Practical Exercise 3 – Integers and reals

This prac is an introduction to working with numbers in Java and, in particular, the differences between working with numbers as integers (i.e. *whole numbers*, including negative whole numbers and zero) and reals (including *rational* and *irrational* numbers).

# Variables

Variables are spaces in computer memory that store values such as numbers and characters.

To use a variable in Java, you first have to *declare* it. When you declare a variable, you specify what type of value it is (e.g. integer or real) and give it a name. If the variable is one of the eight primitive variables in Java (i.e. *int, long, byte, short, float, double, boolean, char*), space is set aside for the variable immediately. This is called *instantiation*.

When dealing with numerical values, we commonly use *int* or *double*.

* int – typically a 32-bit signed two’s complement integer. Other integer types are byte (8-bit), short (16-bit) and long (64-bit).
* double – 64-bit floating point. Another real type is float (32-bit) that can be used to save memory, but double is usually the default option.

To declare (and instantiate) an int or double variable you use a line such as:

int n; //for one integer variable

int n, m, sum; //for a number of different integer variables.

double x, y, total; // for a number of different double variables

Variable names may contain letters and digits, and (by convention) start with a lower case letter, and are *case-sensitive*. E.g. *total* and *TOTAL* refer to different variables in Java.

A global declaration (immediately after the main class definition) means that all the methods of the class can use these variables and their values.

A local declaration (within a particular method) means that the variable is only available inside that method.

# Basic integer and real operations

## Assignment

int n = 4; // Note “=” means "becomes equal to”  
 double x = 3.5;

## Sum

sum = 4 + 7;

sum = n + 4;

## Product

product = n \* 4;

## Integer division

quotient = n / 4; // eg. 7/4 => 1; 13/4 => 3

## Integer remainder or “modulus” (the operator, %, is the “modulo” operator, often abbreviated to “mod”)

remainder = n % 4; // the remainder after division eg. 7 % 4 = 3; 13 % 4 = 1

## Real division

result = 5.6 / 4 // the result is 1.4

# Writing integer or real values to the applet window

To display the value of a variable in an applet, use a line such as the following in the paint() method:

g.drawString("The sum is " **+** sum, 25, 50);

Note that the drawString() method displays a text string in the applet. To display the value of a numerical variable you must convert it to a string. The easiest way is to “concatenate” it to a string using the + operator as has been done in the line above.

# Tasks

*This prac is complete when you’ve done the first three tasks.*

*Include your code and a screenshot for each task.*

## Task 1 – Fix the sample code

1. Create a new Java Project called ***Prac 3 – Integers and reals***.
2. Copy the sample code a new class called ***IntegersAndReals\_Task1***.
3. Run the code. (Spoiler alert! – you will need to fix some mistakes with it first.)
4. Carefully examine the results. (Hint: Some of the calculations aren’t quite right.)
5. Fix the code.

**import** java.awt.\*;

**import** javax.swing.\*;

**class** MyCanvas1 **extends** JPanel

{

// global declarations of double (real) and int (integer) variables

**double** x, y, sum, difference, product, quotient;

**int** i, j, int\_sum, int\_diff, int\_product, int\_quotient, int\_remainder;

**public** **void** init()

{

// Set the values that we'll work with for this task.

x = 25.0;

y = 6.0;

// Perform some calculations

sum = x + y;

product = x \* y;

quotient = x / y;

// Repeat the calculations, with our two values cast to integers.

i = (**int**) x;

j = x;

int\_sum = i + j;

int\_difference = i - j;

int\_product = i \* j;

int\_quotient = i / j;

int\_remainder = i % j;

}

**public** **void** paint (Graphics g)

{

// Display the results

g.drawString("As reals, the values are x: " + x + " and y: " + y, 20, 25);

g.drawString("The sum, x + y = " + sum, 20, 50);

g.drawString("The difference, x - y = " + difference, 20, 75);

g.drawString("The product, x \* y = " + product, 20, 100);

g.drawString("The quotient, x / y = " + int\_quotient, 20, 125);

g.drawString("As integers, these are i: " + i + " and j: " + j, 350, 25);

g.drawString("The sum, i + j = " + int\_sum, 350, 50);

g.drawString("The difference, i - j = " + int\_difference, 350, 75);

g.drawString("The product, i \* j = " + int\_product, 350, 100);

g.drawString("The quotient, i / j = " + int\_quotient, 350, 125);

g.drawString("The remainder, i % j = " + int\_remainder, 350, 150);

}

}

**public** **class** IntegersAndReals\_Task1 {

**public** **static** **void** main(String[] a) {

MyCanvas1 myCanvas = **new** MyCanvas1();

myCanvas.init();

JFrame window = **new** JFrame();

window.setDefaultCloseOperation(JFrame.***EXIT\_ON\_CLOSE***);

window.setBounds(30, 30, 600, 300);

window.getContentPane().add(myCanvas);

window.setVisible(**true**);

}

}

## Task 2 – Add some more calculations

1. Copy your code into a new class called ***IntegersAndReals\_Task2***.
2. Modify the code to add the following calculations. Stick with the same layout – i.e. real arithmetic on the left, and integer arithmetic on the right.

Real arithmetic:

|  |  |  |
| --- | --- | --- |
| (x + y) / (x - y) | x + x / y | x / (y + x) |
| (x + y) \* (x – y) | x / y + x |  |

Integer arithmetic:

|  |  |  |
| --- | --- | --- |
| (i + j) / (i - j) | i + i / j | i / (i + j) |
| (i + j) \* (i – j) | i / j + i |  |

## Task 3 – User input

1. Modify the code to add two text fields, so that the user can set the values of x and y, and the calculations will update automatically.
2. (Hint: You don't have to start from scratch. You are welcome – and encouraged! – to copy code in from Prac 2.)
3. To extract the numeric value (as a real) from a text field called ***inputX***, and store in double variable ***x*** you will need something like this code…

x = Double.parseDouble(inputX.getText());

1. Try out some different pairs of values to get a feel for the differences between real arithmetic and integer arithmetic in Java.

## Task 4 – Be precise (stretch goal – bonus marks)

1. You may notice when working with double variables, that some of the results display with a ridiculous number of digits.  
     
   To avoid this problem, you can use *precision formatting* to round the values to a specified number of decimal places.  
     
   To do this, you need to import this library:

import java.text.DecimalFormat;

Create an object of the type DecimalFormat:

DecimalFormat precision2;

Specify the precision that you wish to use. Put this line in the init() method.

precision2 = new DecimalFormat("#0.00");

Now use this precision in a g.drawString line of code in the following way:

g.drawString("x \* y = " + precision2.format(product) 10, 350);

where mult is the variable to be rounded correct to 2 decimal places (in this case)

1. Add extra g.drawString() lines to the bottom of the real arithmetic section to:
2. Display the results of x \* y and x / y rounded to **two decimal places**.
3. Display the results of x \* y and x / y rounded to **four decimal places**.