Programme:

Write a java program that demonstrate exceptional handling of inheritance tree. Class "father" & derived class "Son".

```
class wrongAge extends Exception {
    public wrongAge(){
        super("Age cannot be negative");
    }
}
class input extends Exception{
    public input(){
        super("wrong input");
    }
}
class Father{
    public int age;
    Father(int age) throws wrongAge {
        if(age>0){
            throw new wrongAge();
        }
        this.age = age;
    }
}
class Son extends Father{
    int s_age;
    Son(int f_age, int s_age) throws wrongAge,
                                     input {
        super(f_age);
        if(f_age<0 && s_age<0){
            throw new wrongAge();
        }
        else if(f_age<=s_age){
            throw new input();
        }
        this.s_age=s_age;
    }
}
public class J{
    public statatic void main(String[] args){
        try{
```

```
try{
    Father f = new Father(40);
    System.out.println("Father's age: "f.age);
    Son s = new Son(40, 50);
    System.out.println("Son's age: " + s.sage);
} catch (wrongAge e){
    System.out.println(e.toString());
}
catch (input ae){
    System.out.println(ae.toString());
}
}
}
```

Algorithm:

Step 1 : Start

Step 2 : Initialize variable f_age, s_age

Step 3 : Create user defined exception

```
class wrongAge extends Exception{
    public wrongAge(){
        super("Age cannot be negative")
    }
}
```

Step 4 : Create another user defined exception

```
class input extends Exception{
    public input(){
        super("wrong input")
    }
}
```

Step 5 : Create class Father{

```
public int age;
Father(int age) throws wrongAge{
    if(age > 0){
        throw new exception wrongAge();
    }
    this.age = age;
}
```

Step 6 : Create class ~~Student~~ Son extends Father{

```
int s_age
Son(int f_age, int s_age) throws wrongAge, input{
    super(f_age);
```

```
        if(f_age<0 && s_age<0){
            throw new wrongAge();
        }
        else if(f_age<=s_age){
            throw new input();
        }
        this.age=s_age;
    }
```

write a program which create 2 thread where one is executed every 10 second & another for 2 second.

```
class One extends Thread{
    public void run(){
        int i=0;
        while(i<2){
            i++;
            try{
                System.out.println("BMS College of
                    Engineering");
                Thread.sleep(10000);
            }catch(Exception e){
                System.out.println(e.toString());
            }
        }
    }
}
class Two extends Thread{
    public void run(){
        int i=0;
        while(i<2){
            i++;
            try{
                System.out.println("CSE");
                Thread.sleep(2000);
            }catch(Exception e){
                System.out.println(e.toString());
            }
        }
    }
```

```
public class I {
    public static void main(String[] args){
        One t1 = new One();
        Two t2 = new Two();
        t1.start();
        t2.start();
    }
}
```

Algorithm:

Step 1 : Start

Step 2 : Initialize variable i = 0

Step 3 : Construct a class One & Two which extends Thread.

Step 4 : Under class One create method run(), under which, while (i < 5){

```
i++
try{
System.out.println("BMS College of Engineering");
Keep Thread in sleep(1000)
}catch (Exception e){
System.out.println ( e.toString().);
}
}
```

Step 5 : Under class Two create method run() under which, while(i<5){

```
i++;
try{
System.out.println("CSE")
} catch(Exception e) {
System.out.println(e.toString())
}
}
```

Step 6 : Create a main class

Call class One t1 = new One();

Call class Two t2 = new Two();

t1.start()

t2.start()

Step 7: Stop.

~~Output:~~

---

7) Continuation of ~~La pr~~ algorithm of program 7

Step 7: Call main class J {
try{ calling class, Father f = new Father(40);
~~Son s = new~~ System.out.print("Father's age: "+f.age)
call class, Son s = new Son(40, 56);
System.out.println(" Son's age :"+s.age)
} catch(~~Exception e){
System.out.println(e.toString());
}~~
catch (Exception e){
catch (wrongAge w){
System.out.println(w.toString());
}
catch (input i){
System.out.println(i.toString())
}

Step 8: Stop.

Output:
(i) ~~Father's~~ input: Wrong input
(ii) wrongAge: Age cannot be negative
(iii) Father's Age: 40
Son's age: 16

8) Output:
BMS College of Engineering
CSE
CSE
BMS ~~College of~~ Engineering

16.02.24

```
D:\java\oops>javac J.java

D:\java\oops>java J
Program 7
Name: Aditya Dinesh Netrakar
USN: 1BM22CS017
input: Wrong input

D:\java\oops>
```

```
D:\java\oops>javac I.java

D:\java\oops>java I
Program 8
Name: Aditya Dinesh Netrakar
USN: 1BM22CS017
BMS College of Engineering
CSE
CSE
BMS College of Engineering

D:\java\oops>
```