

## Exercises: C# Intro and Basic Syntax

Problems for exercises and homework for the [“Programming Fundamentals Extended” course @ SoftUni](#).

### Problem 1. Debit Card Number

Write a program, which receives **4 integers** on the console and **prints them in 4-digit debit card format**. See the examples below for the appropriate formatting.

#### Examples

Input	Output
12 433 1 5331	0012 0433 0001 5331
9182 4221 12 3	9182 4221 0012 0003
812 321 123 22	0812 0321 0123 0022

### Problem 2. Rectangle Area

Write a program, which calculates a **rectangle's area**, based on its **width** and **height**. The **width** and **height** come as floating point numbers on the console, **formatted to the 2<sup>nd</sup> character after the decimal point**.

#### Examples

Input	Output
2 7	14.00
7 8	56.00
12.33 5	61.65

### Problem 3. Miles to Kilometers

Write a program, which **converts miles to kilometers**. **Format** the output to the **2<sup>nd</sup> decimal place**.

Note: **1 mile == 1.60934 kilometers**

## Examples

Input	Output
60	96.56

Input	Output
1	1.61

Input	Output
52.1113	83.86

## Problem 4. Beverage Labels

Write a program, which reads a food product **name**, **volume**, **energy content per 100ml** and **sugar content per 100ml**. Calculate the **energy** and **sugar content** for the **given volume** and print them on the console in the following format:

- Name – as per the input
- Volume – **integer**, **suffix**ed by “ml” (e.g. “220ml”)
- Energy content – **integer**, **suffix**ed by “kcal” (e.g. “500kcal”)
- Sugar content – **real number**, **suffix**ed by “g” (e.g. “30g”)

## Examples

Input	Output
Nuka-Cola 220 300 70	220ml Nuka-Cola: 660kcal, 154g sugars

Input	Output
Ice Cold Nuka-Cola 250 350 65	250ml Ice Cold Nuka-Cola: 875kcal, 162.5g sugars

Input	Output
Nuka-Cola Quantum 350 600 140	350ml Nuka-Cola Quantum: 2100kcal, 490g sugars

## Problem 5. \* Character Stats

Write a program, which **displays information** about a video game character. You will receive their **name**, **current health**, **maximum health**, **current energy** and **maximum energy** on separate lines. The **current** values will **always** be valid (**equal or lower** than their respective **max** values). Print them in the format as per the examples.

## Examples

Input	Output
Mayro 5 10 9 10	Name: Mayro Health:      .....  Energy:      .....

Input	Output
Bauser 10 10 10 10	Name: Bauser Health:      .....  Energy:      .....

Input	Output
Loogi	Name: Loogi
8	Health:      .....
20	Energy:    .....
2	
14	

Input	Output
Toad	Name: Toad
0	Health:  .....
5	Energy:  .....
0	
10	

## Hints

- You can print a character **multiple** times, using `new string(character, count)`.