Übungsblatt 3
Propädeutikum Informatik

7. Mai 2019

1 Primzahlen berechnen

Schreibe eine Funktion primeNumbers(maximum), die alle Primzahlen bis zum angegebenem Maximum berechnet und in einer Liste zurück gibt.
Benutze Schleifen, um für jede Zahl zwischen 2 und maximum zu prüfen ob es eine Primzahl ist.

Achte auf eine effiziente Implementierung. Auch für ein maximum von 1.000.000 ist eine Ausführung in weit unter einer Minute möglich.

2 Palindrome

Schreibe eine Funktion isPalindrome(mystring), die eine String als Argument übergeben bekommt und prüft, ob es ein Palindrom ist. Also ob der String vorwärts und rückwärts gelesen das selbe Wort ergbibt. Die Funktion soll entsprechend True oder False zurückgeben.

3 Monotone Liste

Eine Liste ist Monoton wachsend, wenn jedes Elemente größer oder gleich allen vorrangegeangen Elementen ist (also alle Elemente mit kleinerem Index). Zum Beispiel ist die Liste [1,2,3,3.0] monoton wachsend, die Liste [1,2,5,4] jedoch nicht. Schreibe eine Funktion monoton(mylist), die prüft ob die gegebene Liste monoton wachsend ist und entsprechend True oder False zurück gibt.

In Listen müssen nicht alle Elemente den selben Datentyp haben. Die Funktion muss nur funktionieren für Listen mit int oder float Daten. Wenn andere Daten in der Liste soll False zurück gegeben werden. Der Datentyp einer Variable kann mit type(variable) überprüft werden.

4 MRA Kompression

MRA ist ein Verfahren, womit Wörter auf ihre phonetische Ähnlichkeitgeprüft werden können. Dazu werden die Wörter wie folgt komprimiert:

- wandle alle Buchstaben in Großbuschstaben um
- reduziert Doppelkonsonanten auf ein Zeichen
- entfernt alle Vokale aus dem Wort, außer wenn ein Vokal am Anfang des Wortes steht.
- Wenn die Länge des phonetischen MRA-Wortes länger als 6 Zeichen ist, wird es auf dieersten und die letzten drei Zeichen reduziert.

Beispiel:

- Basketball - BSKTBL
- Programm - PRGRM

Schreibe eine Funktion mra(mystring) welche den MRA Algorithmus implementiert. Nutze wenn nötig weitere Hilfsfunktionen um den Code übersichtlich zu halten.

5 Umgedrehte Liste

Schreibe eine Funktion reverseList(mylist), die eine Liste als Parameter übergeben bekommt und die Liste einmal umdreht, also das erste mit dem letzten Element tauscht, das zweite mit dem vorletzen, usw. Achte darauf, tatsächlich innerhalb der Liste die Elemente zu tauschen und keine zweite Liste anzulegen.

6 Collatz-Folge

Schriebe ein Funktion collatzNachfolger, welche für einen gegebenen Integer Wert den Nachfolger nach der folgenden Rechenvorschrift[1] berechnet:

$collatz(n) = n/2$, falls n gerade ist
$collatz(n) = (3 * n) + 1$, falls n ungerade ist.

Schreibe außerdem eine Funktions collatzIteration, welche für alle Zahlen zwischen 2 und 100 berechnet, wie oft die Collatz Funktion angewandt werden muss, bis das Ergebnis 1 wird. Das Ergebnis soll für jede Zahl zwischen 2 und 100 auf dem Bildschirm ausgegeben werden und in einer Liste gespeichert und von der Funktion zurückgegeben werden.

[1] `https://en.wikipedia.org/wiki/Collatz_conjecture`