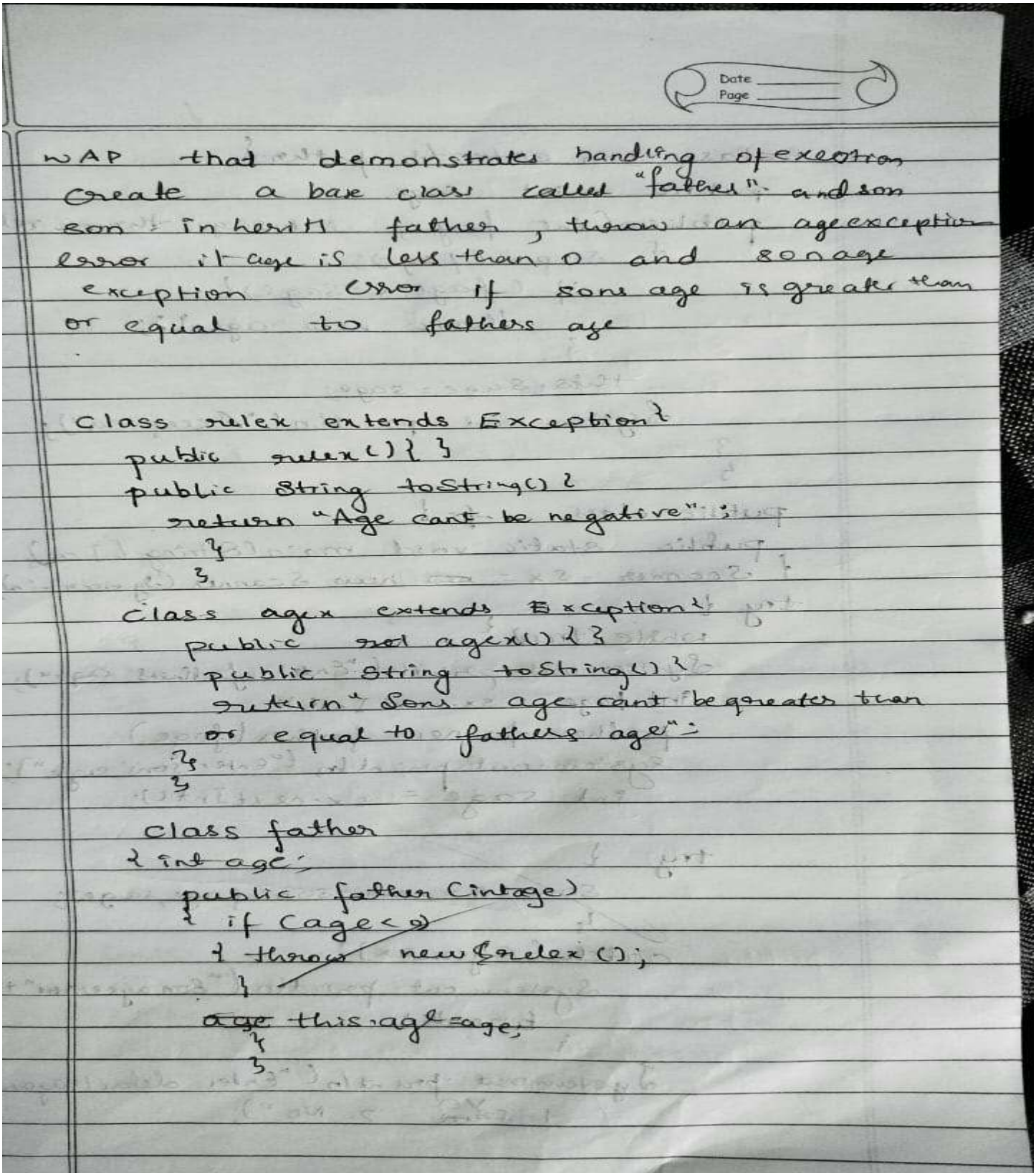
7)Write a program that demonstrates handling of exceptions in inheritance tree. Create a base class called “Father” and derived class called “Son” which extends the base class. In Father class, implement a constructor which takes the age and throws the exception WrongAge( ) when the input age=father’s age



```
WAP that demonstrates handling of exception
Create a base class called "father". and son
son inherits father, throw an age exception
error if age is less than 0 and son age
exception error if sons age is greater than
or equal to fathers age


Class rulex extends Exception{
  public rulex(){ }
  public String toString(){
    return "Age cant be negative";
  }
}
Class agex extends Exception{
  public agex(){}
  public String toString(){
    return "Sons age cant be greater than
    or equal to fathers age";
  }
}

Class father
{ int age;
  public father(int age)
  { if(age<0)
    { throw new rulex();
    }
    age this.age=age;
  }
}
```

```
class son extends father {
Public int sage;
public son(int fage, int sage) throws reles,agex
    { super (fage);
        if (fage <= sage){
            throw new agex();
        }
        this.sage = sage;
    System.out.println("Accepted");
}
}

public class fs {
 public static void main(String [] args)
{ Scanner sx = new Scanner (System.in);
try {
    while (true) {
    System.out.println("Enter fathers age");
     int fage = sx.nextInt();
      father f = new father (fage);
      System.out.println ("Enter sons age");
        int sage = sx.nextInt();

try {
      son s = new son (fage, sage);
    }
catch (agex ex){
```

```
if (choice != 1) {
    break;
  }
}
catch (Exception ex) {
    System.out.println (ex);
}
finally {
    sx.close();
  }
}
```

o/p

Enter fathers age

29

Enter sons age

39

sons age error sons age cant be greater than or equal to father age

Enter fathers age

-30

Age cant be negative

```
import java.util.*;

class relex extends Exception {
    public relex() {}
    public String toString() {
        return "age cant be negative";
    }
}

class agex extends Exception {
    public agex() {}
    public String toString() {
        return "sons age cant be greater than or same as father";
    }
}

class father {
    public int age;

    public father(int age) throws relex {
        if (age < 0) {
            throw new relex();
        }
        this.age = age;
    }
}

class son extends father {
    public son(int fage, int sage) throws agex, relex {
```

```
        super(fage);
        if (fage <= sage) {
            throw new agex();
        }
        age = sage;
        System.out.println("accepted details successfully");
    }
}

class fscombine {
    public static void main(String xx[]) {
        Scanner sx = new Scanner(System.in);
        try {
            while (true) {
                System.out.println("enter fathers age");
                int fage = sx.nextInt();
                father f = new father(fage);
                System.out.println("enter sons age");
                int sage = sx.nextInt();
                son s = new son(fage, sage);
                System.out.println("enter details again 1. yes 2. no");
                int ch = sx.nextInt();
                if (ch != 1) {
                    break;
                }
            }
        } catch (relex ex) {
            System.out.println("Father's age error: " + ex);
```

```
        } catch (agex ex) {
            System.out.println("Son's age error: " + ex);
        }
        sx.close();
    }
}
```

Output: