Text, letter

Description automatically generated

**package** S1;

**public** **class** Book {

**private** String Bname;

**private** String Bid;

**private** String Bauthor;

**private** **int** YOP;

**public** **void** setName(String Bname)

{

**try** {

**if**(Bname.matches("[A-Z a-z -]+"))

{

**this**.Bname=Bname;

}

**else**

{

**throw** **new** InvalidNameException(Bname);

}

}

**catch**(InvalidNameException e)

{

System.***out***.println(e);

}

}

**public** **void** setID(String Bid)

{

**try** {

**if**(Bid.matches("[A-Z a-z 0-9]+"))

{

**this**.Bid=Bid;

}

**else**

{

**throw** **new** InvalidIDException(Bid);

}

}

**catch**(InvalidIDException f)

{

System.***out***.println(f);

}

}

**public** **void** setAuthor(String Bauthor)

{

**try** {

**if**(Bid.matches("[A-Z a-z]+"))

{

**this**.Bauthor=Bauthor;

}

**else**

{

**throw** **new** InvalidAuthorException(Bauthor);

}

}

**catch**(InvalidAuthorException g)

{

System.***out***.println(g);

}

}

**public** **void** setYOP(**int** yop)

{

//autoboxing

Integer obj=yop;

String str=obj.toString();

**int** len=str.length();

**try** {

**if**(len==4)

{

YOP=yop;

}

**else**

{

**throw** **new** InvalidYOPException(yop);

}

}

**catch**(InvalidYOPException h)

{

System.***out***.println(h);

}

}

**public** String getN()

{

**return** Bname;

}

**public** String getid()

{

**return** Bid;

}

**public** String getauthor()

{

**return** Bauthor;

}

**public** **int** getyop()

{

**return** YOP;

}

**public** String toString()

{

**return** String.*format*("Bname=%s%n Bid=%s%n Bauthor=%s%n YOP=%d%n",getN(),getid(),getauthor(),getyop());

}

}

**package** S1;

**public** **class** Demo {

**public** **static** **void** main(String[] args)

{

Book b=**new** Book();

b.setName("James-Backlogs");

b.setID("sc2101");

b.setAuthor("Rohit");

b.setYOP(2022);

System.***out***.println(b);

}

}

**package** S1;

**public** **class** InvalidNameException **extends** Exception {

String type;

InvalidNameException(String type)

{

**this**.type=type;

}

**public** String toString()

{

**return** "InvalidName"+type;

}

}

**class** InvalidIDException **extends** Exception {

String ID;

InvalidIDException(String ID)

{

**this**.ID=ID;

}

**public** String toString()

{

**return** "InvalidID"+ID;

}

}

**class** InvalidAuthorException **extends** Exception {

String Author;

InvalidAuthorException(String Author)

{

**this**.Author=Author;

}

**public** String toString()

{

**return** "InvalidAuthor"+Author;

}

}

**class** InvalidYOPException **extends** Exception {

**int** YOP;

InvalidYOPException(**int** YOP)

{

**this**.YOP=YOP;

}

**public** String toString()

{

**return** "InvalidYOP"+YOP;

}

}

5.)Madhuri wants to develop an app which sorts theStudent Ids based on marks(ascending) if marks are equalsort as per their student ID. The User provides the marksof different students along with student Ids and storesthem in a Vector. Help him out writing the program todevelop the app. (Use comparator interface)

**package** S2;

**public** **class** Student {

**private** **long** ID;

**private** **int** mark1,mark2,mark3,mark4;

Student(**long** ID,**int** mark1,**int** mark2,**int** mark3,**int** mark4)

{

**this**.ID=ID;

**this**.mark1=mark1;

**this**.mark2=mark2;

**this**.mark3=mark3;

**this**.mark4=mark4;

}

**public** **long** getID()

{

**return** ID;

}

**public** **int** Total()

{

**int** total=mark1+mark2+mark3+mark4;

**return** total;

}

**public** String toString()

{

**return** String.*format*("ID=%d%n Mark1=%d%n Mark2=%d%n Mark3=%d\n Mark4=%d\n Total Mark=%d",ID,mark1,mark2,mark3,mark4,Total());

}

}

**package** S2;

**import** java.util.Collections;

**import** java.util.Comparator;

**import** java.util.Vector;

**public** **class** Main {

**public** **static** Vector<Student> *stud*=**new** Vector<Student>();

**public** **static** **void** main(String[] args) {

*stud*.add(**new** Student(90187,70,88,89,78));

*stud*.add(**new** Student(90185,82,64,52,49));

*stud*.add(**new** Student(90184,85,96,40,20));

*stud*.add(**new** Student(90183,71,60,80,68));

Collections.*sort*(*stud*, **new** SortByID());

System.***out***.println("Student Detail based on ID");

**for**(Student cu:*stud*)

{

System.***out***.println(cu);

}

}

}

/\*class SortByID implements Comparator<Student>

{

@Override

public int compare(Student o1, Student o2) {

}

}\*/

**package** S2;

**import** java.util.Comparator;

**public** **class** SortByID **implements** Comparator<Student> {

@Override

**public** **int** compare(Student o1, Student o2) {

**if**(o1.getID()>o2.getID())

**return** 1;

**else** **if**(o1.getID()==o2.getID())

**return** 0;

**else**

**return** -1;

}

}

3.)Enhance the student class ,such that when the Invalid ID ( positive number ) is given it must throw InvakidIDException and InvalidNameException when name has special characters or digits

package p2;

public class Student {

private int id;

public String Name;

public void SetName(String name)

{

boolean m=name.matches("[A-Z a-z]+");

try {

if(m)

{

throw new InvalidNameException(Name);

}

else

{

this.Name=Name;

}

}

catch(InvalidNameException e){

System.out.println(e);

}

}

public void setID(int id)

{

try {

if(id>0)

{

this.id=id;

}

else

{

throw new InvalidIDException(id);

}

}

catch(InvalidIDException e){

System.out.println(e);

}

}

public int getID()

{

return id;

}

public String getName()

{

return Name;

}

public String toString()

{

return String.format("Name=%s%n ID=%d",getName(),getID());

}

}

class InvalidIDException extends Exception{

int id;

InvalidIDException(int id)

{

this.id=id;

}

public String toString()

{

return "Invalid ID"+id;

}

}

class InvalidNameException extends Exception{

String Name;

InvalidNameException(String Name)

{

this.Name=Name;

}

public String toString()

{

return "Invalid Name"+Name;

}

}

package p2;

public class Demo {

public static void main(String[] args) {

Student s=new Student();

s.setID(187);

s.SetName("Niranjan");

System.out.println(s);

}

}