EINFACHE GRAFIKPROGRAMMIERUNG

Untertitel

GRAFIKPRIMITIVE

- Grundlage: Turtle Grafik
- Grundmetapher:
  - Zeichenstift auf Zeichenfläche
  - Stift wird bewegt
- Zeichnen von:
  - Punkten
  - Strecken
  - Text

BEISPIEL I

Vorarbeit:
- Projekt öffnen
- Neues Verzeichnis: Grafik

```
import turtle

alex=turtle.Turtle()          #Zeichenobjekt erzeugen

alex.fd(50)                   #Vorwärts 50LE
alex.left(20)                 #20° links drehen
turtle.exitonclick()          #Bildschirm nach Ende anzeigen
```

Alternative (alt)

```
from turtle import *          # Grafikbefehle importieren
forward(50)                   # 50 LE vorwärts
left(30)                      # Um 90° links drehen
forward(30)                   # 30 LE vorwärts
mainloop()                    # zeigt Bildschirm weiterhin an
```

GRAFIKBEFEHLE

- Bewegung:
  - forward() | fd()
  - backward() | bk() | back()
  - right() | rt()
  - left () | lt()
  - goto()
  - write()
  - circle(radius)

GRAFIKBEFEHLE

- Zustandsabfragen
  - `position()`
  - `xcor()`
  - `ycor()`
- Beispiel:
  - `if schildkroete.xcor < 30:`

GRAFIKPRIMTIVE

- Stiftkontrolle
  - pendown()
  - penup()
  - pencolor()
  - pensize()
- Damit Bewegung ohne Zeichnen möglich
  - „gestrichelte Linien zeichnen“

QUELLEN

- Quellen und Dokumentation:
- https://docs.python.org/3.0/library/turtle.html

AUFGABE 2

1. Zeichnet ein Quadrat mit einer Seitenlänge von 100LE
2. Zeichnet ein Quadrat mit einer einzugebenden Seitenlänge

```
Import turtle
laenge=int(input("Seitenlänge?"))
alex=turtle.Turtle()
alex.fd(laenge)
alex.rt(90)
alex.fd(laenge)
alex.rt(90)
alex.fd(laenge)
alex.rt(90)
alex.fd(laenge)
alex.rt(90)
turtle.exitonclick()
```

```
alex=turtle.Turtle()
for _ in range(0,4):
    alex.fd(laenge)
    alex.rt(90)
turtle.exitonclick()
```

Umsetzung mit Grafikobjekt
(Instanz einer Schildkröte)

BEISPIELE

```
import turtle
star = turtle.Turtle()
for i in range(5):
   star.forward(50)
   star.right(144)
turtle.done()
```

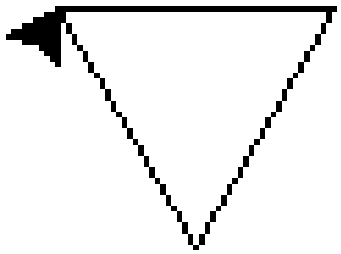
```
import turtle
spiral = turtle.Turtle()
for i in range(20):
    spiral.forward(i * 10)
    spiral.right(144)
turtle.done()
```

```
import turtle
painter = turtle.Turtle()
painter.pencolor("blue")
for i in range(50):
    painter.forward(50)
    painter.left(123)
painter.pencolor("red")
for i in range(50):
    painter.forward(100)
    painter.left(123)
turtle.done()
```

AUFGABE 3

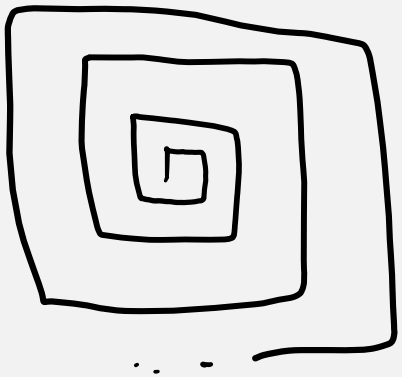
- Zeichnet ein gleichseitiges Dreieck, die Seitenlänge soll dabei vom Nutzer eingegeben werden können.
  Zusatz: Farbwahl ☺

```
import turtle
seitenlaenge=int(input("Seitenlänge?"))
bob=turtle.Turtle()
bob.goto(0,0)
bob.fd(seitenlaenge)
bob.rt(120)
bob.fd(seitenlaenge)
bob.rt(120)
bob.fd(seitenlaenge)
bob.rt(60)
turtle.exitonclick()
```



AUFGABE 4

Einfach:



Schwer:

Schreibe ein Programm, das einen Startpunkt (x und y Koordinaten) sowie einen Endpunkt einliest

- Das Programm soll eine gestrichelte Linie zwischen den beiden Punkten zeichnen
- Erweiterung:
  - „Stricheldicke“ soll eingestellt werden können
  - „Stricheldicke“ soll „perfekt“ eingeteilt sein

```
import turtle
import math

#Einlesen von Start und Endpunkt

startx=int(input("x-Wert Startpunkt?"))
starty=int(input("y-Wert Startpunkt?"))
endx=int(input("x-Wert Endpunkt?"))
endy=int(input("y-Wert Endpunkt?"))

#Winkel und Länge bestimmen
winkel=math.degrees(math.tan((abs(endy-starty))/(abs(endx-startx))))
laenge=math.sqrt(((endy-starty)**2)+((endx-startx))**2)

print(winkel)
print(laenge)

bob=turtle.Turtle()
bob.goto(startx, starty)
bob.rt(winkel)

for _ in range(0,10):
    bob.fd(laenge/20)
    bob.penup()
    bob.fd(laenge/20)
    bob.pendown()
turtle.exitonclick()
```