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

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

# Numerik

## Serie 1

1. a) Ausrechnen der ersten Formal mittels Taschenrechner:

| | |
|---|---|
| $10^1$ | 1,99999847691328769880290124792 57 |
| $10^2$ | 1,99999998476912904932780850903 58 |
| $10^3$ | 1,99999999984769129011051202417 73 |
| $10^4$ | 1,99999999999847691290106684751 66 |
| $10^5$ | 1,99999999999998476912901067041 63 |
| $10^6$ | 1,99999999999999984769129010644 01 |
| $10^7$ | 1,99999999999999999847707392868 44 |
| $10^8$ | 1,99999999999999999911410935551 00 |
| $10^9$ | 1,99999999999999999999994489994 11 |
| $10^{10}$ | 1,99999999999999998045473526370 04 |
| $10^{11}$ | 1,99999999999999999999300995082 39 |
| $10^{12}$ | 1,99999999997166323811398111552 7 |
| $10^{13}$ | 1,99999999852694857151878242558 8 |
| $10^{14}$ | 1,99999999997169335712069389279 7 |
| $10^{15}$ | 1,99999999966695142339900782474 3 |
| $10^{16}$ | 1,99842168574089841611683526307 72 |
| $10^{17}$ | 1,99884925816104586748339153758 67 |
| $10^{18}$ | 1,99473178535448897479519111502 01 |
| $10^{19}$ | Division durch 0 |

Man kann deutlich erkennen wie das Ergebnis erst der 2 nähert, dann wieder kleiner wird. Am Ende produziert der Taschenrechner durch die interne Rundung eine Division durch Null.

b) Wir vereinfachen die Formel:

$$\begin{aligned} f(x) &= \frac{\sin^2 x}{1-\cos x} \qquad \left\| \cdot (1+\cos x) \right. \\ &= \frac{\sin^2 x(1+\cos x)}{(1-\cos^2} \\ &= 1+\cos x \end{aligned}$$

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

Vergleich der alten und der neuen Formel:

| i | Alte Formel | Neue Formel | Differenz |
|---|---|---|---|
| 0 | 2.0000000000000000 | 2.0000000000000000 | 0.0000000000000000 |
| 1 | 1.8775825618903734 | 1.8775825618903728 | -0.0000000000000007 |
| 2 | 1.9689124217106415 | 1.9689124217106446 | 0.0000000000000031 |
| 3 | 1.9921976672293171 | 1.9921976672293291 | 0.0000000000000120 |
| 4 | 1.9980475107000650 | 1.9980475107000992 | 0.0000000000000342 |
| 5 | 1.9995117584852764 | 1.9995117584851365 | -0.0000000000001399 |
| 6 | 1.9998779321704798 | 1.9998779321710067 | 0.0000000000005269 |
| 7 | 1.9999694825735379 | 1.9999694825770951 | 0.0000000000035572 |
| 8 | 1.9999923706067624 | 1.9999923706151699 | 0.0000000000084075 |
| 9 | 1.9999980926132492 | 1.9999980926519734 | 0.0000000000387241 |
| 10 | 1.9999995230076641 | 1.9999995231628795 | 0.0000000001552154 |
| 11 | 1.9999998801698313 | 1.9999998807907127 | 0.0000000006208813 |
| 12 | 1.9999999677141509 | 1.9999999701976776 | 0.0000000024835267 |
| 13 | 1.9999999900658927 | 1.9999999925494194 | 0.0000000024835267 |
| 14 | 1.9999999975164733 | 1.9999999981373549 | 0.0000000006208816 |
| 15 | 1.9999999993791184 | 1.9999999995343387 | 0.0000000001552203 |
| 16 | 1.9999999998447797 | 1.9999999998835847 | 0.0000000000388050 |
| 17 | 1.9999999999611950 | 1.9999999999708962 | 0.0000000000097011 |
| 18 | 1.9999999999902989 | 1.9999999999927240 | 0.0000000000024252 |
| 19 | 1.9999999999975748 | 1.9999999999981810 | 0.0000000000006062 |
| 20 | 1.9999999999993938 | 1.9999999999995453 | 0.0000000000001514 |
| 21 | 1.9999999999998486 | 1.9999999999998863 | 0.0000000000000377 |
| 22 | 1.9999999999999623 | 1.9999999999999716 | 0.0000000000000093 |
| 23 | 1.9999999999999907 | 1.9999999999999929 | 0.0000000000000022 |
| 24 | 1.9999999999999978 | 1.9999999999999982 | 0.0000000000000004 |
| 25 | 1.9999999999999996 | 1.9999999999999996 | 0.0000000000000000 |
| 26 | 2.0000000000000000 | 2.0000000000000000 | 0.0000000000000000 |
| 27 | 2.0000000000000000 | 2.0000000000000000 | 0.0000000000000000 |
| 28 | 2.0000000000000000 | 2.0000000000000000 | 0.0000000000000000 |
| 29 | 2.0000000000000000 | 2.0000000000000000 | 0.0000000000000000 |
| 30 | 2.0000000000000000 | 2.0000000000000000 | 0.0000000000000000 |

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

Der zugehörige C-Quellcode:

```
#include <stdio.h>
#include <math.h>


int main()
{
  double ya, yn, x, diff;
  int i=0;

  printf(" i   Alte Formel          Neue Formel          Differenz\n");

  for(i=0; i<=30; i++)
  {
    if (i == 0) {
      ya= 2;
      yn= 2;
      diff= 0;
    } else {
      x= pow(0.5,i);
      ya= (sin(x) * sin(x)) / (1 - cos(x));
      yn= 1 + cos(x);
      diff= yn-ya;
    }

    printf("%2d & %.16f & %.16f & %.16f \\\\\n", i, ya, yn, diff);
  }

  return 0;
}
```

2. a) Sei $0 \leq t \leq 0$, $k$ geht gegen $+\infty$. Dann strebt $t^k$ gegen 0. Da $e^t$ von $k$ unabhängig ist, wird das Integral mit wachsendem $k$ immer kleiner.
Somit gilt $I_k > I_k + 1 > 0$, $k \in \mathbb{N}$.

b) $I_k$ genügt der Rekursion, da
$I_1 = \int_0^1 te^t\,dt = te^t - \int_0^1 e^t\,dt$
$I_2 = t^2e^t - \int_0^1 2te^t\,dt$
$I_3 = t^3e^t - \int_0^1 3t^2e^t\,dt = t^3e^t - 3t^3e^t + \int_0^1 6te^t\,dt$
.
.

Man kann zeigen, das $I_k + 1$ jeweils durch die Multiplikation von i $I_k$ mit $e - (k + 1)$ gebildet werden kann.

c) Der Rekursionsalortihmus in C implementiert:

```
#include <stdio.h>
#include <math.h>

#define E       2.71828182845904523536

double rec_vorw(int k) {
  if (k == 0)
    return (E - 1);
  else
    return (E - (k * rec_vorw(k-1)));
}

double rec_rueck(int k, int stop, double Ik) {
  if (k == stop)
    return Ik;
  else
    return (rec_rueck(k-1, stop, ((Ik - E) / -k)));
}

int main()
{
  int i;

  printf(" k  Vorwaerts           Rueckwaerts\n");
  for(i=0; i<=31; i++) {
    printf("%2d   %.16f  %.16f\n", i, rec_vorw(i), rec_rueck(63, i, 0), i);
  }

  return 0;
}
```

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

... und seine Ergebnisse:

| k | "Vorwärts" | "Rückwärts" |
|---|---|---|
| 0 | 1.7182818284590451 | 1.7182818284590451 |
| 1 | 1.0000000000000000 | 1.0000000000000000 |
| 2 | 0.7182818284590451 | 0.7182818284590452 |
| 3 | 0.5634363430819098 | 0.5634363430819095 |
| 4 | 0.4645364561314058 | 0.4645364561314071 |
| 5 | 0.3955995478020160 | 0.3955995478020096 |
| 6 | 0.3446845416469491 | 0.3446845416469874 |
| 7 | 0.3054900369304017 | 0.3054900369301337 |
| 8 | 0.2743615330158318 | 0.2743615330179761 |
| 9 | 0.2490280313165592 | 0.2490280312972603 |
| 10 | 0.2280015152934536 | 0.2280015154864418 |
| 11 | 0.2102651602310557 | 0.2102651581081854 |
| 12 | 0.1950999056863769 | 0.1950999311608206 |
| 13 | 0.1819830545361452 | 0.1819827233683770 |
| 14 | 0.1705190649530128 | 0.1705237013017674 |
| 15 | 0.1604958541638526 | 0.1604263089325340 |
| 16 | 0.1503481618374036 | 0.1514608855385012 |
| 17 | 0.1623630772231834 | 0.1434467743045252 |
| 18 | -0.2042535615582568 | 0.1362398909775906 |
| 19 | 6.5990994980659243 | 0.1297238998848237 |
| 20 | -129.2637081328594491 | 0.1238038307625699 |
| 21 | 2717.2561526185072580 | 0.1184013824450763 |
| 22 | -59776.9170757786996546 | 0.1134514146673663 |
| 23 | 1374871.8110247384756804 | 0.1088992911096191 |
| 24 | -32996920.7463118955492973 | 0.1046988418281864 |
| 25 | 824923021.3760792016983032 | 0.1008107827543861 |
| 26 | -21447998553.0597763061523438 | 0.0972014768450063 |
| 27 | 579095960935.3322753906250000 | 0.0938419536438755 |
| 28 | -16214686906186.5859375000000000 | 0.0907071264305313 |
| 29 | 470225920279413.6875000000000000 | 0.0877751619736370 |
| 30 | -14106777608382408.0000000000000000 | 0.0850269692499366 |
| 31 | 437310105859854656.0000000000000000 | 0.0824457817110112 |

Man sieht deutlich, das es bei dem ersten Algorithmus zu einem Fehler ab $k > 18$ kommt. Da der Fehler so stark ausfällt, vermuten wir einen Fehler durch Überlauf.