Prozedurale Programmierung – Übung 6

WS 2022/23 *Prof. Dr. Johannes Jurgovsky* Hochschule **Rosenheim** University of Applied Sciences

In der Community wird ein (fast leeres) Projekt in der Datei „loops.zip“ bereitgestellt.

Aufgabe 1

Entwickeln Sie ein C-Programm, das durch folgenden Bildschirmaufbau definiert wird:

```
------------------------------------------------
Exp: Berechnung von einfachen Funktionen
------------------------------------------------
Bitte positive obere Grenze eingeben (ganzzahlig <= 20):  10
-------------------------------------------------------------------------------
   i    1/i Summe(1/i)                  i!           1/i!  Naeherung e
-------------------------------------------------------------------------------
   1 1.0000  1.0000000                   1 1.00000000000000 2.000000000000
   2 0.5000  1.5000000                   2 0.50000000000000 2.500000000000
   3 0.3333  1.8333333                   6 0.16666666666667 2.666666666667
   4 0.2500  2.0833333                  24 0.04166666666667 2.708333333333
   5 0.2000  2.2833333                 120 0.00833333333333 2.716666666667
   6 0.1667  2.4500000                 720 0.00138888888889 2.718055555556
   7 0.1429  2.5928571                5040 0.00019841269841 2.718253968254
   8 0.1250  2.7178571               40320 0.00002480158730 2.718278769841
   9 0.1111  2.8289683              362880 0.00000275573192 2.718281525573
  10 0.1000  2.9289683             3628800 0.00000027557319 2.718281801146
```

Dabei ist „Summe (1/i)“ = $\sum_{j=1}^{i} \frac{1}{j}$ und für die Eulersche Zahl e näherungsweise die i-te Näherung $e_i = \sum_{j=0}^{i} \frac{1}{j!}$

Beachten Sie dabei folgende Randbedingungen:

- Die Maximale Anzahl der Tabellenzeilen ist in der symbolischen Variable `MAX_LIMIT` definiert. (20)
- die Eingabe der Obergrenze `limit` soll auf den gültigen Bereich getestet werden
  (also ob für Grenze `limit` gilt: 1 <= `limit` <= 20). Ist die eingegebene Zahl nicht im gültigen Bereich, soll
  - eine entsprechende Fehlermeldung ausgegeben werden
  - der Benutzer so lange weiter zur Eingabe aufgefordert werden, bis die Bedingung erfüllt ist.

  Die Eingabe der Obergrenze soll in einer separaten Funktion erfolgen, die durch folgenden Prototyp definiert ist: `int grenzeEinlesen(int cMax);`
  Hierbei ist `cMax` der größte erlaubte Wert für die obere Grenze (im Beispiel oben also der Wert von `MAX_LIMIT`)
  Die Funktion liefert die eingegebene Grenze zurück.
- Die Werte der Tabelle sollen in einem Array vom Typ `struct Row_s[MAX_LIMIT]` gekapselt werden. In der loop.c finden Sie bereits die passende globale Variable `rows` mit diesem Typ. Die Attribute des Datentyps müssen Sie noch selbst definieren.
- Bei der Berechnung der Tabellenzeilen gehen Sie wie folgt vor:

- Implementieren Sie eine Funktion `void calculateRow(int i)`, die zu einem gegebenen `i` die entsprechenden Werte der i-ten Zeile berechnet und diese Werte direkt in das globale Array `rows` schreibt.
- Implementieren Sie eine Funktion `void calculateTable(int limit)`. Hierbei ist `limit` die vorher eingegebene obere Grenze. Diese Funktion soll für jede Tabellenzeile die Funktion `calculateRow` aufrufen.
- Implementieren Sie eine Funktion `void printTable(int limit)` , die die in `rows` gespeicherten Werte als Tabelle (wie oben gezeigt) ausgibt.

Aufgabe 2 (optional)

Erweitern Sie Ihr Programm um eine Funktion, die die Anzahl der Ziffern der Zahl `!i` bestimmt. Diese Funktion ist durch folgenden Prototyp definiert: `int numDigits(unsigned long x)`
Erweitern Sie den Datentyp `struct Row_s` um eine weitere Variable und geben Sie auch diese bei der Ausgabe aus.

Beispiele:

- Der Wert von !3 ist 6. Die Anzahl der Ziffern ist 1.
- Der Wert von !6 ist 720. Die Anzahl der Ziffern ist 3.
- Der Wert von !11 ist 39916800. Die Anzahl der Ziffern ist 8.