Q1. Write a class **ConversionSystem** to perform the two operations: first find the integer from a given string and second to convert Polar coordinates to Cartesian coordinates. The classes **BinarytoDecimal** and **DecimaltoBinary** inherit the class **ConversionSystem.** Perform the following operations based on the input and the options specified below:

1. Use function *conversionNumber(A,R)* inside class **ConversionSystem**
   # Returns the Cartesian coordinates (x,y) of the given polar coordinates of the point (R,A) where R is the radial coordinate and A is the angular coordinate often called the polar angle. *(Upto Three Decimal Points).*
   Note:
   convert A degree to its Radian as
   A(in radian) = A (in degrees) * (pi / 180). (Use pi =3.14)
   New coordinates x and y are calculated as x = R * cosA and y = R * sinA.

2. Based on input A and R perform the following conversion. *(Do Not Use Inbuilt Function for Conversion)*
   - ***If A = 0 and R = 1***
     Use function *conversionNumber(S)* inside class **ConversionSystem**
     #Returns the integer part from the given input String
   - ***If A = 2 and R = 10***
     Use function *conversionNumber(S)* inside class **BinarytoDecimal**
     #Returns a Decimal number after conversion of the given number from Binary to decimal
   - ***If A= 10 and R = 2***
     Use function *conversionNumber(S)* inside class **DecimaltoBinary**
     #Returns a Binary number after conversion of the given number from Decimal to Binary

Write a java program **DemonstrateNumberSystem** to test functionality of the **ConversionSystem**

Input Format

- First line takes input option either i or j
- Based on the option, Second Line takes input in the following format where *i* and *j* are options
  - If option is *i,* Input format is <S A R> *where S is a String, A is the current base of the number system and R is the base to which number is to be converted.*
  - If option is *j,* Input format is <A R> *where A is the polar angle in degree (in polar coordinates) and R (the radial coordinate).*

Output Format

- If input is valid then,
  - For option *i,* print the converted number
  - For option *j,* print the Cartesian coordinates *as <x space followed by y>.* upto three decimal points.
- If not a valid input, print **WRONGOUTPUT**

<u>Sample Input</u>

Input	Output
i INTEGER100 0 1	100
i 7 10 2	111
i 1010 2 10	10
j 30 3	2.598 1.499

Q2. Create a class hierarchy of shapes and write a Java program that computes the amount of paint needed to paint different objects. The hierarchy will consist of a parent class **Shape** with three derived classes - **Sphere, Rectangle, and Cylinder**.

1. Write a class **Shape** with the following specification:
   - A method **area** (no function parameters) which returns 0.00
2. Class **Sphere** inherits class **Shape**.
   - A sphere has a radius
   - The method **area** is overridden, and returns the computed area of the sphere rounded to two decimal places (formula:***4*pi*(radius^2)***).
3. Class **Rectangle** inherits class **Shape**.
   - A rectangle is defined by its length and width.
   - The method **area** is overridden, and returns the computed area of the rectangle rounded to two decimal places (formula: ***length * width***).
4. Class **Cylinder** extends class **Shape**.
   - A cylinder is defined by a radius and height
   - The method **area** is overridden, and returns the computed area of the cylinder rounded to two decimal places (formula: ***pi*(radius^2)*height)***.

(Use pi =3.14)

Create a test class **Paint** that contains

- coverage (***Coverage** is defined as the rate at which a coating spreads on a material and is usually expressed in units of square meters per liter*).
- A method **getAmount** computes the amount of paint needed to paint a shape.

  The amount of the paint = The area of the shape ÷(divided by) the coverage for the paint. If the calculated amount is a floating point value, then return the floor value.

Constraints

Length, width, height, radius, coverage are non negative integers.

Input Format

The input line is given as a space separated list in the format:

<Character> <Integer> <Parameters>

<Character> can be (S or R or C)

<Integer> represents the coverage

- ***S** : If the <Character> is **S** then:* <Parameters> is a space separated list where the first parameter is the *name of the shape* (single word) and the second parameter is an integer which is the *radius* of the Sphere.
- ***R** : If the <Character> is **R** then,* <Parameters> is a space separated list where the first parameter is the *name of the shape* (single word), the second and the third parameters are integers representing *length* and *width* of the *rectangle* respectively.
- ***C**: If the <Character> is **C** then,* <Parameters> is a space separated list where the first parameter is the *name of the shape* (single word), the second and the third parameters are integers representing *radius* and *height* of the *cylinder* respectively.

Output Format

- If the <Character> in input is **S** then, display the *name of the shape* and *the amount of paint* required to paint the sphere separated by a space.
- If the <Character> in input is **R** then, display the *name of the shape* and *the amount of paint* required to paint the Rectangle.
- If the <Character> in input is **C** then, display the *name of the shape* and *the amount of paint* required to paint the cylinder.
- Any input not following the mentioned Input Format should display “INVALID”

Sample Input

INPUTS	*OUTPUTS*
R 350 Window 90 63	*Window 16*

Q3. Write a Java program to read a positive integer *S* (greater than zero) and a value *N* (*N* can be integer, float or character). Use **constructor overloading** to compute and display values based on the input as follows.

a. If *N* is an integer then compute Nth power of *S* and display the computed value. If *N* < *0*, then print "INVALID".

b. If *N* is float, then compute (N*S) and print the ceil value. If *N* < 0, then print 0.

c. If *N* is a character then return the String, which is character *N* concatenated *S* times. If *S* is greater than 10, then print "OVERFLOW"

Constraints

S is a positive integer.

N can be integer, float or character.

Input Format

The First line of the input contains a positive integer *S*.
The second line of the input contains a value *N* .

Output Format

The output prints the result after computation based on the value of *N*.

Sample Input

INPUTS	*OUTPUTS*
10 5	100000
7 2.12	15
3 p	ppp