```
class WrongAgeException extends Exception {
    public WrongAgeException(String message) {
        super(message);
    }
}
class Father {
    int age;
    public Father(int age) throws WrongAgeException {
        if (age < 0) {
            throw new WrongAgeException("Father's age cannot be negative!");
        }
        this.age = age;
    }
}
class Son extends Father {
    int sonAge;
    public Son(int fatherAge, int sonAge) throws WrongAgeException {
        super(fatherAge);
        if (sonAge < 0) {
           throw new WrongAgeException("Son's age cannot be negative!");
        }
        if (sonAge >= fatherAge) {
           throw new WrongAgeException("Son's age cannot be greater than or equal to Father's age!");
        }
        this.sonAge = sonAge;
    }
}

public class Demo {
    public static void main(String[] args) {
        try {
            Father father = new Father(40);
            Son son = new Son(40, 18);
            System.out.println("Father's age: " + father.age);
            System.out.println("Son's age: " + son.sonAge);
            Father invalidFather = new Father(-5);
        }

        catch (WrongAgeException e){
            System.out.println("Exception caught: " + e.getMessage());
        }

        try {
            Son invalidSon = new Son(35, 36);
        }

        catch (WrongAgeException e){
            System.out.println("Exception caught: " + e.getMessage());}
    }
}
```

```
D:\24BECS409>javac Demo.java

D:\24BECS409>java Demo
Father's age: 40
Son's age: 18
Exception caught: Father's age cannot be negative!
Exception caught: Son's age cannot be greater than or equal to Father's age!

D:\24BECS409>
```

21/11/24

Week - 7

Write a program that demonstrates handling of exceptions in inheritance tree. Create a base class called "Father" & derived class called "son" which extends the base class. In Father class, implement a constructor which takes the age & throws the exception WrongAge when the i/p age < 0. In son class, implement a constructor that uses both father and Son's age & throws an exception if son's age is >= father's age.

=>
```
class WrongAgeException extends Exception{
  public WrongAgeException (String message){
    super(message);
  }
}

class Father{
  int age;
  public Father (int age) throws WrongAgeException{
    if (age <0 ){
      throw new WrongAgeException ("Fathers's
        age cannot be negative");
    }
    this.age = age;
  }
}

class Son Extends Father{
  int sonAge;
  public Son (int fatherAge, int SonAge)
```
}

```
throws WrongAgeException {
    super(fatherAge);
    if (sonAge < 0) {
        throws new WrongAgeException("Son's age
            cannot be negative!");
    }
    if (sonAge >= fatherAge) {
        throw new WrongAgeException("Son's
            age cannot be greater than or equal
            to Father's age");
    }
    this.sonAge = sonAge;
}
}
public class Demo {
    public static void main(String xx[]) {
        try {
            Father father = new Father(40);
            Son son = new Son(40, 18);
            System.out.println("Fathers age:" + father.age);
            System.out.println("Son's age:" + son.sonAge);
            Father InvalidFather = new Father(-5);
        }
        catch(WrongAgeException e) {
            System.out.println("Exception caught:" + e.get-
                                                Message());}

        try {
            Son invalidSon = new Son(35, 36);
        }
        catch(WrongAgeException e) {
            System.out.println("Exception caught:" +
                e.getMessage());}
        }
}
```

PAGE NO :
DATE :

21/11/24

Output :

Father's age : 40
Son's age : 18
Exception Caught : Father's age cannot be Negative.
Exception Caught : Son's age cannot be greater than or equal to Father's age.