```
public class PointXY {
…
void add(PointXY p)
   {
      X=p.getX()+X;
      Y=p.getY()+Y;
   }
   }
```


- Instead of first declaring a variable of the class and initializing it, you can create an instance of the class in the parentheses of the calling method.
- To do this, you may need a constructor that can specify the values of the fields of the class so the argument can be rightfully initialized.

```
public class PointXY {
…
void copy(PointXY p)
   {
      X=p.getX();
      Y=p.getY();
   }
}
```


OO19

Check lab manual



OO19

PointXY

public String coordinateType="cartesian ";
int X;
int Y;

void initPointXY()
int getX()
int getY()
String getCordinateType()
void setX(int X)
void setY(int Y)
void setCoordinateType(String st)
void add(PointXY p)
void copy(PointXY p)

LineXY

PointXY p1=new PointXY();
PointXY p2=new PointXY();
Color color;

LineXY()
LineXY(PointXY P1,PointXY P2, Color C)
void SetP1(PointXY p)
void SetP2(PointXY p)
void setColor(Color C)
PointXY getP1()
PointXY getP2()
double getLength()
Color getColor()

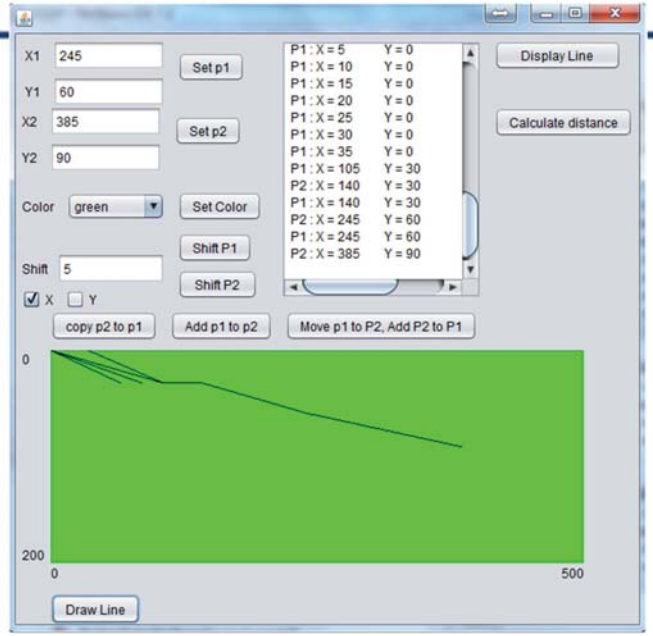
Returning a Class From its Own Method

- You can create a method in a class that returns an instance of the class.
- To start, on the left side of the method,

```
class Point {
      Point MethodName() {
      }
}
```

- There are various ways you can deal with the method. If you want to return a new value of the class, you can declare an instance of the class, initialize it, and then return it
- Remember that, to call the method, if it is not static, you will need to declare an instance of the class from where you are calling the method.

OO20 Check lab manual



OO20

PointXY

public String coordinateType="cartesian ";
int X;
int Y;

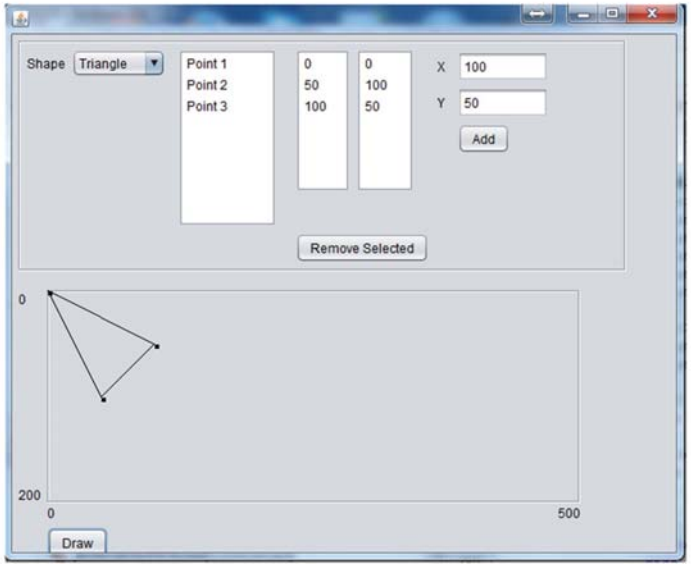
void initPointXY()
int getX()
int getY()
String getCordinateType()
void setX(int X)
void setY(int Y)
void setCoordinateType(String st)
void add(PointXY p)
void copy(PointXY p)
PointXY shift(int s,boolean x, boolean y)

LineXY

PointXY p1=new PointXY();
PointXY p2=new PointXY();
Color color;

LineXY()
LineXY(PointXY P1,PointXY P2, Color C)
void SetP1(PointXY p)
void SetP2(PointXY p)
void setColor(Color C)
PointXY getP1()
PointXY getP2()
double getLength()
Color getColor()

Check Lab manual

- 

- **PointXY**

public String coordinateType="cartesian ";
int X;
int Y;

void initPointXY()
int getX()
int getY()
String getCordinateType()
void setX(int X)
void setY(int Y)
void setCoordinateType(String st)
void add(PointXY p)
void copy(PointXY p)
PointXY shift(int s,boolean x, boolean y)

Thanks,

See you next Lecture, isA