Mathematik am Computer

7. Übung: Matlab, Teil IV

Marcus Grote und Helmut Harbrecht

Universität Basel

6.–9. November 2023

Übersicht

Die while-Schleife

```
while Bedingung
      Befehle
end
```

1. Falls zu Beginn der while-Schleife die Bedingung gilt, so werden alle Befehle zwischen `while` und `end` ausgeführt.
2. Es wird nun wieder geprüft, ob immer noch die Bedingung gilt. Falls ja, so werden wieder alle Befehle zwischen `while` und `end` ausgeführt.
3. Dies wiederholt sich solange bis die Bedingung nicht erfüllt ist.

Die while-Schleife

Anmerkungen:

- Gilt die Bedingung am Anfang nicht, so werden alle Befehle innerhalb der while-Schleife übersprungen.

Beispiel: Die Bedingung ist $t < 1$.

- Es muss der Variable t vor der Abfrage in `while t < 1` ein Wert zugewiesen sein.
- Die Abfrage $t < 1$ liefert als Ergebnis entweder 0 (`false`) oder 1 (`true`). Die Befehle in der Schleife werden solange wiederholt ausgeführt, bis der Wert der logischen Abfrage $t < 1$ gleich 0 (`false`) ist.
- Gilt $t < 1$ immer, so bricht der Programmablauf niemals ab. Deswegen wird der Wert von t normalerweise innerhalb der Schleife geändert.
- Jede logische Abfrage ist erlaubt, also auch z.B. `while 0<=t && t<1` oder `while c == 15`.

Die while-Schleife

Beispiel

INPUT

```
Sum = 0;
cnt = 0;
while Sum < 10
    if rand(1) > 0.5
        Sum = Sum + 1;
    else
        Sum = Sum - 1;
    end
    cnt = cnt + 1;
end
disp(['S: ' num2str(Sum)])
disp(['C: ' num2str(cnt)])
```

OUTPUT

```
S: 10
C: 292
```

```
S: 10
C: 5238
```

```
S: 10
C: 366
```

```
S: 10
C: 82
```

⋮

Magische Quadrate

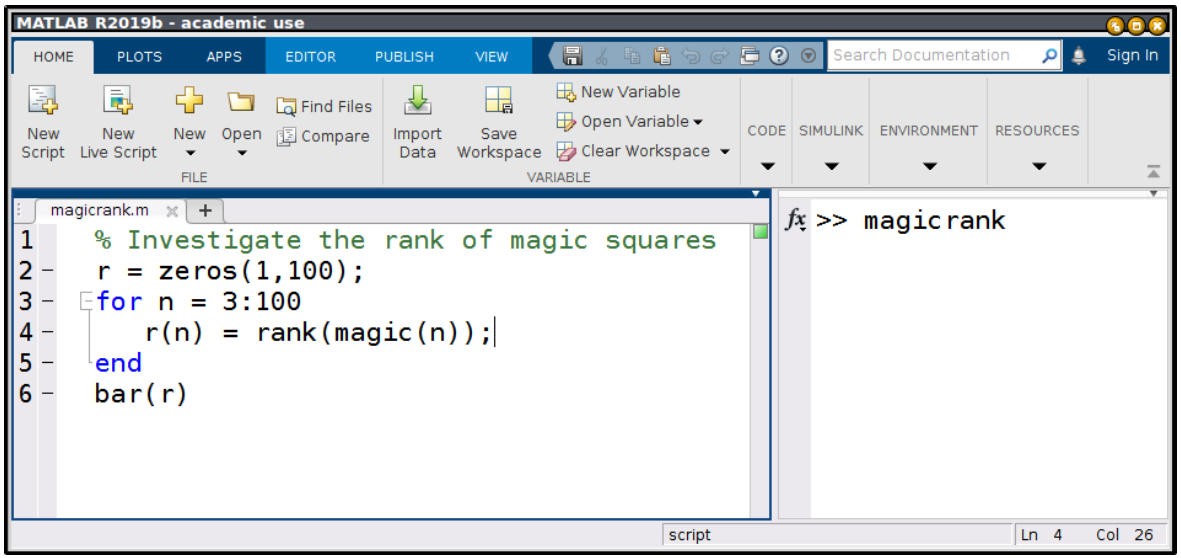
Eine $(n \times n)$ Matrix A ist ein magisches Quadrat, falls die Zeilen- bzw. Spaltensummen konstant sind, z.B.

$$A = \begin{pmatrix} 8 & 1 & 6 \\ 3 & 5 & 7 \\ 4 & 9 & 2 \end{pmatrix}$$

`magic(n)` erzeugt in MATLAB ein $(n \times n)$ magisches Quadrat.

Skripte

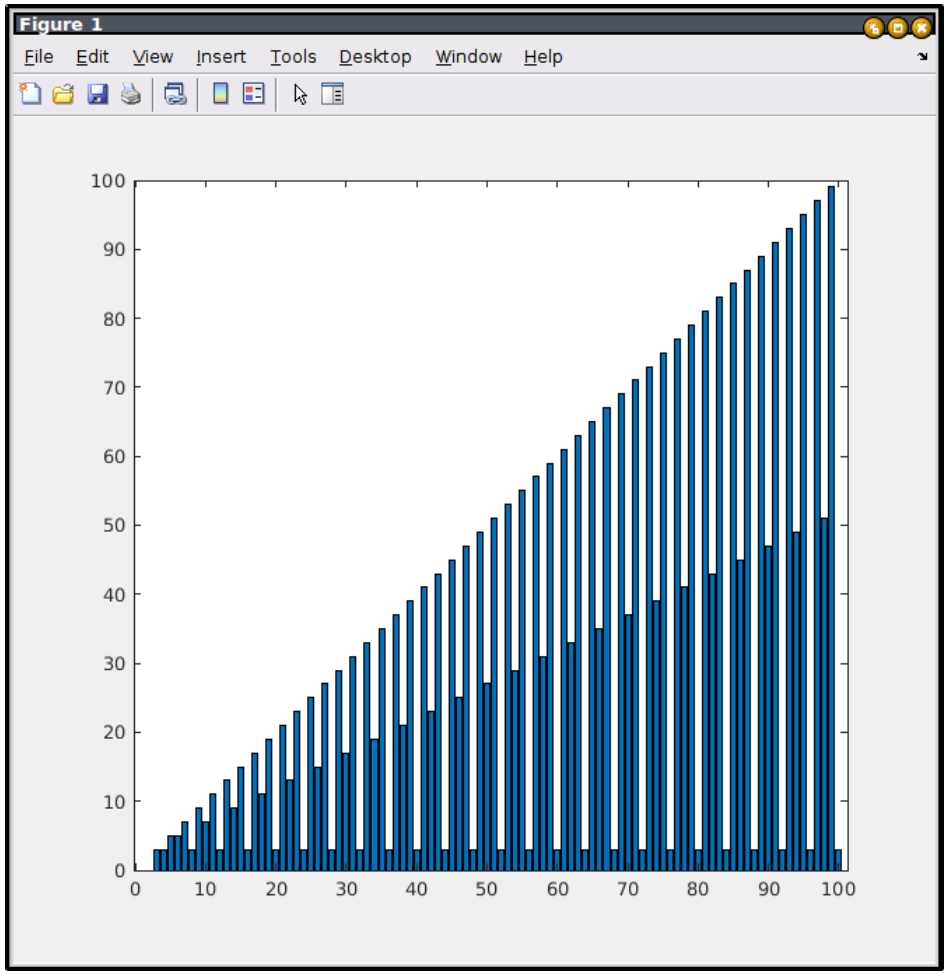
Magische Quadrate



Eingabe von `magicrank` im Command Window führt das Skript `magicrank.m` aus.

Skripte

Magische Quadrate: Ausgabe



Skripte und Funktionen

Einschränkungen

Skripte führen nach Aufruf die darin enthaltenen Befehle sequenziell aus. Dabei ist zu beachten:

Skripte haben keinen abgekapselten Workspace.

Die Variablen im Haupt-Workspace können durch ein Skript initialisiert, verändert und gelöscht werden.

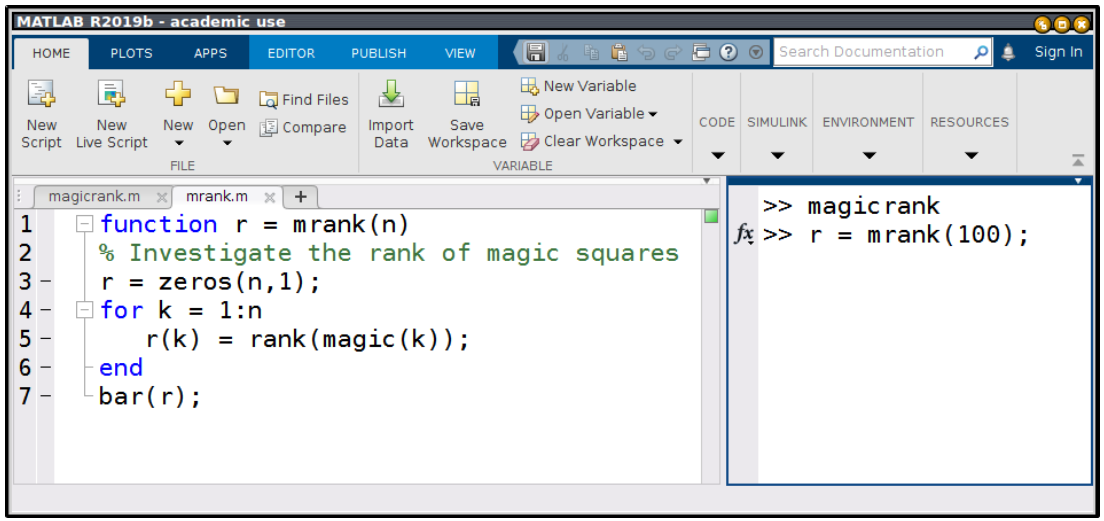
Angenommen wir wollen rechnen, ohne die Werte im Haupt-Workspace zu verändern:

Wir brauchen also einen temporären Workspace

- $\rightarrow$ in den ausgewählte Parameter übergeben werden, und
- $\leftarrow$ aus dem Werte und Ergebnisse zurückgenommen werden

Lösung: Funktionen.

Funktionen



Die Eingabe von `mrank(100)` liefert dasselbe Ergebnis.

Funktionen

- Dateiname und Funktionsname sollen übereinstimmen.
- Die Dateierweiterung ist `.m`.
- Alle Variablen die innerhalb einer Funktion bearbeitet werden, sind lokal, d.h. in einem temporären, abgekapselten Workspace.
- Die Funktion `function r = mrank(n)` bedarf eines Eingabeparameters und gibt eine Variable als Ergebnis zurück.
- Eingabe mehrerer Parameter und Rückgabe mehrerer Variablen ist möglich.

  Beispiel:

```
function [V1 , V2, V3] = nFunktion(P1, P2)
```

Aufruf:

```
X=1; Y=2;
[A,B,C] = nFunktion(X,Y);
```

Funktionen

Aufruf:

1. `mrank(100)` führt die Funktion aus und gibt den Rückgabewert aus.
2. `mrank(100);` führt die Funktion aus und gibt den Rückgabewert nicht aus.
3. `z = mrank(100);` führt die Funktion aus und speichert den Rückgabewert in z.
4. `z = mrank(100)` führt die Funktion aus, speichert den Rückgabewert in z und gibt ihn aus.

Globale Variablen

Variablen, die direkt im „Command Window“ oder durch eine Befehlsabfolge in einem Skript definiert werden, sind global.

Auf diese Variablen kann vom „Command Window“ und von jedem Skript aus zugegriffen werden.

Die Variablen und ihre Werte existieren so lange, bis diese gelöscht oder geändert werden.

Lokale Variablen

- Variablen in Funktionen sind lokal, das heisst, sie sind nur innerhalb dieser Funktion bekannt.

  Sobald der Funktionsaufruf endet, werden die Variablen und ihre Werte gelöscht.

- Funktionen kennen keine globalen Variablen, deswegen:
  - können Funktionen keine globalen Variablen modifizieren,
  - müssen alle benötigten Werte als Parameter übergeben werden,
  - müssen alle nach Ende des Funktionsaufruf weiter benötigten Werte als Rückgabewerte zurückgegeben werden.