**Skript**

**Programmieren mit dem TI-Inspire**

**von**

**Georg Sahliger**

**in Zusammenarbeit mit der ProgrammierAG**

**Gymnasium Mainz-Oberstadt**

Aktueller Stand

24.12. 2017

Inhaltsverzeichnis

[0 Vorwort 2](#_Toc501874924)

[1. Einführung 3](#_Toc501874925)

[2. Befehle für den Taschenrechner 5](#_Toc501874926)

[3. Variablen eingeben und ausgeben 7](#_Toc501874927)

[4. Text eingeben und If-Anweisung einsetzen 8](#_Toc501874928)

[5. Mathematische Vorlagen verwenden 9](#_Toc501874929)

[6. For-Schleifen verwenden. 10](#_Toc501874930)

[7. While-Schleifen und Zufallszahlen verwenden. 11](#_Toc501874931)

[8. Programme mit anderen Applikationen verbinden. 12](#_Toc501874932)

# 0 Vorwort

Der TI-Inspire ist ein programmierbarer Taschenrechner. Dies ist eine gute Funktion, um Schülern ein klein wenig das Programmieren näherzubringen. Die Programmiersprache nennt sich TI Basic. Daneben ist der Taschenrechner auch noch Lua-fähig. Mit dieser Sprache lassen sich z.B. Spiele auf dem TI-Inspire programmieren.

In diesem Skript soll es aber vor allem um TI Basic gehen. Damit lassen sich einfache Matheprogramme im Unterricht realisieren.

Leider bin ich keine Informatiker und leider finden sich im Internet nicht so viele hilfreiche Seiten, so dass ich hier nur aufschreiben kann, was ich bisher entdeckt habe. Für Tipps und weitere Hinweise bin ich sehr dankbar.

Das Skript ist so aufgebaut, dass man einzelne Seiten herauskopieren und direkt im Unterricht einsetzen kann. Dabei sollten alle Schüler das Arbeitsblatt „Einführung in das Programmieren“ bearbeiten. Die Reihenfolge der anderen Programme ist nicht zwingend. Es empfiehlt sich das Einführungsblatt zu laminieren und als Hilfe bereitzuhalten, falls den Schülern das Erstellen und Aufrufen von Programmen nicht präsent ist.

Ungeübtere Schüler programmieren die Programme auf den Arbeitsblättern zunächst einmal nach. Die geübteren Schüler können die Programme variieren oder neue Programmideen ausprobieren.

Die Beispielprogramme und weitere Links finden sich auf [www.Sahliger.net](http://www.Sahliger.net)

Es empfiehlt sich die Lehrersoftware bzw. die Schülersoftware von TI Nspire zu nutzen, um z.B. Die Programme von der Homepage auf die Rechner zu übertragen. Ansonsten sit die bei diesen kleinen Programmen auch nicht unbedingt nötig. Alle Befehle findet man auch zum Anklicken im Taschenrechnermenü.

Die Programme auch am Taschenrechner (Handheld) tippen und per Kabel und mit Hilfe des Befehls „Senden“ an andere Handhelds verteilen. Den Befehl „Senden“ findet man bei [DOK] , [1:Datei] [6:Senden].

Ich hoffe, allen, die sich für das Programmieren interessieren, hiermit einen kleinen Einstieg zu ermöglichen.

# Einführung

|  |  |
| --- | --- |
| **Programmieren**  **mit dem Ti-Inspire** | |
| Aufgabe: Erstelle ein Programm! Das Programm soll zwei Zahlen abfragen, anschließend beide addieren und die Summe ausgeben. | |
| Öffne eine Calculatorseite, [menu], [9: Funktionen und Programme], [1: Programmeditor], [1: Neu]. Mit [enter] abschließen. |  |
| Nenne das Programm „Addition“ und bestätige mit [enter]. |  |
| Unter dem Wort „Prgm“ kannst du nun Befehle eingeben.  Request fragt eine Variable ab. „Enter a“ ist kein Befehl, sondern der Text, der bei der Abfrage angezeigt werden soll. Anstatt „Enter a“ hätte man auch „Gib a ein:“ schreiben können.  Die Anführungsstriche “ “ findest du bei [ctrl] und    Disp zeigt etwas auf dem Bildschirm an.  Weitere Befehle, die man eingeben kann, findet man unter [menu] |  |
| Wenn das Programm fertig ist, muss es überprüft und gespeichert werden. Hierzu [menu],  [2: Syntax überprüfen & speichern,  [1: Syntax überprüfen & speichern. Ctrl +B] +  Mit [enter] abschließen. |  |
| Dann „addition()“ eingeben.  Die Klammer ist hierbei wichtig!  Möchte man das Programm noch einmal starten, muss man nicht noch einmal „addition ()“ eingeben.  Es genügt, mit der Pfeiltaste des Touchpads nach oben zu klicken und auf [enter] zu drücken.  Man kann das Programm auch starten, indem man auf einer Calculatorseite die Taste VAR drückt und das Programm auswählt. |  |
| Nach Eingabe aller Variablen, führt der Rechner das Programm aus. |  |

# Befehle für den Taschenrechner

|  |  |  |
| --- | --- | --- |
|  | | |
| Befehl | Erklärung | Beispiel |
| := | Ordnet einer Variablen eine Zahl oder ein Wort zu. | c:= 1+ 2 |
| Disp | Zeigt ein Wort oder den Wert einer Variablen an | Disp "Ergebnis",c  Disp a,"cm" |
| Disp "" | Erzeugt eine Leerzeile |  |
| DispAt | Schreibt in eine bestimmte Zeile. Hier Zeile 2 | DispAt 2,"Gib dein Lieblingsfach ein" |
| Request | Liest eine Zahl ein | Request "Zahl",a  Request a |
| RequestStr | Liest einen Text ein | Request "Wie heißt du?",a |
| Text | Gibt einen Text aus und einen OK Knopf, mit dem man weiterklicken kann. | Text „Starte mit OK!“ |
| Wait 2 | Wartet für 2 Sekunden |  |
| © | Fügt man dieses Zeichen ein, kann man das Programm kommentieren. | ©Diese Zeile macht… |
| getKey() | Warte auf eine Eingabe |  |
| Define | Definiert einen Variable |  |
| If Then EndIf | Wenn-Dann Anweisung | If a>b Then  c:=a-b  Disp "Differenz:",c  EndIf |
| If …. then … Else ….. Then ….. Endif |  | If a < b  then Disp „a ist kleiner“  Elseif a>b  then Disp „b ist kleiner“  else Disp „a ist gleich b“  Endif |
| For k, 1, 10, 1:  :EndFor | Schreibt 10 mal den Text „Hello World“  „Für k = 1 bis 10 schreibe „Hello World!“  Die letzte 1 steht für die Schrittweite.  Die beiden Zahlen können auch Variablen sein. | For k,1,10  Disp Hello World  EndFor  For k,1,b  c:=a+c  EndFor |
| While  EndWhile | Erzeugt eine Schleife, die ausgeführt solange eine Bedingung erfüllt ist.  Dabei wird die Bedingung geprüft, bevor die Schleife gestartet wird.  < > Steht für ungleich. | While w<>10  w:=w+1  ….  EndWhile:Disp "Richtige: " |
| :Loop: :EndLoop |  |  |
| Try: :Else: :EndTry |  |  |
| Lbl und Goto | Mit lbl kann man bestimmte Stellen im Programm markieren und mit Goto dann dort direkt hinspringen. | If x = 1 Goto Stelle1  \* \*  \*  Lbl Stelle1 |
| Die folgenden Befehle sind keine reinen Programmierbefehle, sondern z.B. im Menu „Calculator“ zu finden. Aber die Befehle können auch genutzt werden. | | |
| Rand() | Erzeugt eine Zufallszahl zwischen 0 und 1 |  |
| RandInt(1,6) | Erzeugt eine ganze Zufallszahl zwischen 1 und 6 |  |
| RandInt(1,6,10) | Erzeugt zehn ganze Zufallszahl zwischen 1 und 6 |  |
| Randseed() | Sorgt für eine Anfangszahl. Ohne diese wäre die Zufallszahlen immer gleich. | Randseed(12) |

# Variablen eingeben und ausgeben

|  |  |  |
| --- | --- | --- |
| **PRGMULTI** | | **Klasse 7** |
| Beschreibung: Zwei Variablen werden eingelesen und miteinander multipliziert.  Lernziele: Im ersten Programm in diesem Skript haben wir Variablen mit dem Requestbefehl eingelesen. Nun sehen wir, wie wir dies auch direkt machen können.  Ebenso soll das Programm in einer anderen Ansicht geöffnet werden. | | |
| Öffne im Startbildschirm [1: Neues]  und wähle dann [9: Programmeditor hinzufügen] aus. |  | |
| **Programmcode**  Define prgmulti(a,b)=  Prgm  c:=a ∙ b  Disp "Das Produkt ergibt: ",c  EndPrgm |  | |
| Nun kann man einen Calculatorseite öffnen. Dazu klickt man auf [on] und wählt die Calculatorseite aus.  Dort drückt man nun auf [VAR] und wählt das Programm aus. Startet man nun das Programm, so gibt man die Variablen direkt in der Klammer ein. |  | |
| Variationen, weiterführende Ideen: Weitere Grundrechenarten, Flächenberechnungen | | |

# Text eingeben und If-Anweisung einsetzen

|  |  |  |
| --- | --- | --- |
| **PRGMULTI** | | **Klasse 7** |
| Beschreibung: Der Schüler wird gefragt, ob Mathe sein Lieblingsfach ist. Bejaht er dies, erhält er ein Lob.  Lernziele: Text einlesen. Den Textbefehl kennen lernen und eine If-Anweisung benutzen. | | |
| **Programmcode**  Define prglieb()=  Prgm  Text "Das folgende Programm fragt nach deinem Lieblingsfach."  RequestStr "Ist Mathe dein Lieblingsfach?",a  If a="ja" or a="Ja" Then  Disp "Das ist toll!☺"  Else  Disp "Schade. "  EndIf  EndPrgm | | |
| Die Befehle findet man auch immer unter [Menu]  Der Taschenrechner verfügt über verschiedene Sonderzeichen, die natürlich auch alle eingebaut werden können. Beispielsweise: 😊 |  | |
| Öffnet man nun das Programm, sieht das folgendermaßen aus: |  | |
| Variationen, weiterführende Ideen: Quiz, bei dem richtige Antworten gelobt werden oder man Punkte erhält. Verschiedene Blödsinnsprogramme. | | |

# Mathematische Vorlagen verwenden

|  |  |  |
| --- | --- | --- |
| **PRGWurzel** | | **Klasse 7** |
| Beschreibung: Von einer Zahl wird die Wurzel gezogen und sowohl der positive als auch der negative Wert ausgegeben.  Lernziele: Sonderzeichen verwenden. Programm in Notes öffnen. | | |
| Öffne im Startbildschirm [1: Neues]  und wähle dann die gelbe Applikation [Notes] aus.  Tippe den nebenstehenden Text ab. So kannst du erklären, was das Programm macht.  Die Umlaute also hier den Buchstaben „ö“ erhältst du, wenn du „o“ tippst und dann rechts die Taste mit der Fahne antippst.  Füge nun eine Mathebox ein. Hier kannst du später das Programm öffnen. Die Mathe box findest du unter [Menu] [3:Einfügen] [1:Mathebox]  Öffne nun auf der Startseite eine Calculatorseite  und schreibe das Programm. |  | |
| **Programmcode**  Define prgwurzel()=  Prgm  Request "Zahl eingeben:",a  b:=√(a)  Disp "Lösung:"  Disp b,"und",−b  Disp ""  Disp "IL={",b,",",−b,"}"  EndPrgm | Anmerkungen:  Das Wurzelzeichne findet man bei den Mathematischen Sonderzeichen.  Disp " " Erzeugt eine Leerzeile  Die Ausgabe als Lösungsmenge ist etwas umständlich. IL besteht aus einem I und einen L | |
| Nun kannst du wieder auf 1.1 die Note-Seite von eben klicken. Gehe auf die Mathebox und starte das  Programm. | | |
| Variationen, weiterführende Ideen: Quadratzahlen bzw. höhere Potenzen berechnen, Betrag ausgeben. | | |

# For-Schleifen verwenden.

|  |  |  |
| --- | --- | --- |
| **PRGPot** | | **Klasse 9** |
| Beschreibung: Das Programm berechnet Potenzen. Man gibt die Basis und die Hochzahl ein und das Programm berechnet die Potenz.  Lernziele: For-Schleife, Programmkommentare eingeben. | | |
| Beispiel. Bei komplizierteren Programmen empfiehlt es sich zunächst mal eine Beispielaufgabe zu rechnen bzw. ein Schaubild zu zeichnen. | | |
| Das Programm soll c = 25 ausrechnen.   1. Schritt: c = 2 ∙ 1 = 2 2. Schritt: x = 2 ∙ 2 = 4 3. Schritt: x = 2 ∙ 4 = 8 4. Schritt: x = 2 ∙ 8 = 16 5. Schritt: x = 2 ∙ 16 = 32 6. Schritt: x = 2 ∙ 32 = 64   Nun kann man schon ein Muster erkennen. | Allgemein formuliert: c = ak   1. Schritt: c = a ∙ 1 = 2 2. Schritt: c = a ∙ 2 = 4 3. Schritt: c = a ∙ 4 = 8   Der Wert a wird immer mit dem vorherigen Ergebnis multipliziert und das ganze k mal.  Daher lautet die Befehlszeile:  c := a ∙ c und das k mal | |
| **Programmcode**  Define prgpot()=  PrgmRequest "Zahl",aRequest "Hoch",bc:=1 © Setzt den Wert von c immer auf 1.  For k,1,b c:=a ∙ cEndFor  Disp "Ergebnis",cEndPrgm | | |
| Anmerkungen  © Dieses Zeichen ermöglicht es, das Programm zu kommentieren.  Sollte man aus Versehen oder absichtlich eine Endlosschleife programmiert haben, muss man lange auf den OnKnopf drücken. | | |
| Variationen, weiterführende Ideen:  -10mal einen bestimmten Satz schreiben.  - Programm zur Addition schreiben.  - Numerische Programme zur Wurze oder zum Logarithmus oder zur Nullstellenfindung schreiben. | | |

# While-Schleifen und Zufallszahlen verwenden.

|  |  |
| --- | --- |
| **PRGWurf** | **Klasse 7** |
| Beschreibung: Es wird solange gewürfelt, bis eine 6 erscheint. Dann ist das Spiel zu Ende.  Lernziele: While-Schleife, Zufallszahlerzeugen. | |
| Vorbemerkung: While Schleife ist eine Schleife, die solange ausgeführt wird, solange eine bestimmte Bedingung erfüllt ist. Die Abfrage für die Bedingung steht am Anfang vor der Schleife. Hier wird also solange gewürfelt, solange noch keine 6 gewürfelt wurde. Wird eine 6 gewürfelt, wird die Schleife nicht mehr ausgeführt und das Programm geht weiter. | |
| **Programmcode**  Define prgwurf()=  PrgmText "Wer als Erstes eine Sechs würfelt, hat gewonnen. Bereit?"  c:=1  While c≠6 Text "Würfeln!" Wait 1  c:=randInt(1,6) Disp "Wurf",cEndWhile  Disp "Gewonnen!!!"EndPrgm | |
| c := 1 soll verhindern, dass die Variable schon zufällig 6 ist, bevor gewürfelt wird.  Wait 1 wartet eine Sekunde bevor es weitergeht, um das Spiel zu verlangsamen. | |
| Variationen, weiterführende Ideen:   * Spiel für zwei Spieler: Wer als erster eine bestimmte Summe erreicht. Würfelt ein Spieler eine 6 bekommt er eine bestimmte Punktzahl abgezogen. * Mäxchen * Roulette | |

# Programme mit anderen Applikationen verbinden.

|  |  |  |
| --- | --- | --- |
| **PRGLinkfunk1** | | **Klasse 8** |
| Beschreibung: Per Zufall bestimmt das Programm m und b. Anschließend wird die Funktion in einem Koordinatensystem gezeichnet.  Lernziele: Programme in andere Applikationen zeichnen verwenden, Seiten gruppieren. | | |
| Vorbereitung.  Öffne eine Graph-Seite. Dann öffne eine Calculatorseite. Gehe auf [Menu] [9: Programmeditor].  Nun ist die Calculatorseite geteilt. Diese Gruppierung wollen wir aufheben und die ersten beiden Seiten gruppieren. Hierzu gehe auf [CTRL] und Pfeil nach oben. Unter [dok] und [5:Seitenlayout] kann man Gruppierungen aufheben und neu zusammensetzen. Das Dokument sollte wie folgt aussehen:  Letzte Vorbereitung: Im Fenster „Graph“ die [TAB] Taste klicken und f1 anklicken, damit die Funktion auch angezeigt wird. | | |
| Das Programm besteht aus zwei Seiten. | | |
|  |  | |
| **Anmerkungen**  Möchte man auch noch weitere Applikationen verbinden, so ist dies leicht möglich. Man kann man z.B. eine Tabelle öffnen und sich die Werte der Funktion dort anzeigen lassen. | | |
| Variationen, weiterführende Ideen:   * Quadratische Funktionen zeichnen lassen | | |

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Skriptende

# 