Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Lecturer: Prof. B. Gärtner

Informatik für Mathematiker und Physiker HS16

Exercise Sheet 5

Submission deadline: 15:15 - Tuesday 25th October, 2016
Course URL: http://lec.inf.ethz.ch/ifmp/2016/

Assignment 1 – Floating Point Quiz (5 points) [similar: Exam Summer 2016 (5.), Exam Spring 2015 (3.)]

a) Is the *decimal* number 1.5 an element of $F^*(2,3,-1,2)$?

b) What is the *decimal* value of the largest number in $F^*(2,3,-1,2)$?

c) How many numbers does $F^*(2,3,-1,2)$ represent?

d) Assuming IEEE standard 754, what is the value of a, and why?

```
unsigned int a = 1.5f;
```

e) State the type and value of the following expression:

```
4.0 * 5 / 8
```

Assignment 2 – Floating Point Computations (4 points) [similar: Exam Summer 2016 (5.c)]

a) Compute the exact binary value of the decimal value 0.3

b) Compute the exact binary value of the decimal value 11.7

c) Compute the following expressions as the parentheses suggest, representing each intermediate result (and the final result) in the normalized floating point number system $F^*(2,3,-3,3)$ according to the rules of computing with floating point numbers. If necessary, use the following rounding mode: if there is a 1 directly behind the last significant digit, round up, and if there is a 0, round down.[1]

$(0.25 + 0.25) + 4$

	decimal		binary
	0.25		
+	0.25		
=			
+	4		
=		←	

$(4 + 0.25) + 0.25$

	decimal		binary
	4		
+	0.25		
=			
+	0.25		
=		←	

[1] Example for the rounding mode: in F^*, the binary represented number $1.01\underline{0}..$ is rounded down to 1.01, while $1.01\underline{1}..$ is rounded up to 1.10

Assignment 3 – Points on Line? (4 points)

Write a C++ program point_on_line.cpp that determines if a point (x, y) is on a line g or not. The line g is defined as: $g(x) = 2.1 \cdot x + 0.5$. Use floating point variables of type double to do the calculation. Furthermore, since the computer might have to round the floating point numbers, you can not just compare them for equality using ==. Therefore, when you want to compare two floating point numbers, treat them as equal as soon as their difference[2] is less than 0.000001. Furthermore, the first input to your program is the x-coordinate of the point and the second input is the y-coordinate of the point.

I/O-Examples (Explanation: http://lec.inf.ethz.ch/ifmp/2016/codeboard.html)

```
0.25
3.91
Point is not on line
```

```
-110
-230.5
Point is on line
```

Submission: https://codeboard.ethz.ch/ifmp16E5T3

Assignment 4 – Rounding (4 points)

[Skript-Aufgabe 90]

a) Implement the following function. You may assume that the type double complies with the IEEE standard 754. The function is only required to work correctly, if the nearest integer is in the value range of the type int. For example, the number 3.25 is rounded to 3, the number 3.75 is rounded to 4, the number 3.5 is rounded to 4, and the number -3.5 is rounded to -4.

```
// PRE: x is roundable to a number in the value range of type int and
//      the truncated number is representable as a double again
// POST: return value is the integer nearest to x, or the one further
//       away from 0 if x lies right in between two integers.
int round (double x);
```

b) Write a program round.cpp which inputs a number of type double from the user, then rounds this number using your function from a), and then outputs the rounded number.

I/O-Examples (Explanation: http://lec.inf.ethz.ch/ifmp/2016/codeboard.html)

[2] Remark: This leads to an absolute tolerance. When the numbers were large, this approach would no longer be suitable. But for simplicity, this exercise can be solved with an absolute tolerance.

```
2.7
The rounded number is 3
```

Submission: https://codeboard.ethz.ch/ifmp16E5T4

Challenge - Finitely-Representable Numbers (8 points)

[Skript-Aufgabe 76]

Are you asking yourself why we are limiting our considerations to base-2 representations while you already have an intuition for the general case? If so, feel free to try this week's challenge, exercise 76 from the script!