1 Kleinstes gemeinsames Vielfaches

Lösungsidee:  
Das kleinste gemeinsame Vielfache wird folgend berechnet:   
num1 \* num2 / GCT (greatest common divisor von num1 und num 2)  
Nur positive Zahlen sind erlaubt.  
  
Testfälle:   
(10, 10); (10, 4); (170, -60); (23, 0)  
  
C-Program code: Siehe beigelegtes leastCommonMultiple.c/.h file.   
Tests: Siehe beigelegtes main.c file.

2 Modularisierung

Lösungsidee:  
Schnittstelle für folgende string Functionen schreiben.

**string\_len**  
Solange kein Terminierungszeichen auftritt durch das char Array iterieren und einen Counter bei jeder Iteration um 1 hinaufzählen.  
  
Testfälle:  
"TestThisLength", "", "1"

**string\_cmp**  
Hier wird buchstabenweise überprüft ob das aktuelle Zeichen größer, kleiner oder gleich ist. Siehe Angabe.  
  
Testfälle:   
("TestString", "TestString"); ("Test", "Tasten"); ("Tast", "Testen"); ("Test", "TestString");  
("TestString", "Test"); ("A", "B"); ("B", "A"); ("", "")

**string\_toUpperCase**  
String Länge ermitteln und beim der Iteration auf einen kleinen Buchstaben überprüfen, falls einer existiert wird mit folgender Berechnung der Buchstabe groß gemacht und überschrieben: „str[i] - 'a' + 'A'“   
  
Testfälle:  
"TestString", "", "56asd", " "

**string\_copy**  
Zuerst genug Speicher für den destination String reservieren, dann Zeichen für Zeichen vom source String hinüber kopieren. Am Ende noch das Terminierungszeichen beim destination String hinzufügen.  
  
Testfälle: (der zu kopierende string)  
"TestString", "42 Hahaa", " ", ""

**string\_subStr**  
Hier wird, mit einem start Offset und einem end Offset, der destination String vom source String hinausgeschnitten. Dabei muss man beachten, dass die Offsets passen d.h. end Offset darf nicht kleiner sein als start Offset, beide Offsets dürfen nicht größer sein als der source String selber und der delta Offset (endOffs - startOffs) muss >= 0 sein. Zurückgegeben wird die anzahl der kopierten Zeichen und bei fehlschlag der Funktion, wegen den bevor angegebenen Gründen, wird -1 zurückgegeben.  
  
Testfälle:  
("TestString", 2, 5); ("TestString", 0, 9); ("TestString", 0, 15); ("TestString", -3, 9); ("TestString", 6, 3); ("TestString", 0, 0); ("TestString", 5, 5);

**string\_revert**  
Hier wird einfach, indem man das letzte Zeichen (j = length of string – 1 -> wird bei jeder Iteration dekrementiert) und das erste Zeichen (i = 0 -> wird bei jeder Iteration inkrementiert) miteinander tauscht, der String reverted. Dabei wird das gemacht solange i < j ist.  
  
Testfälle:  
"RevertThisString", "Revert This String", "username: xdorphine56", "", " "

C-Program code: Siehe beigelegtes stringfunc.c/.h file.  
Tests: Siehe beigelegtes main.c file.