# More Exercises: Data Types and Variables

Additional exercises for the [Python Fundamentals Course@SoftUni](https://softuni.bg/trainings/4222/programming-fundamentals-with-python-september-2023).

Submit your solutions in the SoftUni judge system at <https://judge.softuni.org/Contests/1723>.

***Note: All the exercises are excluded from your homework!***

1. **Exchange Integers**

Read two integer numbers and, after that, **exchange their values**. Print the variable values before and after the exchange, as shown below:

**Examples**

|  |  |
| --- | --- |
| **Input** | **Output** |
| 5  10 | Before:  a = 5  b = 10  After:  a = 10  b = 5 |
| 10  20 | Before:  a = 10  b = 20  After:  a = 20  b = 10 |

**Hints**

You may use a **temporary variable** to remember the old value of **a**, then assign the value of **b** to **a**, then assign the value of the temporary variable to **b**.

## Prime Number Checker

Write a program to check if a number is **prime**. A prime number is a natural number greater than 1, not a product of two smaller natural numbers. For example, the only ways of writing 5 as a product, 1 × 5 or 5 × 1, involve 5 itself.

The **input** comes as an integer number.

The **output** should be **True** if the number is prime and **False** otherwise.

**Examples**

|  |  |
| --- | --- |
| **Input** | **Output** |
| 7 | True |
| 8 | False |
| 81 | False |

## Decrypting Messages

On the **first line**, you will receive a **key** (**integer**). On the **second line**, you will receive **n** – the number of **lines**, which will **follow**.On the next **n lines** – you will receive a **lower** and an **uppercase** letter per line.

You should **add the key** to **each of the characters** and append them to a **message**. In the end, **print the** **decrypted message**.

### Examples

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Input** | **Output** |  | **Input** | **Output** |
| **3**  **7**  P  l  c  q  R  k  f | SoftUni |  | **1**  **7**  C  d  b  q  x  o  s | Decrypt |

## Balanced Brackets

On the **first line**, you will receive **n** – the number of lines, which will follow. On the following **n** lines, you will receive **one** of the following:

* Opening bracket – "(",
* Closing bracket – ")" or
* **Random string**

Your task is to find out if the **brackets** are **balanced**. That means after every **opening** bracket should follow a **closing** one. Nested parentheses are **not valid**, and if, for example, **two** **consecutive opening brackets** exist, the expression should be marked as **unbalanced**. You should print "BALANCED" if the parentheses are balanced and "UNBALANCED" otherwise.

### Examples

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Input** | **Output** |  | **Input** | **Output** |
| **8**  (  5 + 10  )  \* 2 +  (  5  )  -12 | BALANCED |  | **6**  12 \*  )  10 + 2 -  (  5 + 10  ) | UNBALANCED |