16/12/23

PROGRAM-7

Write a program that demonstrates handling of exceptions in inheritance tree.

```
import java.util.Scanner;
class WrongAge extends Exception
{
    public WrongAge (String message){
        super(message);
    }
}
class father{
    int fatherAge;
    public father (int fatherAge)
throws WrongAge {
        if (fatherAge<0) {
            throw new WrongAge ("Age cannot
            be negative");
    }
    this.fatherAge = fatherAge;
    }
}
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 must
        be less than father's age");
    }
    this.sonAge = sonAge;
    }
}
```

DATE: / /
PAGE NO: Usha Gold

```
public class fatherson {
    public static void main (String[] args ){
        Scanner sc = new Scanner (System.in );
        System.out.println ("Enter father's age and
            son's age ");
        int fa = sc.nextInt();
        int sa =  sc.nextInt();
      try {
          Son s = new Son (fa,sa);
          System.out.println (" father's age : " +
                                s.fatherAge );
      System.out.println ("son's age : "+s.sonAge );
        } catch (WrongAge e ) {
            System.out.println (" error : " e.getMessage());
    }
  }
}
```

=> Algorithm

Step 1: Start
Step 2: Create Extend WrongAge to Exception
Step 3: Create a class father
Step 4: Initialise fatherAge,
Step 5: Class Extend class son to class father.
Step 6: Initialise SonAge, Create a method Son initialise fatherage, sonage. It throws WrongAge exception if the son's age is greater than or equal to the father's age
Step 7: Main method in the fatherson class takes input for the father's age and son's age.
Step 8: Create son object & then prints the ages
Step 9: exit