# Lab: Data Types and Variables

Problems for in-class lab for the ["C# Fundamentals" course @ SoftUni](https://softuni.bg/trainings/3836/programming-fundamentals-with-csharp-september-2022)  
You can check your solutions in [Judge](https://judge.softuni.org/Contests/1192/Data-Types-and-Variables-Lab)

# Integer and Real Numbers

## Convert Meters to Kilometers

You will be given an integer that will be a distance in meters. Create a program that converts meters to kilometers formatted to the second decimal point.

using System;

namespace \_01.\_Convert\_Meters\_to\_Kilometers

{

class Program

{

static void Main(string[] args)

{

int meters = int.Parse(Console.ReadLine());

float km = meters / 1000.0f;

Console.WriteLine($”{km:F2}”);

}

}

}

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 1852 | 1.85 |
| 798 | 0.80 |

## Pounds to Dollars

Create a program that converts British pounds to US dollars formatted to the 3rd decimal point.

1 British Pound = 1.31 Dollars

using System;

namespace \_02.\_Pounds\_to\_Dollars

{

class Program

{

static void Main(string[] args)

{

double pounds = double.Parse(Console.ReadLine());

double dollars = pounds \* 1.31;

Console.WriteLine($"{dollars:F3}");

}

}

}

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 80 | 104.800 |
| 39 | 51.090 |

## Exact Sum of Real Numbers

Create a program to enter n numbers and calculate and print their **exact sum** (without rounding).

using System;

namespace \_03.\_Exact\_Sum\_of\_Real\_Numbers

{

class Program

{

static void Main(string[] args)

{

int n = int.Parse(Console.ReadLine());

decimal sum = 0;

for (int i = 0; i < n; i++)

{

decimal inputNum = decimal.Parse(Console.ReadLine());

sum += inputNum;

}

Console.WriteLine(sum);

}

}

}

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 3  1000000000000000000  5  10 | 1000000000000000015 |
| 2  0.00000000003  333333333333.3 | 333333333333.30000000003 |

### Hints

Use Decimal to not lose precision.

# Data Types and Type Conversion

## Centuries to Minutes

Create a program to enter an integer number of **centuries** and convert it to **years**, **days**, **hours** and **minutes**.

using System;

namespace \_04.\_Centuries\_to\_Minutes

{

class Program

{

static void Main(string[] args)

{

int centuries = int.Parse(Console.ReadLine());

int years = centuries\*100;

int days = (int)(years \* 365.2422);

long hours = days \* 24;

long minutes = hours \* 60;

Console.WriteLine($"{centuries} centuries = {years} years = {days} days = {hours} hours = {minutes} minutes");

}

}

}

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 1 | 1 centuries = 100 years = 36524 days = 876576 hours = 52594560 minutes |
| 5 | 5 centuries = 500 years = 182621 days = 4382904 hours = 262974240 minutes |

### Hints

* Use appropriate data types to fit the result after each data conversion.
* Assume that a year has 365.2422 days on average ([the Tropical year](https://en.wikipedia.org/wiki/Tropical_year)).

## Special Numbers

A **number** is **special** when its **sum of digits is 5, 7 or 11**.

Write a program to read an integer n and for all numbers in the range **1…n** to print the number and if it is special or not (True / False).

using System;

namespace \_05.\_Special\_Numbers

{

class Program

{

static void Main(string[] args)

{

int number = int.Parse(Console.ReadLine());

int sum = 0;

for (int i = 1; i <=number; i++)

{

sum = 0;

int currentNum = i;

while (currentNum > 0)

{

sum+=currentNum% 10;

currentNum /= 10;

}

if(sum==5||sum==7||sum==11)

{ Console.WriteLine($"{i} -> True"); }

else

{ Console.WriteLine($"{i} -> False"); }

}

}

}

}

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 7 | 1 -> False  2 -> False  3 -> False  4 -> False  5 -> True  6 -> False  7 -> True |
| 15 | 1 -> False  2 -> False  3 -> False  4 -> False  5 -> True  6 -> False  7 -> True  8 -> False  9 -> False  10 -> False  11 -> False  12 -> False  13 -> False  14 -> True  15 -> False |

### Hints

To calculate the sum of digits of given number num, you might repeat the following: sum the last digit (num % 10) and remove it (num = num / 10) until num reaches 0.

## Reversed Chars

Create a program that takes 3 lines of characters and prints them in reversed order with a space between them.

using System;

namespace \_06.\_Reversed\_Chars

{

class Program

{

static void Main(string[] args)

{

char firstChar = char.Parse(Console.ReadLine());

char secondChar = char.Parse(Console.ReadLine());

char thirdChar = char.Parse(Console.ReadLine());

Console.WriteLine($"{thirdChar} {secondChar} {firstChar}");

}

}

}

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| A  B  C | C B A |
| 1  L  & | & L 1 |

## Concat Names

Read two names and a delimiter. Print the names joined by the delimiter.

using System;

namespace \_07.\_Concat\_Names

{

class Program

{

static void Main(string[] args)

{

string firstName = Console.ReadLine();

string lastName = Console.ReadLine();

string delimiter = Console.ReadLine();

string result = firstName + delimiter + lastName;

Console.WriteLine(result);

}

}

}

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| John  Smith  -> | John->Smith |
| Jan  White  <-> | Jan<->White |
| Linda  Terry  => | Linda=>Terry |

## Town Info

You will be given 3 lines of input. On the first line, you will be given the name of the town, on the second – the population and on the third the area. Use the correct data types and print the result in the following format:

"**Town {town name} has population of {population} and area {area} square km**".

using System;

namespace \_08.\_Town\_Info

{

class Program

{

static void Main(string[] args)

{

string townName = Console.ReadLine();

int population = int.Parse(Console.ReadLine());

int area = int.Parse(Console.ReadLine());

Console.WriteLine($"Town {townName} has population of {population} and area {area} square km.");

}

}

}

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| Sofia  1286383  492 | Town Sofia has population of 1286383 and area 492 square km. |
| Kaliningrad  437456  223 | Town Kaliningrad has population of 437456 and area 223 square km. |

## Chars to String

Create a program that reads 3 lines of input. On each line, you get a single character. Combine all the characters into one string and print it on the console.

using System;

namespace \_09.\_Chars\_to\_String

{

class Program

{

static void Main(string[] args)

{

char firstChar = char.Parse(Console.ReadLine());

char secondChar = char.Parse(Console.ReadLine());

char thirdChar = char.Parse(Console.ReadLine());

Console.WriteLine($"{firstChar}{secondChar}{thirdChar}");

}

}

}

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| a  b  c | abc |
| %  2  o | %2o |
| 1  5  p | 15p |

## Lower or Upper

Create a program that prints whether a given character is upper-case or lower case.

using System;

namespace \_10.\_Lower\_or\_Upper

{

class Program

{

static void Main(string[] args)

{

char letter = char.Parse(Console.ReadLine());

if ((int)letter>=65&& (int)letter<=90)

{ Console.WriteLine("upper-case"); }

else if ((int)letter >= 97&& (int)letter <= 122)

{ Console.WriteLine("lower-case"); }

}

}

}

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| L | upper-case |
| f | lower-case |

# Variables

## Refactor Volume of Pyramid

You are given a **working code** that finds the **volume of a pyramid**. However, you should consider that the variables exceed their optimum span and have improper naming. Also, search for variables that **have multiple purposes**.

using System;

namespace \_11.\_Refactor\_Volume\_of\_Pyramid

{

class Program

{

static void Main(string[] args)

{

Console.Write("Length: ");

double length = double.Parse(Console.ReadLine());

Console.Write("Width: ");

double width = double.Parse(Console.ReadLine());

Console.Write("Height: ");

double heigth = double.Parse(Console.ReadLine());

double V = ((length \* width) \*heigth) / 3;

Console.WriteLine($"Pyramid Volume: {V:f2}");

}

}

}

### Hints

* **Reduce the span** of the variables by declaring them at the moment they receive a value, not before
* Rename your variables to **represent their** real **purpose** (example: "dul" should become length, etc.)
* Search for variables that have multiple purposes. If you find any, **introduce a new variable**.

|  |
| --- |
| **Sample Code** |
| double dul, sh, V = 0;  Console.WriteLine("Length: ");  dul = double.Parse(Console.ReadLine());  Console.WriteLine("Width: ");  sh = double.Parse(Console.ReadLine());  Console.WriteLine("Heigth: ");  V = double.Parse(Console.ReadLine());  V = (dul + sh + V) / 3;  Console.WriteLine($"Pyramid Volume: {V:f2}"); |

## Refactor Special Numbers

You are given a **working code** that is a solution to **Problem 5**. **Special Numbers**. However, the variables are **improperly named, declared before** they are needed and some of them are used for multiple purposes. Without using your previous solution, **modify the code,** so that it is **easy to read and understand**.

using System;

namespace \_05.\_Special\_Numbers

{

class Program

{

static void Main(string[] args)

{

int number = int.Parse(Console.ReadLine());

int sum = 0;

for (int i = 1; i <=number; i++)

{

sum = 0;

int currentNum = i;

while (currentNum > 0)

{

sum+=currentNum% 10;

currentNum /= 10;

}

if(sum==5||sum==7||sum==11)

{ Console.WriteLine($"{i} -> True"); }

else

{ Console.WriteLine($"{i} -> False"); }

}

}

}

}

|  |
| --- |
| **Sample Code** |
| int kolkko = int.Parse(Console.ReadLine());  int obshto = 0;  int takova = 0;  bool toe = false;  for (int ch = 1; ch <= kolkko; ch++)  {  takova = ch;  while (ch > 0)  {  obshto += ch % 10;  ch = ch / 10;  }  toe = (obshto == 5) || (obshto == 7) || (obshto == 11);  Console.WriteLine("{0} -> {1}", takova, toe);  obshto = 0;  ch = takova;  } |

### Hints

* Reduce the span of the variables by declaring them at the moment they receive a value, not before.
* Rename your variables to represent their real purpose (example: "toe" should become isSpecialNum, etc.).
* Search for variables that have multiple purposes. If you find any, introduce a new variable.