Gruppe 1
Freitag 13:15-14:45, B-Woche
(Dr. B. Schulze)

Claudia Heinich, 8966108
Sven Pohl, 8968075
Arne Brutschy, 8964813

# Numerik

## Serie 3

5. a)

$$\begin{aligned} y &= fl\big(fl(0.234 * fl(0.156 + 0.833)) - fl(fl(0.234 * 0.156) + fl(0.234 * 0.833))\big) \\ &= fl\big(fl(0.234 * 0.989) - fl(0.037 + 0.195)\big) \\ &= fl\big(0.231 - 0.232\big) \\ &= -0.001 \end{aligned}$$

Der reale Wert wäre $y = 0$.

b)

$$\begin{aligned} y &= fl\big(1 - fl(0.601 * fl(\frac{1}{0.601}))\big) \\ &= fl\big(1 - fl(0.601 * 1.664)\big) \\ &= fl\big(1 - 1.000\big) \\ &= 0 \end{aligned}$$

Dies ist der richtige Wert.

c)

$$\begin{aligned} y &= fl\big(|0.388| - fl(\sqrt{fl(0.388^2)})\big) \\ &= fl\big(|0.388| - fl(\sqrt{0.151})\big) \\ &= fl\big(0.388 - 0.389\big) \\ &= -0.001 \end{aligned}$$

Der reale Wert wäre wiederum $y = 0$.

6. a) Man kann deutlich sehen das es sich bei der benutzten Architektur um einen PC handelt. Hier werden 53 Bits für die Mantisse verwendet, wie im IEEE-Standard 754 für `double` Zahlen gefordert.

   b) Beim Optimieren schaltet der Compiler von `double` Arithmetik auf auf `long double` um, da der mathematische Coprozessor `x87` intern ausschliesslich mit `long double` rechnet. Das kann man auch an den 64 Bits der Mantisse sehen, sie weisen deutlich auf das extended double Format der `x86` Architektur hin.

Claudia Heinich, 8966108
Sven Pohl, 8968075
Arne Brutschy, 8964813

c) Wir nutzen `volatile` um das Optimieren der `double` Variablen zu verhindern:

```
#include <stdio.h>

int main() {
#include <stdio.h>

int main() {
  volatile double umach, sum;
  int             length;

  umach= 1.0;
  length= 0;
  sum= 2.0;

  while (sum > 1.0) {
    umach= 0.5 * umach;
    length+= 1;
    sum= 1.0 + umach;
  }

  umach= 2.0 * umach;
  printf("LENGTH=%2d UMACH=%22.15e\n", length, umach);

  return 0;
}
```