Exercises: Functions and Forms

Problems for exercise for the "PHP Fundamentals" course @ SoftUni.

You can check your solutions in Judge.

I. Functions

1. Smallest of Three Numbers

Write a function to print the smallest of three integer numbers. Use appropriate name for the function.

Examples

Input	Output
2 5 3	2
600 342 123	123
25 21 4	4

2. Vowels Count

Write a function that receives a single string and prints the count of the vowels. Use appropriate name for the function.

Examples

Input	Output
SoftUni	3
Cats	1
JS	0

3. Characters in Range

Write a function that receives two characters and prints on a single line all the characters in between them according to ASCII.

Examples

Input	Output
a d	b c

# :	$ % & ' ( ) * + , - . / 0 1 2 3 4 5 6 7 8 9
C #	$ % & ' ( ) * + , - . / 0 1 2 3 4 5 6 7 8 9 : ; < = > ? @ A B

4. Password Validator

Write a program that checks if a given password is valid. Password rules are:

- **6 – 10 characters (inclusive)**
- **Consists only of letters and digits**
- **Have at least 2 digits**

If a password is valid print "Password is valid". If it is not valid, for every unfulfilled rule print a message:

- **"Password must be between 6 and 10 characters"**
- **"Password must consist only of letters and digits"**
- **"Password must have at least 2 digits"**

Examples

Input	Output
`logIn`	Password must be between 6 and 10 characters Password must have at least 2 digits
`MyPass123`	`Password is valid`
`Pa$s$s`	Password must consist only of letters and digits Password must have at least 2 digits

Hints

Write a function for each rule.

5. Add and Subtract

You will receive 3 **integers.** Write a function **`sum`** to get the sum of the first two integers and **`subtract`** function that subtracts the third integer from the result from the Sum function.

Examples

Input	Output
23 6 10	19
1 17 30	-12
42 58 100	0

6. Middle Characters

You will receive a single string. Write a function that prints the middle character. If the length of the string is even there are two middle characters.

Examples

Input	Output
aString	r
someText	eT
3245	24

7. NxN Matrix

Write a function that receives a single integer **n** and prints **N x N** matrix with that number.

Examples

Input	Output
3	3 3 3 3 3 3 3 3 3
7	7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7
2	2 2 2 2

8. Factorial Division

Read two **integer** numbers. Calculate factorial of each number. **Divide the first result** by the **second** and print the division **formatted to the second decimal point**.

Examples

Input	Output
5 2	60.00

Input	Output
6 2	360.00

9. Palindrome Integers

A palindrome is a number which reads the same backward as forward, such as 323 or 1001. Write a program which reads a positive integer numbers until you receive "**End**", for each number print whether the number is palindrome or not.

Examples

Input	Output
123 323 421 121 END	false true false true

Input	Output
32 2 232 1010 END	false true true false

10. Top Number

A top number is an integer that holds the following properties:

- Its **sum of digits is divisible by 8**, e.g. 8, 16, 88.
- Holds at least **one odd digit**, e.g. 232, 707, 87578.

Write a program to print all master numbers in the range **[1…n]**.

Examples

Input	Output
50	17 35

Input	Output
100	17 35 53 71 79 97

11. *Array Manipulator

Trifon has finally become a junior developer and has received his first task. It's about manipulating an array of integers. He is not quite happy about it, since he hates manipulating arrays. They are going to pay him a lot of money, though, and he is willing to give somebody half of it if to help him do his job. You, on the other hand, love arrays (and money) so you decide to try your luck.

The array may be manipulated by one of the following commands

- **exchange {index}** – splits the array **after** the given index, and exchanges the places of the two resulting sub-arrays. E.g. [1, 2, 3, 4, 5] -> exchange 2 -> result: [4, 5, 1, 2, 3]
  - If the index is outside the boundaries of the array, print "**Invalid index**"
- **max even/odd**– returns the **INDEX** of the max even/odd element -> [1, 4, 8, 2, 3] -> **max odd** -> print **4**
- **min even/odd** – returns the **INDEX** of the min even/odd element -> [1, 4, 8, 2, 3] -> **min even** > print **3**
  - If there are two or more equal **min/max** elements, return the index of the **rightmost** one
  - If a **min/max even/odd** element **cannot** be found, print **"No matches"**
- **first {count} even/odd**– returns the first {count} elements -> [1, 8, 2, 3] -> **first 2 even** -> print [**8, 2]**
- **last {count} even/odd** – returns the last {count} elements -> [1, 8, 2, 3] -> **last 2 odd** -> print [**1, 3]**
  - If the count is greater than the array length, print "**Invalid count**"

    - If there are **not enough** elements to satisfy the count, print as many as you can. If there are **zero even/odd** elements, print an empty array "[]"
- **end** – stop taking input and print the final state of the array

Input

- The input data should be read from the console.
- On the first line, the initial array is received as a line of integers, separated by a single space
- On the next lines, until the command "**end**" is received, you will receive the array manipulation commands
- The input data will always be valid and in the format described. There is no need to check it explicitly.

Output

- The output should be printed on the console.
- On a separate line, print the output of the corresponding command
- On the last line, print the final array in **square brackets** with its elements separated by a comma and a space
- See the examples below to get a better understanding of your task

Constraints

- The **number of input lines** will be in the range [2 … 50].
- The **array elements** will be integers in the range [0 … 1000].
- The **number of elements** will be in the range [1 .. 50]
- The **split index** will be an integer in the range [-2^{31} … $2^{31} - 1$]
- **first/last count** will be an integer in the range [1 … $2^{31} - 1$]
- There will **not** be redundant whitespace anywhere in the input
- Allowed working time for your program: 0.1 seconds. Allowed memory: 16 MB.

Examples

Input	Output
1 3 5 7 9 exchange 1 exchange 5 exchange 3 end	Invalid index [3, 5, 7, 9, 1]
1 1000 10 100 1000 max even max odd end	4 0 [1, 1000, 10, 100, 1000]
1 113 11 1123 min odd min even end	0 No matches [1, 113, 11, 1123]
1 8 2 3 10 5 first 0 odd first 4 even first 7 odd first 2 odd end	[] [8, 2, 10] Invalid count [1, 3] [1, 8, 2, 3, 10, 5]

1 8 2 3 10 5 last 3 even last 1 odd last 7 odd last 4 odd end	[8, 2, 10] [5] Invalid count [1, 3, 5] [1, 8, 2, 3, 10, 5]
6 10 7 4 9 14 19 2 22 45 exchange 4 max even min even first 3 odd exchange 22 exchange 6 min odd min even last 7 even end	3 2 [19, 45, 7] Invalid index 0 5 [4, 14, 2, 22, 6, 10] [7, 4, 9, 14, 19, 2, 22, 45, 6, 10]

II. Forms

12. *Even Numbers from 1 to N

You are given a number **num**. Write a PHP script that loops through all of the numbers from **1** to **num** and prints only the even ones. The input comes as a parameter named **num**. The parameter **num** will hold a positive integer.

Skeleton

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>PHP Form</title>

</head>
<body>
<form>
    N: <input type="text" name="num" />
    <input type="submit" />
</form>
<!--Write your PHP Script here-->
</body>
</html>
```

Examples

Parameter name	Input	Output
num	5	2 4

Input	Output
2	2

13. *Prime Numbers from N to 1

You are given a number **num**. Write a PHP script that prints only the **prime** numbers from **num to 1**. A prime number is a number that can be divided only by 1 and itself. **1 is not a prime number.** The input comes as a parameter named **num**. The parameter **num** will hold a positive integer.

Skeleton

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>PHP Form</title>

</head>
<body>
<form>
    N: <input type="text" name="num" />
    <input type="submit" />
</form>
<!--Write your PHP Script here-->
</body>
</html>
```

Examples

Parameter name	Input	Output
num	10	7 5 3 2

Input	Output
20	19 17 13 11 7 5 3 2