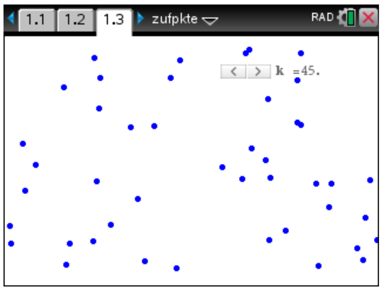
In dieser Einheit wirst du lernen, wie man **Listen** in Programmen einsetzen kann, um interessante Streudiagramme zu erzeugen.

Lernziele:

- Einsatz von **Listen** in Programmen
- Mit **Listen** im Programm Streudiagramme erzeugen



TI-Nspire™ Basic kennt keine direkten Anweisungen zur Erzeugung von Grafiken. Ein Programm kann Funktionen zum Plotten und für Streudiagramme produzieren, aber keine Punkte direkt plotten.

In dieser Lektion werden wir drei wichtige Ideen verfolgen:

1. mit Listen in einem Programm arbeiten,
2. auf einer Graphs-Seite ein Streudiagramm dieser Listen darstellen, und
3. 'dynamische' Programme verwenden, die 'auf Abruf' laufen.

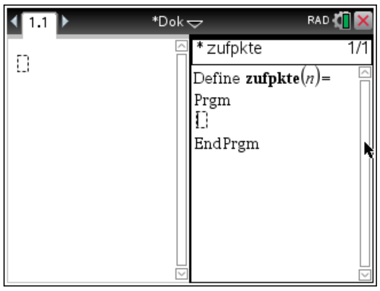
Listen definieren

Listen werden zwischen geschlungenen Klammern { } geschrieben. Um eine leere Liste zu erzeugen verwende eine Zuweisung wie etwa **mliste:={ }** ({} = ctrl)) mit keinem Listenelement zwischen den Klammern.

Die Elemente der Liste (die Werte zwischen den Klammern) werden über eckige Klammern angesprochen. So meint man mit **mliste[3]** das dritte Element von **mliste**.

Man kann Elemente zu einer Liste hinzufügen, indem man einen Wert gleich hinter ihren letzten Wert ablegt. Wenn z.B. eine Liste **mliste** die drei Elemente 12, 7 und 2 enthält, kann man ein weiteres Element als Viertes hinzufügen. mliste[4]:=17 ergibt dann mliste = {12, 7, 3, 17}. dim(Listenname) ergibt die Anzahl der Elemente der Liste und wird oft so dazu verwendet, ein Element in eine Liste aufzunehmen: **mliste[dim(mliste)+1):=**<*neues Element*>.

Programmierung von Zufallspunkten



Wir werden nun ein Programm schreiben, das ein Muster von Zufallspunkten auf der Graphs-Seite erzeugt.

1. Beginne ein neues Programm im Editor (wir wollen es **zufpkte** nennen) mit dem Argument *n*.
   - *n* ist die Anzahl der zu erzeugenden Punkte.
   - Das Programm produziert (oder erneuert) zwei 'globale' Listen von Zufallszahlen, mit denen wir das Streudiagramm auf einer Graphs-Seite plotten werden.

Zufallszahlengeneratoren

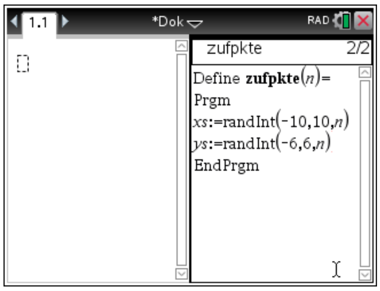
Im TI-Nspire™ CX gibt es mehrere Zufallszahlfunktionen. Die beiden am häufigsten gebrauchten sind **rand()** und **randInt( )**.

- **rand**() erzeugt eine Zufallszahl zwischen 0 und 1.
- **randInt**(*a*,*b*) erzeugt eine ganze Zufallszahl aus dem Intervall [a,b]. **randInt**(1,6) erzeugt eine ganze Zufallszahl von 1 bis 6. Probiere das im *Calculator*.

Wir machen den ersten Versuch, Listen von Zufallszahlen zu erzeugen:

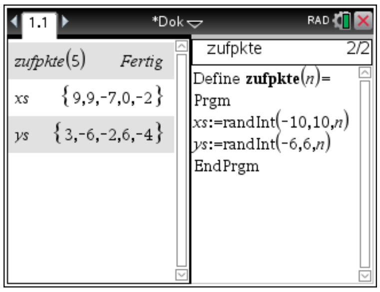
xs := randInt(-10, 10, n)

ys := randInt(-6, 6, n)



2. Das dritte Argument in den beiden Funktionen lässt die **randInt**-Funktion eine Liste von *n* Werten im gewählten Intervall erzeugen, anstelle von nur einer einzigen.

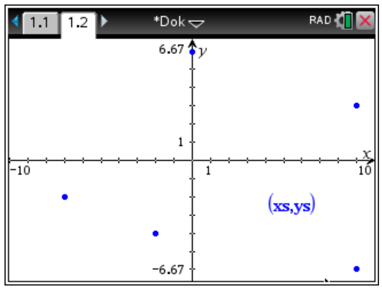
3. Speichere das Programm mit ctrl B. Führe nun das Programm im *Calculator* mit einem kleinen Wert für *n* aus und sieh dir die Listen ***xs*** und ***ys*** an. Deine Werte werden wahrscheinlich nicht mit den rechts abgebildeten übereinstimmen.



4. Füge eine Graphs-Seite im Problem ein und erzeuge ein Streudiagramm von **(*xs*,*ys*)** über ***menu> Graph Eingabe/Bearbeitung> Streudiagramm***.

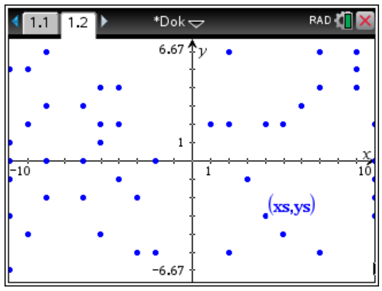
5. Gib ***xs*** für die x-Werte (bei x←) und ***ys*** eine Zeile tiefer für die y-Werte (bei y←) ein.

6. Schließe ab mit enter.



Siehst du nun, warum wir diese Intervalle für die Zufallszahlen gewählt haben? (Überprüfe die Standardvorgaben für die Fenstereinstellungen!)

Nachdem wir das Programm erfolgreich getestet haben, wechsle wieder zum *Calculator* und führe das Programm mit einem größeren Argument *n* (aber nicht zu groß) aus. Sieh dir das Ergebnis im Graphs-Fenster an.



zufpkte(50)