```python
import os                 # wegen 'cls'
import sys                # wegen Anzahl der Scheiben als Parameter
import time               # für die Anweisung 'sleep' für Pause

#----------------------------------------------------------------------------

from ctypes import *

STD_OUTPUT_HANDLE = -11

class COORD(Structure):
    pass

COORD._fields_ = [("X", c_short), ("Y", c_short)]

def print_at(r, c, s):
    h = windll.kernel32.GetStdHandle(STD_OUTPUT_HANDLE)
    windll.kernel32.SetConsoleCursorPosition(h, COORD(c, r))

    c = s.encode("windows-1252")
    windll.kernel32.WriteConsoleA(h, c_char_p(c), len(c), None, None)

#print_at(6, 3, "Hello")

# Quelle:  https://rosettacode.org/wiki/Terminal_control/Cursor_positioning#Python

#----------------------------------------------------------------------------

### Konstanten ###

my_disc = ["                       ",
           "          ###          ",
           "         XXXXX         ",
           "        &&&&&&&        ",
           "       %%%%%%%%%       ",
           "      $$$$$$$$$$$      ",
           "     #############     ",
           "    XXXXXXXXXXXXXXX    ",
           "   &&&&&&&&&&&&&&&&&   ",
           "  %%%%%%%%%%%%%%%%%%%  ",
           " $$$$$$$$$$$$$$$$$$$$$ ",
           "#######################"]

boden   = "
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~~~~"

aktuelle_disc = 0                                            # nur zur "Initialisierung"

anzahl_bewegungen = 0                                        # Initialisierung


### Klassendefinitionen ###

class Scheibe():
    def __init__(self, nummer_scheibe):
        Scheibe.form = my_disc[nummer_scheibe]

    def zeichnen(self, stab, scheiben_nr_auf_stab):
        pos_rechts =  37 * stab - 34           # stab_a = 1, stab_b = 2, stab_c = 3
        pos_oben   =  17 - scheiben_nr_auf_stab
        print_at(pos_oben, pos_rechts, self.form)

    def loeschen(self, stab, scheiben_nr_auf_stab):
        pos_rechts =  37 * stab - 34           # stab_a = 1, stab_b = 2, stab_c = 3
        pos_oben   =  17 - scheiben_nr_auf_stab
        print_at(pos_oben, pos_rechts, my_disc[0])


class Stab():
    def __init__(self,nr):
```

```
        self.anzahl = 0
        self.nummer = nr        # z.B.: stab_a = 1, stab_b = 2, stab_c = 3
        self.scheiben = []      # Liste von Scheiben, die auf diesem Stab sind

    def scheibe_aufnehmen(self, nummer_scheibe):
        self.scheiben.append(nummer_scheibe)
        self.anzahl += 1                        # Anzahl um 1 erhöhen
        hilf = Scheibe(nummer_scheibe)
        stab_nr = self.nummer
        scheiben_nr_auf_stab = self.anzahl
        hilf.zeichnen(stab_nr, scheiben_nr_auf_stab)

    def scheibe_abgeben(self):
        aktuelle_scheibe = self.scheiben.pop() # Rückgabe ist Nummer der Scheibe (int)
        stab_nr = self.nummer
        scheiben_nr_auf_stab = self.anzahl
        hilf = Scheibe(scheiben_nr_auf_stab)
        hilf.loeschen(stab_nr, scheiben_nr_auf_stab)
        self.anzahl -= 1                        # Anzahl um 1 verkleinert
        return aktuelle_scheibe                 # das ist die Nummer der Scheibe / my_disc


class StabX(Stab):
    laenge = 0

    def __init__(self,nr):
        super().__init__(nr)

    def zeichne_stab(self):
        for i in range(StabX.laenge, 0, -1):
            print_at(17-i, self.nummer * 37 - 23, "|")  # siehe bei Scheibe.zeichnen

    def scheibe_aufnehmen(self, nummer_scheibe):
        super().scheibe_aufnehmen(nummer_scheibe)
        self.zeichne_stab()

    def scheibe_abgeben(self):
        aktuelle_scheibe = super().scheibe_abgeben()
        self.zeichne_stab()
        return aktuelle_scheibe



### siehe  https://erasmus-reinhold-gymnasium.de/info/rekursion/hanoi.html#programm
def bewege (anzahl,stab_a,stab_b,stab_c):
    if anzahl > 0:
        bewege(anzahl-1,stab_a,stab_c,stab_b)
        aktuelle_disc = stab_a.scheibe_abgeben()
        stab_c.scheibe_aufnehmen(aktuelle_disc)
        global anzahl_bewegungen
        anzahl_bewegungen += 1                  # Anzahl um 1 erhöhen
        time.sleep(5/10)                        # sonst ist das Programm viel zu schnell
        bewege(anzahl-1,stab_b,stab_a,stab_c)



######## Hauptprogramm ########

# Bildschirm löschen
os.system('cls')

# Quelle: https://rosettacode.org/wiki/Terminal_control/Hiding_the_cursor#Python
print("\x1b[?25l")                              # Cursor ausschalten

# wir lesen die Anzahl der Scheiben ein
anzahl_scheiben = int(sys.argv[1])

# wir zeichen den Boden, wo die 3 Stäbe und damit die Scheiben drauf gelegt werden
print_at(17, 0, boden)


# Wir erstellen die 3 Stäbe (also die Instanzen = Objekte)
```

```
stab_a = StabX(1)
stab_b = StabX(2)
stab_c = StabX(3)


# wir setzen die Länge der Stäbe auf anzahl_scheiben + 1
StabX.laenge = anzahl_scheiben + 1

# wir legen die Scheiben auf den stab_a
for i in range(anzahl_scheiben, 0, -1):
    stab_a.scheibe_aufnehmen(i)


stab_b.zeichne_stab()
stab_c.zeichne_stab()
time.sleep(2)                                   # 2s zum Anschauen, bevor es losgeht


# jetzt läuft die Verscheibung der Scheiben ab = das eigentliche Programm
bewege(anzahl_scheiben,stab_a,stab_b,stab_c)


# diese Anweisungen dient insbesondere dazu, dass der Pfad am Ende unter den Türmen ist
time.sleep(5/10)                                # Pause 1/2 sekunde
print_at(22, 3, "Bin fertig!\n")
print("   Anzahl der Bewegungen:", anzahl_bewegungen)
print("\x1b[?25h")                              # Cursor wieder anschalten
```