

```

class WrongAge extends Exception {
    String message;

    WrongAge(String message) {
        this.message = message;
    }

    public String toString() {
        return "WrongAge Exception: " + message;
    }
}

class Father {
    int fAge;
    Father(int age) throws WrongAge {
        if (age < 0) {
            throw new WrongAge("Father's age cannot be negative!");
        }
        fAge = age;
    }
}

class Son extends Father {
    int sAge;

    Son(int fAge, int sAge) throws WrongAge {
        super(fAge);

        if (sAge < 0) {
            throw new WrongAge("Son's age cannot be negative!");
        }
        if (sAge >= fAge) {
            throw new WrongAge("Son's age cannot be greater than or equal to Father's age!");
        }

        this.sAge = sAge;
    }
}

public class FatherSon {
    public static void main(String[] args) {
        try {
            Father father1 = new Father(40);
            Son son1 = new Son(40, 20);
            System.out.println("Father's age: " + father1.fAge + ", Son's age: " + son1.sAge);

            Father father2 = new Father(-5);
        }
        catch (WrongAge e) {
            System.out.println(e);
        }

        try {

```

```
        Son son2 = new Son(35, 40);
    }
    catch (WrongAge e) {
        System.out.println(e);
    }

    try {
        Son son3 = new Son(50, -10);
    }
    catch (WrongAge e) {
        System.out.println(e);
    }
}
}
```

```
D:\1BM23CS321>java FatherSon.java
Father's age: 40, Son's age: 20
WrongAge Exception: Father's age cannot be negative!
WrongAge Exception: Son's age cannot be greater than or equal to Father's age!
WrongAge Exception: Son's age cannot be negative!
```

# LAB PROGRAM - 7

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

```
class WrongAge extends Exception {
    String message;
    WrongAge (String message) {
        this.message = message;
    }
    public String toString() {
        return "Wrong Age Exception: " + message;
    }
}

class Father {
    int fAge;
    Father (int age) throws WrongAge {
        if (age < 0) {
            throw new WrongAge ("Father's age cannot be negative");
        }
        fAge = age;
    }
}

class Son extends Father {
    int sAge;
    Son (int fAge, int sAge) throws WrongAge {
        super (fAge);
        if (sAge < 0) {
            throw new WrongAge ("Son's Age is never");
        }
    }
}
```

```

        if (sAge >= fAge) {
            throw new WrongAge ("Son's age can't be greater than Father's Age");
        }
        this.sAge = sAge;
    }
}

```

```

public class FatherSon {
    public static void main (String[] args) {

```

```

        try {

```

```

            Father father1 = new Father (40);

```

```

            Son son1 = new Son (40, 20);

```

```

            System.out.println ("Father's Age: " + fAge);

```

```

            System.out.println ("Son's Age: " + sAge);

```

```

            Father father2 = new Father (35);

```

```

        } catch (WrongAge e) {
            System.out.println (e);

```

```

        } try {

```

```

            Son son2 = new Son (35, 40);

```

```

        } catch (WrongAge e)

```

```

            System.out.println (e);

```

```

        }

```



```

try {
    son sons = new Son (50, -10);
}
catch (WrongAge e) {
    System.out.println(e)
}
}
}

```

Output :

Father's Age: 40

Son's Age : 20

WrongAge Exception: Father's age cannot be negative.

WrongAge Exception: Son's Age cannot be greater than Father's Age.

~~WrongAge~~ Exception: Son's Age cannot be negative.

o/b seen  
 of this  
 21/11/21