Output:
Enter Father Age.
5
Enter son Age
10
Error! Son's Age cannot be greater than father's age.

Enter Father Age.
-2
Error! Age cannot be negative.

Enter Father Age.
45
Enter Son Age.
-2
Error! Age cannot be negative.

Lab-8

Write a program which creates two threads, one thread displaying "BMS College of Engineering" once every 10 seconds and another displaying "CSE" once every two seconds.

```
import java.lang.*;
class DisplayMessage extends Thread
{   String message;
    long interval;

DisplayMessage(String message, long interval)
   { this.message = message;
     this.interval = interval;
   }
public void run()
 { try()
   { while(true)
     { System.out.println(message);
       Thread.sleep(interval);
     }
   } catch ( InterruptedException)
       { S.O.P(Thread.currentThread().getName() +
                    "interrupted ");
       }
   }
public class ThreadDemo
 { public static void main (String[] args)
   {
     DisplayMessage Thread thread1 =
     new DisplayMessage ("BMSCollege of Eng", 10000);
     DisplayMessage thread2 =
       new Display("CSE", 2000);

     thread1.setName ("Thread1");
     thread2.setName ("Thread2");

     thread1.start();
     thread2.start();
```

Rotate RC.

```
try
{  Thread.sleep(30000);
} catch (InterruptedException e)
   { S.O.P ("Main thread interrupted");
   }

thread1.interrupt(); //interrupt
thread2.interrupt();

S.O.P("Main thread exiting");
}
}
```

Output :-

```
BMS College of Engineering
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
Multi thread exiting
Thread 2 interrupted
Thread 1 interrupted
```

6-2-24

Name: Sai Krutin C R
USN: 1BM22CS232

8 Demonstrate inter process communication and deadlo

```
class Q {
int n;
synchronized int get()
{  System.out.println("Got " +n);
   return n;
}
synchronized void put(int n)
{  this.n = n;
   System.out.println("Put " + n);
}
}
class Producer implements Runnable
{  Q q;
   Producer(Q q)
   {  this.q = q;
      new Thread(this, "Producer").start();
   }
public void run()
{  int i = 0;
   while(i < 15)
   {  q.put(i++); }
}}
class Consumer implements Runnable
{  Q q;
   Consumer(Q q)
   {  this.q = q;
      new Thread(this, "Consumer").start(); }
public void run()
   {  int i = 0;
      while(i < 15) {  int r = q.get(); i++; }}
```

```
class PC
{ public static void main(String args[])
  {   Q q = new Q();
      new Producer(q);
      new Consumer(q);
      System.out.println("Press Control-C to stop");
  }
}
```

Output:-

Put: 1
Got: 1
Got: 1
Put: 2
Got: 1
Got: 2
Got: 1
Put: 3
Got: 1
Got: 3
Put: 2
Put: 4
Put: 3
Got: 4
Put: 4
Put: 5
Put: 5
Got: 5
Put: 6
Put: 7
Got: 7

9 Program on deadlock