Program 7

Q] Write a program that demonstrates handling of exceptions in inheritance tree. Create a base class called “Father” and derived class called “Son” which extends the base class. In Father class, implement a constructor which takes the age and throws the exception WrongAge( ) when the input age<0. In Son class, implement a constructor that uses both father and son’s age and throws an exception if son’s age is >=father’s age.

CODE:

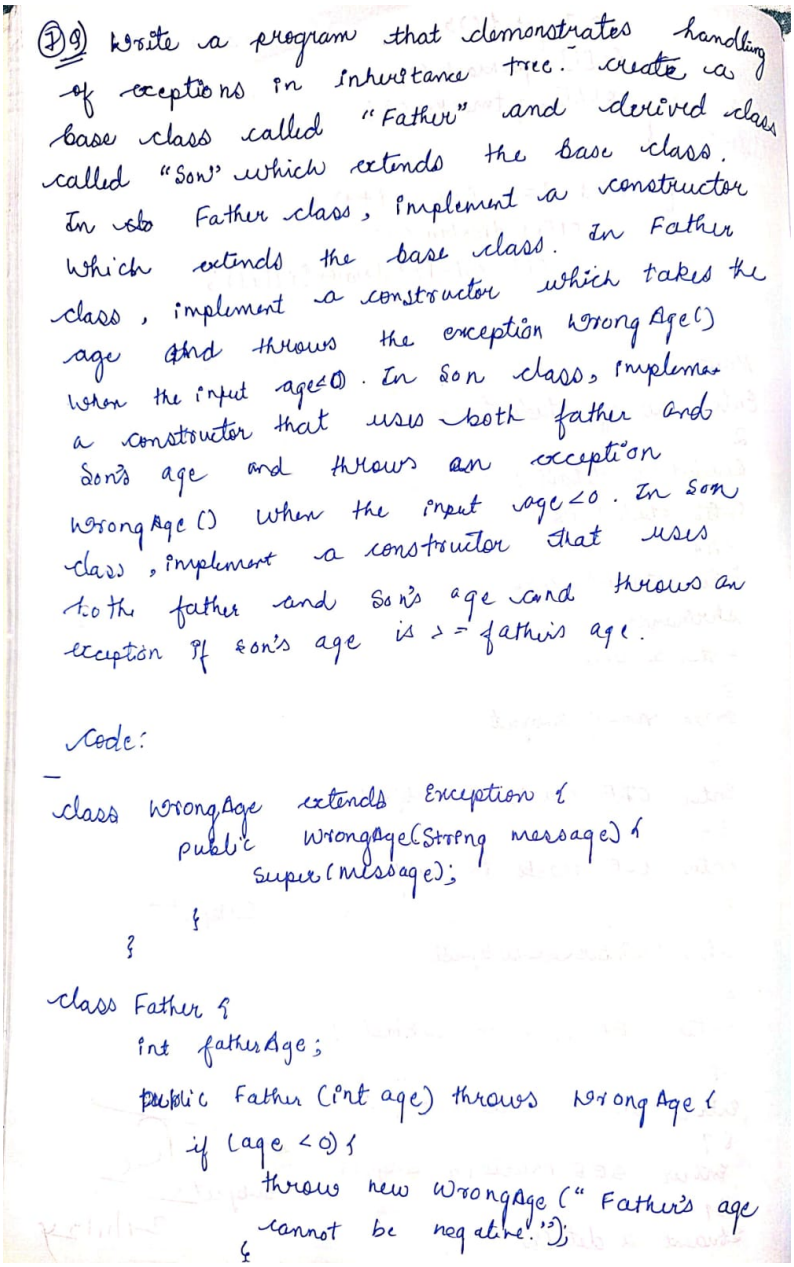
```
class WrongAgeException extends Exception {
    public WrongAgeException(String message) {
        super(message);
    }
}

class Father {
    int age;

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

class Son extends Father {
    int sonAge;

    public Son(int fatherAge, int sonAge) throws WrongAgeExcepti
        super(fatherAge);
```

```
        if (sonAge >= fatherAge) {
            throw new WrongAgeException("Son's age cannot be gre
             equal to Father's age");
        }
        this.sonAge = sonAge;
    }
}

public class ExceptionInheritanceDemo {
    public static void main(String[] args) {
        try {
            Father father = new Father(45);
            Son son = new Son(45, 20);
            System.out.println("Father's age: " + father.age);
            System.out.println("Son's age: " + son.sonAge);
        } catch (WrongAgeException e) {
            System.out.println("Exception: " + e.getMessage());
        }

        try {
            Son invalidSon = new Son(30, 35);
        } catch (WrongAgeException e) {
            System.out.println("Exception: " + e.getMessage());
        }

        try {
            Father invalidFather = new Father(-5);
        } catch (WrongAgeException e) {
            System.out.println("Exception: " + e.getMessage());
        }
    }
}
```

OUTPUT:

```
Father's age: 45
Son's age: 20
Exception: Son's age cannot be greater than or equal to Father's
Exception: Father's age cannot be negative
```

OBSERVATION:



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

Code:

```
class WrongAge extends Exception {
    public WrongAge(String message) {
        super(message);
    }
}

class Father {
    int fatherAge;
    public Father(int age) throws WrongAge {
        if (age < 0) {
            throw new WrongAge("Father's age
            cannot be negative.");
```

```
        this.fatherAge = age;
    }
}
class Son extends Father {
    int sonAge;
    public Son(int fatherAge, int sonAge)
    throws WrongAge {
        super(fatherAge);
        if (sonAge >= fatherAge) {
            throw new WrongAge("Son's age can't be
            greater or equals father Age");
        }
        this.sonAge = sonAge;
    }
}
public class Main {
    PSVM() {
    try {
        Son son3 = new Son(-1, 10);
        Son son1 = new Son(50, 25);
        Sout("Father's Age:" + son1.fatherAge
        + "Son's Age:" + son1.sonAge);
    }
    catch (WrongAge e) {
        Sout("Exception caught:" + e.getMessage());
    }
}
```

~~Output~~

// output

Son age can't be negative

Son age can't be greater than father's age.