# Exercise: Nested Loops and Methods

Test your tasks in the Judge system: <https://judge.softuni.org/Contests/4422>

# Nested Loops

## Stupid Passwords

Write a program that:

* Reads **an integer number N** from the console
* **Generates all possible passwords** consisting of the following 3 parts:
  + The **first** part is an **even** number in the range [2…**N**]
  + The **second** digit is an **odd** number in the range [1…**N**]
  + The **third** is the **product** of the first two
* The **output** holds all possible passwords



### **Example**

|  |  |
| --- | --- |
| **Input** | **Output** |
| 11 | 212 236 2510 2714 2918 21122 414 4312 4520 4728 4936 41144 616 6318 6530 6742 6954 61166 818 8324 8540 8756 8972 81188 10110 10330 10550 10770 10990 1011110 |
| 5 | 212 236 2510 414 4312 4520 |
| 6 | 212 236 2510 414 4312 4520 616 6318 6530 |

## Magic Numbers

Write a program that:

* Reads **an integer number N** from the console
* Find all **3-digit** **magic numbers**

**Note:** A magic number is a three-digit number where the **product of its digits equals the value of N**. If you have a three-digit number abc, where a, b and c are its digits, it is considered a magic number of order N, if a \* b \* c = N.

### **Example**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Input** | **Output** |  | **Input** | **Output** |  | **Input** | **Output** |
| 1 | 111 | 3 | 113 131 311 | 7 | 117 171 711 |

## Unique PIN Codes

Write a program to **generate PIN codes** following certain rules:

* + Read **3 integer digits**: **max1**, **max2**, **max3** (each is an upper limit)
  + Generate unique 3-digit **PIN codes**, matching the following:
  + Each digit is **within the following range**:
    - First digit: **[1 … max1]**
    - Second digit: **[1 … max2]**
    - Third digit: **[1 … max3]**
  + The **first** and the **third** **digit** must be **even**
  + The **second** **digit** must be a **prime** **number** in the range [**2…7**]
  + Print the PIN codes, each on separate line

### **Example**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Input** | **Output** |  | **Input** | **Output** |  | **Input** | **Output** |
| 3  5  5 | 222  224  232  234  252  254 | 4  4  6 | 222  224  226  232  234  236  422  424  426  432  434  436 | 3  3  5 | 222  224  232  234 |

## Letter Combinations

Write a program to generate **3-letter combinations** under certain conditions:

* + Read a start letter **s**, end letter **e** and excluded letter **x**
  + **Generate all combinations of 3 letters** in the range [**s**…**e**], excluding **x**
  + Print all **combinations on the first line**
  + Print **combinations** **count on the second line**

### **Example**

|  |  |
| --- | --- |
| **Input** | **Output** |
| a  c  b | aaa aac aca acc caa cac cca ccc  8 |

|  |  |
| --- | --- |
| **Input** | **Output** |
| d  g  e | ddd ddf ddg dfd dff dfg dgd dgf dgg fdd fdf fdg ffd fff ffg fgd fgf fgg gdd gdf gdg gfd gff gfg ggd ggf ggg  27 |

## Happy Numbers

Write a program that:

* Reads **an integer number N** from the console
* Generate all **4-digit happy numbers {d1}{d2}{d3}{d4}** for given integer **N**:
  + A number is happy if **d1** + **d2** == **d3** + **d4** == **N**
  + Print all happy numbers

### **Example**

|  |  |
| --- | --- |
| **Input** | **Output** |
| 3 | 1203 1212 1221 1230 2103 2112 2121 2130 3003 3012 3021 3030 |
| 4 | 1304 1313 1322 1331 1340 2204 2213 2222 2231 2240 3104 3113 3122 3131 3140 4004 4013 4022 4031 4040 |

# Methods

## Vowels Count

Write a program that:

* Read a **text (string)** from the console
* Create a method that receives a **text**
* Find the **count of the vowels** contained in the text
* Print the **count of the vowels** in the text

### **Example**

|  |  |
| --- | --- |
| **Input** | **Output** |
| SoftUni | 3 |
| Cats | 1 |
| JS | 0 |

## Factorial Division

Write a program that:

* Read **two integers** numbers from the console **in range [1…10]**
* Calculate the [**factorial**](https://en.wikipedia.org/wiki/Factorial) of each number
* Dividethe **first calculated factorial** by the **second calculated factorial** (integer division)
* Print the result of the division

### **Example**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Input** | **Output** |  | **Input** | **Output** |
| 5  2 | 60 |  | 6  2 | 360 |

## Multiplication Sign

Write a program that:

* Reads **three integer numbers** (num1, num2 and num3) from the console
* Finds if num1 \* num2 \* num3 (the product) is **negative**, **positive or zero**
* Print:
  + **negative** - if the product is smaller than 0
  + **positive** - if the product is bigger than 0
  + **zero** - if the product is equals to 0

**Note**: Try to do this **WITHOUT** multiplying the three numbers.

### **Example**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Input** | **Output** |  | **Input** | **Output** |  | **Input** | **Output** |
| 2  3  -1 | negative | 2  3  1 | positive | 7  0  1 | zero |

## Password Validator

Write a program that checks if a given password is **valid**.

The password validation **rules** are:

* It should contain **6 – 10 characters (inclusive)**
* It should contain **only letters and digits**
* It should contain **at least 2 digits**

If it is **not valid**, for any unfulfilled rule **print the corresponding message**:

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

### **Exaзmple**

|  |  |
| --- | --- |
| **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 |