Lektion 5 : Verwendung des Moduls ti_system

Übung 1 : Arbeiten mit Listen

In dieser ersten Übung der Lektion 5 wird gezeigt, wie man mittels des Moduls **ti_system** Listen in Python importieren und auch aus Python exportieren kann.

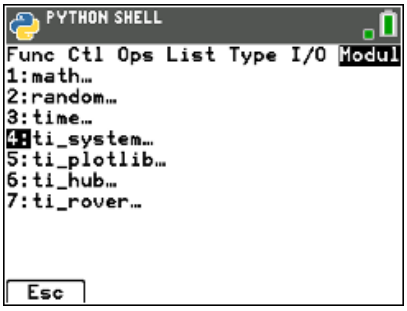
Lernziele :

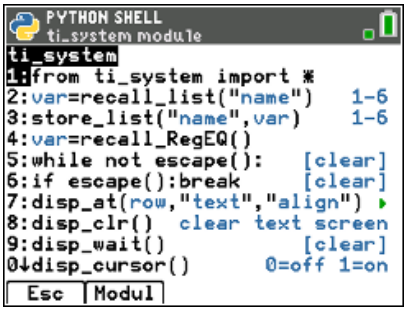
- Importieren, exportieren und erstellen von Listen.
- Wiederholung der Grafikbefehle aus Lektion 4.

Das Modul **ti_system** enthält Befehle zur Kommunikation mit dem grafischen Taschenrechner.
In dieser ersten Übung werden davon nur zwei verwendet :

2 : var=recall_list(« name ») und **3 : store_list(« name »,var)**

Die anderen Befehle werden in späteren Übungen verwendet.



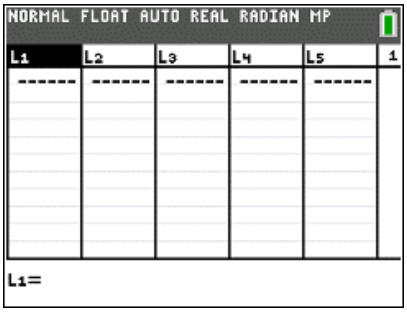


Zuerst sollen im Taschenrechner zwei Listen erstellt und grafisch dargestellt werden.

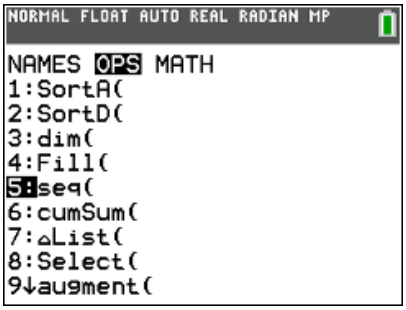
1 : Eingeben von Werten in den Taschenrechner

a) Erstellen von zwei Listen L_1 und L_2 :

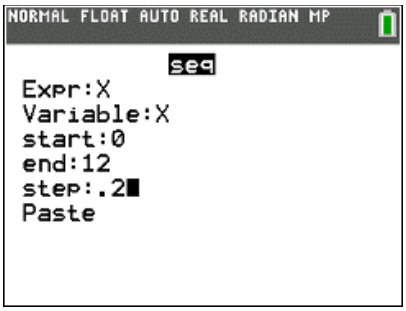
- Zunächst sollten alle vorhandenen Listen gelöscht werden. Dazu muss der Taschenrechner im üblichen Modus sein. Mit [2nd]+ kommt man ins Menü **MEMORY**. Dort muss der Punkt **4 :ClrAllLists** gewählt werden.
- Dann wechselt man mit [stat] **1 :Edit** in den Listeneditor. Der Cursor wird in den Listenkopf bewegt wie dargestellt.

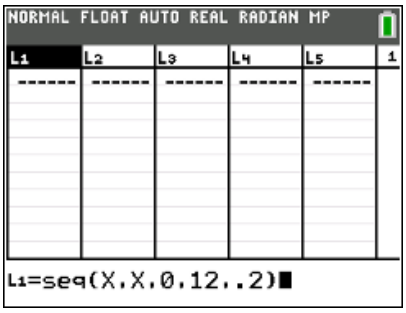


- Die Liste **L_1** soll eine Zahlenfolge von 0 bis 12 enthalten, die x-Werte der Funktion: $x \mapsto -x^2/2 + 3x + 1$
- Die Funktionswerte werden dann in **L_2** erzeugt. **L_1** und **L_2** bilden die Grundlage der grafischen Darstellung.
- Die Tasten [2nd][stat] führen zum Listenmenü **list**. Hier sollte der Punkt **5 :seq(** angewählt werden.

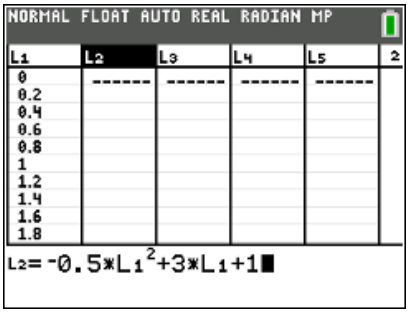


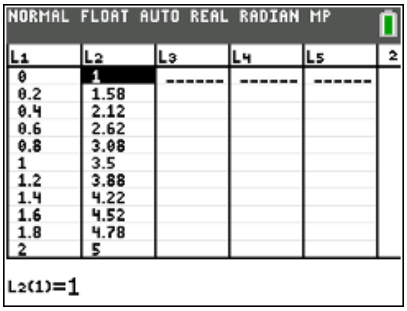
- Man erhält eine Maske, die wie nebenstehend ausgefüllt werden sollte. Damit wird eine Punktfolge für das Intervall [0 ;12] mit der Schrittweite 0,2 erzeugt. Mit **Paste** werden die Angaben in die klassische Darstellung für Folgen umgewandelt und in den Listeneditor kopiert. [enter] erzeugt dann die Liste **L_1**.



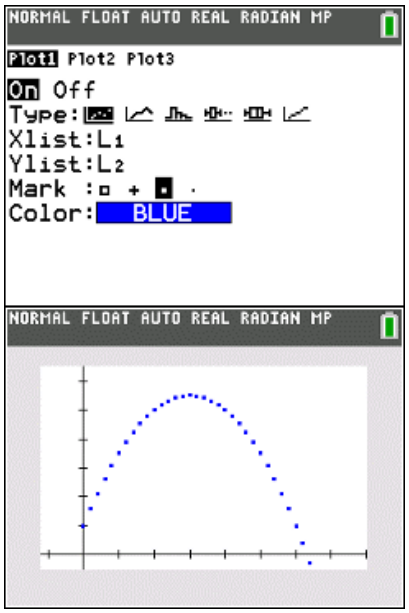


- Nun wird der Cursor zu **L_2** bewegt und der Funktionsterm eingegeben. Dabei muss man das **x** durch den Listennamen ersetzen. Den Listennamen muss man zwingend dem Menü **list** entnehmen.
- Aus $x \mapsto -x^2/2 + 3x + 1$ wird so L_2=-0.5*L_1^2+3*L_1+1.
- [enter] führt dann die Berechnung durch.





b) Grafische Darstellung



- Über 2nd y= gelangt man ins Menü **StatPlot**. Hier sollte **Plot1** mit den nebenstehenden Einstellungen ausgewählt werden.
- Anschließend kann man über zoom **9 :ZoomStat** die Grafik darstellen lassen ; sie sieht aber noch nicht sehr gut aus. Besser ist es, man verwendet die **Window**-Einstellungen X_{min} = -1.2 ; X_{max} = 8 ; Y_{min}= -0.5 und Y_{max}= 6.5.
- **Hinweis** : Das Zeichnen aller Funktionsgraphen sollte deaktiviert sein!

2) Importieren der Werte in ein Python-Programm :

- Anlage eines neuen Programmes mit dem Namen U5SB1.
- Die Module **ti_system** und **ti_plotlib** importieren (Reihenfolge egal).
- Zwei leere Listenvariablen **lx** und **ly** erzeugen.
- Die leeren Listen **lx** und **ly** füllen mit den Inhalten von **L_1** und **L_2**. Dazu wird der Befehl **2:var=recall_list(« name »)** aus dem Modul **ti_system** verwendet. Der **« name »** ist der **Listenname**, also **1** oder **2 (ohne L !).**
- Lässt man nun das Programm laufen, so kann man sich durch Drücken von vars **OK** und enter den Inhalt der Listen **lx** und **ly** ansehen.

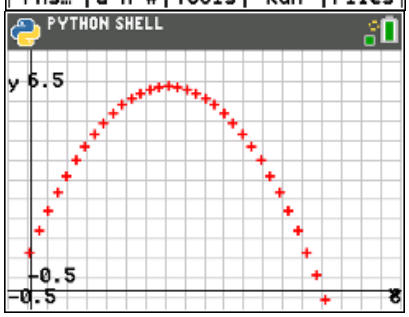
Lehrertipp : Die Erstellung von leeren Listen lx=[ ] und ly=[ ] ist nicht unbedingt erforderlich, da sie beim Abruf der Listen L_1 und L_2 erstellt werden. Es ist jedoch sinnvoll, die Gewohnheiten beizubehalten, die notwendig sind, wenn das **ti_system**-Modul nicht verwenden wird.

Grafische Darstellung

Zunächst einmal muss die grafische Darstellung eingerichtet werden wie abgebildet.

```
EDITOR: U5SB1
PROGRAM LINE 0015
lx=recall_list("1")
ly=recall_list("2")
#Grafik
plt.cls()
plt.grid(1,1,"solid")
plt.window(-0.5,8,-0.5,6.5)
plt.axes("on")
plt.labels("x","y",12,2)
plt.color(255,0,0)
plt.scatter(lx,ly,"+")
plt.show_plot()_
Fns… | a A # | Tools | Run | Files
```

Dann kann man das Programm laufen lassen.



3) Exportieren von Werten aus einem Python-Programm

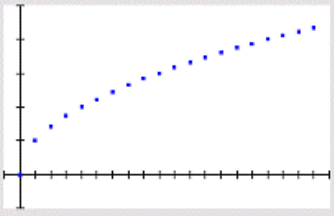
- Ein neues Programm mit dem Namen **U5SB11** anlegen.
- Die drei angegebenen Module importieren (Reihenfolge beliebig).
- Die Funktion **data(a,b,h)** programmieren.
- Es werden zwei leere Listen **x** und **y** angelegt, die später Werte aus dem Intervall **[a ; b]** enthalten, die mit der Schrittweite **h** erstellt werden.

```
EDITOR: U5SB11
PROGRAM LINE 0010
from math import *
from ti_system import *
def data(a,b,h):
  x=[]
  y=[]
  for i in range(a,b,h):
    x.append(i)
    y.append(sqrt(x[i]))
  store_list("1",x)
  store_list("2",y)_
Fns… | a A # | Tools | Run | Files
```

Lehrertipp : Durch die Erstellung von zwei leeren Listen wird vermieden, dass bei der Ausführung des Programmes ein Fehler gemeldet wird.

- Liste **y** enthält die Quadratwurzel der Werte aus Liste **x**.
- Die Listen werden nun mit einer **For**-Schleife gefüllt. Die **append**-Befehle bewirken, dass neue Elemente an die Liste angehängt werden. Die Listen haben also eine dynamische Länge.
- Die Listen werden mit dem Befehl **store_list(**« name »,**var)** in den Taschenrechner übertragen. **var** ist der Listenname im Programm, « **name** » die Nummer der Liste im Taschenrechner.

```
PYTHON SHELL
>>> # Shell Reinitialized
>>> # Running U5SB11
>>> from U5SB11 import *
>>> data(0,20,1)
>>>
Fns… | a A # | Tools | Editor | Files
NORMAL FLOAT AUTO REAL RADIAN MP
```



Hinweis: Vorsicht mit der Einrückung, denn die **store_list**-Anweisungen müssen nicht in der Schleife stehen. Man kann das Menü **Tools (f3)** und dann **2 :Indent ◄** verwenden, um eine Versatzebene zu löschen. Noch einfacher geht es mit zweimal [del].

- Nun kann man das Programm laufen lassen. Scheinbar tut sich nichts.
- Erst wenn man Python verlässt kann man sich das Ergebnis in einem StatPlot L_1 gegen L_2 ansehen.

Lehrertipp : Die Ausgabe in Listen wird interessant im Zusammenhang mit den Sensoren des Microcontrollers **TI-Innovator** & **TI-Rover**.