Task 1

Write a *.java* program to find the prime numbers between the given interval. The lower and upper bound of the integral should be given as user input. Use Fermat's primality test to check primality. The algorithm can be written as follows:

- **Inputs:** n: a value to test for primality where $n > 3$;
  k: a parameter that determines the number of times to test for primality

- **Output:** composite if n is composite, otherwise probably prime
  Repeat k times:

    - Pick an integer a randomly in the range $[2, n-2]$
    - If $a^{n-1} \not\equiv 1 \ (mod \ n)$, then return composite

  If composite is never returned: return probably prime

Here is the way to generate random numbers

```
Random rn = new Random();
int randomNum = rn.nextInt((max − min) + 1) + min;
```

by adding

```
import java.util.Random;
```

in the beginning of the program.

Task 2 (Need for Speed)

Mahmut is a truck driver; driving an old, slow, rusty and falling apart truck. Recently, the needle on the speedometer fell off. He glued it back on, but he might have placed it at the wrong angle. Thus, when the speedometer reads s, his true speed is $s + c$, where c is an unknown constant (possibly negative).

Mahmut made a careful record of a recent journey and wants to use this to compute c. The journey considered of n segments. In the ith segment, he traveled a distance of d_i and the speedometer read s_i for the entire segment. The whole journey took time t. Help Mahmut by computing c.

Note that while Mahmut's speedometer might have negative readings, his true speed should be greater than zero for each segment of the journey.

The input starts with two integers n $(1 \leq n \leq 1000)$, the number of sections in Mahmut's journey, and t $(1 \leq t \leq 10^6)$, the total time. This is followed by $2n$ lines, each two describing one segment of Mahmut's journey while i^{th} of these lines contains two integers d_i $(1 \leq d_i \leq 1000)$ and s_i

($|s_i| \leq 1000$), the distance and speedometer reading for the ith segment of the journey.

Write a *.java* program to find the constant c. Time is specified in hours, distance in km, and speed in km per hour. An example program input-output should be like

```
>> INPUT
>> Please enter the number of sections: 3
>> Please enter the total time: 5
>> Please enter the distance of the segment 1: 4
>> Please enter the speedometer reading of segment 1: −1
>> Please enter the distance of the segment 2: 4
>> Please enter the speedometer reading of segment 2: 0
>> Please enter the distance of the segment 3: 10
>> Please enter the speedometer reading of segment 3: 3
>> OUTPUT
>> c=3.000000
>>
>> INPUT
>> Please enter the number of sections: 4
>> Please enter the total time: 10
>> Please enter the distance of the segment 1: 5
>> Please enter the speedometer reading of segment 1: 3
>> Please enter the distance of the segment 2: 2
>> Please enter the speedometer reading of segment 2: 2
>> Please enter the distance of the segment 3: 3
>> Please enter the speedometer reading of segment 3: 6
>> Please enter the distance of the segment 4: 3
>> Please enter the speedometer reading of segment 4: 1
>> OUTPUT
>> c=−0.508653
```

Submission Information

LabWorks are evaluated within the recitation hour. You may work with your friends. Collaboration is strongly recommended. However, each student should be able to present his/her program. Submissions after the recitation hour will not be evaluated.