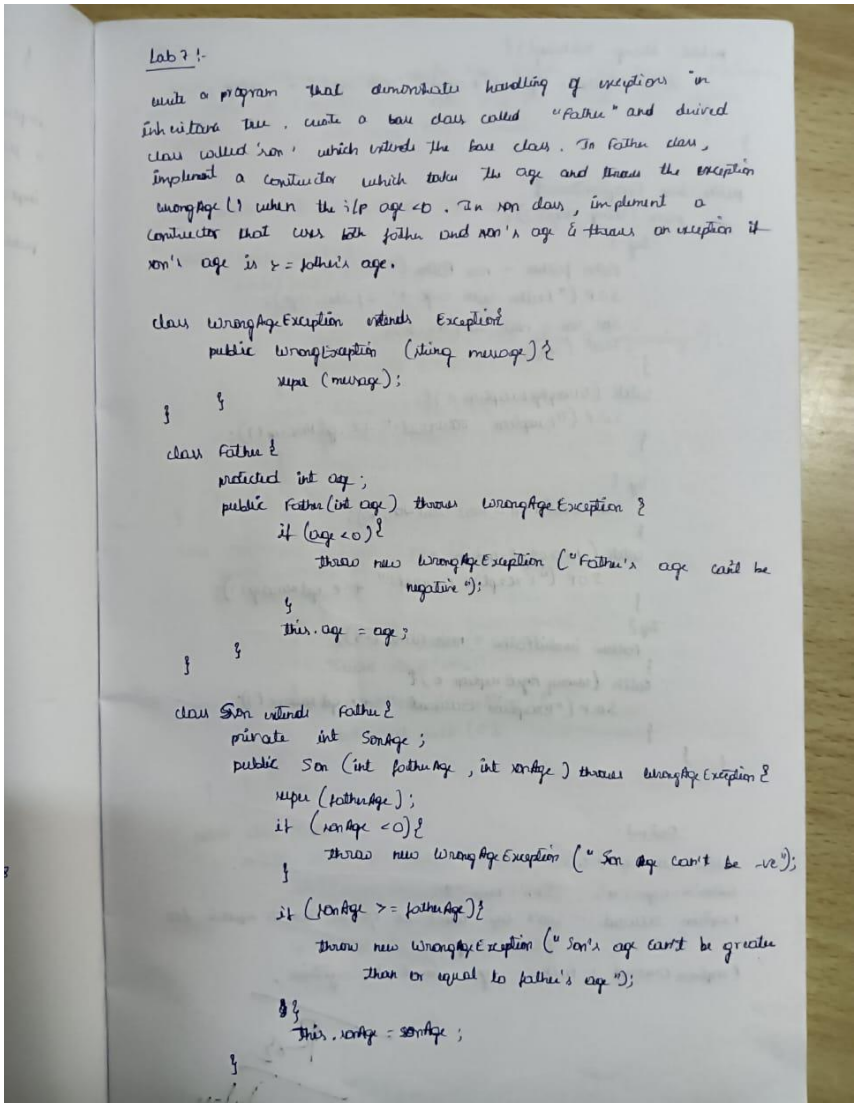
Lab 7:

7. 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.



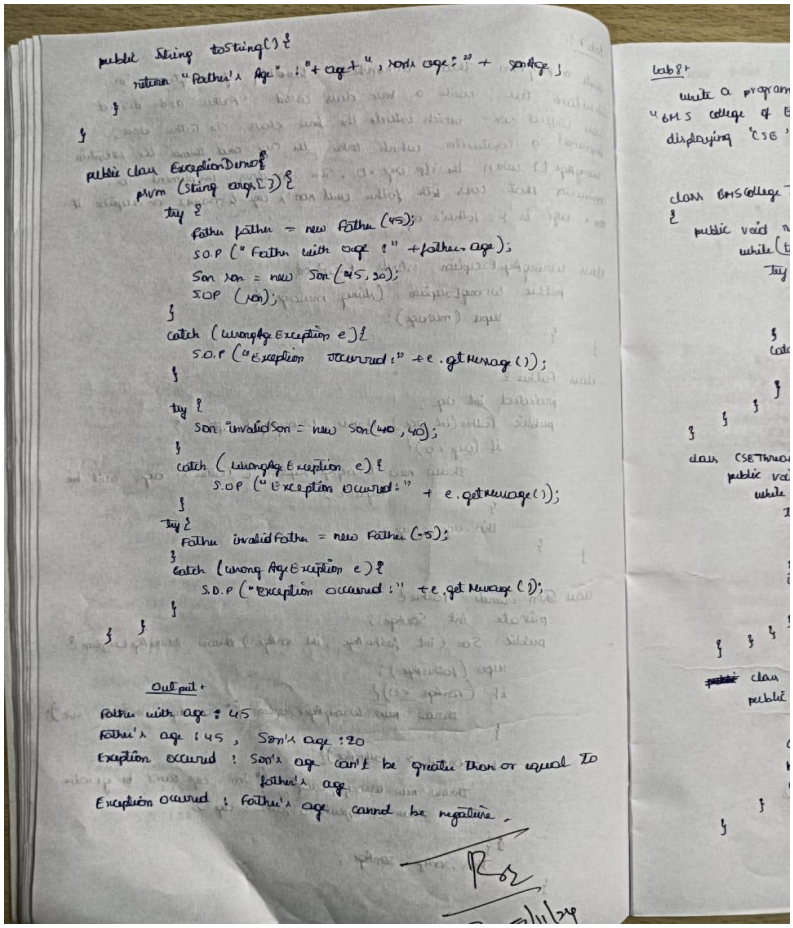
Lab 7 :-

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 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 WrongException (String message){
        super (message);
    }
}

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

class Son extends Father{
    private int SonAge;
    public Son (int fatherAge, int sonAge) throws WrongAgeException{
        super (fatherAge);
        if (sonAge <0){
            throw new WrongAgeException ("Son age can't be -ve");
        }
        if (sonAge >= fatherAge){
            throw new WrongAgeException ("Son's age can't be greater
                than or equal to father's age");
        }
        this.sonAge = sonAge;
    }
```



Code:

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

class Father {
    protected 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 {
    private 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;
    }

    @Override
    public String toString() {
        return "Father's Age: " + age + ", Son's Age: " + sonAge;
    }
}

public class ExceptionInheritanceDemo {
    public static void main(String[] args) {
        try {
            Father father = new Father(45);
            System.out.println("Father created with age: " + father.age);

            Son son = new Son(45, 20);
```

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

        try {
            Son invalidSon = new Son(40, 40);
        } catch (WrongAgeException e) {
            System.err.println("Exception occurred: " + e.getMessage());
        }

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

Output:

```
D:\Java_Lab_Programs>javac ExceptionInheritanceDemo.java

D:\Java_Lab_Programs>java ExceptionInheritanceDemo
Father created with age: 45
Father's Age: 45, Son's Age: 20
Exception occurred: Son's age cannot be greater than or equal to Father's age.
Exception occurred: Father's age cannot be negative.
```