# Homework: PHP Syntax

## Student Information

You will be given 3 lines of input – student name, age and average grade. Your task is to print all the info about the student in the following format: "Name: {student name}, Age: {student age}, Grade: {student grade}".

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| John  15  5.40 | Name: John, Age: 18, Grade: 5.40 |
| Steve  16  2.50 | Name: Steve, Age: 16, Grade: 2.50 |
| Marry  12  6.00 | Name: Marry, Age: 12, Grade: 6.00 |

## Sum Two Numbers

Write a PHP script **SumTwoNumbers.php** that decleares two variables, **firstNumber** and **secondNumber.** They should hold integer or floating-point numbers (hard-coded values). Print their **sum** in the output in the format shown in the examples below. The numbers should be **rounded to the second** number after the decimal point. Find in Internet how to **round** a given number in PHP. Examples:

|  |  |
| --- | --- |
| **Input** | **Output** |
| 2  5 | $firstNumber + $secondNumber = 2 + 5 = 7.00 |
| 1.567808  0.356 | $firstNumber + $secondNumber = 1.567808 + 0.356 = 1.92 |
| 1234.5678  333 | $firstNumber + $secondNumber = 1234.5678 + 333 = 1567.57 |

## Foreign Languages

Write a program, which prints the language, that a given country speaks. You can receive only the following combinations: English **is spoken** in England and USA; Spanish **is spoken** in Spain, Argentina and Mexico; for the others**,** we should print "unknown".

### Input

You will receive a **single country name** on a **single line**.

### Output

**Print** the **language**, which the country **speaks**, or if it is **unknown** for your program, print **"**unknown**"**.

### Examples

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Input** | **Output** |  | **Input** | **Output** |
| USA | English |  | Germany | unknown |

## Interval of Numbers

Write a program, which takes **two numbers** as input and prints the **interval of numbers between them**, **starting** from the **smaller one** and **ending** with the **larger** one.

### Input

You will receive **two lines**. Each of them will contain **one integer**.

### Output

Print all the numbers separated on **new lines**.

### Constraints

* The numbers, which you receive will be in the interval **[0…100]**.
* The two numbers, which you take as an input will **not be equal**.

### Examples

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Input** | **Output** |  | **Input** | **Output** |
| 42  48 | 42  43  44  45  46  47  48 | 100  14 | 14  15  16  *continues...*  98  99  100 |

## Sum of Odd Numbers

Write a program that prints the next **n** **odd numbers** (starting from 1) and on the **last row** prints the **sum of them**.

### Input

On the first line, you will receive a number – **n**. This number shows how many **odd numbers** you should print.

### Output

Print the next **n** odd numbers, starting from **1**, separated by **new lines**. On the last line, print the **sum** of these numbers.

### Constraints

* **n** will be in the interval **[1…100]**

### Examples

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Input** | **Output** |  | **Input** | **Output** |
| 5 | 1  3  5  7  9  Sum: 25 |  | 3 | 1  3  5  Sum: 9 |

## Multiplication Table

You will receive an **integer** as an input from the console. Print the **10 times table** for this integer. See the examples below for more information.

### Output

Print every row of the table in the following format:

{theInteger} X {times} = {product}

### Constraints

* The integer will be in the interval **[1…100]**

### Examples

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Input** | **Output** |  | **Input** | **Output** |
| 5 | 5 X 1 = 5  5 X 2 = 10  5 X 3 = 15  5 X 4 = 20  5 X 5 = 25  5 X 6 = 30  5 X 7 = 35  5 X 8 = 40  5 X 9 = 45  5 X 10 = 50 |  | 2 | 2 X 1 = 2  2 X 2 = 4  2 X 3 = 6  2 X 4 = 8  2 X 5 = 10  2 X 6 = 12  2 X 7 = 14  2 X 8 = 16  2 X 9 = 18  2 X 10 = 20 |

## Multiplication Table 2.0

Rewrite you program so it can receive the **multiplier from the console**. Print the **table from the given multiplier to 10**. If the given multiplier is **more than 10** - print only one row with the **integer**, the given **multiplier** and the **product**. See the examples below for more information.

### Output

Print every row of the table in the following format:

{theInteger} X {times} = {product}

### Constraints

* The integer will be in the interval **[1…100]**

### Examples

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Input** | **Output** |  | **Input** | **Output** |  | **Input** | **Output** |
| 5  1 | 5 X 1 = 5  5 X 2 = 10  5 X 3 = 15  5 X 4 = 20  5 X 5 = 25  5 X 6 = 30  5 X 7 = 35  5 X 8 = 40  5 X 9 = 45  5 X 10 = 50 |  | 2  5 | 2 X 5 = 10  2 X 6 = 12  2 X 7 = 14  2 X 8 = 16  2 X 9 = 18  2 X 10 = 20 |  | 2  14 | 1. X 14 = 28 |

## Restaurant Discount

A restaurant wants to automate their reservation process. They need a program that reads the **count of people** andthe **package** from the console and calculates **how much** the customer should **pay** to book the place.

Different halls have different prices:

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Small Hall** | **Terrace** | **Great Hall** |
| **Price** | 2500$ | 5000$ | 7500$ |
| **Capacity** | 50 | 100 | 120 |

The restaurant has **discounts** depending on the **service package,** which the group wants.

You can see the discounts in the table below:

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Normal** | **Gold** | **Platinum** |
| **Discount** | 5% | 10% | 15% |
| **Price** | 500$ | 750$ | 1000$ |

You should **add** the **price** of the **package** to the **price** of the **hall**. The discount is calculated based on the **hall’s price + package’s price**.

Example: The group has **10 people** and wants the **gold package 🡺 $292.50 per person**:

* **10 people** <= 50 🡺 they get the **Small Hall** 🡺 $2500
* Gold package 🡺 **$750**, **10%** discount on the entire purchase
* Total price: **$2500 + $750** = **$3250**
* Discount: $3250 - **10% discount** = $2925
* Price per person: $2925 / **10 people** = **$292.50 per person**

You should print **which hall** is the **most suitable** for the group and the **price per person**. If the group is **bigger than 120** people – print “**We do not have an appropriate hall.**”.

### Input

* First line: the **group size** as an integer.
* Second line: the **type** of the **package** as **string**

### Output

Print the output in the following format:

|  |
| --- |
| We can offer you the {hallName}  The price per person is {price}$ |

**Format** the **price** to the **2nd decimal place**.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 20  Gold | We can offer you the **Small Hall**  The price per person is **146.25$** |
| 90  Platinum | We can offer you the **Terrace**  The price per person is **56.67$** |
| 150  Normal | We do not have an appropriate hall. |

## Problem 9. String Length

Write a program that reads from the console a string of maximum 20 characters. If the length of the string is less than 20, the rest of the characters should be filled with \*. Print the resulting string on the console.

Examples:

|  |  |
| --- | --- |
| **Input** | **Output** |
| Welcome to SoftUni! | Welcome to SoftUni!\* |
| a regular expression (abbreviated regex or regexp and sometimes called a rational expression) is a sequence of characters that forms a search pattern | a regular expression |
| PHP | PHP\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* |

## Problem 10. Find the Letter

Write a program, which receives a string and prints the index of a given letter in the string. The tricky part is that you will have to find **not** the first letter, but the **nthletter**.

### Input

* On the first line, you will receive the **string** you are going to search through.
* On the **second** line, you will receive an **array** with exactly **two** **elements**:
  + The **first** element will be the **letter**, which you have to search for.
  + The **second** element will be an **integer N**, showing us which **occurrence** ofthe **letter** we are searching for.

**Example**: If we receive the string “Programming is awesome!” and on the next line we receive the array “m 3”. We should find the **third** occurrence of the letter ‘**m**’. It can be found on **20th** index.

### Output

If the nth occurrence of the letter is present in the string, print **the index** of that occurrence.

If the letter is **not present** in the string, or there are **less occurrences** than **N**, print:

* “Find the letter yourself!”

*Note: the* ***comparison*** *should be* ***case******sensitive****.*

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| Programming is awesome!  m 3 | 20 |
| Strings, strings everywhere...  e 5 | Find the letter yourself! |

## Problem 11. Digits, letters and other

Write a program that receives a single string and on the first line prints all the digits, on the second – all the letters, and on the third – all the other characters. There will always be at least one digit, one letter and one other characters.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| Agd#53Dfg^&4F53 | 53453  AgdDfgF  #^& |

### Hints

* Read the input.
* Do the same for the letters and other chars
  + Find something like isDigit method for the letters.

## Problem 12. \*Non-Repeating Digits

Write a PHP script **NonRepeatingDigits.php** that declares an integer variable **N**, and then finds all 3-digit numbers that are less or equal than **N (<= N)** and consist of unique digits. Print "no" if no such numbers exist. Examples:

|  |  |
| --- | --- |
| **Input** | **Output** |
| 1234 | 102, 103, 104, 105, 106, 107, 108, 109, 120, 123, 124, 125, 126, 127, 128, 129, 130, 132, 134, 135, …, 980, 981, 982, 983, 984, 985, 986, 987 |
| 145 | 102, 103, 104, 105, 106, 107, 108, 109, 120, 123, 124, 125, 126, 127, 128, 129, 130, 132, 134, 135, 136, 137, 138, 139, 140, 142, 143, 145 |
| 15 | no |
| 247 | 102, 103, 104, 105, 106, 107, 108, 109, 120, 123, 124, 125, 126, 127, 128, 129, 130, 132, 134, 135, 136, 137, 138, 139, 140, 142, 143, 145, 146, 147, 148, 149, 150, 152, 153, 154, 156, 157, 158, 159, 160, 162, 163, 164, 165, 167, 168, 169, 170, 172, 173, 174, 175, 176, 178, 179, 180, 182, 183, 184,  185, 186, 187, 189, 190, 192, 193, 194, 195, 196, 197, 198, 201, 203, 204, 205, 206, 207, 208, 209, 210, 213, 214, 215, 216, 217, 218, 219, 230, 231, 234, 235, 236, 237, 238, 239, 240, 241, 243, 245, 246, 247 |

## Problem 13. \*Lazy Sundays

Write a PHP script **LazySundays.php** which prints the **dates** of all Sundays in the current month. Example:

|  |  |
| --- | --- |
| **Examples** | **Output** |
| August | 5rd 08, 2018  12rd 08, 2018  19rd 08, 2018  26rd 08, 2018 |
| December | 2rd 12, 2018  9rd 12, 2018  16rd 12, 2018  23rd 12, 2018  30rd 12, 2018 |
| October | 7rd 10, 2018  14rd 10, 2018  21rd 10, 2018  28rd 10, 2018 |

## Problem 14. \* \*Time Until New Year

Write a PHP script **TimeUntilNewYear.php.** Use the built-in function **getdate()** to get the current date and time. Print how many **hours, minutes and seconds** are left until New Year and how many **days, hours, minutes and seconds** are leftin a **counter format** . Look at examples below to get a better idea. The **examples** show the current date and time in **"d-m-Y G:i:s"** format. Use **the current time**. Check for date/time formats in PHP.   
(Note: Keep [the Spring/Autumn time shifts](http://en.wikipedia.org/wiki/Daylight_saving_time#Procedure) in mind in your calculations.)

|  |  |
| --- | --- |
| **Examples** | **Output** |
| 12-08-2014 13:07:09 | Hours until new year : 3394  Minutes until new year : 203692  Seconds until new year : 12221571  Days:Hours:Minutes:Seconds 141:10:52:51 |
| 12-08-2014 11:08:47 | Hours until new year : 3396  Minutes until new year : 203811  Seconds until new year : 12228673  Days:Hours:Minutes:Seconds 141:12:51:13 |