**Exercise 1:-**

**Snail**

The purpose of this exercise is to train you in usage of simple integer operations.

Estimated workload of this exercise is *30 min*.

**Description**

Consider a snail travels up a tree a feet each day. Then snail slides down b feet each night. Height of the tree is h feet.

Please, proceed to [Snail](/Prince779/snail/-/blob/master/src/main/java/com/epam/rd/autotasks/snail/Snail.java) class and write a program that prints number of days for the snail to reach the top of the tree.

Program reads a, b, h line by line. Input values are guaranteed to be positive integers.

If the snail cannot reach the top of the tree, print the message Impossible.

**Examples**

Input:

4

2

14

Output:

6

Input:

4

3

10

Output:

7

Input:

4

4

10

Output:

Impossible

Input:

4

4

1

Output:

1

**Exercise 2:-**

**Pizza Split**

The purpose of this exercise is to familiarize you with basic conditional and cyclic operations.

Estimated workload of this exercise is *20 minutes*.

**Description**

Please, proceed to [PizzaSplit](/Prince779/pizza-split/-/blob/master/src/main/java/com/epam/rd/autotasks/pizzasplit/PizzaSplit.java) class. The program must read two values from System.in:

* number of people;
* number of pieces per pizza.

It is guaranteed that this values are positive integers.

Then the program must print the minimum number of pizzas (not zero) so that everyone has an equal number of slices and no slice is left.

**Example**

Input: 12 8

Output: 3

**Exercise 3:-**

**Go dutch**

The purpose of this exercise is to familiarize you with basic conditional and cyclic operations.

Estimated workload of this exercise is *20 minutes*.

**Description**

Consider a company of friends visiting a restaurant. They decided to equally split the bill.

Friends decided to add 10 percent of the bill total amount as tips. Then they cover the total payment in equal parts.

Please, proceed to [GoDutch](/Prince779/go-dutch/-/blob/master/src/main/java/com/epam/rd/autotasks/godutch/GoDutch.java) class and write a program that reads a bill total amount and a number of friends, and then prints part to pay.

Consider some details:

* Program must read data from System.in
* Bill total amount cannot be negative. If input value is negative, the program stops, printing: Bill total amount cannot be negative
* Number of friends cannot be negative or zero. If input value is, then the program stops, printing: Number of friends cannot be negative or zero
* Bill total amount, number of friends and part to pay are integers

**Example**

Input:

1000

5

Output:

220

**Exercise 4:-**

**Cycle Swap**

The purpose of this exercise is to train you to work with arrays.

Estimated workload of this exercise is *30 min*.

**Description**

Please, proceed to [CycleSwap](/Prince779/cycle-swap/-/blob/master/src/main/java/com/epam/rd/autotasks/CycleSwap.java) class and implement its static methods:

* void cycleSwap(int[] array)  
  Shifts all the elements in the given array to the right by 1 position.  
  In this case, the last array element becomes first.  
  For example, 1 3 2 7 4 becomes 4 1 3 2 7.
* void cycleSwap(int[] array, int shift)  
  Shift all the elements in the given array to the right in the cycle manner by shift positions.  
  Shift value is guaranteed to be non-negative and not bigger than the array length.  
  For example, 1 3 2 7 4 with a shift of 3 becomes 2 7 4 1 3.

For a greater challenge, try not using loops in your code (not mandatory).

Note that input array may be empty.

**Examples**

Code Sample:

...

int[] array = new int[]{1, 3, 2, 7, 4};

CycleSwap.cycleSwap(array);

System.out.println(Arrays.toString(array));

Output:

[4, 1, 3, 2, 7]

Code Sample:

...

int[] array = new int[]{1, 3, 2, 7, 4};

CycleSwap.cycleSwap(array, 2);

System.out.println(Arrays.toString(array));

Output:

[7, 4, 1, 3, 2]

Code Sample:

...

int[] array = new int[]{1, 3, 2, 7, 4};

CycleSwap.cycleSwap(array, 5);

System.out.println(Arrays.toString(array));

Output:

[1, 3, 2, 7, 4]