Q) Write a program that demonstrates handling of exception in inheritance tree. Create a base class called "father" and derived class called "son" which extends the base class. In father class, implement a constructer which takes the age and throws the exception WrongAge() when the input age<0. In son class, implement a constructor that cases both father and son's age and throws an exception if son's age is >= father's age.

```
import java.util.scanner;
class wrongAgeException extends Exception{
        public wrongAgeException(String message){
                super(message);
        }
}
class father {
        private int age;
        public father(int age) throws WrongException{
            if(age<0) {
            }
        } this.age = age;
    }

class son extends father{
        private int sonAge;
        public son (int fatherAge, int sonAge) throws WrongAgeException{
            super(fatherAge);
            if(sonAge>=FatherAge) {
              throw new WrongAgeException ("Son's age can't be greater
         than or equal to father's age");
            }
            this. sonAge = sonAge;
        }
}
public class Main {
            public static void main (String[] args) {
             Scanner scanner = new scanner (system. in);
             System.out.print ("Enter father's age : ");
             int fatherAge = scanner.nextInt().
             System. out.print ("Enter son's age:");
              int sonAge = Scanner.nextInt();
              try{
                 Son son = new son(fatherAge, sonAge);
                  System.out.println("son object created succefully.");
              }
              catch (WrongAgeException e) {
                     System.out.println("Error: " + e.getMessage());
              }
            }
}
```

output:

```
enter father Age : 25
   enter son's age : 12
   son object created succefully
```

19/2/24

```
import java.util.Scanner;


class WrongAge extends Exception

{

public WrongAge(String str)

{

super(str);

}

}



class Father

{

int fatherAge;

Father(int fatherAge) throws WrongAge

{

this.fatherAge = fatherAge;


if(fatherAge<0)

throw new WrongAge("Father's Age cannot be less than zero");


}

}


class Son extends Father

{

int sonAge;

Son(int fatherAge,int sonAge) throws WrongAge

{

super(fatherAge);

this.sonAge = sonAge;
if(sonAge<0)
throw new WrongAge("Son's Age cannot be less than zero");


if(sonAge>=fatherAge)
throw new WrongAge("Son's age cannot be greater than or equal to Father's age");


}
}


class AgeExceptionMain
{
public static void main(String[] args)
{
Scanner input = new Scanner(System.in);


System.out.print("Enter father's age: ");
int fatherAge = input.nextInt();


System.out.print("Enter son's age: ");
int sonAge = input.nextInt();


try{
Son son = new Son(fatherAge,sonAge);
System.out.println("Father's and Son's age are valid");


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

Output:

```
Enter father's age: 50

Enter son's age: 20

Father's and Son's age are valid


Enter father's age: 50

Enter son's age: 60

Exception: WrongAge: Son's age cannot be greater than or equal to Father's age


Enter father's age: -1

Enter son's age: 20

Exception: WrongAge: Father's Age cannot be less than zero


Enter father's age: 0

Enter son's age: 20

Exception: WrongAge: Son's age cannot be greater than or equal to Father's age


Enter father's age: 0

Enter son's age: -1

Exception: WrongAge: Son's Age cannot be less than zero
```