# Lab: Data Types and Variables

Problems for exercises and homework for the ["Programming Fundamentals" course @ SoftUni.](https://softuni.bg/trainings/3951/programming-fundamentals-with-java-january-2023)

You can check your solutions in [Judge.](https://judge.softuni.org/Contests/1227)

# Integer and Real Numbers

## Convert Meters to Kilometers

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

### Examples

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

import java.util.Scanner;  
  
public class demo {  
 public static void main(String[] args) {  
 Scanner scan = new Scanner(System.*in*);  
  
 int meters = Integer.*parseInt*(scan.nextLine());  
  
 double convertToKilometers = (meters \* 1.0) / 1000;  
  
 System.*out*.printf("%.2f", convertToKilometers);  
 }  
}

## Pounds to Dollars

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

1 British Pound = 1.36 Dollars

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 80 | 108.800 |
| 39 | 53.040 |

import java.util.Scanner;  
  
public class demo {  
 public static void main(String[] args) {  
 Scanner scan = new Scanner(System.*in*);  
  
 double britishPound = Double.*parseDouble*(scan.nextLine());  
  
 double convertToUsd = britishPound \* 1.36;  
  
 System.*out*.printf("%.3f", convertToUsd);  
 }  
}

## Exact Sum of Real Numbers

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

### Examples

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

### Hints

Use BigDecimal not to lose precision.

import java.math.BigDecimal;  
import java.util.Scanner;  
  
public class demo {  
 public static void main(String[] args) {  
 Scanner scan = new Scanner(System.*in*);  
  
 int num = Integer.*parseInt*(scan.nextLine());  
  
 BigDecimal sum = new BigDecimal(0);  
  
 for (int i = 0; i < num; i++) {  
 BigDecimal currentNumber = new BigDecimal(scan.nextLine());  
  
 sum = sum.add(currentNumber);  
 }  
 System.*out*.println(sum);  
 }  
}

# Data Types and Type Conversion

## 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.**".

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| Sofia  1286383  492 | Town Sofia has population of 1286383 and area 492 square km. |
| Plovdiv  343424  101 | Town Plovdiv has population of 343424 and area 101 square km. |

import java.math.BigDecimal;  
import java.util.Scanner;  
  
public class demo {  
 public static void main(String[] args) {  
 Scanner scan = new Scanner(System.*in*);  
  
 String nameOfCity = scan.nextLine();  
 int population = Integer.*parseInt*(scan.nextLine());  
 int area = Integer.*parseInt*(scan.nextLine());  
  
 System.*out*.printf("Town %s has population of %d and area %d square km.",  
 nameOfCity,population,area);  
 }  
}

## Concat Names

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

### Examples

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

import java.math.BigDecimal;  
import java.util.Scanner;  
  
public class demo {  
 public static void main(String[] args) {  
 Scanner scan = new Scanner(System.*in*);  
  
 String firstName = scan.nextLine();  
 String secondName = scan.nextLine();  
 String delimiter = scan.nextLine();  
  
 System.*out*.println(firstName + delimiter + secondName);  
 }  
}

## Chars to String

Write 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.

### Examples

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

import java.math.BigDecimal;  
import java.util.Scanner;  
  
public class demo {  
 public static void main(String[] args) {  
 Scanner scan = new Scanner(System.*in*);  
  
 char firstSymbol = scan.nextLine().charAt(0);  
 char secondSymbol = scan.nextLine().charAt(0);  
 char thirdSymbol = scan.nextLine().charAt(0);  
  
 System.*out*.printf("%s%s%s", firstSymbol, secondSymbol, thirdSymbol);  
 }  
}

## Reversed Chars

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

### Examples

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

import java.math.BigDecimal;  
import java.util.Scanner;  
  
public class demo {  
 public static void main(String[] args) {  
 Scanner scan = new Scanner(System.*in*);  
  
 char firstSymbol = scan.nextLine().charAt(0);  
 char secondSymbol = scan.nextLine().charAt(0);  
 char thirdSymbol = scan.nextLine().charAt(0);  
  
 System.*out*.printf("%c %c %c", thirdSymbol, secondSymbol, firstSymbol);  
 }  
}

## Lower or Upper

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

### Examples

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

import java.math.BigDecimal;  
import java.util.Scanner;  
  
public class demo {  
 public static void main(String[] args) {  
 Scanner scan = new Scanner(System.*in*);  
  
 char symbol = scan.nextLine().charAt(0);  
  
 if (Character.*isUpperCase*(symbol)){  
 System.*out*.println("upper-case");  
 }else if (Character.*isLowerCase*(symbol)){  
 System.*out*.println("lower-case");  
 }  
  
 /\* if (symbol >= 'A' && symbol <= 'Z'){  
 System.out.println("upper-case");  
 }else if (symbol >= 'a' && symbol <= 'z'){  
 System.out.println("lower-case");  
 }\*/  
 }  
}

## Centuries to Minutes

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

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 1 | 1 centuries = 100 years = 36524 days = 876581 hours = 52594877 minutes |
| 5 | 5 centuries = 500 years = 182621 days = 4382906 hours = 262974384 minutes |

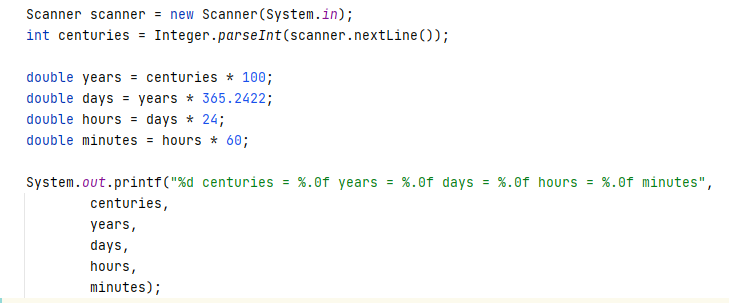
### Hints

* 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)).

### Solution

You might help yourself with the code below:

import java.math.BigDecimal;  
import java.util.Scanner;  
  
public class demo {  
 public static void main(String[] args) {  
 Scanner scan = new Scanner(System.*in*);  
  
 byte centuries = Byte.*parseByte*(scan.nextLine());  
  
 int years = centuries \* 100;  
 double days = years \* 365.2422;  
 double hours = days \* 24;  
 double minutes = hours \* 60;  
  
 System.*out*.printf("%d centuries = %d years = %.0f days = %.0f hours = %.0f minutes",  
 centuries, years, Math.*abs*(days), Math.*abs*(hours), Math.*abs*(minutes));  
  
 }  
}



## 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).

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 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 |
| 9 | 1 -> False  2 -> False  3 -> False  4 -> False  5 -> True  6 -> False  7 -> True  8 -> False  9 -> 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 (sum = sum / 10) until the num reaches 0.

import java.math.BigDecimal;  
import java.util.Scanner;  
  
public class demo {  
 public static void main(String[] args) {  
 Scanner scan = new Scanner(System.*in*);  
  
 int number = Integer.*parseInt*(scan.nextLine());  
  
 for (int i = 1; i <= number; i++) {  
 int currentNumber = i;  
  
 int sum = 0;  
  
 while (currentNumber > 0){  
 sum = sum + (currentNumber % 10);  
 currentNumber = currentNumber / 10;  
 }  
  
 boolean isSpecialNumber = sum == 5 || sum == 7 || sum == 11;  
  
 if (isSpecialNumber){  
 System.*out*.printf("%d -> True%n", i);  
 }else {  
 System.*out*.printf("%d -> False%n", i);  
 }  
  
 }  
  
 }  
}

# 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**.

### Code

|  |
| --- |
| **Sample Code** |
| Scanner scanner = **new** Scanner(System.***in***);  **double** dul, sh, V = 0; System.***out***.print(**"Length: "**); dul = Double.*parseDouble*(scanner.nextLine()); System.***out***.print(**"Width: "**); sh = Double.*parseDouble*(scanner.nextLine()); System.***out***.print(**"Height: "**); V = Double.*parseDouble*(scanner.nextLine()); V = (dul \* sh \* V) / 3; System.***out***.printf(**"Pyramid Volume: %.2f"**, V); |

### 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**.

import java.math.BigDecimal;  
import java.util.Scanner;  
  
public class demo {  
 public static void main(String[] args) {  
 Scanner scan = new Scanner(System.*in*);  
  
 double length,width ,height = 0;  
  
 System.*out*.print("Length: ");  
 length = Double.*parseDouble*(scan.nextLine());  
 System.*out*.print("Width: ");  
 width = Double.*parseDouble*(scan.nextLine());  
 System.*out*.print("Height: ");  
 height = Double.*parseDouble*(scan.nextLine());  
 height = (length \* width \* height) / 3;  
 System.*out*.printf("Pyramid Volume: %.2f", height);  
  
 }  
}

## Refactor Special Numbers

You are given a **working code** that is a solution to **Problem 9. Special Numbers**. However, the variables are **improperly named, declared before** they are needed, and some of them are used for multiple things. Without using your previous solution, **modify the code** so that it is **easy to read and understand**.

### Code

|  |
| --- |
| **Sample Code** |
| Scanner scanner = **new** Scanner(System.***in***);  **int** kolkko = Integer.*parseInt*(scanner.nextLine()); **int** obshto = 0; **int** takova = 0; **boolean** 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);  System.***out***.printf(**"%d -> %b%n"**, 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

import java.math.BigDecimal;  
import java.util.Scanner;  
  
public class demo {  
 public static void main(String[] args) {  
 Scanner scan = new Scanner(System.*in*);  
  
 int number = Integer.*parseInt*(scan.nextLine());  
  
 for (int i = 1; i <= number; i++) {  
 int currentNumber = i;  
  
 int sum = 0;  
  
 while (currentNumber > 0) {  
  
 sum += currentNumber % 10;  
 currentNumber = currentNumber / 10;  
 }  
  
 boolean isSpecialNum = sum == 5 || sum == 7 || sum == 11;  
 if (isSpecialNum){  
 System.*out*.printf("%d -> True%n", i);  
 }else {  
 System.*out*.printf("%d -> False%n", i);  
 }  
 }  
  
  
 }  
}