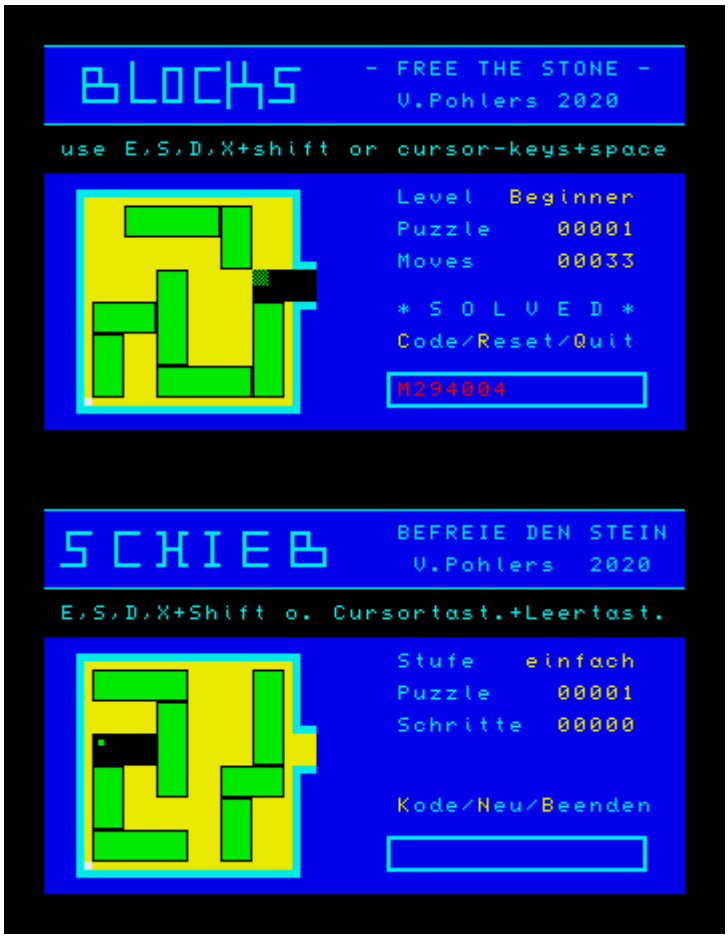
Blocks

BLOCKS bzw. SCHIEB ist ein Schiebepuzzle-Spiel für den Z9001/KC87, das ich 2020 programmiert habe. Die Idee dazu stammt von der App „Unblock Me“ von kiragames.

Die große Version enthält **1000 Puzzles in 4 Schwierigkeitsstufen**, die kleine Version 110 Puzzles in 4 Schwierigkeitsstufen.



Dateiname	BLOCKS.COM bzw. SCHIEB.COM
Laden in	OS
Programmstandort	300h-3FFFh
OS-Kommando	BLOCKS bzw. SCHIEB

Das Programm wird im OS geladen. Es wird kein RAM-Modul und kein BASIC benötigt.

Download

- blocks-bin.zip

Programm Version 15.1.2021

Bedienung

Ziel des Spiels ist es, den gefüllten Spielstein aus dem Spielfeld herauszuschieben. Dazu müssen die Spielsteine (Blöcke) verschoben werden. Waagerechte liegende Steine können nur waagerecht verschoben werden, senkrechte Steine nur senkrecht.

Die Spielsteine werden mit dem „Wordstar-Kreuz“ angewählt und und mit Shift+Wordstar-Tasten verschoben.
Alternativ kann man die Cursortasten nutzen. Dann gibt es eine abweichende Bedienung. Mit Space oder Enter wechselt man zwischen Selektieren und Bewegen.[1)]

```
     E              ohne Shift - Stein auswählen              hoch
Leertaste wechselt zwischen
     ^              mit Shift  - Stein bewegen                  ^
Stein auswählen und bewegen
     |                                                         |
S <--+--> D                                            links <--+--> rechts
     |                                                         |
     v                                                         v
     X                                                       runter
```

Die intelligente Steuerung soll zum nächstliegenden Stein in der gewünschten Richtung wechseln.

Das klappt meist, manchmal ist die Bedienung aber unlogisch 😕

Mit 'A' kann man die Spielsteine der Reihe nach durchlaufen.

'R' setzt das aktuelle Puzzle zurück.

'C' startet die Codeeingabe.

'Q' beendet das Spiel und kehrt zum OS zurück.

Hat man ein Puzzle gelöst, erhält man einen individuellen **Code** für das nächste Puzzle. Diesen sollte man notieren. Durch Eingabe eines gültigen Codes springt man zum Puzzle und muss nicht wieder von vorn anfangen!

Und hier die Codes für die jeweils ersten 5 Puzzle je Schwierigkeitsgrad. (Die Codes variieren je nach Rechner!)

```
Level 0 - Beginner 00050
Y408000 A118006 M2AC009 Y43000C A144013

Level 1 - Intermediate 00030
J228003 B134005 F1C400B B15000E Z460011

Level 2 - Advanced 00020
S348003 G1D0005 O2E8008 C17000D O308010

Level 3 - Expert 00010
```

```
D164002 D174007 D180008 D19000D H22C013
```

Internes

Das Spiel an sich ist nicht sonderlich kompliziert zu programmieren. Es enthält keine komplexen Algorithmen. Über ein paar Sachen muss man sich aber trotzdem Gedanken machen:

- Die Orignal-App ist 16 MByte groß und nutzt eine Datenbank mit den Puzzles. Wie komprimiert man die Puzzles so, dass möglichst viele in 16K RAM passen?
- Zum Weiterspielen sollte man den Spielstand speichern. Ich habe mich für die Eingabe der geschafften Puzzle-Nummer entschieden. Dazu musste ich mir ein Verfahren ausdenken, um Puzzle-Codes zu erzeugen und auch wieder zu dekodieren. Die Codes sollen möglichst zufällig sein, damit sie nicht erraten werden können.
- Die Original-App wird mit dem Finger bedient. Wie baut man eine sinnvolle Steuerung mit Tasten?[2)]
- Die Oberfläche wurde mit Paintbox erstellt, anstatt diese zu programmieren. Das komprimierte Bild wird einfach eingebunden.

[1)]
Cursortasten mit Shift zum Bewegen ist leider nicht ohne eigene Tastaturroutine machbar.
[2)]
Ich ermittle die nächstliegende Taste in der gewünschten Richtung mittels Berechnung der euklidischen Distanz (x1-x0)^2+(y1-y0)^2. Das klappt meist, manchmal ist die Bedienung aber unlogisch, da vom oberen linken Punkt eines Steins ausgehend gerechnet wird und nicht vom Schwerpunkt aus.