MY CREATION ,MY DREAM.

" DON'T TRY TO MEMORISE THIS PRPOGRAMS,

JUST TRY TO UNDERSTAND"

--------------------------------------------------------------------------------------------------------------

Lecture 1:

First program in java:

Program:

class First

{

public static void main(String [] args)

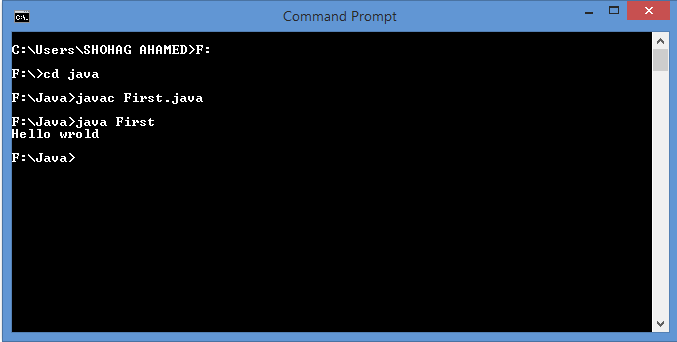
{

System.out.println("Hello wrold");

}

}

Output:



Lecture 2:

Concept of Class and object in java:

Program:

class Box

{

Private int length,breadth ,height;

public void setdimention(intl,intb,int h)

{

length=l;

breadth=b;

height=h;

}

public void showdimention()

{

System.out.println("L="+length);

System.out.println("B="+breadth);

System.out.println("H="+height);

}

}

class Example

{

public static void main(String [] args)

{

Box b1=new Box();

b1.setdimention(12,8,15);

b1.showdimention();

}

}

Output:



Lecture 3:

Concept of static variable,static function and static class in java:

Program:

class Account

{

privateintaccount\_no,balance;

private static introi;

public void set\_data(inta,int b)

{

account\_no=a;

balance=b;}

public static void set\_value(int r)

{

roi=r;}

public static class Test

{

public static String bank\_name="SONALY BANK";}

public void show\_data()

{

System.out.println("ACCOUNT NO:"+account\_no);

System.out.println("BALANCE:"+balance);

System.out.println("RATE OF INTEREST:"+roi);

}

}

public class Hello

{

public static void main(String [] args)

{

Account A1=new Account();

A1.set\_data(1010019876,200000);

Account.set\_value(5);

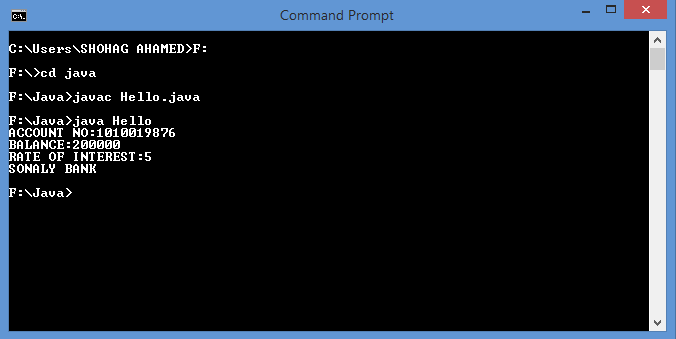
A1.show\_data();

System.out.println(Account.Test.bank\_name);

}

}

Output:



Lecture 4.1:

Concept of Command line argument(string type)in java:

Program:

class Echo

{

public static void main(String []args)

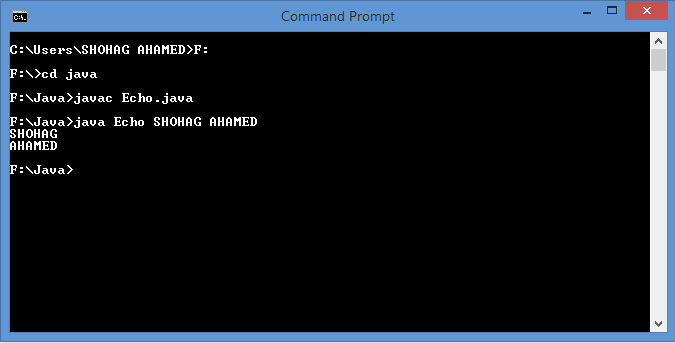
{

for(inti=0;i<args.length;i++)

System.out.println(args[i]);}

}

Output:



Lecture 4.2:

Concept of command line argument(int type) in java:

Program:

classwrold

{

public static void main(String []args)

{ int s=0;

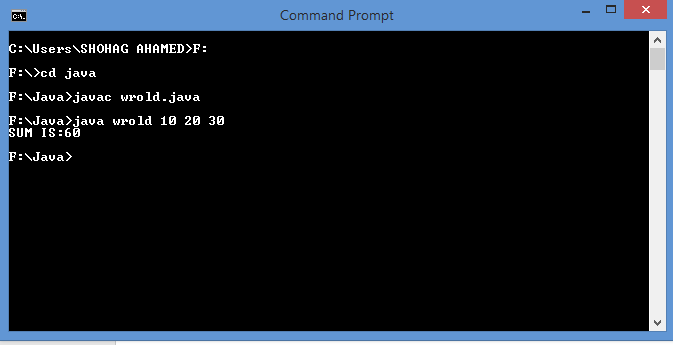
for(inti=0;i<args.length;i++)

s=s+Integer.parseInt(args[i]);

System.out.println("SUM IS:"+s);}

}

Output:



Lecture 5:

Concept of packages in java:

Program:

Package wrold;

Class HelloWrold

{

public static void main(String []args)

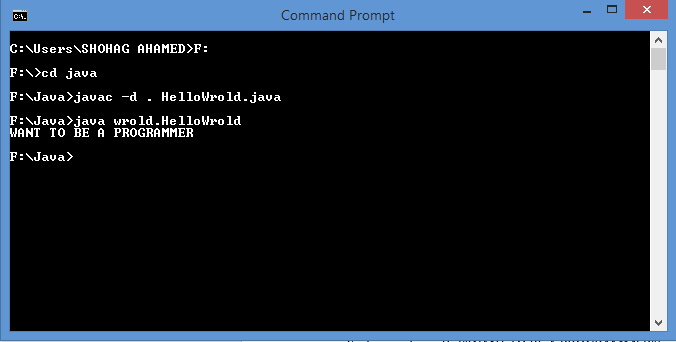
{

System.out.println("WANT TO BE A PROGRAMMER");

}

}

Output:



Lecture 6:

Use of import keyword(import of user defined class):

Program:

package pack2;

public class Student

{

Private int IDno;

private String Name;

public void SetID(int r)

{IDno=r;}

public void SetName(String n)

{Name=n;}

Public int getID()

{return(IDno);}

public String getName()

{return(Name);}

}

package pack1;

import pack2.Student;

public class Example

{ public static void main(String []args)

{ Student S=new Student();

S.SetID(16030);

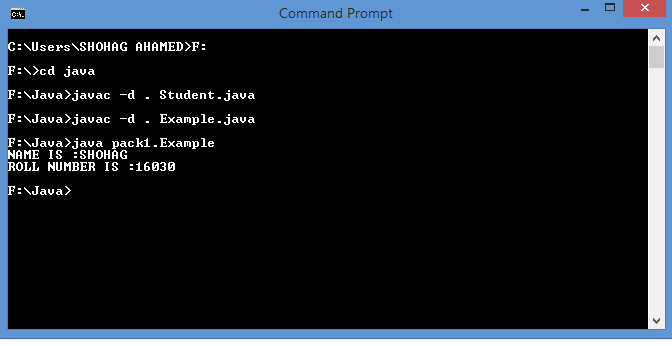
S.SetName("SHOHAG");

System.out.println("NAME IS :"+S.getName());

System.out.println("ROLL NUMBER IS :"+S.getID());}

}

Output:



Lecture 7:

Constructor in java:

Program:

class Box

{

Private int l,b,h;

public Box()

{

System.out.println("WANT OT BE A PROGRAMMER");}

public Box(intx,inty,int z)

{ l=x;b=y;h=z;}

public void showData()

{

System.out.println("LENGTH:"+l);

System.out.println("BREATH:"+b);

System.out.println("HEIGHT:"+h);

}

public static void main(String []args)

{

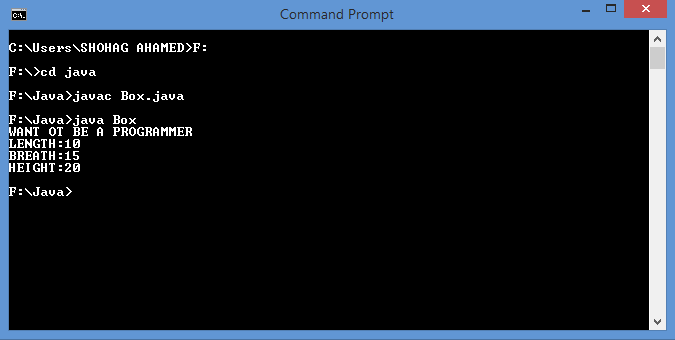
Box b1=new Box();

Box b2=new Box(10,15,20);

b2.showData();}

}

Output:



Lecture 8.1:

Single inheritance in java:

Program:

public class person

{

private String name;

private int age;

public void SetName(String n)

{ name=n;}

public void SetAge(int a)

{ age=a;}

public String getname()

{ return(name);}

Public int getage()

{ return(age);}

}

public class Student extends person

{ private int ID;

public void SetID(int r)

{ ID=r;}

publicintgetID()

{ return(ID);}

}

public class Example

{

public static void main(String [] args)

{

Student s1=new Student();

s1.SetName("SHOHAG");

s1.SetAge(21);

s1.SetID(16030);

System.out.println("NAME:"+s1.getname());

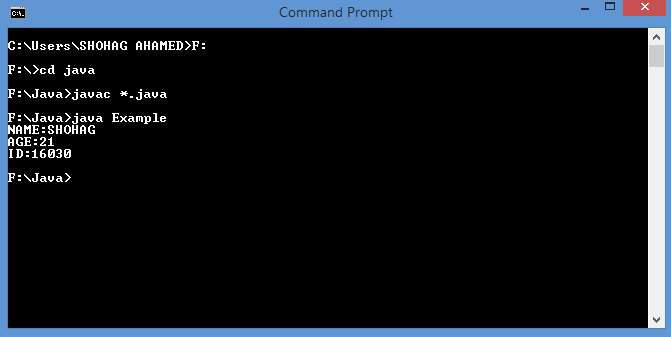
System.out.println("AGE:"+s1.getage());

System.out.println("ID:"+s1.getID());

}

}

Output:



Lecture 8.2:

Multilevel inheritance in java:

Program:

class Car{

public void vehicleType()

{

System.out.println("Vehicle Type: Car");

}

}

classMaruti extends Car{

public void brand()

{

System.out.println("Brand: Maruti");

}

public void speed()

{

System.out.println("Max: 90Kmph");

}

}

public class Maruti800 extends Maruti{

public void speed()

{

System.out.println("Max: 80Kmph");

}

public static void main(String args[])

{

Maruti800 obj=new Maruti800();

obj.vehicleType();

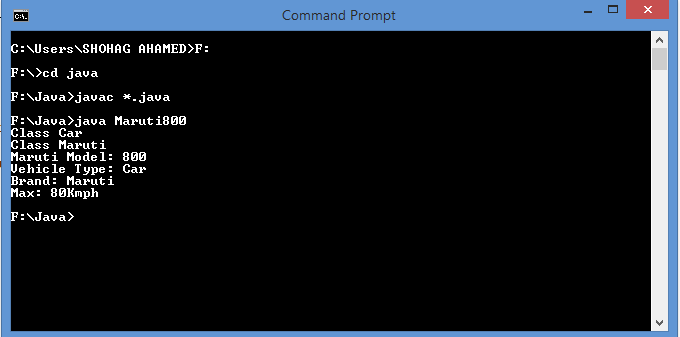
obj.brand();

obj.speed();

}

}

Outuput:



Lecture 8.3:

Hierchical inheritance in java:

public class Information

{

private String name;

privateint age;

public void setname(String n)

{ name=n;}

public void setage(int a)

{ age=a;}

public String getname()

{ return(name);}

publicintgetage()

{ return(age);}

}

public class Student extends Information

{public void fun()

{System.out.println("STUDENT'S INFORMATION:");}

}

public class Teacher extends Information

{

public void function()

{ System.out.println("TEACHER'S INFORMATION");}

public static void main(String []args)

{ Teacher t=new Teacher();

Student s=new Student();

t.setname("ZIA-UR-RAHMAN");

t.setage(30);

s.setname("SHOHAG");

s.setage(20);

t.function();

System.out.println("TEACHERS NAME:"+t.getname());

System.out.println("TEACHERS AGE:"+t.getage());

s.fun();

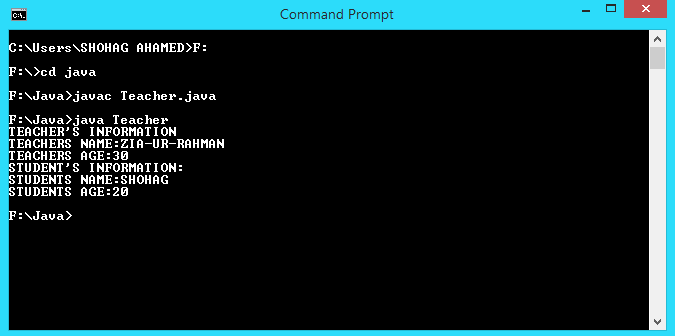
System.out.println("STUDENTS NAME:"+s.getname());

System.out.println("STUDENTS AGE:"+s.getage());

}

}

Output:



Program:

Output:

Lecture 9.1:

Instance inatiazation block in java:

Program:

public class Test

{

private int x;

{

System.out.println("Instance inatialization block:"+x);

x=5;}

public Test()

{System.out.println("constructor x:"+x);}

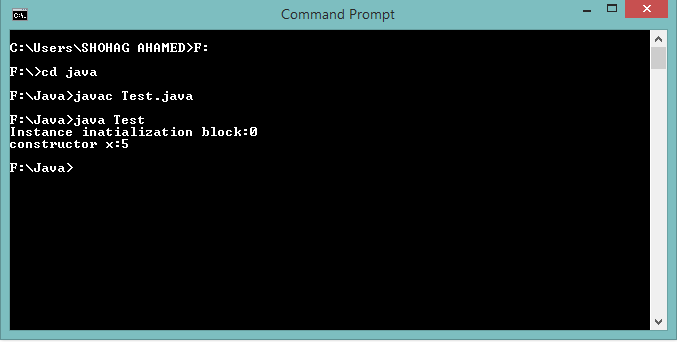
public static void main(String []args)

{

Test t=new Test();

}

Output:



Lecture 9.2:

Static inatiazation block in java:

Program:

public class Test

{

private static int x=5;

static

{

System.out.println("Static inatialization block:"+x);

x=5;}

public static void main(String []args)

{

new Test();

}

}

Output:

Lecture 10.1:

//overloaded function can stay in same class or different class//

Function overloading in javaProgram:

class A

{

public void fun(int x)

{System.out.println("class A");}

}

class B extends A

{

public void fun(intx,int y)

{System.out.println("class B");}

}

class Example

{

public static void main(String []args)

{

B b=new B();

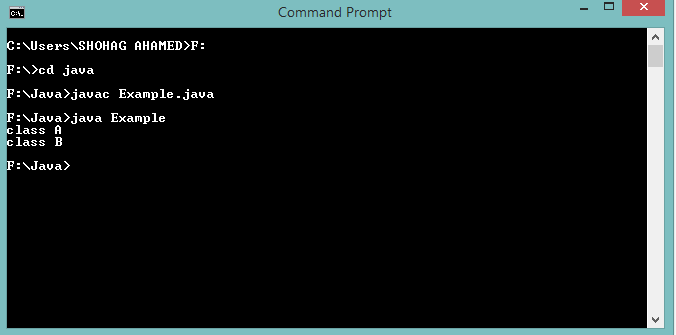
b.fun(5);

b.fun(4,7);

}

}

Output:



Lecture 10.2:

//functions must be stay in different calss//

Function overriding or Method overriding in java:

Program:

class A

{

public void fun(int x)

{System.out.println("class A");}

}

class B extends A

{

public void fun(int x)

{System.out.println("class B");}

}

class Example

{

public static void main(String []args)

{

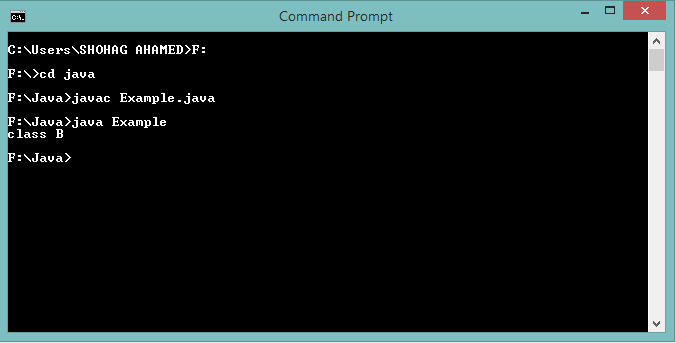
B b=new B();

b.fun(5);

}

}

Output:



Lecture 11:

Use of final keyword in java:

program:

class Example

{

private final int x=5; //final instance variable ,if the value is not initialize this step, it contain blank//

private final static int x=5; //final static variable

{ x=5;

System.out.println(“X:”+x) } // instance initialization block

public Example ()

{ x=5;

System.out.println(“X:”+x) } //initialization with constructor

static

{ x=5;

System.out.println(“X:”+x) } //static initialization block

Public static void main(String []args)

{

Example e=new Example();}

}

class final dummy()//final class(this class is not inheritable)

{

Public final void fun()

…………..}

Class voidmoredummy extends dummy

{ public void fun()

……….}//final method or class(function overriding is impossible)

Lecture 12.1:

this keyword in java:

program:

public class Example

{

Private int l,b,h;

public void setvalue(intl,intb,int h)

{ this.l=l;this.b=b;this.h=h;}

}

public static void main(String []args)

{

Example e=new Example();

e.setvalue(4,5,6);

}

}

Lecture 12.2:

this and super keyword in java:

program:

class A

{ publicint z;

public void f1()

{ System.out.println("class A,z:"+z);}

}

class B extends A

{publicint z;

public void f1()

{

{ System.out.println("class B,z:"+z);}}

public void f2()

{

int z;

z=9;

super.z=2;

super.f1();

this.z=4;

this.f1();

System.out.println("Z:"+z);

}

}

class Example

{

public static void main(String []args)

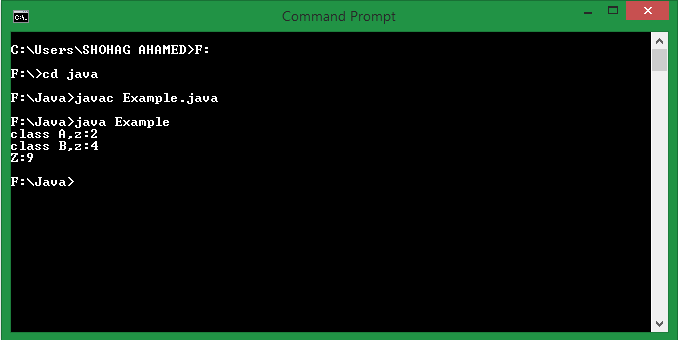
{

B b=new B();

b.f2();

}

}

output:

Lecture 13.1:

static member function in inheritance:

program:

class Parent

{

public static void f1()

{ System.out.println("hello wrold");}

}

class Child extends Parent

{

public void f()

{ System.out.println("WANT TO BE A PROGRAMMER");}

}

class Example

{

public static void main(String []args)

{Child c=new Child();

c.f1();

c.f();}

}

output:



Lecture 13.2:

Function hiding(use static keyword) and overriding in inheritance:

Program:

//if use static then it is called function hiding,else function overriding//

{

class Parent()

public static void f1

{ System.out.println("hello wrold");}

}

class Child extends Parent

{

public static void f1()

{ System.out.println("WANT TO BE A PROGRAMMER");}

}

class Example

{

public static void main(String []args)

{Child c=new Child();

c.f1();}

}

Output:



Lecture 13.3:

Program:

//static member variables do not inherite//

class Parent

{

public static int y=4;

}

class Child extends Parent

{

static

{y=5;}

}

class Example

{

public static void main(String []args)

{

System.out.println("y="+Child.y);}

}

Output:



//constructor do not inherited//

Lccture 14.1:

Constructor in inheritance in java:(defult constructor by super() )

Program:

class A

{ publicint x;

public A()

{System.out.println("A");}

}

class B extends A

{ publicint y;

public B()//child class constructor first call the parent class constructor then itself

{System.out.println("B");}

}

class Example

{ public static void main(String []args)

{ B b=new B();}

}

Output:



Lccture 14.2:

Constructor in inheritance in java:(parameterized constructor )

Program:

class A

{ publicint x;

public A(int a)

{

x=a;

System.out.println("x:"+x);

System.out.println("A");

}

}

class B extends A

{ publicint y;

public B()

{

super(4);

System.out.println("B");}

}

class Example

{ public static void main(String []args)

{ B b=new B();}

}

Output:



Lceture 14.3:

Constructor chaining in java:

Program:

class A

{ publicint x;

public A()

{

System.out.println("A 1");

}

}

class B extends A

{ publicint y;

public B()

{

this(4);

System.out.println("B 1");}

public B(int b)

{

System.out.println("B 2");}

}

class Example

{ public static void main(String []args)

{ B b=new B();}

}

Output:



Lceture 15.1:

Abstract class in java:

Program:

abstract public class information

{

private String name;

privateint age;

public void setname(String n)

{ name=n;}

public void setage(int a)

{ age=a;}

public String getname()

{ return(name);}

publicintgetage()

{ return(age);}

}

public class Student extends Information

{public void fun()

{System.out.println("STUDENT'S INFORMATION:");}

}

public class Teacher extends Information

{

public void function()

{ System.out.println("TEACHER'S INFORMATION");}

public static void main(String []args)

{ Teacher t=new Teacher();

Student s=new Student();

t.setname("ZIA-UR-RAHMAN");

t.setage(30);

s.setname("SHOHAG");

s.setage(20);

t.function();

System.out.println("TEACHERS NAME:"+t.getname());

System.out.println("TEACHERS AGE:"+t.getage());

s.fun();

System.out.println("STUDENTS NAME:"+s.getname());

System.out.println("STUDENTS AGE:"+s.getage());

}

}

Output:



Lceture 15.2:

How to direct access the elements of abstract class:

//it is impossible to creat an object of an abstract class but it is possible to creat an reference variable of an abstract class//(same case for interface)

abstract public class Information

{

private String name;

privateint age;

public void setname(String n)

{ name=n;}

public void setage(int a)

{ age=a;}

public String getname()

{ return(name);}

publicintgetage()

{ return(age);}

}

public class Student extends Information

{public void fun()

{

System.out.println(“don’t creatobj for understanding”);

}

}

class Example

{public static void main(String []args)

{ Information i=new Student();//access child clss obj. reference in abstaractclss

reference variable//

i.setname("SHOHAG");

i.setage(20);

System.out.println("STUDENTS NAME:"+i.getname());

System.out.println("STUDENTS AGE:"+i.getage());

}

}

Output:



Lecture 15.3:

Abstract method in java:

Program:

abstract class Account

{

abstract void CalculateInterest();

}

class Savings extends Account

{

public void CalculateInterest()

{ System.out.println("LINE MUST BE EXECUTE");}

public static void main(String []args)

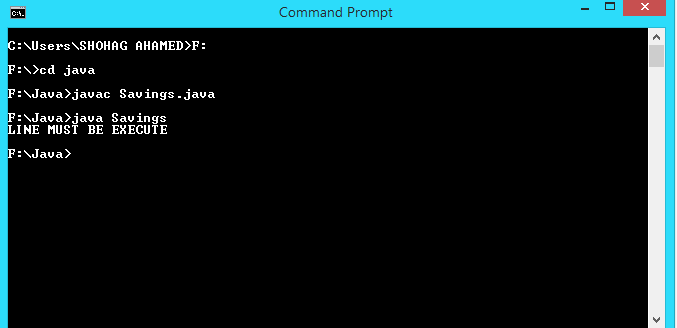
{ Savings s=new Savings();

s.CalculateInterest();

}

}

Output:



Lecture 16.1:

Interface in java:

Program:

interface I1

{

void fun1();

}

interface I2

{

void fun2();

}

class A implements I1,I2

{

public void fun1()

{ System.out.println("this is fun1");}

public void fun2()

{ System.out.println("this is fun2");}

}

class Example

{

public static void main(String args[])

{

A obj=new A();

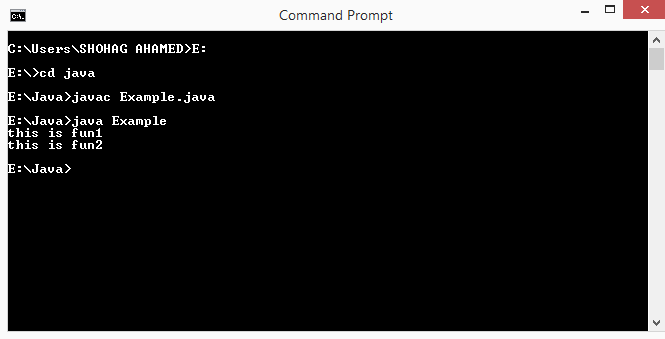
obj.fun1();

obj.fun2();

}

}

Output:



Lecture 16.2:

Multiple inheritance of interface:

Program:

interface I1

{

void fun1();

}

interface I2

{

void fun2();

}

interface I3 extends I1,I2

{

void fun3();

}

class A implements I3

{

public void fun1()

{ System.out.println("this is fun1");}

public void fun2()

{ System.out.println("this is fun2");}

public void fun3()

{ System.out.println("this is fun3");}

}

class Example

{

public static void main(String args[])

{

A obj=new A();

obj.fun1();

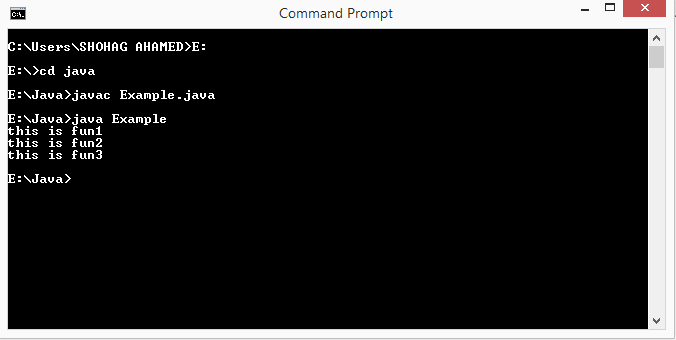
obj.fun2();

obj.fun3();

}

}

Output:



Lecture 17:

Taking input from keyboard in java:

Program:

Import java.util.Scanner;

class Example

{

public static void main(String args[])

{

intn,rem,r,a,rev=0;

Scanner obj=new Scanner(System.in);

System.out.print("Enter a number:");

n=obj.nextInt();

a=n;

while(a>0)

{

rem=a%10;

rev=(rev\*10)+rem;

a=a/10;

}

if(rev==n)

System.out.println("Number is palindrom");

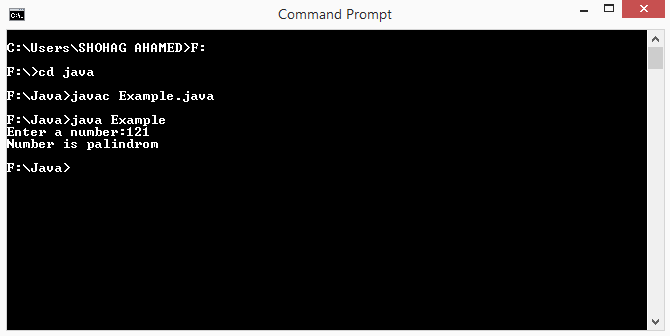
else

System.out.println("Number is not palindrom");

}

}

Output:



Lecture on:Exception Handling in java:

i)Default through default catch…

code:(Arithmatic Exception)

class Example

{

public static void main(String args[])

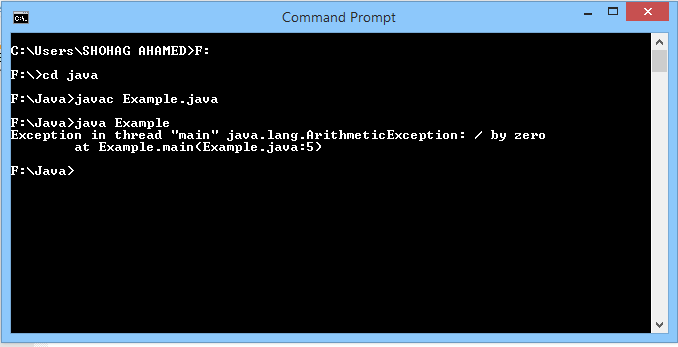
{

System.out.println("Result:"+3/0);

}

}

output:



Code:(nullpointer Exception)

class Example

{

public static void main(String args[])

{

String s1=null;

System.out.println("Frist line");

System.out.println("string length:"+s1.length());

System.out.println("Last line");

}

}

Output:

ii)default through our catch…

code:

output:

Topic must be added:

1.Array class..

2.String class..

3.Exception handling..

4.Thread

5.File handling

Enough for 2nd year 1st semester…….

Will be updated as soon as possible…..

-----------------------------------------------------------------------------------------------------------

PERFORMED BY

SHOHAG(ICT-‘13)