UNIVERSIKUM

Wahlfachkurse Schuljahr 2010/11

Ferienkurs 2. bis 6. Mai. 2011
Schulhaus Birch

Programmieren mit SCRATCH

An fünf Tagen wirst du mit Scratch in die Geheimnisse des Programmierens eingeweiht. Du bewegst Kobolde auf der Bühne und erfindest mit ihnen lustige Spiele. Du lernst, Folgen von Computerbefehlen zu planen, kreativ und vorausschauend zu denken.

Die Idee

- Kinder sind spielerisch
- Kinder sind neugierig: sie wollen wissen und verstehen

Mit jeder Übung schaffen die Schülerinnen und Schüler ein vorzeigbares Ergebnis, über das sie sich freuen können, und auf das sie stolz sein werden. Sie sollen dabei aber auch ganz nebenbei das systematische Programmieren entdecken und ein tieferes Verständnis erwerben für das, was sie tun.

Auf einer Forschungsreise am Computer erhalten die Schülerinnen und Schüler einen Einblick in die Tätigkeit des Informatikers und entdecken fundamentale Begriffe und Techniken der Programmentwicklung:

- Methode, Algorithmus und Programm
- Grundlagen des korrekten logischen Denkens
- Variable, arithmetische Operation, logische Operation
- Sequentieller Ablauf, Iteration, Selektion, Unterprogramm, interaktives System, Multitasking

Herzlichen Dank an Prof. J. Hromkovič, Giovanni Serafini und Bernd Gärtner vom Ausbildungs- und Beratungszentrum für Informatikunterricht (ABZ) der ETH Zürich für ihre Unterstützung, die kritische Durchsicht meiner Manuskripte und ihre wertvollen Hinweise. Einige Aufgaben haben wir vom Kurs "Wie bunte Schildkröten laufen lernen" vom ABZ übernommen oder uns davon inspirieren lassen.

Liebe Schülerin, lieber Schüler

Du hast dich für den Kurs Programmieren mit Scratch angemeldet, vielleicht weil du gerne selbst Spiele programmieren willst, oder weil du dich für Informatik interessierst. Programmieren gehört zum Werkzeug eines jeden Informatikers, auch wenn das Programmieren alleine noch keinen Informatiker ausmacht.

Dieser Kurs ist eher das, was man eine Werkstatt nennt. Ich werde selten vor der Klasse stehen und Erklärungen abgeben. Ihr werdet erleben, was Programmieren ist. "*Learning by doing*", übersetzt: "tu etwas und lerne dabei".

Im ersten Projekt wirst du mit einfachen Übungen Scratch kennen lernen. Mit dem Projekt 2 beginnt die eigentliche Programmierwerkstatt: in jedem Projekt brauchst du, was du vorher gelernt hast. Am Schluss kannst du eigene Projekte entwickeln, wie es dir gefällt.

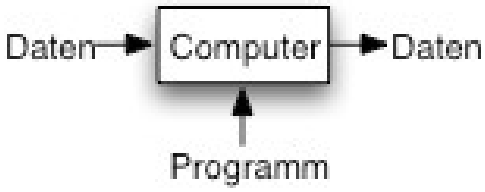
Wir halten uns an die englische Version von Scratch, denn Englisch ist die Sprache der Informatik. Es ist ja auch eure Sprache: *gamen*, *chatten*, *fooden*, *shoppen*, *cool* ...

Anspruchsvollere Aufgaben sind mit  gekennzeichnet. Sammle Mäuse!

François Louis Nicolet

Programme

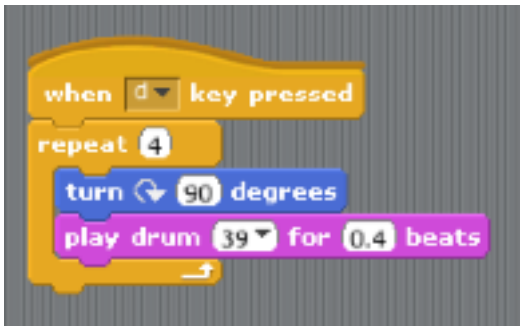
Einen Computer kann man für beliebige Aufgaben mit Daten einsetzen und **selbständig arbeiten** lassen.
Ein Programm beschreibt, wie eine Aufgabe gelöst werden soll.



Ein **Programm** ist eine **Folge von Computerbefehlen**.
Ein Computerbefehl ist eine Anweisung, die der Computer ausführen kann.

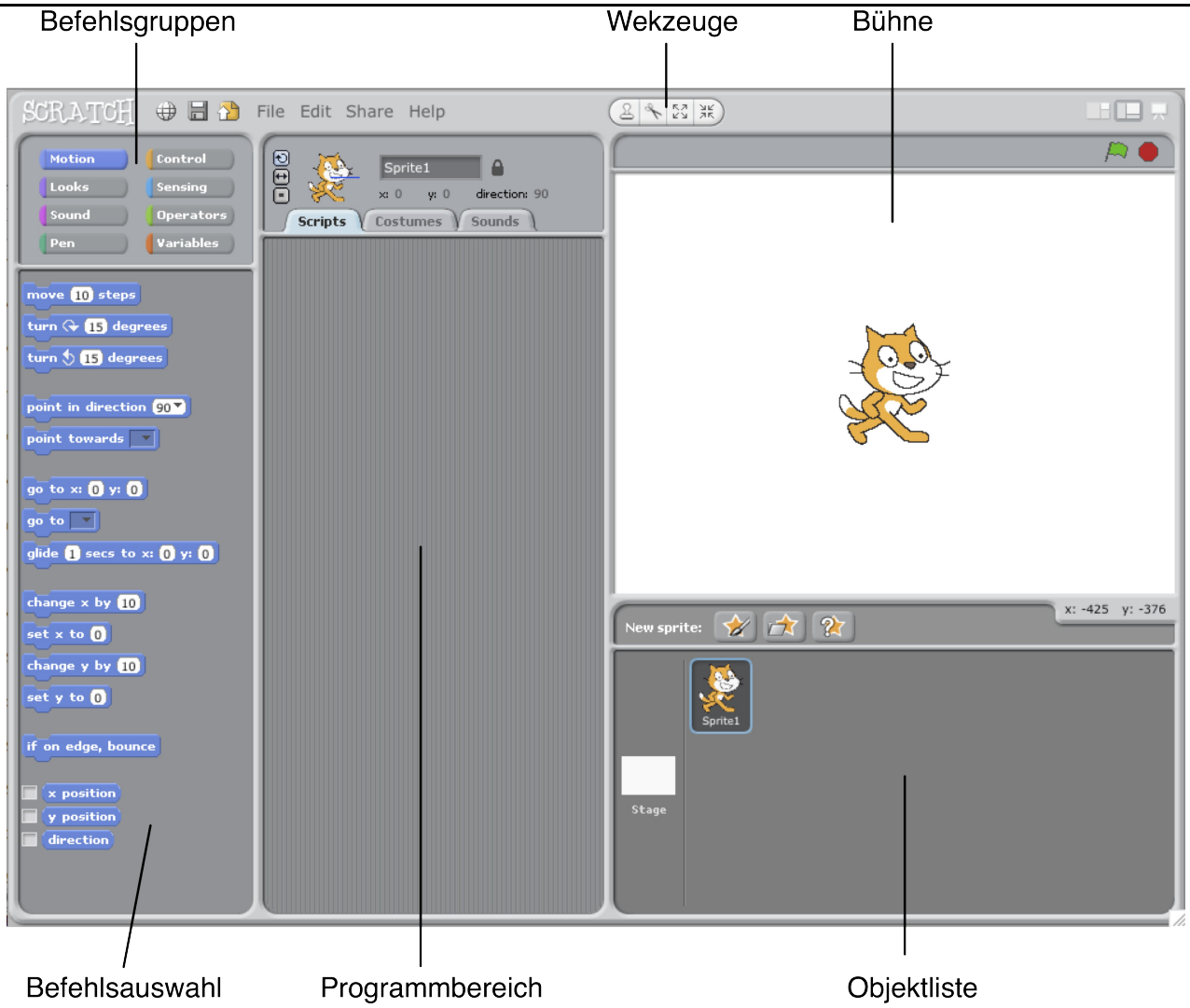
Der Computer kennt nur wenige Befehle. Alle komplizierten Tätigkeiten, die der Computer ausführen soll, müssen wir aus diesen einfachen Computerbefehlen zusammensetzen. Das ist nicht immer einfach. Es gibt Programme, die aus Millionen von Befehlen zusammengesetzt sind. In dieser Werkstatt entdeckst du, wie man dabei die Übersicht nicht verliert.

Scratch



Scratch wurde vom Massachussets Institute of Technology (MIT) entwickelt, um Acht- bis Sechzehnjährige in die Programmierung einzuführen. Mit Scratch baut man Programme durch Zusammensetzen von Bausteinen, so wie mit Teilen eines Puzzle oder Legosteinen. In den meisten Computersprachen *schreibt* man die einzelnen Computerbefehle als *Text*. Mit Scratch *schiebt* man die Computerbefehle mit der Maus untereinander.

Scratch kannst du von der Adresse
`http://scratch.mit.edu/download`
herunterladen und auf deinem Computer installieren.



Das Scratch-Programmfenster

Die **Bühne** ist der Platz, wo Geschichten, Spiele und Animationen aufgeführt werden. Die Schauspieler auf der Bühne heissen ***Sprite***, lustige, zuweilen auch zu bösen Streichen aufgelegte Zwerge oder Hausgeister, Kobolde. Du kannst das Aussehen der Sprites verändern. Du kannst ihnen befehlen sich zu bewegen, Musik zu spielen oder mit anderen Sprites zu spielen. Jedem Sprite kannst du einen Namen geben. Dem Sprite, der wie eine Katze aussieht, wollen wir den Namen ***Büsi*** geben.

Um einen Sprite zu programmieren, schiebst du Befehlsblöcke aus der **Befehlsauswahl** in den **Programmbereich**. In der Blockpalette sind die Blöcke in 8 Gruppen eingeteilt.

Bewegung
Aussehen
Klang
Malstift
Steuerung
Fühlen
Operationen
Variablen

Projekt 1: Auf der Bühne

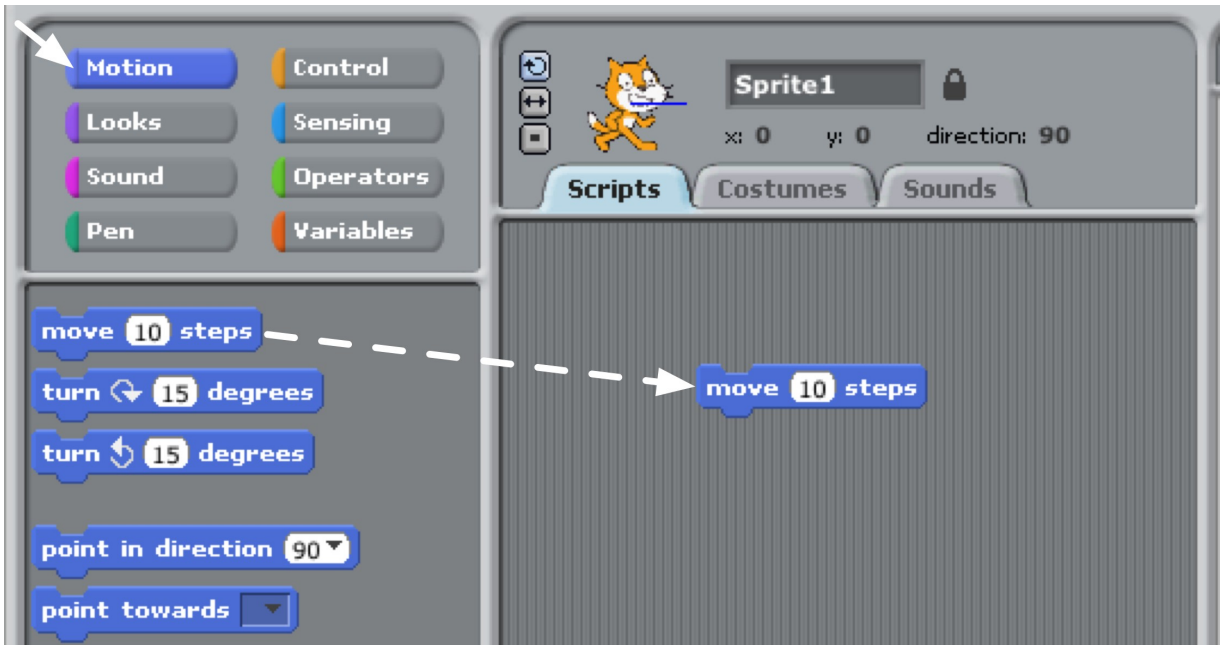
Mit diesem Projekt entdeckst du, wie man mit Scratch Bewegung auf die Bühne bringt.

Aufgabe 1: Erste Schritte

Schiebe den Befehl 

(gehe 10 Schritt vorwärts) aus der Gruppe 

(Bewegung) in den Programmbereich:



Klicke auf den Befehl, den du in den Programmbereich geschoben hast. Was geschieht auf der Bühne?

Klicke mehrmals auf den Befehl und beobachte! Du kannst Sprite mit der Maus wieder in die Mitte der Bühne zurückschieben, bevor er die Bühne verlassen hat.

Aufgabe 2: Füge Musik zu

Schiebe den Befehl 

(spiele Schlagzeug) in den Programmbereich, und befestige ihn unter dem **move**-Befehl:

Klicke, sieh und höre!

Du kannst verschiedene Schlagzeuge auswählen, wenn du auf das kleine schwarze Dreieck klickst:

Aufgabe 3: Das grüne Fähnchen

Schiebe den Befehl 

über den ersten Befehl auf der Bühne. Jetzt wird das Programm mit Klick auf das grühne Fähnchen oben rechts über der Bühne gestartet.

Der Befehl 

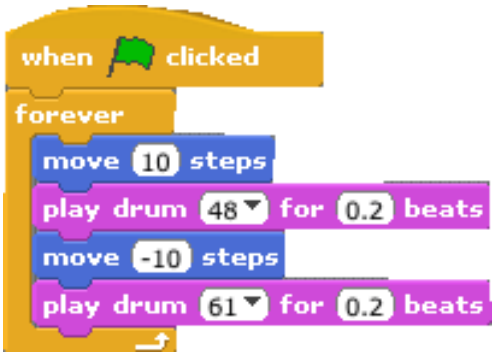
heisst in Scratch *Programmhut*.

Die Vorteile wirst du mit den folgenden Projekten sehen.

Aufgabe 4: Tanze!

Füge einen weiteren **move**-Befehl an, diesmal mit **-10** anstelle von **10**. Füge noch ein **play drum**-Befehl, mit einem anderen Schlagzeug an. Klicke!

Aufgabe 5: Ohne Ende



Hole den **forever**-Befehl (*forever* = ewig, immer wieder) aus der Gruppe **Control**. Du kannst ihn so schieben, dass er die anderen Befehle umklammert. Sprite wiederholt die vier Befehle immer wieder.

Klicke auf das grüne Fähnchen, um den Tanz zu starten. Du kannst die Vorstellung stoppen mit Klick auf dem roten Knopf  über der Bühne neben dem grünen Fähnchen.

Aufgabe 6: Start mit Taste

In der Gruppe **Looks** findest du den Befehl 

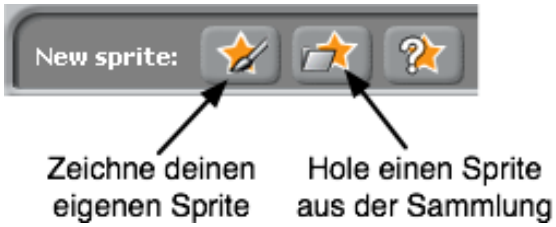
Schiebe ihn in den Programmbereich. Darüber schiebst du den Programmhut:

Drücke nun die Leertaste (*space key*) deiner Tastatur und beobachte!

Statt *space* kannst du eine andere Taste wählen.

Aufgabe 8: Ein zweiter Sprite

Unter der Bühne findest du drei Flächen mit je einem Stern. Damit bringst du neue Sprites auf die Bühne:



Ganz oben über der Bühne findest du  . Mit der Schere kannst du *Büsi* entfernen.

Aufgabe 9: Neuer Tanz


m1

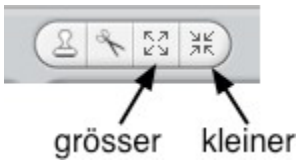
Erfinde einen neuen Tanz mit neuer Musik und, wenn du willst, mit einen anderen Sprite!

Projekt 2: *Büsi* zeichnet

*Im Projekt 1 hast du gespielt und dabei entdeckt, was man mit Scratch machen kann. Und jetzt geht es richtig los! Du wirst entdecken, was Programmieren ist, und wie man **richtig** programmiert. Viel Erfolg!*

Aufgabe 1

Für unsere Zwecke ist *Büsi* zu gross. Mach es kleiner: Oben in der Mitte des Scratch-Fensters findest du kleine Flächen (engl. *buttons*). Sie haben die folgende Bedeutung:

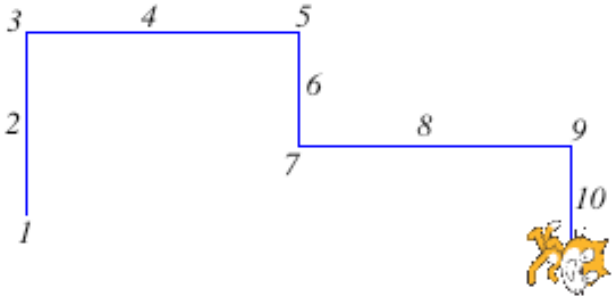


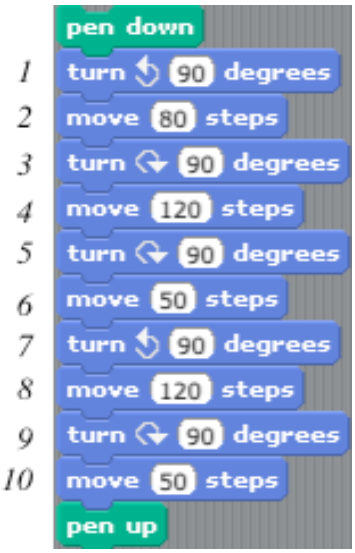
Mit dem Befehl pen down aus der Befehlsgruppe Pen senkt *Büsi* seinen Zeichenstift auf das Papier und mit pen up hebt er den Stift wieder. So wie du es beim Zeichnen und Schreiben auch tust.

Aufgabe 2

Schiebe Befehle in den Programmbereich, um das folgende Programm zusammenzusetzen und klicke es an, um es zu starten.

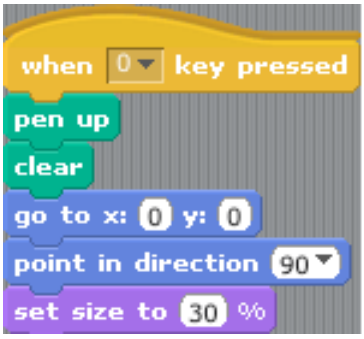
Hast du folgendes Bild gezeichnet?





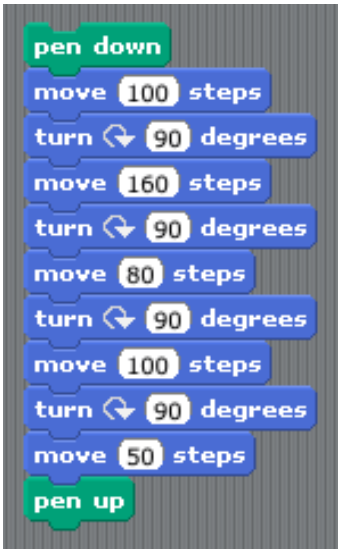
Aufgabe 3

Für die folgenden Aufgaben brauchst du ein Programm, welches alles Gezeichnete löscht und *Büsi* wieder in die Mitte der Bühne setzt:



Programm wird mit der Taste "0" gestartet
hebe den Zeichenstift hoch
lösche alle Zeichnungen auf der Bühne
gehe in die Mitte der Bühne
schaue nach rechts
verkleinere dich

Dieses Programm wirst du in späteren Aufgaben wieder brauchen. Am Besten speicherst du alle deine Projekte.



Aufgabe 4

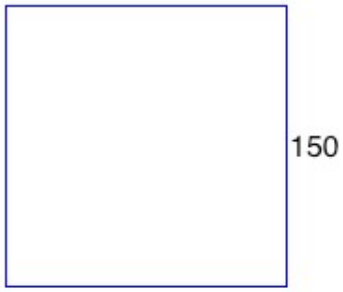
Setze das Programm (links) zusammen und führe es aus.

Zeichne hier unten das entstandene Bild und beschreibe (wie in Aufgabe 1 in diesem Projekt) was jeder Befehl verursacht hat.

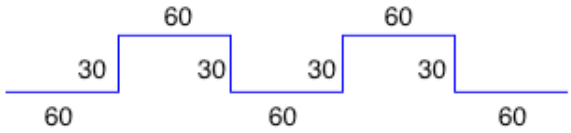
Aufgabe 5 m2

Setze Programme zusammen, die die folgenden Bilder zeichnen. Die Längenangaben sind **steps** (Schritte) des Befehls **move ☐ steps**.

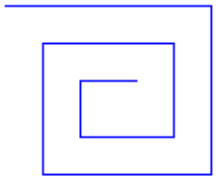
a)



b)

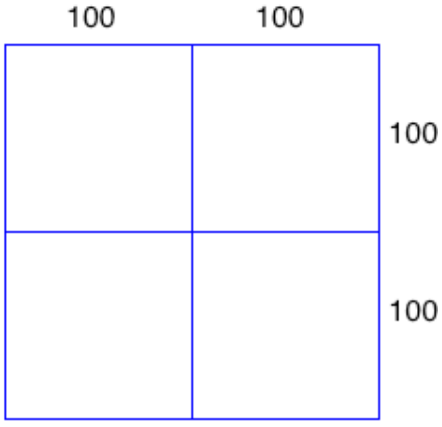


c)



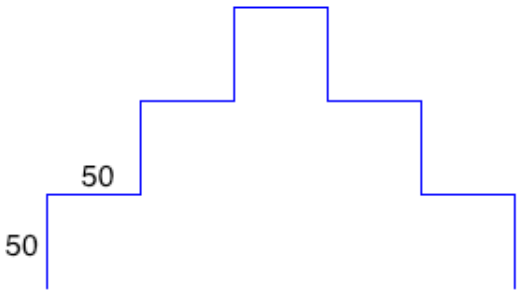
Die Grösse darfst du selbst wählen

d)



Aufgabe 6

Setze ein Programm zusammen, das dieses Bild zeichnet:



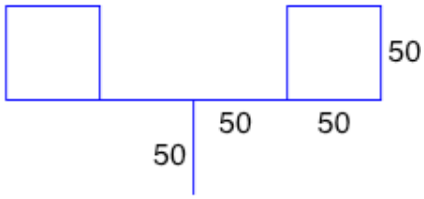
Schaffst du es, dein Programm so zu ändern, dass es nur die Befehle 

und 

verwendet?

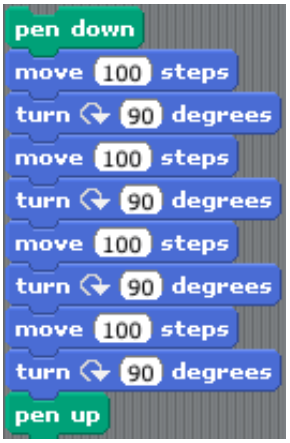
Aufgabe 7  m3

Büsi will dieses Bild zeichnen. Kannst du helfen?



- Einen Computer kann man für beliebige Aufgaben einsetzen und **selbständig arbeiten** lassen. Wie der Computer die Aufgabe Schritt für Schritt lösen soll, ist in einem Programm beschrieben.
- Ein **Programm** ist eine Folge von Computerbefehlen.
- Ein **Computerbefehl** ist eine Anweisung, die der Computer ausführen kann.

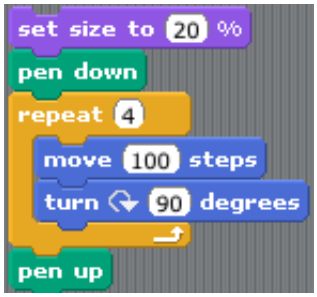
Projekt 3: Wiederholen



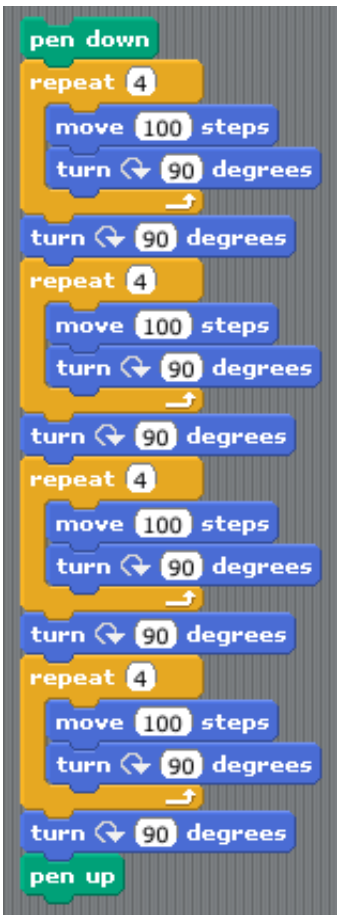
Dieses Programm aus Projekt 2 zeichnet ein Quadrat.

Die beiden Befehle

werden viermal hintereinander ausgeführt. Es wäre einfacher, dem Computer zu befehlen, diese zwei Befehle viermal zu wiederholen. Genau das wollen wir tun:



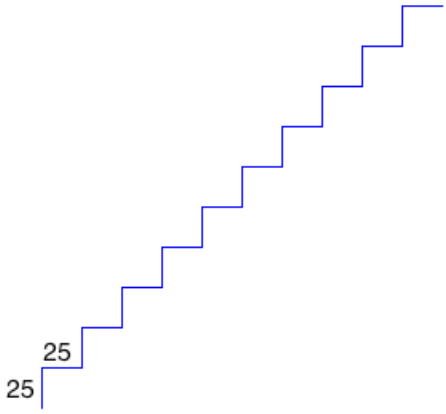
repeat heisst wiederholen. Die beiden Befehle **move** und **turn**, die innerhalb der **repeat**-Klammer eingefügt sind, führt Sprite viermal nacheinander aus.



Aufgabe 1

Was zeichnet dieses Programm?

Kannst du es kürzer machen?



Aufgabe 2: Treppen

a) Zeichne eine Treppe mit 10 Stufen der Grösse 20.

b) Zeichne eine Treppe mit 5 Stufen der Grösse 50.

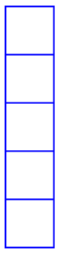
c) Zeichne eine Treppe mit 20 Stufen der Grösse 10.

Aufgabe 3 m4

Zeichne die folgenden Bilder:

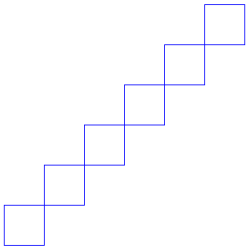
a) Jedes Quadrat mit Seitenlänge 20

b) Jedes Quadrat mit Seitenlänge 30



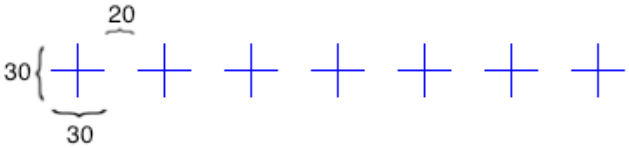
Aufgabe 4 m5

Zeichne mit einem Programm das folgende Bild (jedes Quadrat mit Seitenlänge 60:



Aufgabe 5 m6

Setze ein Programm für das folgende Bild zusammen:



Aufgabe 7: Tanz

In dieser Aufgabe zeige ich dir, wie du Sam zum Tanzen bringen kannst.

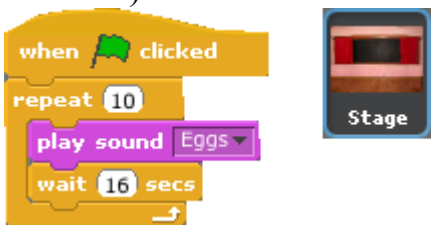
Du brauchst einen *Tänzer*. Du klickst auf den Sprite in der Sprite-Liste, dann auf den mittleren Stern  oberhalb der Sprite-Liste und öffnest den Ordner **People**. Wähle z.B. *Sam*. Über dem Programmbereich ersetzst du den Namen **Sprite 2** durch **Sam**. Den bisherigen Sprite *Büsi* kannst du löschen.

Du brauchst auch ein *Bühnenbild*. Klicke die mit **Stage** (Bühne) angeschriebene Fläche, links von der Sprite-Liste. Über dem Programmbereich klickst du auf **Backgrounds** (Hintergründe), dann auf **Import**. Jetzt kannst du den Ordner **Indoors** öffnen und ein geeignetes Bühnenbild aussuchen. Nimm **Chalkboard** (Wandtafel).

Sam soll jetzt tanzen! Klicke auf Sam in der Sprite-Liste. Wir setzen das *Programm 1* (siehe unten) gemeinsam zusammen.

Musik brauchst du auch noch. Die gehört zur Bühne. Klicke auf **Stage** neben der Sprite-Liste. Dein Musik-Programm ist das *Programm 2* (siehe unten).

Programm 1: Sam Tanzt



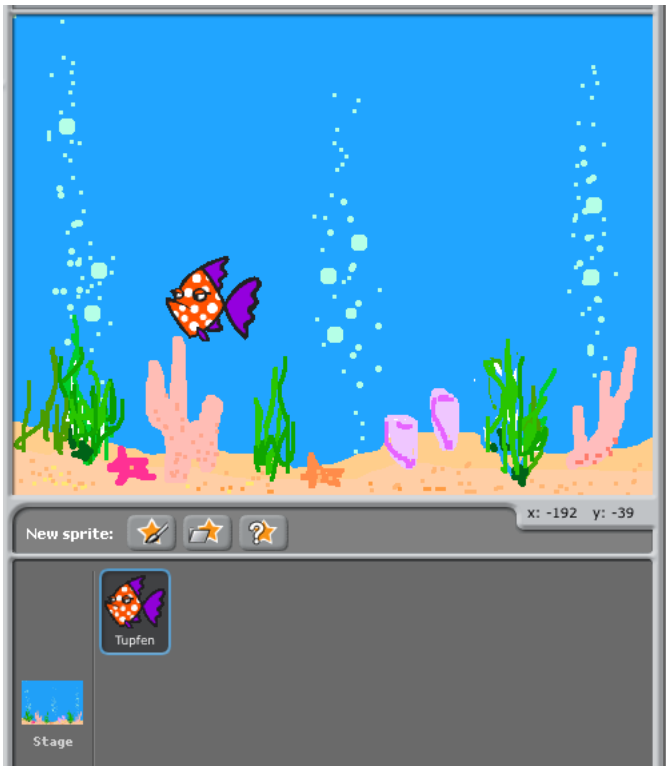
Programm 2: Musik auf der Bühne

Mit Klick auf das grüne Fähnchen werden zwei Programme gleichzeitig gestartet und laufen gleichzeitig ab. In der Informatik nennt man das Multiprocessing.

Aufgabe 8: Dein eigener Tanz  m7

Entwickle einen eigenen Tanz. Suche dir eine Tänzerin oder einen Tänzer nach deinem Geschmack aus. Erfinde Tanzschritte und die Musik dazu!

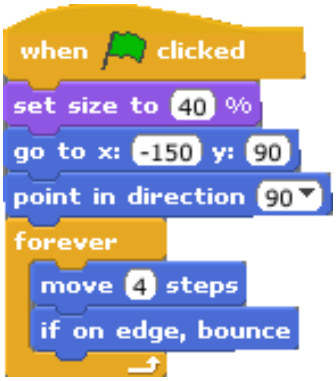
Aufgabe 9: Aquarium



Klicke auf **Stage** unter der Bühne, auf **Backgrounds** und **Import** über dem Programmbereich und hole den Aquarium-Hintergrund aus dem Ordner **Nature**.

Hole einen Fisch aus der Sammlung: Klicke auf den Fisch unter der Bühne, dann auf den mittleren Stern, ebenfalls unter der Bühne und suche einen Fisch im Ordner **Animals**. Gib dem Fisch einen Namen und lösche *Büsi*.

Der Fisch soll nach folgendem Programm schwimmen:

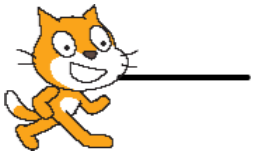


Hole jetzt weitere Fische aus der Sammlung, gib jedem einen Namen und setze für jeden ein eigenes Schwimmprogramm zusammen.

Projekt 4: Winkel und Vielecke

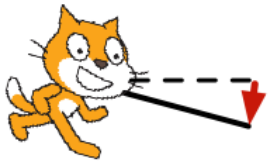
Drehen

Sprite läuft immer in die Richtung, in die er gerade schaut.



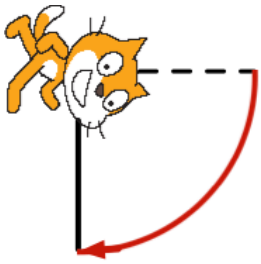
Mit dem Befehl 

dreht sich Sprite um 15° nach rechts.



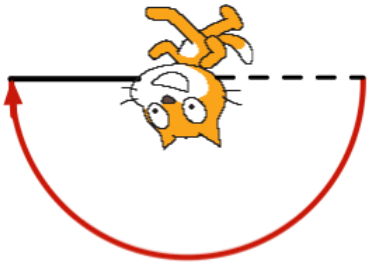
Mit dem Befehl 

dreht sich Sprite um 90° nach rechts. Dies entspricht einem Viertelkreis



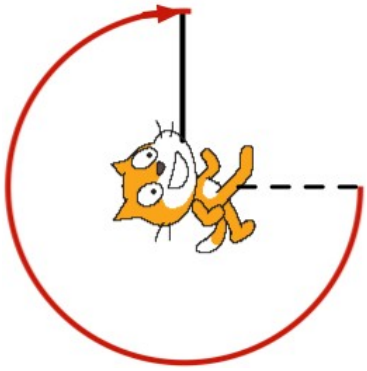
Mit dem Befehl 

dreht sich Sprite um 180° nach rechts. Dies entspricht einer halben Umdrehung.



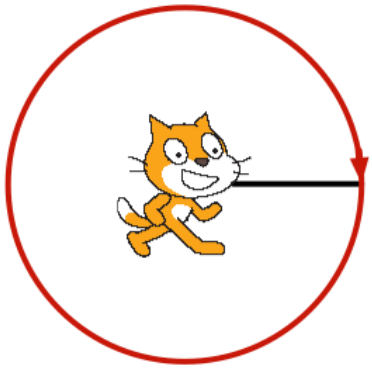
Mit dem Befehl 

dreht sich Sprite um 270° nach rechts. Dies entspricht einer dreiviertelsdrehung.



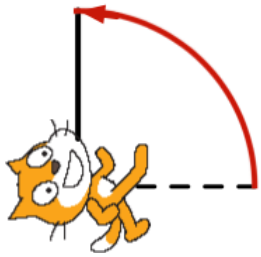
Mit dem Befehl 

dreht sich Sprite um 360° nach rechts. Dies entspricht einer ganzen Umdrehung.



Mit dem Befehl

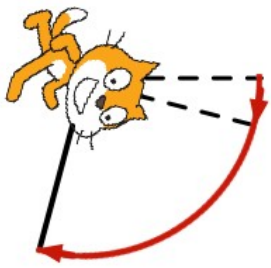
dreht sich Sprite um 90° nach links.



Beachte, dass sich das Drehen nach links und rechts auf die Sicht von Sprite bezieht, wie das folgende Beispiel mit den Befehlen 

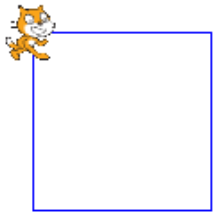
und 

zeigt.



Aufgabe 1: Viereck und Sechseck

Im Projekt 3, Aufgabe 1 hast du ein Quadrat gezeichnet:



Am Anfang schaut *Büsi* nach rechts. Nachdem es das Quadrat gezeichnet hat, schaut es wieder nach rechts. Es hat beim Zeichnen eine ganze Umdrehung gemacht. Eine ganze Umdrehung, das sind 360°. In jeder der vier Ecken hat er sich um 90° nach rechts gedreht: eine Viertelsdrehung. *360° : 4 = 90°*

Wenn *Büsi* ein regelmässiges 6-Eck zeichnen soll, dann muss es sich in jeder der 6 Ecken um eine Sechstels-Drehung drehen: *360° : 6 = 60°*.

Setze die Programme zusammen für ein Viereck und für ein Sechseck!

Aufgabe 2: Regelmässiges Zwölfeck

Setze ein Programm für ein regelmässiges 12-Eck zusammen!

Aufgabe 3: Sprite rechnet

Sprite soll nun ein regelmässiges 7-Eck zeichnen. Das hat 7 Ecken und 7 Seiten. In jeder Ecke dreht sich Sprite um

360° : 7 = ?

Du hast keinen Taschenrechner und Mühe mit dem Dividieren? Lass doch den Computer die Division ausführen! Du findest in der Befehlsgruppe Operators die Rechenoperation 360 / 7. Diese Rechenoperation kannst du im **turn … degrees**-Befehl einsetzen:

Setze das Programm zusammen und führe es aus!

Aufgabe 4  m8

a) Zeichne den Stern mit 8 Strahlen der Länge 150.

b) Gelingt dir ein Stern mit 16 Strahlen?

Aufgabe 5  m9

Zeichne ein regelmässiges 360-Eck. Wähle die Seitenlänge so, dass das Vieleck ganz auf die Bühne passt. Auf dem Bildschirm sieht das 360-Eck aus wie ein Kreis!

Ein Kreis ist ein regelmässiges Vieleck mit sehr vielen Ecken.

Projekt 5: Bedingung

Bedingung

Frage: Ist es schon zu spät? – *Antwort:* nein.
Frage: Sind deine Eltern einverstanden? – *Antwort:* ja.
Frage: Hast du genug Geld dabei? – *Antwort:* ja.
Frage: Bist du älter als 14? – *Antwort:* nein.

Diese Fragen kann man nur mit *ja* oder *nein* beantworten. Eine solche Frage kann man als *Bedingung* verwenden:

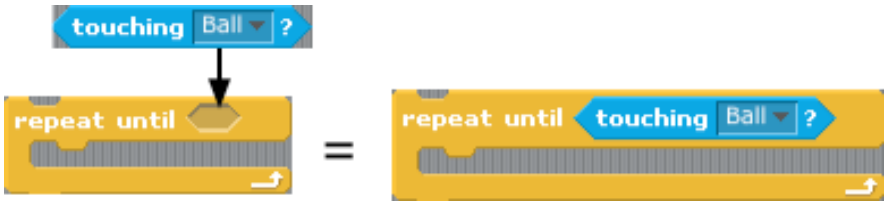
Wenn deine Eltern einverstanden sind, darfst du zu uns kommen (sonst nicht).
Wenn es schon zu spät ist, bleiben wir zu Hause (sonst nicht).
Wenn du genug Geld hast, kannst du dir den Computer kaufen (sonst nicht).
Wenn du älter bist als 14, darfst du ins Kino (sonst nicht).

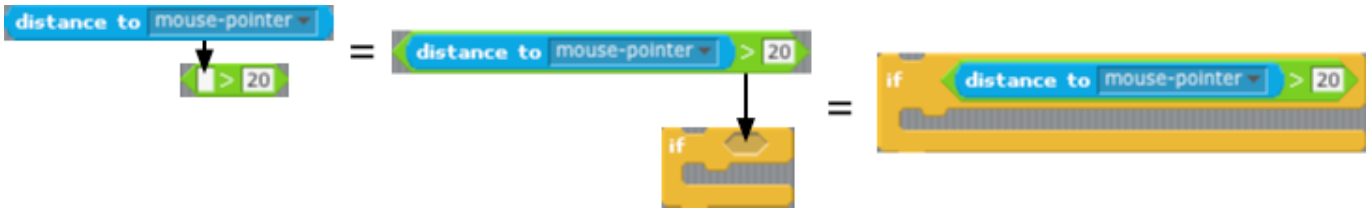
Eine Bedingung trifft entweder zu oder nicht: sie ist entweder *wahr* oder *falsch*. In der Programmierung heissen Befehle, die an eine Bedingung geknüpft sind, *bedingte Befehle*. Sie werden nur ausgeführt, wenn die Bedingung erfüllt (*wahr*) ist.

In allen Programmiersprachen gibt es die Möglichkeit, einen oder mehrere Befehle bedingt auszuführen.

Einige Bedingungen:

So können sie zu bedingten Anweisungen zusammengesetzt werden:





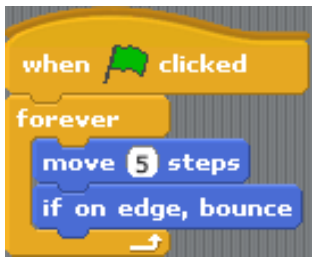
Mit Bedingungen kannst du Spiele programmieren, in denen Sprites auf Berührung eines anderen Sprites oder auf Tasten reagieren.

Aufgabe 1: Bühnenrand

Im Projekt 1, Aufgabe 1 haben wir den Befehl **move 10 steps** in den Programmbereich gezogen. Das tun wir wieder, und Klicken mehrmals hintereinander auf den Befehl. Dabei musst du aufpassen, dass *Büsi* nicht über den Bühnenrand wegläuft. Das kann man mit dem Befehl 

(wenn du am Bühnenrand bist, kehre um) verhindern. Das ist ein *bedingter Befehl*: *Büsi* kehrt nur um, wenn es am Bühnenrand ist, sonst nicht.

Setze das folgende Programm zusammen, führe es aus, und beobachte, wie sich *Büsi* verhält!



on edge ist die Bedingung, **bounce** ist der bedingte Befehl.

Mit diesem Programm zeige ich dir noch etwas. Über dem Programmbereich findest du *Büsi* mit seinem Namen und links davon drei kleine Quadrate zum anklicken. Mit diesen kannst du bestimmen, wie sich *Büsi* am Bühnenrand verhalten soll.

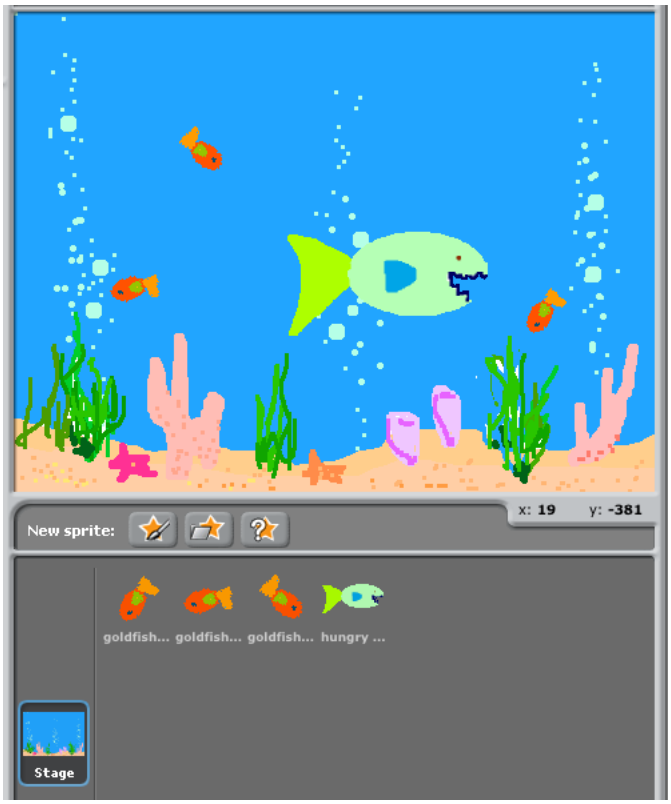

Er dreht sich ganz um und fährt auf dem Kopf zurück,
Er dreht sich um 180° und und fährt auf den Füssen zurück,
Er dreht sich überhaupt nicht um und fährt rückwärts zurück.

Probiere es aus!

Aufgabe 2: Jagd im Aquarium

a) Du brauchst ein Aquarium wie im Projekt 3, mit drei kleinen Goldfischen und einem grossen grünen Raubfisch. Wähle für dieses Projekt genau diese Fische.

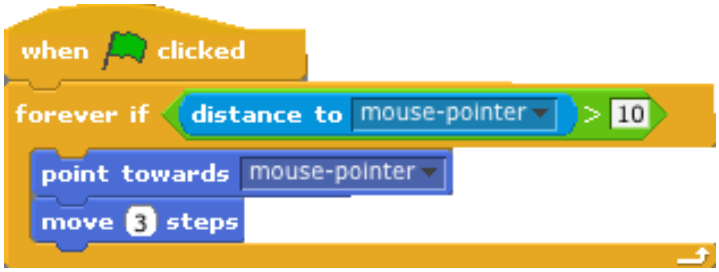
b) Für jeden der kleinen *Goldfische* setzst du das folgende Programm zusammen:



Damit jeder Goldfisch auf seine Art schwimmt, benutzst du Zufallszahlen:
pick random -5 to 5 "nimm eine beliebige Zahl zwischen -5 und 5" (*random* = zufällig).

c) Und nun zum *grünen Raubfisch*. Auch für ihn setzst du ein Schwimmprogramm zusammen.

Du ***steuerst*** den grünen Raubfisch mit dem Mauspfeil: point towards mouse-pointer (ziele in Richtung Mauspfeil), aber nur solange er nicht zu nahe an den Mauspfeil kommt: distance to mouse-pointer.



Es ist wichtig, dass du den Befehl forever if distance to mouse-pointer > 10 gut verstehst!

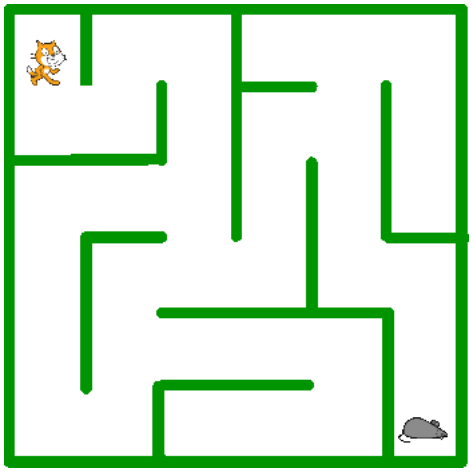
d) Der grüne Raubfisch will die kleinen Goldfische fressen.

Wenn der Raubfisch mit seinen schwarzen Zähnen einen Goldfisch berührt, frisst er ihn: color ■ is touching ■ ? (die Farbe ■ berührt ■ ?). Dann verschwindet der Goldfisch. Nach 3 Sekunden erscheint er an einem zufälligen Ort (**pick random**). Das führt zu folgendem Programmteil für die Goldfische:

Füge diesen Programmteil in jedes Goldfischprogramm und teste, ob es tut, was es soll!

In dieser Aufgabe *steuerst* du den Raubfisch mit der Maus.
Computer werden oft zum *Steuern* von Abläufen eingesetzt.

Aufgabe 3: Labyrinth m10



Male als Bühnenhintergrund ein Labyrinth. Du brauchst zwei Sprites: das uns bekannte *Büsi* und eine Maus. Am Anfang ist *Büsi* oben links, und die Maus sitzt unten rechts (Bild rechts).

Das Spiel beginnt mit Klick auf das grüne Fähnchen. *Büsi* und Maus gehen in ihre Startpositionen.

Die Maus springt im Labyrinth umher. Sie läuft geradeaus (

) bis sie an eine Wand stösst (

). Da dreht sich sich (

) und läuft geradeaus weiter bis sie wieder an eine Wand stösst.

Wenn sie von der Katze erwischt wird (

), verschwindet sie und das Spiel ist zu Ende.

Büsi steuerst du mit den vier Pfeiltasten 

.

Büsi soll die Maus erwischen. Wenn *Büsi* an eine Wand stösst, muss es zurück in die Startposition oben links.

Aufgabe 4: Dein Labyrinth m11

Erweitere das Spiel!

> *So wie du im Labyrinth die Katze Büsi mit den Pfeiltasten steuerst, werden Weltraumrakten, Flugzeuge (Autopilot!), Maschinen, Produktionsabläufe und vieles mehr mit Computern gesteuert.*

Projekt 6 Variable

Aufgabe 1: *Büsi* zählt

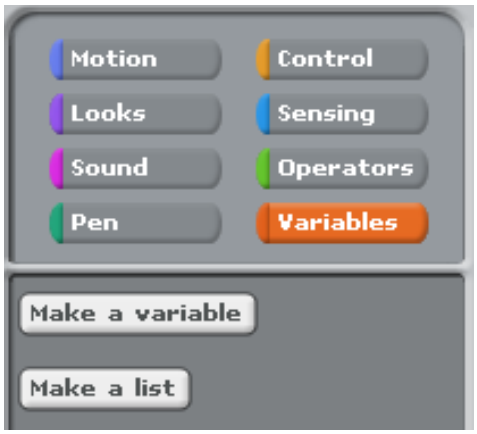
Das ist eine ganz einfache Aufgabe: *Büsi* soll zählen, wie oft es angeklickt wird. Dafür brauchst du einen ***Zähler***. Das Spiel soll damit beginnen, dass der Zähler auf **0** gesetzt wird. Dann soll bei jedem Klick auf *Büsi* der Zähler um **1** erhöht werden.

Der Zähler zeigt also bei Spielbeginn **0**, dann bei jedem Klick auf *Büsi* **1**, **2**, **3,** ...

In der Programmierung nennt man so etwas eine Variable. Und da in einem Programm oft mehrere Variablen vorkommmen können, gibt man jeder Variablen einen *Namen*.

> Eine Variable hält eine Zahl fest.
> Eine Variable hat einen Namen.

In der Befehlsgruppe **Variables** klickst du auf **Make a variable** (mache eine Variable).



Es erscheint das Fenster **Variable name?** Gib ihr den Namen **score** – oder wenn es dir besser gefällt **Punkte**.
Klicke **OK**.

Setze für *Büsi* zwei Programme zusammen.

Das erste Programm wird mit Klick auf dem grünen Fähnchen gestartet und setzt den Zähler auf **0**.

Das zweite Programm wird mit Klick auf *Büsi* gestartet und erhöht den Zähler um **1**.

Aufgabe 2: Labyrinth

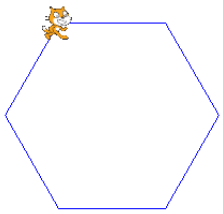
Du kannst das Labyrinth-Programm von Projekt 5 erweitern, damit es zählt, wie oft *Büsi* die Maus erwischt. Wohin gehört der Befehl 

und wohin gehört der Befehl 

?

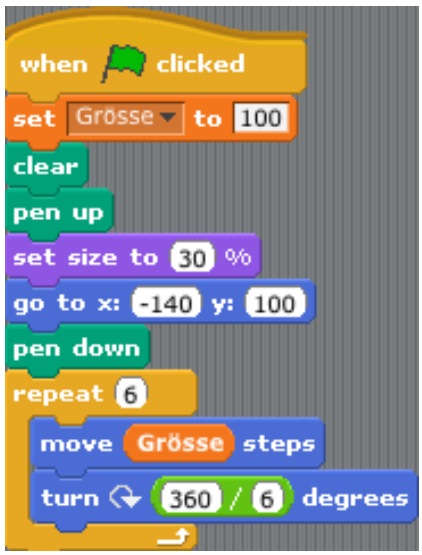
Aufgabe 3: Ein Sechseck

Im Projekt 4 hast du *Büsi* ein Sechseck zeichnen lassen. *Büsi* läuft einen 100er-Schritt und dreht anschliessend um eine Sechsteldrehung nach rechts. Das wiederholt es sechmal.



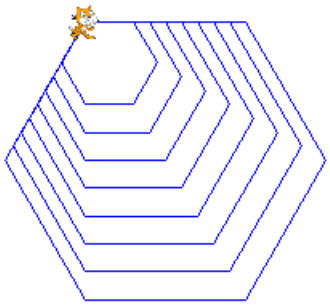
Der Befehl **move 100 steps** bestimmt die Grösse des Sechsecks. Um verschieden grosse Sechsecke zu zeichnen, ersetzst du die **100** durch eine Variable, der du den Namen **Grösse** gibst.

Jetzt kannst du verschieden grosse Sechsecke zeichnen.



Aufgabe 4: Geschachtelte Sechsecke m12

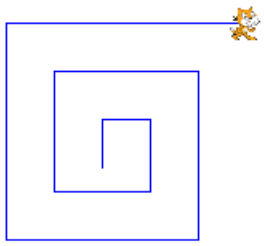
Büsi soll mehrere ineinandergeschachtelte Sechsecke zeichnen. Du brauchst zwei Variablen: **Grösse** bestimmt die Grösse (die Seitenlänge) eines Sechsecks (wie in der Aufgabe 3) und **Anzahl** bestimmt die Anzahl ineinandergeschatelter Sechsecke. Die beiden Variablen werden, wie in Aufgabe 3, am Anfang des Programmes zu den gewünschen Werten gesetzt. Dann zeichnet das Programm die Sechsecke indem es die Wiederholung **repeat 6** wiederholt. Das erste Sechseck hat die Grösse wie in **Grösse** angegeben, dann wird für das nächste Sechseck der Variablen **Grösse** 10 dazuaddiert.



Setze das Programm zusammen!

Aufgabe 5 m13

Setze ein Programm zusammen, das diese Schnecke zeichnet. Das Aussehen der Zeichnung wird durch Variablen bestimmt.



Projekt 7: Unterprogramme

Sprites können einander zurufen. So kann ein Programm ein anderes Programm veranlassen etwas zu tun, oder ein Sprite kann ein anderes Sprite veranlassen etwas zu tun.

So kann eine grössere Aufgabe in Teilaufgaben aufgeteilt werden. Die einzelnen Teilprogramme werden damit einfacher und übersichtlicher.

Teilaufgaben werden durch **Unterprogramme** gelöst.

Aufgabe 1: Fischjagd im Aquarium  m14

Im Projekt 5 hast du ein Aquarium programmiert, mit drei Goldfischen und einem Raubfisch. Dieses Programm sollst du erweitern.

a) Der Raubfisch hält das Maul offen, solange er hungrig ist. Hat er einen Goldfisch erwischt, macht er das Maul zu. Du brauchst für den Raubfisch deshalb ein zweites Kostüm.
Das geht so:
Klicke auf den grünen Raubfisch in der Sprite-Liste unter der Bühne.

Klicke auf **Costumes**, dann auf **Import.** !

Wähle im Fenster **Import Costumes** den grünen Fisch mit geschlossenem Maul aus.

OK.

b) Wenn ein Goldfisch vom Raubfisch erwischt wird, ruft er "**erwischt!**". In der Befehlsgruppe **control** findest du den Befehl

.

```
if color ■ is touching ■ ?
  broadcast erwischt!
  hide
  wait 3 secs
  go to x:-200 y: pick random -200 to 200
  show
```

Erweitere die drei Goldfisch-Programme!

c) Der grüne Raubfisch erhält ein zusätzliches Programm mit dem Programmhut


```
when I receive erwischt!   (Programmhut)
    play sound chomp
    switch to costume Maul-zu
    wait 0.5 secs
    switch to costume Maul-auf
```

Setze das Programm zusammen und teste es!

d) Wenn du willst, soll der Raubfisch den Fisch kauen:

```
when I receive erwischt!   (Programmhut)
    play sound chomp
    repeat 3
            switch to costume Maul-zu
            wait 0.3 secs
            switch to costume Maul-auf
```

Setze die Programme zusammen und spiele damit. Zur Erinnerung: den Raubfisch steuerst du mit dem Mauspfeil.

Jeder Goldfisch sendet die Botschaft (broadcast) "erwischt", wenn er mit den Zähnen des Raubfischs in Berührung kommt. Diese Botschaft startet das Raubfisch-Programm mit dem Hut 

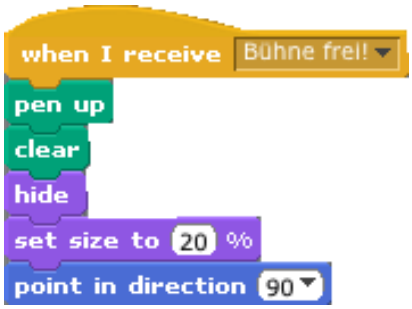
erwischt.

Eine Botschaft kann ein Programm starten.

Mit Unterprogrammen teilt der erfahrene Programmierer eine komplexe Aufgabe in mehrere einfache Aufgaben auf. Damit wird die Lösung leichter und übersichtlicher.

Projekt 8: Blume

Bei dieser Übung teilst du eine komplexe Aufgabe mit **Unterprogrammen** (Projekt 7) in einfache Teilaufgaben auf.



Aufgabe 1. Als erstes setzst du ein **Aufräumprogramm** zusammen. Es soll mit der Botschaft "Bühne frei" gestartet werden.

Aufgabe 2: Kreis

Das erste Programm *Bild 2* macht zuerst die Bühne frei (Botschaft "Bühne frei") und ruft das zweite Programm (Botschaft "zeichne Kreis") das einen Kreis zeichnet. Beim Zeichnen des Kreises macht *Büsi* eine **volle Umdrehung** (360°), denn es hat sich 360 Mal um 1° gedreht und schaut wieder in die Startrichtung. Setze das Programm zusammen und starte es!

Aufgabe 3.

Das Programm von Aufgabe 2 zeichnet langsam. Versuche es mit Drehungen von 10° statt nur 1°. *Büsi* dreht nur noch 36 mal (36 · 10° = 360°) und macht Schritte von 15 statt 1.5. *Bild 3*. Jetzt läuft *Büsi* schneller, aber der Kreis ist nicht mehr ganz so rund.

Bild 2: Kreis

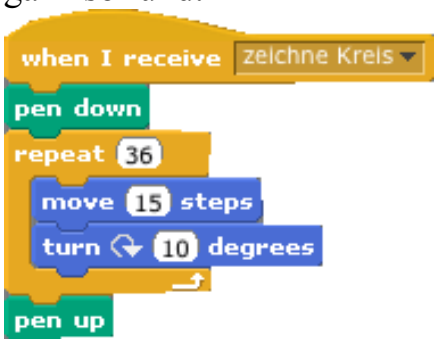
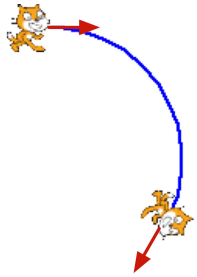


Bild 3: Kreis, schneller

Aufgabe 4: Teilkreis

In dieser Aufgabe soll *Büsi* nicht einen Kreis zeichnen, sondern einen Teilkreis. Es zeichnet nicht 36 mal 10° (=360°) wie in *Bild 2b*, sondern nur 12 mal 10° (= 120°). *Bild 4*.



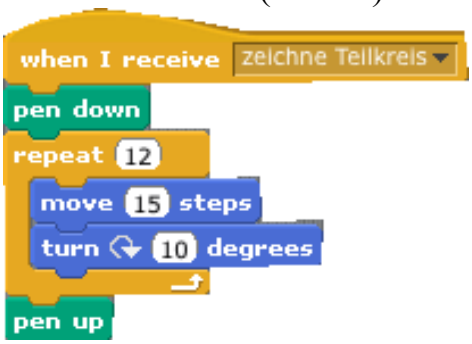


Bild 4: Teilkreis

Aufgabe 5: Blatt.

Büsi zeichnet aus zwei Teilkreisen ein Blatt (*Bild 5*). Es zeichnet zuerst den Teilkreis wie in Aufgabe 4, dreht um 60° (warum 60°?) und zeichnet einen zweiten Teilkreis. Setze das Programm selber zusammen!

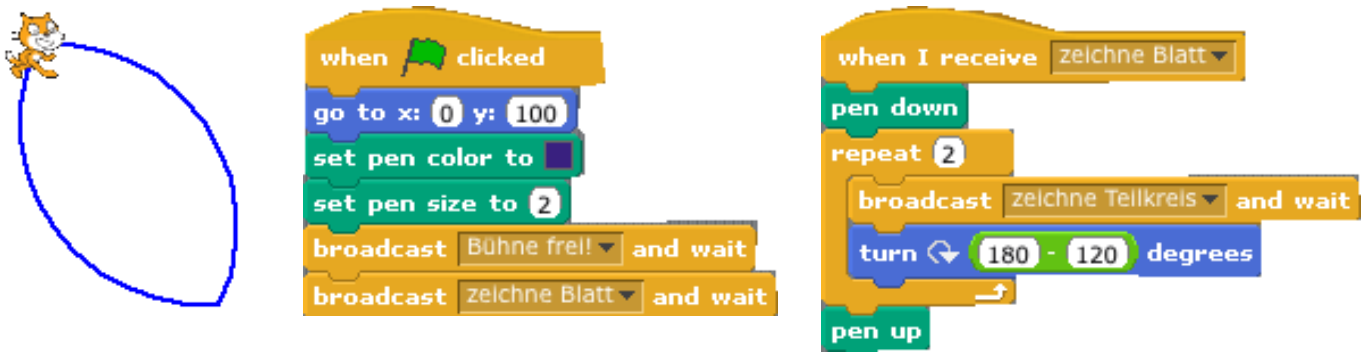
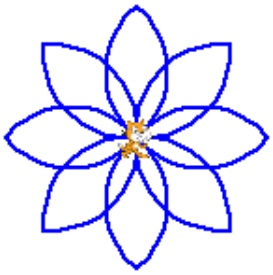


Bild 5: Blatt

Aufgabe 6: Blume.



Eine Blume besteht aus mehreren Blättern, die im Kreis angeordnet sind. Setze ein Progamm zusammen, welches durch wiederholten Aufruf von broadcast zeichne Blatt and wait eine Blume mit 8 Blättern zeichnet.

Du hast in deinem Projekt jetzt für jede Teilaufgabe ein Programm. So ist dein Projekt übersichtlich: Man sieht, dass das Hauptprogramm (grünes Fähnchen) die Bühne frei macht und anschliessend eine Blume zeichnet.

Das Programm "zeiche Blume" *ruft das Programm* "zeichne Blatt" *so oft auf, wie die Blume Blätter haben soll.*

Man sieht, dass das Programm "zeichne Blatt" *zweimal das Programm* "zeichne Teilkreis" *aufruft, um ein Blatt zu zeichnen.*

Aufgabe 7: Wieviel Blätter soll die Blume haben?
Das Programm "zeichne Blume" vom Projekt 6 enthält ein repeat, um 8 mal ein Blatt zu zeichnen.

Schaffe eine Variable "Anzahl Blätter" und ersetze die 8 im repeat durch die Variable. Bei "Anzahl Blätter 8" dreht *Büsi* nach jedem Blatt um ein Achtel einer vollen Drehung (ein Achtel von 360°), also um 360° : 8 = 45°. Mit einer anderen Anzahl Blätter muss *Büsi* nach jedem Blatt um 360° : Anzahl Blätter drehen.

Die Anzahl Blätter wird am Anfang des Hauptprogramms (grünes Fähnchen) bestimmt.

Aufgabe 8: Lange und kurze Blätter.

Im Programm "zeichne Teilkreis" hast du ein 

. 15 bestimmt die Länge der Blätter. Auch dafür definierst du eine Variable und gibst ihr den Namen Blattlänge. Nun kannst du die 15 im move-Befehl durch die Variable ersetzen.

Zeichne Blumen mit Blattlängen von 4 bis 15!

Aufgabe 9: Breite und schmale Blätter.  m15

Im Programm "zeichne Teilkreis" wiederholt *Büsi* 12 mal die Befehle 

und 

. *(Bild 4)*. *Büsi* dreht sich 12 mal um 15°; wenn es mit dem Teilkreis fertig ist, hat er sich um 12 · 15° = 180° gedreht. Mit nur 10 Wiederholungen dreht er sich um 10 · 15° = 150°; der Teilkreis ist weniger gekrümmt und das Blatt wird schlanker. Mit 15 Wiederholungen dreht sich *Büsi* 15 · 15° = 225°, der Teilkreis wird krümmer und das Blatt wird breiter.

Definiere eine weitere Variable, der du den Namen "Blattbreite" gibst. Ersetze die 15 durch Blattbreite im repeat-Befehl des Programms "zeichne Teilkreis".

Im Programm "zeichne Blatt" wird die Drehung an der Blattspitze duch den Befehl 

bewirkt. 180° ist der Halbkreis und 120° ist die Drehung, die *Büsi* beim Zeichnen des Teilkreises schon ausgeführt hat. Diese Berechnung ersetzst du durch 

.

Projekt 9: Zaubergarten   m16

Mit diesem Projekt hast du eine ziemlich komplexe Aufgabe. Du wirst sie dank Unterprogramm-Technik spielend lösen. Viel Vorarbeit hast du schon mit Projekt 8 gemacht.

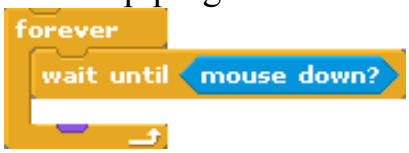
Jedesmal, wenn du auf die Bühne klickst, erscheint ein kleiner Schmetterling und zeichnet eine Blume, genau dort, wo du geklickt hast. Jedesmal eine andere Blume:

- mehr oder weniger Blätter
- andere Blattgrösse, Blattbreite
- andere Farben

So kann man einen ganzen Blumengarten zusammensetzen.

Einige Tipps

Tipp 1: Das Hauptprogramm wartet, bis auf die Bühne geklickt wird:



Tipp 2: Genau dort, wo geklickt wird, soll eine Blume erscheinen. Die "Sensing"-Werte 

und 

geben an, wo genau geklickt worden ist.

Tipp 3: Eine Blume hat einen Stiel. Dafür ist ein weiteres Programm "zeichne Stiel" notwendig. Dieses Programm kann den Befehl

(*glide* = gleite) enthalten.

Tipp 4: Bei jedem Klick soll das Programm eine andere Blume zeichnen: grössere, mehr oder weniger Blätter, kleinere, verschiedene Farben. Da helfen *Zufallszahlen*. Weisst du noch was sind Zufallszahlen sind? Wenn du zum Beispiel würfelst, bekommst du Zufallszahlen von 1 bis 6. Zufallszahlen (*random* = Zufall) sind auch im Projekt 5, Aufgabe 2 vorgekommen.

Mit 

zeichnest du eine achtblättrige Blume. Mit

zeichnest du Blumen mit 6 bis 12 Blättern.