Lab Exercises for Week 6

Just two lab exercises this week to give you some time to catch up on exercises from the previous weeks.

1. Implement a very simple calculator that interacts with the user to get two integer values and then repeatedly offers the options of addition, subtraction, multiplication, integer division (including remainder on integer division), floating point division or quit. The program should offer a simple text-based interface where each option is identified by a number e.g.

   1: add
   2: subtract
   3: multiply
   4: integer division
   5: floating point division
   6: quit

   The program should terminate when the quit option is selected. e.g. if the two values input are 14 and 5 and then the add option is chosen, then the result displayed should be:

   add: 14.0 + 5.0 = 19.0

   If this is followed by choosing the subtract option, the result displayed should be:

   subtract: 14.0 – 5.0 = 9.0

   Hints:

   - Use **TextIO.getlnInt()** to read the menu choices and when reading the numbers for integer division
   - Use **TextIO.getlnDouble()** to read all the other numbers
   - use a `do while` loop which exits when the user chooses the quit option
   - use a `switch` statement within the `do while` loop with a `case` for each possible option that the user can input

   [We will update this program at a later stage to use methods to carry out the calculator functions and display the results.]

2. Write a program that repeatedly prompts for and reads in a line of text that consists of one or more words until the user enters 'no'. The program should go through the string a character and break it up into words. The words should be output one per line. A word is defined to be a sequence of letters. Any characters in the input that are not letters should be discarded. For example, if the user inputs the line:

   He said, "That's not a good idea."

then the output of the program should be

```
He
said
that
s
not
a
good
idea
```

An improved version of the program would list “that’s” as a single word. An apostrophe can be considered to be part of a word if there is a letter on each side of the apostrophe. To test whether a character is a letter, you might use *(ch >= ’a’ && ch <= ’z’) || (ch >= ’A’ && ch <= ’Z’)*. However, this only works in English and similar languages. A better choice is to call the standard function *Character.isLetter(ch)*, which returns a *boolean* value of *true* if *ch* is a letter and *false* if it is not. This works for any Unicode character.