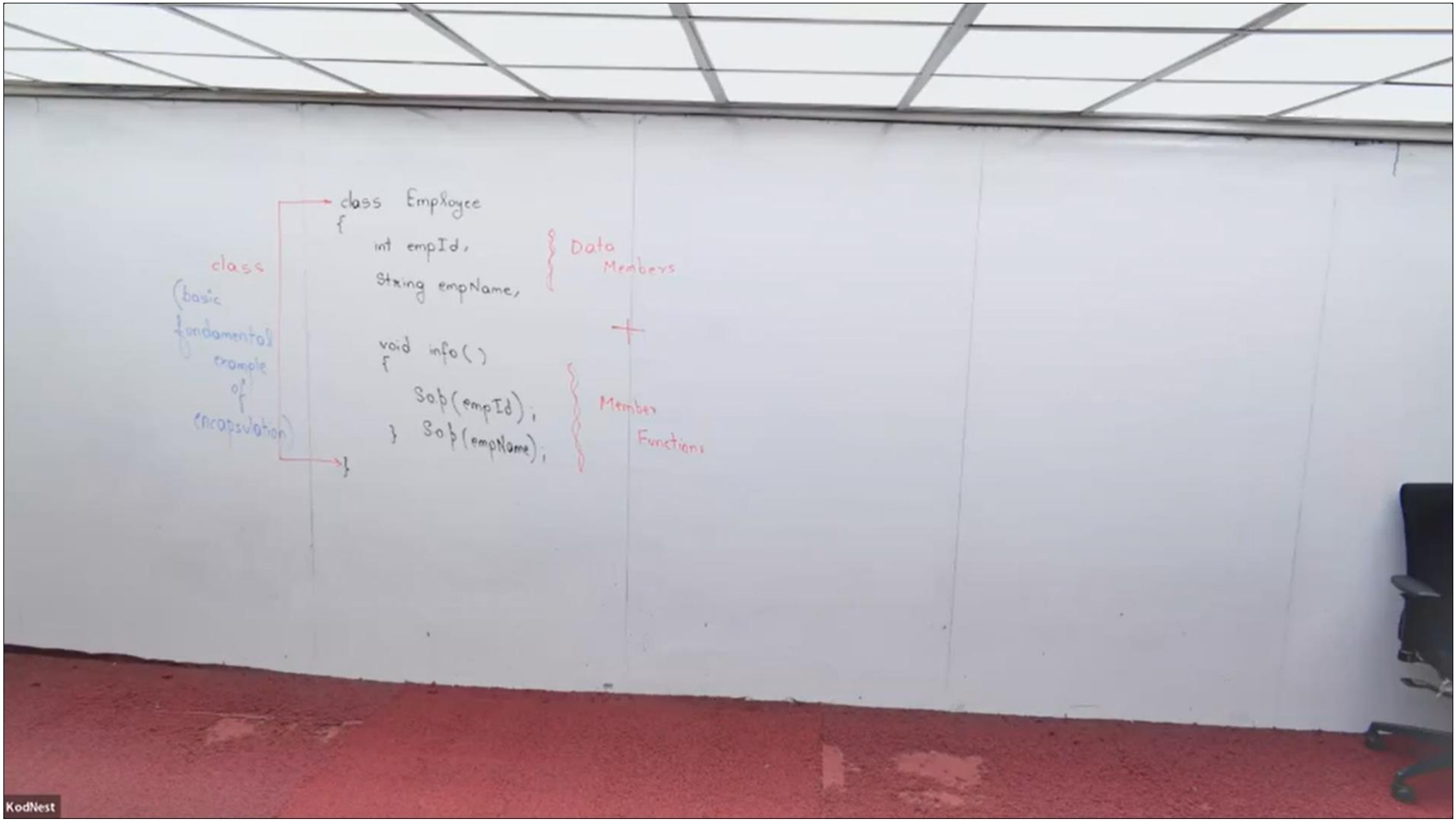

class Employee
{
int empId,
String empName,
void info()
{
Sop(empId);
} Sop(empName);
}
Data Members
+
Member Functions
class
(basic fundamental example of encapsulation)
KodNest

OOP

creating an object :- SS1

class :- A class is a blue print of an object which consists of properties (Data members) and behaviors) member function of an object

object :- Is the programming representation of the class and its memory is allocated in the heap segment.

Java is an object oriented programming language.
every element in java will be represented as class and object.

* as it is object oriented programming language it follows oops principle. i.e

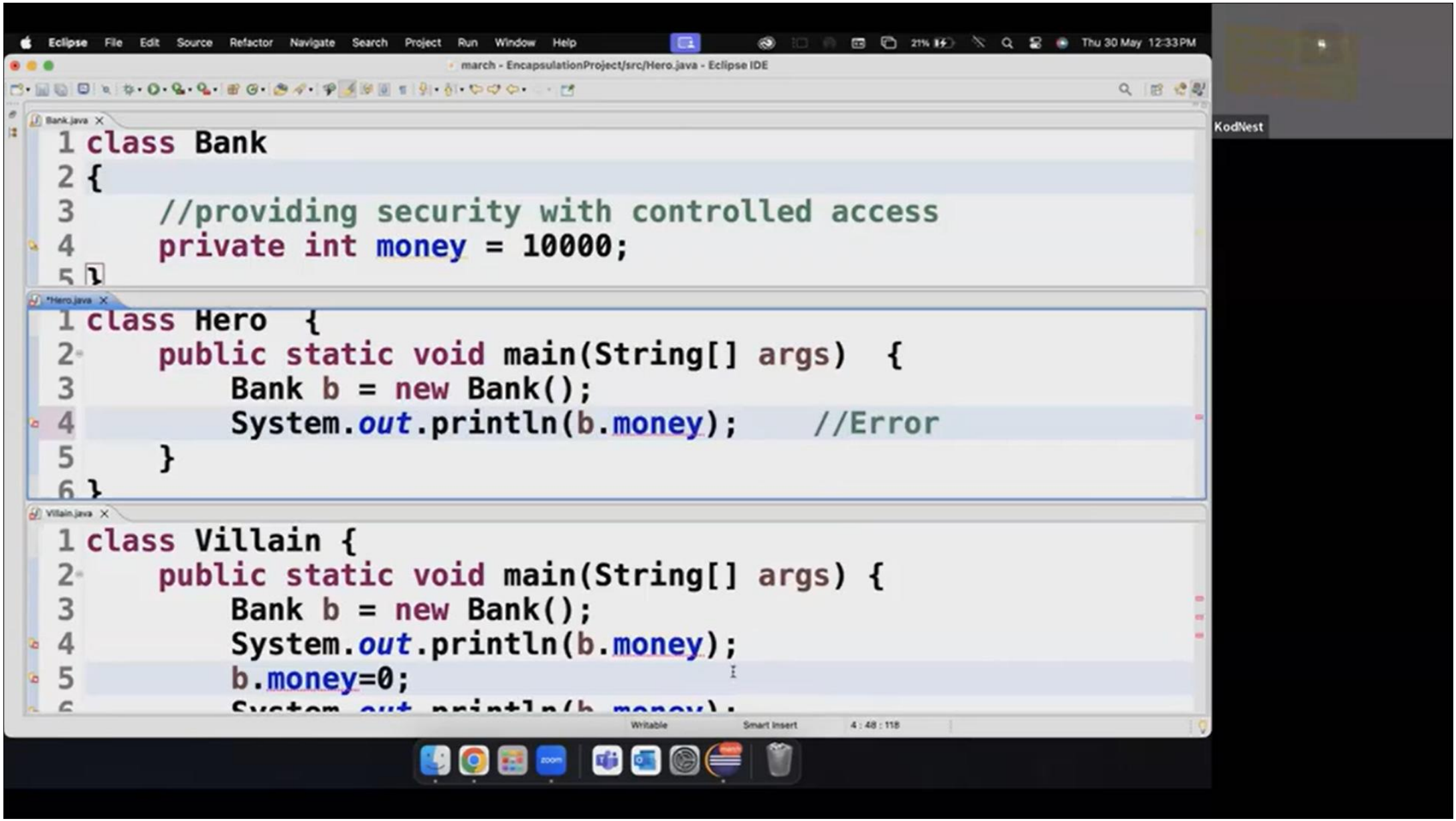
(i) Encapsulation ~~inheritance~~
(2) ~~Polymorpt~~ Inheritence
(3) polymorphism
(4) abstraction.

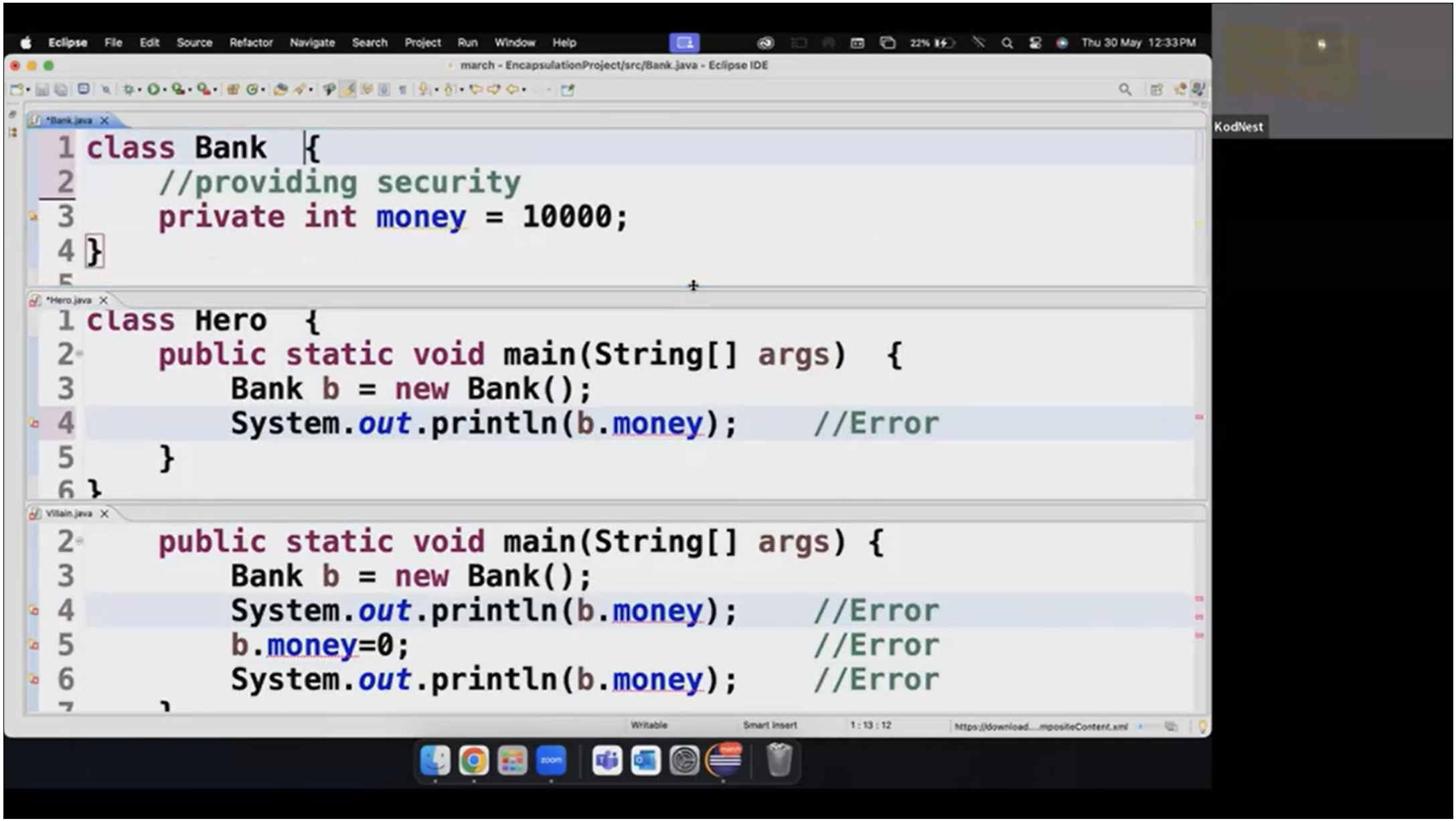
(1) Encapsulation:- It is a process of binding the data member's with member functie

A Java class is a basic fundamental example of Encapsulations :.

Encapsulation using Private key word and Public setters and getters.

54


Eclipse File Edit Source Refactor Navigate Search Project Run Window Help
Thu 30 May 12:33 PM
march - EncapsulationProject/src/Hero.java - Eclipse IDE
KodNest
Bank.java
class Bank
{
//providing security with controlled access
private int money = 10000;
}
*Hero.java
class Hero {
public static void main(String[] args) {
Bank b = new Bank();
System.out.println(b.money); //Error
}
}
Villain.java
class Villain {
public static void main(String[] args) {
Bank b = new Bank();
System.out.println(b.money);
b.money=0;
Writable
Smart Insert
4 : 48 : 118


Eclipse File Edit Source Refactor Navigate Search Project Run Window Help
Thu 30 May 12:33 PM
march - EncapsulationProject/src/Bank.java - Eclipse IDE
KodNest
*Bank.java
class Bank {
//providing security
private int money = 10000;
}
*Hero.java
class Hero {
public static void main(String[] args) {
Bank b = new Bank();
System.out.println(b.money); //Error
}
}
Villain.java
public static void main(String[] args) {
Bank b = new Bank();
System.out.println(b.money); //Error
b.money=0; //Error
System.out.println(b.money); //Error
Writable
Smart Insert
1 : 13 : 12

Controlled access.

Encapsulation is a process of providing security to the data member present in a class

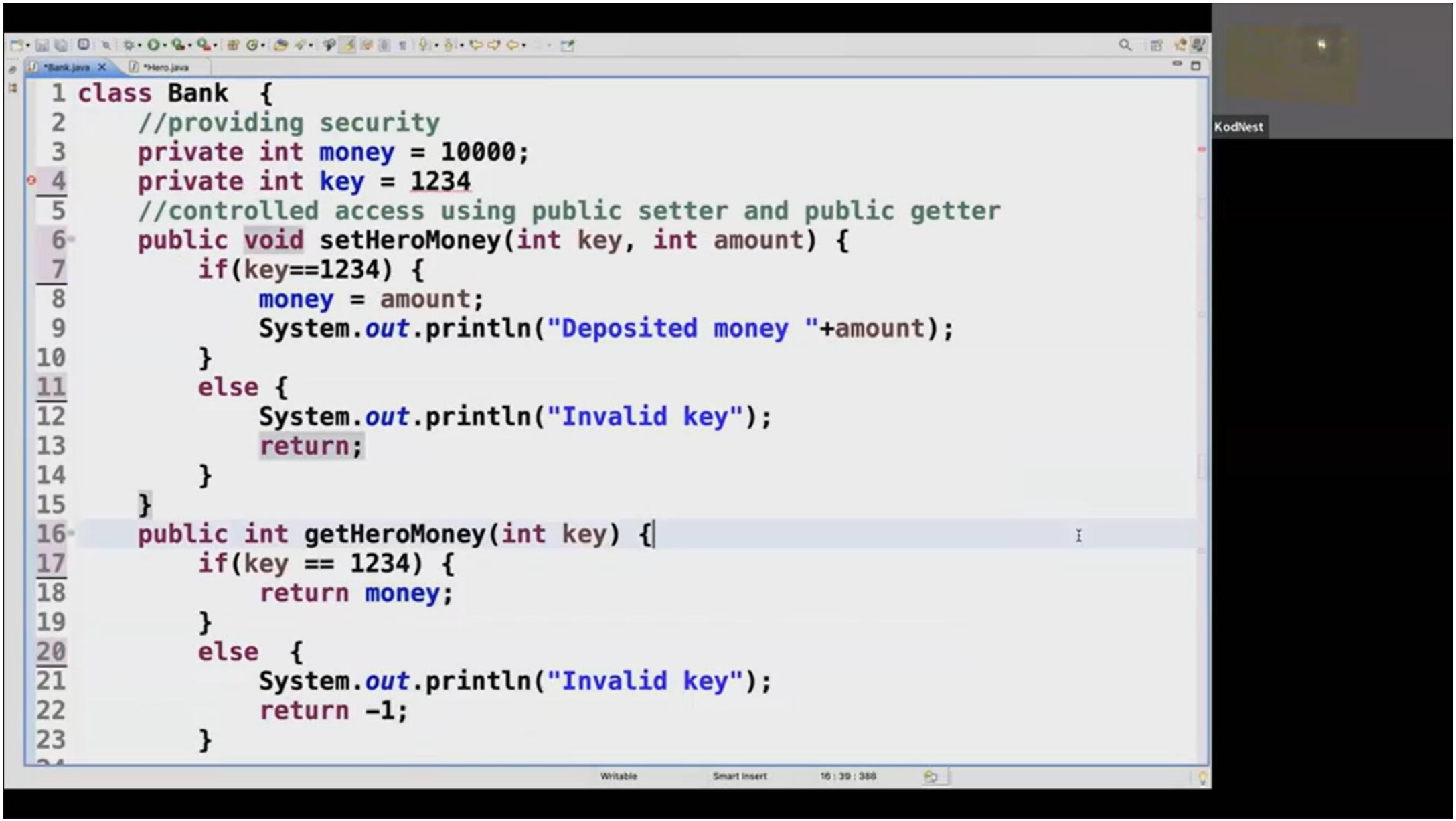
SS ->

If money variable is not declared as private then anyone outside the bank class can access the bank class hence for providing the security private keyword is used.

If any data member is declared as private then it can not be accessed outside the class that's why we are getting errors in hero class and villane class

Encapsulation does not mean only providing security whether it means providing the security using private keyword along with controlled access using public setter method and public getter methods.

1. controlled access.
2. ~~Priv~~ security.

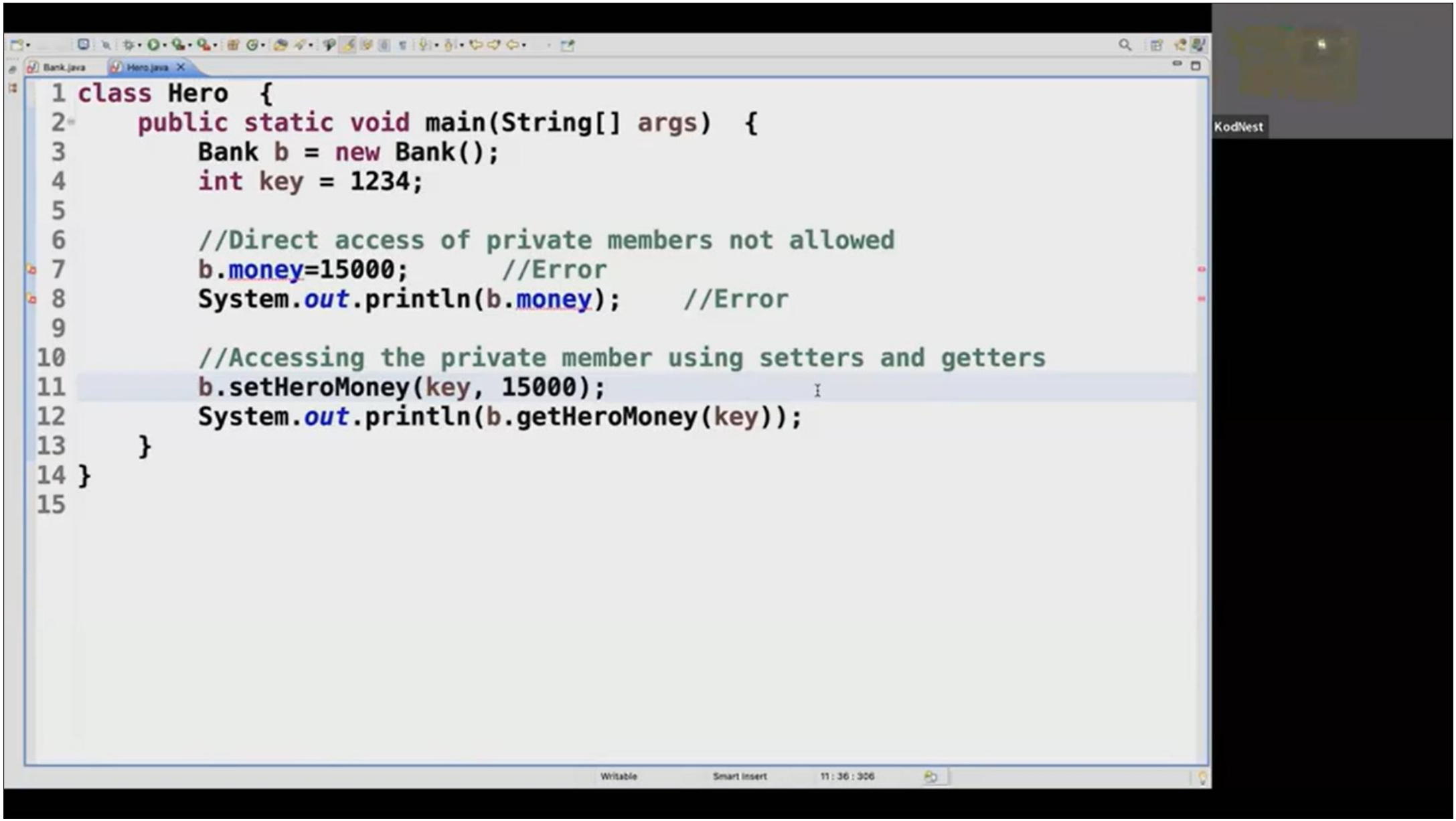


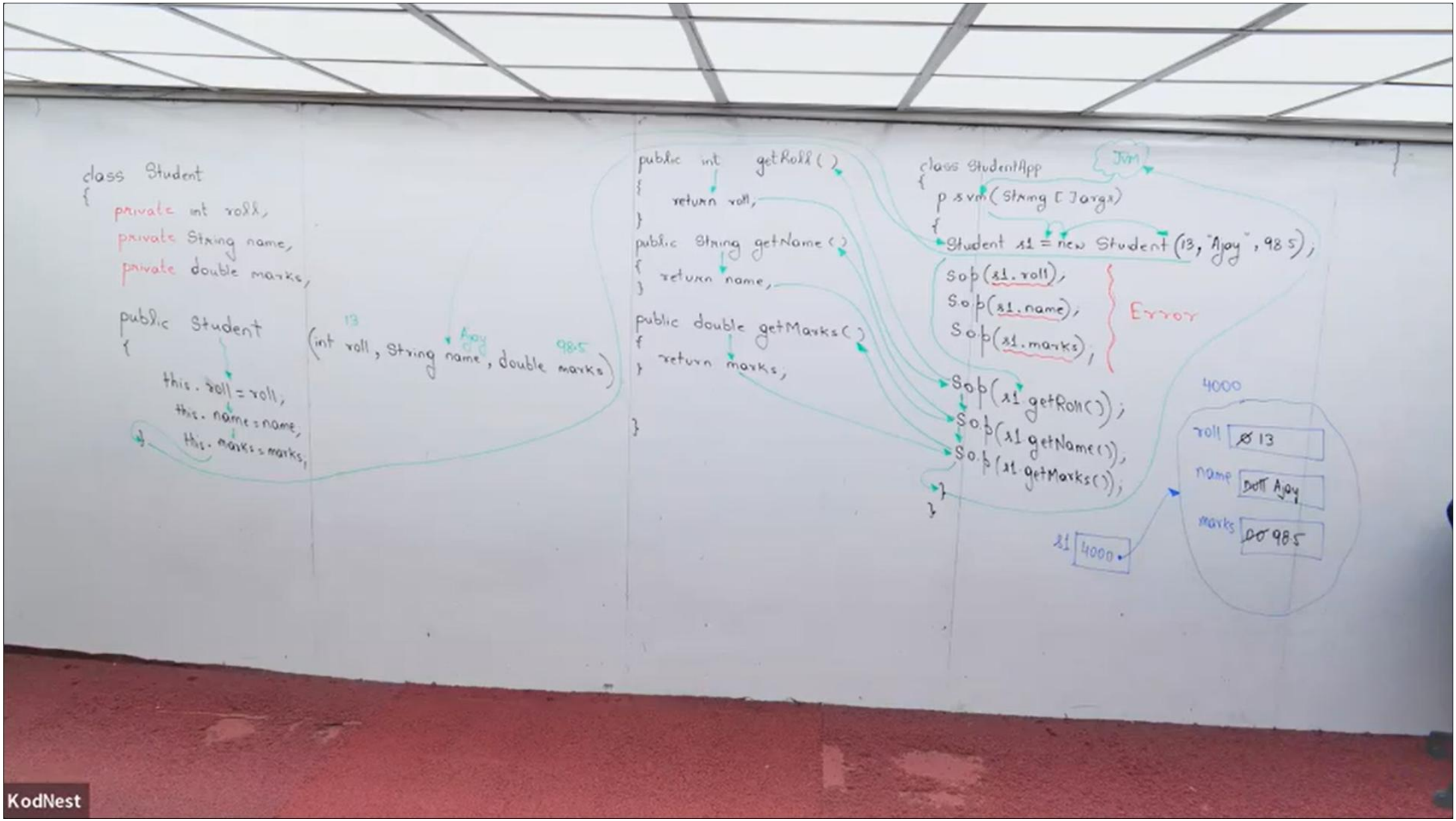

```java
class Bank  {
    //providing security
    private int money = 10000;
    private int key = 1234
    //controlled access using public setter and public getter
    public void setHeroMoney(int key, int amount) {
        if(key==1234) {
            money = amount;
            System.out.println("Deposited money "+amount);
        }
        else {
            System.out.println("Invalid key");
            return;
        }
    }
    public int getHeroMoney(int key) {
        if(key == 1234) {
            return money;
        }
        else  {
            System.out.println("Invalid key");
            return -1;
        }
```

S.4:. Complete. setter . getter :-

2# Inheritance :- Is the process. Inheriting. or. accuring. the properties of One class into another class.

In Java. inheritance can be achtive. extends. key word.


Bank.java
Hero.java
1 class Hero {
2 public static void main(String[] args) {
3 Bank b = new Bank();
4 int key = 1234;
5
6 //Direct access of private members not allowed
7 b.money=15000; //Error
8 System.out.println(b.money); //Error
9
10 //Accessing the private member using setters and getters
11 b.setHeroMoney(key, 15000);
12 System.out.println(b.getHeroMoney(key));
13 }
14 }
15
KodNest
Writable
Smart Insert
11 : 36 : 306


class Student
{
private int roll,
private String name,
private double marks,
public Student
{
this.roll = roll;
this.name = name;
this.marks = marks;
}
(int roll, String name, double marks)
13
Ajay
98.5
public int getRoll()
{
return roll,
}
public String getName()
{
return name,
}
public double getMarks()
{
return marks,
}
}
class StudentApp
{
p.s.v.m(String[]args)
{
Student s1 = new Student(13, "Ajay", 98.5);
S.o.p(s1.roll),
S.o.p(s1.name);
S.o.p(s1.marks),
Error
S.o.p(s1.getRoll());
S.o.p(s1.getName());
S.o.p(s1.getMarks());
}
}
Jvm
4000
roll 13
name Ajay
marks 98.5
s1 4000