



Einführung in die Informatik I

Übung 4

1 Matrix-Zugriff per Colon(:)-Operator

Erzeugen Sie eine 5x5-Matrix $A = (a_{ij})_{5 \times 5}$ durch die folgende Vorschrift für Elemente

$$a_{ij} = j + 5(i - 1)$$

Diskutieren Sie die Ergebnisse der folgenden Zugriffsoperationen:

- $A(2, :)$, $A(:, 5)$ und $A(:, :)$
- $A(1, 1 : 2 : 5)$
- $A(3 : -1 : 1, 5 : -1 : 3)$

2 Matrizen: Überlagerung der Vektoren

Schreiben Sie eine MATLAB-Funktion namens `Quersumme`, die als Parameter eine quadratische ($n \times n$) Matrix M erhält. Die Funktion soll prüfen, ob die Summe einer Zeile mit der jeweiligen Summe der Spalte übereinstimmt. Bei Übereinstimmung wird in einen Vektor y eine 1 eingetragen, andernfalls eine 0.

Bsp.: 3x3-Matrix:

$$M = \begin{bmatrix} 8 & 3 & 6 \\ 4 & 8 & 7 \\ 4 & 9 & 5 \end{bmatrix} \rightarrow \text{Ergebnis: } y = [0 \ 0 \ 1]$$

$$\begin{aligned} \sum \text{Spalte1} &\neq \sum \text{Zeile1} \rightarrow y(1) = 0 \\ \sum \text{Spalte2} &\neq \sum \text{Zeile2} \rightarrow y(2) = 0 \\ \sum \text{Spalte3} &= \sum \text{Zeile3} \rightarrow y(3) = 1 \end{aligned}$$

Hinweis: Verwenden Sie die `sum`- und `size`-Befehle.

3 Fehlerfinden

- In der Vorlesung wurden verschiedene Fehlerarten vorgestellt. Nennen Sie die Ihnen bekannten Fehlerarten, die Reaktion des Compilers und geben sie ein Beispiel dafür an.



- ii. Suchen Sie die Stellen in folgenden Funktionen, die eine Fehler- oder Warnmeldung verursachen. Bestimmen Sie die **Zeile(n) mit einem Fehler** und **begründen** Sie Ihre Entscheidung. Es ist auch möglich, dass eine oder mehrere Funktionen **fehlerfrei** sein können. Schreiben Sie als Begründung FEHLERFREI, wenn das Skript vollständig funktioniert.

<pre>function v = f1(x,n) v = zeros(n,1); for i = 1:round(n/2)+3x v(i) = n * x; end</pre>	<pre>function x = f2(b) if (cos(b)^2-(sin(b+1))*b) > 0 x = 1 else x = 0 end</pre>	1	2
<pre>function acht = f3(neun, eins) v = [eins;3.14]; zwei=round(v(2)); if pi > v(zwei) drei = 1 + zwei; else fuenf = 2 + zwei; end acht = drei + fuenf;</pre>	<pre>function y = f4(b) y = 1; while x + y < 10 x = y + 1; y = x + y; end</pre>	3	4
<pre>function beta=f5(alpha) if length(pi) = length(2*pi) beta = -alpha; elseif length(pi) < length(2*pi) beta = +alpha; else beta = alpha+alpha/2; end end</pre>	<pre>function x=f6(u) if length(u+f6) > 3 x=u+log10(u); else x=u-log10(u); end end</pre>	5	6
<pre>function x=f7(t) x=zeros(length(t),length(t)); for i=1:size(x,1) for j=i:length(t)+1 x(i,j)=t(j); end end end</pre>	<pre>function x=f8(y) f=y+3; x=sqrt((2+y)/(cos(3*f)+y))+1; x=x+1; end</pre>	7	8
<pre>function x=f9(y) h=y+3; x=sin((2+y/ln(h))/(tan(h*z)+pi)); x=x+1; end</pre>	<pre>function x=f10(s) if length(s) > 3 x=1; else x=0; end end</pre>	9	10



<pre>function x=f11(w) for i=1:length(w) w(i)=w(i)+log(w(i)); end x=w;</pre>	11	<pre>function x=f12(y) z=y + factorial(5); x=diff((log(y))/(pi*z)+y)+1; x=x+1; end</pre>	12
--	----	--	----

4 Sieb des Eratosthenes

Das Sieb des Eratosthenes ist eine einfache Methode zur Bestimmung aller Primzahlen p im Intervall $2 \leq p \leq n$. Dazu werden die Zahlen 2 bis n in eine Reihe geschrieben. Sodann wird die erste nicht durchgestrichene Zahl p (zu Beginn also 2) genommen und jedes Vielfache von p , außer p selbst, durchgestrichen:

2 3 ~~4~~ 5 ~~6~~ 7 ~~8~~ 9 ~~10~~ 11 ~~12~~ ...

Nun wird die nächste nicht durchgestrichene Zahl gesucht (also 3) und deren Vielfache durchgestrichen:

2 3 ~~4~~ 5 ~~6~~ 7 ~~8~~ 9 ~~10~~ 11 ~~12~~ ...

Sobald die Vielfachen aller Zahlen der Liste durchgestrichen wurden, sind alle Primzahlen bis n gefunden.

Aufgabenstellung: Programmieren Sie das Sieb des Eratosthenes als Funktion namens `Eratosthenes`, die den Algorithmus durchführt.

Hinweis: Den Rest der Division zweier Zahlen erhalten Sie mit dem `rem`-Befehl.

5 Zusatzaufgabe

- Schreiben Sie ein Skript, welches angibt, wie oft gewürfelt wird bis ein Würfel zwei Mal hintereinander eine Sechs zeigt.
- Schreiben Sie ein Skript, welches angibt, wie oft für ein Sechserpasch gewürfelt werden muss.