**Exception Handling**

**+try ,catch,finally**

//psvm

Scanner in =new Scanner();

Int a ,b;

Sout(“enter a ”);

a=in.nextInt();

Sout(“enter b”);

b=in.nextInt();

try{

Sout(“a/b”+(a/b));

}

catch(Arithmetic Exception e){

sout(“divide by zero exception”);

}

finally{

Sout(a+”,”+b+”,”+c);

}

Note: In order to know whats the error   
catch(Arithmetic Exception e){

sout(“divide by zero exception”);

e.printStackTrace();

}

**+Throw an own Expection**

try{

//check balance

If(bal<=100){

throw new NullPointerException(“Min balance”);

}

}

catch(NullPointerException e){

sout(Deposit some fund);

}

**+throw Exception**

Psvm throws Intrrupted Exception{

sout(new Date());

Thread.sleep(1000);

sout(new Date());

}

.......

.

.

..

**+multiCatch**

//psvm

Scanner in =new Scanner();

Int a ,b;

Sout(“enter a ”);

a=in.nextInt();

Sout(“enter b”);

b=in.nextInt();

try{

Sout(“a/b”+(a/b));

}

catch(Arithmetic Exception e){

sout(“divide by zero exception”);

}

catch(InputMismatchException e){

sout(“enter carefully”);

}

finally{

Sout(a+”,”+b+”,”+c);

}

.

.

.

.

**Java Abstraction**

**Abstraction Class**

An abstract class is one that can’t be instantiated. All other functionality of the class exists ; and its fields ,methods and constructor are accessed in the same manner. You just can’t create an instant of the abstract class

A parent class contains common functionalities of a collection of child classes, but the parent class itself is too abstract to be used in its own.

Rules of **Abstraction Class/Methods**

* If a class contains an abstract method, then class must be abstract as well.
* Any child class must either override the abstract method or declare itself abstract

abstract class College{

int student =3200;

}

class khec extends College{

String principal =”Er.Sujan Maka”;

}

Psvm

///College c=new College();

///Sout (c.student);

Khec k =new Khec();

Sout(k.student);

Sout(k.pricipal);