Object Oriented III

Dr. Ahmed ElShafee

Array of objects

- The card class
- There are 52 cards in a deck; each belongs to one of four suits and one of
- 13 ranks. The suits are Spades, Hearts, Diamonds and Clubs (in descending
- order in Bridge). The ranks are Ace, 2, 3, 4, 5, 6, 7, 8, 9, 10, Jack, Queen and
- King.

- After creating a class as follow

```
class Card
{
int suit, rank;
public Card() {
        this.suit = 0; this.rank = 0;
}
public Card(int suit, int rank) {
        this.suit = suit; this.rank = rank;
}}
```

- You can simple create an object of Card class

```
Card threeOfClubs = new Card(0, 3);
```


- Or simple create the whole set of cards

```
Card[] cards = new Card[52];
```

- Keep on mind to define a mapping between a sequence of numbers and the things you want to represent.

```
Spades → 3
Hearts → 2
Diamonds → 1
Clubs → 0
```

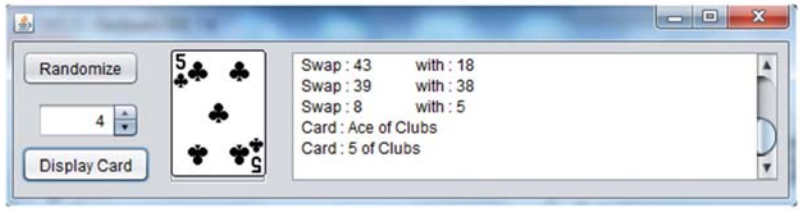
```
Jack → 11
Queen → 12
King → 13
```

OO22

Check Lab Manual

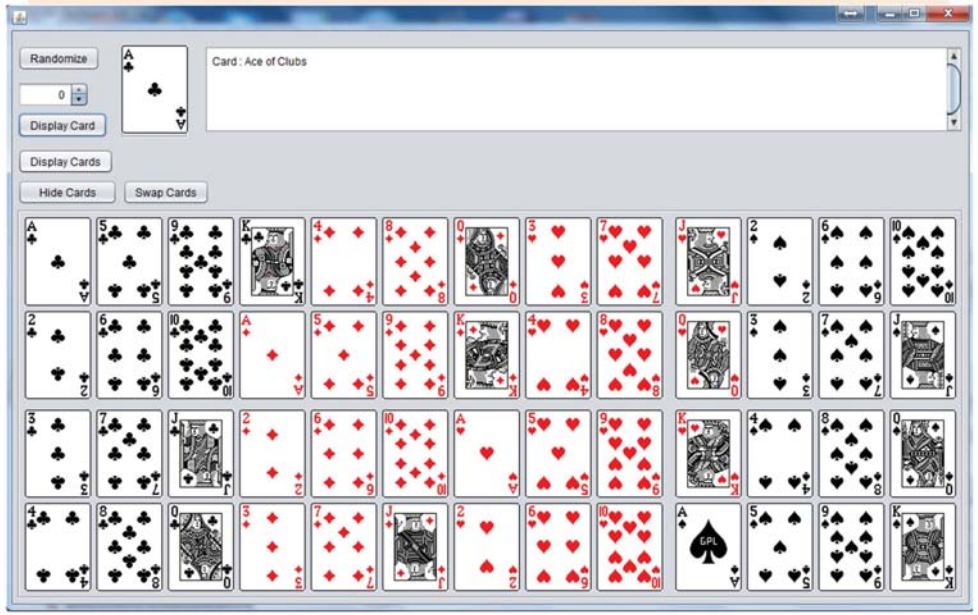
OO23

Check Lab Manual



OO24

Check Lab Manual



OO25

Check Lab Manual

Assignment

NewsPaper
String name=""; int balance; double netPrice; double sellPrice;
NewsPaper() void initNewsPaper(String name, int balance, double netPrice,double sellPrice) int getBalance() double getNetPrice() double getSellPrice() void sellNewsPaper()

NewsPapersBooth

double costInHand=0;
double sellInHand=0;
double profit=0;
NewsPaper[] np=new NewsPaper[5];

NewsPapersBooth() void setNewPaperData(int item,String name, int balance, double netPrice,double sellPrice) void sellNewsPeper(int item) double getNewsPeperBalance(int item) double getCostInHand() double getSellInHand() double getProfit()

Serialization (cont,..)

ObjectInputStream

The ObjectInputStream class enables you to read Java objects from InputStream's instead of only bytes.

You wrap an InputStream in a ObjectInputStream and then you can read objects from it.

```
ObjectInputStream input = new ObjectInputStream(
                              new FileInputStream("object.data"));

MyClass object = (MyClass) input.readObject();
//etc.

input.close();
```


Serialization (cont,..)

For this example to work the object you read must be an instance of MyClass, and must have been serialized into the file "object.data" via an ObjectOutputStream.

Serialization (cont,..)

ObjectOutputStream

The ObjectOutputStream class enables you to write Java objects to OutputStream's instead of only bytes.

You wrap an OutputStream in a ObjectOutputStream and then you can write objects to it.

```
ObjectOutputStream output = new ObjectOutputStream(
                              new FileOutputStream("object.data"));

MyClass object = new MyClass();

output.writeObject(object);
//etc.

output.close();
```

Serialization (cont,..)

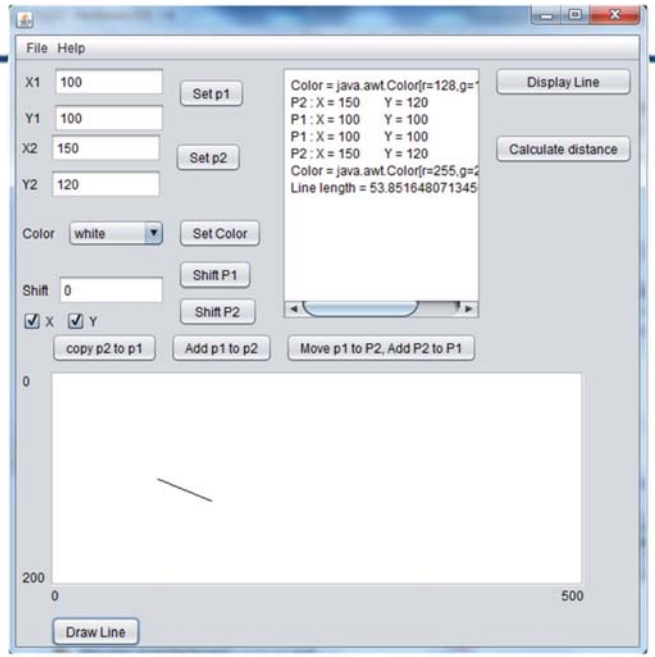
- This serialized object can now be read via an ObjectInputStream.
- To make objects from a class implements Serializable
- Add the “implements Serializable”, in class definition

```
public class LineXY implements Serializable {

    PointXY p1;
    PointXY p2;
    Color color;
}
```


OO26

Check lab manual



OO26

-

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)
```

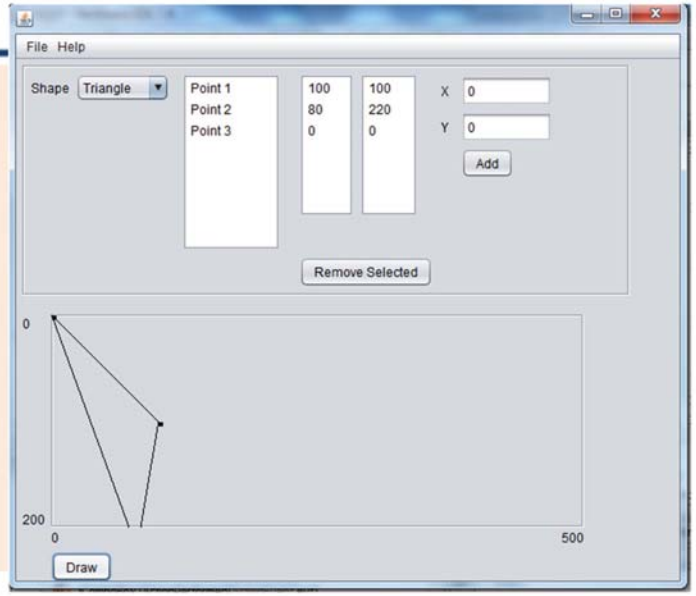
LineXY

```
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()
```


OO27

Check lab manual



OO27

-

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)

Public and Private Fields

- In a class, you can control how fields are accessed outside the class.
- You can create some fields that can be accessed outside the class and you can create other fields that other classes cannot access.
- To create a field that can be accessed by only members of the same class, precede its data type with the **private** keyword.

```
public class House{
String propertyNumber;
private String kitchenCharacteristics;
byte Stories;
int bedrooms;
private boolean bathroomNeedCleaning;
double Value;
}
```


- To create a field that can be accessed by members of the same class and members of other classes, precede its data type with the **public** keyword.

```
public class House{
public String propertyNumber;
private String kitchenCharacteristics;
byte Stories;
public int bedrooms;
private boolean bathroomNeedCleaning;
double Value;
}
```


- by default, the members of a Java class are public.
- This means that if you do not use the public or the private keyword, a field is automatically made public.

Public and Private Methods

- you can control the access level of a method.
- To do this, you can precede the return value or void of a method with **private** or **public**.
- When a method has been marked as private, it can be accessed by methods of the same class but other classes cannot call that method.
- If a method was created as public, other methods of the same class can access it and instances of other classes can call it.
- the methods of a class are public by default.
- Therefore, if you want to hide a member of a class, you must explicitly mark it as private.

Accessing private fields/methods

- We saw that you could create (or make) a field as private to prevent its access from the outside of its class. Here is an example:

```
public class Square {
private double side;
}
```

- After making a field private, other classes can neither specify its value not retrieve
- To solve this problem, you create a property for the field. There are two types of properties: a reader and a writer.

Read property

- A property is referred to as *read* if its role is only to make available the value of the field it represents.
- To create a read property , create a method with the following characteristics:

  a. Because the method is used to communicate with the outside world, it must be created as **public**

  b. The method must return the same type of value as the field it represents

  c. The name starts with *get* and ends with the name of the field with the first letter in upper case

  d. The method does not take any argument

  e. The method returns the field it represents


```
public class Square{
private double side;
// This is a read property
public double getSide() {
return side;
}
}
```

- A read property is also referred to as read-only property because the clients of the
- class can only retrieve the value of the property but they cannot change it.
- Once a read property has been created, other classes can access it

A Property Writer

- instead of, retrieving the value of a field of a class, you may want external classes to be able to change the value of that member.
- Continuing with our policy of hiding a field as private, you can create another type of property.
- a property is referred to as write if it can change (or write) the value of its corresponding field.
- To build a write property, create a method using the following characteristics:

a. Because the method is used to communicate with the outside world, it must be created as public

b. Because the method is used by outside classes to assign a value to the field, the method is (or should be) created as **void**

c. The name starts with *set* and ends with the name of the field with the first letter in upper case

d. The method takes one argument that is the same type as the field it refers to

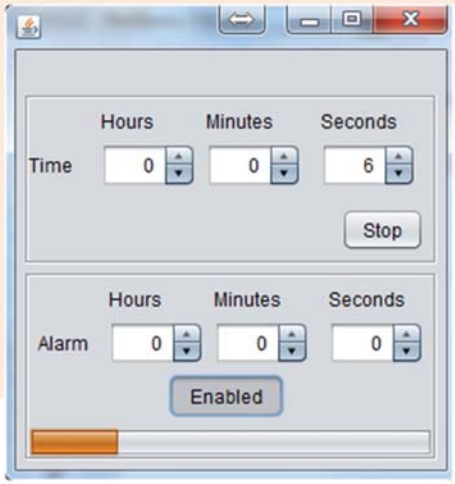
e. In the body of the method, assign the argument to the field it represents


```
public class Square {
private double side;
// This is a write property
public void setSide(double value) {
side = value;
}
}
```

- As you see, clients of a class can change the corresponding field of a member variable through the property writer.
- Based on this relationship, it is not unusual for a client of a class to make an attempt to "mess" with a field.
- For example, an external object can assign an invalid value to a member variable.

OO28

Check Lab manual



-

CTime
private int s,m,h;
CTime() CTime(int h, int m, int s) void tick() int getMinutes() int getSeconds() int getHours() void setHours(int h) void setMinutes(int m) void setSeconds(int s) long convert2Minutes() Boolean compareToM(CTime ct)