PROGRAMS

- **Execute the 3 programs given below and submit a printout of the same by 18.11.2019**

1) Create a function that takes in n, a, b and returns the number of values raised to the nth power lie in the range [a, b], inclusive.

Examples

PowerRanger (2, 49, 65) ➞ 2

2 squares lie between 48 and 65, 49 (7^2) and 64 (8^2)

PowerRanger(3, 1, 27) ➞ 3

3 cubes lie between 1 and 27, 1 (1^3), 8 (2^3) and 27 (3^3)

PowerRanger(10, 1, 5) ➞ 1

1 value raised to the 10th power lies between 1 and 5, 1 (1^10)

PowerRanger(5, 31, 33) ➞ 1

PowerRanger(4, 250, 1300) ➞ 3

Notes

- Remember that the range is inclusive.
- a < b will always be true.
- Consider only integer values

2) Create a function that takes a string as an input. The string is in the form of a number and converts and returns the integer value. Use this function to input two strings and add their integral values.

Examples:

Input two strings:

“1453”

“4321”

After converting and adding, the sum = 5774

3) Create a function toBinary( ), that takes a decimal number and returns the binary conversion (also an integer). Input a string and print the binary code of each character’s ASCII value separated by a space.

Examples

Enter a string:

“ABC”

1000001   1000010   1000011

Modulo and time difference

Dealing with remainders may cause heavy headache to novice programmers. Let us write a simple program which has this operation for its core to study integer division better. At the same time we'll have some practice in handing dates - which sometimes gives headache even to experienced coders.

In arithmetic, the remainder (or modulus) is the amount "left over" after performing the division of two integers which do not divide evenly (from Wiki). This task will provide further practice with modulo operation.

Suppose, we are given two timestamps - for example, when the train or ferry boat starts its travel and when it finishes. This may look like:

```
start: May 3, 17:08:30
end  : May 8, 12:54:15
```

and we are curious to know, how much time (in days, hours, minutes and seconds) is spent in traveling (perhaps, to choose faster variant). How this could be achieved?

One of the easiest way is:

- convert both timestamps to big numbers, representing seconds from the beginning of the month (or year, or century);
- calculate their difference - i.e. travel time in seconds;
- convert this difference back to days, hours, minutes and seconds.

First operation could be performed by multiplying minutes by `60` and hours by `60*60` etc. and summing all values up.
The third operation should be performed on contrary by several divisions with keeping remainders.

In this task we are given several pair of timestamps. We suppose that both dates in the pair are always in the same month, so only number of day will be given. We want to calculate difference between timestamps in each pair.

Input data: the first line contains number of test-cases, other lines contain test-cases themselves. Each test-case contains `8` numbers, `4` for each timestamp: `day1 hour1 min1 sec1 day2 hour2 min2 sec2` (second timestamp will always be later than first).
Answer: for each test-case you are to output difference as following `(days hours minutes seconds)` - please note brackets - separated by spaces.

Example:

```
input data:
3
1 0 0 0 2 3 4 5
5 3 23 22 24 4 20 45
8 4 6 47 9 11 51 13

answer:
(1 3 4 5) (19 0 57 23) (1 7 44 26)
```