# More Exercise: Data Types and Variables

Additional exercises for the [Python Fundamentals Course @SoftUni](https://softuni.bg/trainings/2442/python-fundamentals-september-2019). Submit your solutions in the SoftUni judge system at <https://judge.softuni.bg/Contests/1723>

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

## Biggest of 3 Numbers

Write a program that finds the **biggest of 3 numbers**.

The **input** comes as 3 integers.

The **output** is the biggest from the input numbers.

**Examples**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Input** | **Output** |  | **Input** | **Output** |  | **Input** | **Output** |  |
| -2  7  3 | 7 |  | 130  5  99 | 130 | 43  44  45 | 45 |

1. **Exchange Integers**

Read two integer numbers and after that **exchange their values** by using some programming logic. 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 |

**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 (only wholly divisible by itself and one).

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

The **output** should be **true** for prime number and **false** otherwise.

**Examples**

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

## Decrypting Messages

You will receive a **key** (**integer**) and **n** characters afterward. Add the key to each of the characters and append them to a **message**. At the end print the message, which you decrypted.

### Input

* On the **first line**, you will receive the **key**
* On the **second line**, you will receive **n** – the number of **lines**, which will **follow**
* On the next **n lines** – you will receive **lower** and **uppercase** characters from the **Latin** alphabet

### Output

Print the **decrypted message**.

### Constraints

* The **key** will be in the interval **[0…20]**
* **n** will be in the interval **[1…20]**
* The **characters** will always be **upper** or **lower**-case letters from the **English alphabet**
* You will receive **one** **letter** per **line**

### 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

You will receive **n** lines. On **those** **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 **closing** bracket should follow an **opening** one. Nested parentheses are **not valid**, and if **two** **consecutive opening brackets** exist, the expression should be marked as **unbalanced**.

### Input

* On the **first line**, you will receive **n** – the number of lines, which will follow
* On the next **n** lines, you will receive “(”, “)” or **another** string

### Output

You have to print “BALANCED”, if the parentheses are balanced and “UNBALANCED” otherwise.

### Constraints

* **n** will be in the interval **[1…20]**
* The length of the stings will be between **[1…100]** characters

### Examples

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