

TIC- TAC - TOE

TIC - TAC – TOE

Ein kleines DOS-Spiel in Assembler

Inhalt

1.Code Listing

2.Erklärung des Spiels

2.0 Spielregeln

2.1 Bedienung des Programms

3.Realisierung

3.0 Setzen der Zeichen

3.1 Auswerten des Spiels

3.2 Ende des Spiels

4.Autor

1. Code Listing

MASM SYNTAX

Datei: ttt.asm:

```
; Title: Tic Tac Toe
; Author: Rouven Maier
; File: ttt.asm

.MODEL SMALL
.STACK 256

.DATA
;Der Titel Samt Rahmen
msg001 DB 201,205,205,205,205,205,205,205,205,205,205,205,187,13,10
        DB 186,84,73,67,32,84,65,67,32,84,79,69,186,13,10,0

;Messages fuer Statusausgabe und Abfragen
msg002 DB "An der Reihe: ",13,10,0
msg003 DB "Erneut spielen? (J/N):",13,10,0
msg004 DB " hat Gewonnen!",13,10,0
msg005 DB "Unentschieden! ",13,10,0

;ASCII-Codes fuer die Tabelle
data001 DB 204,205,205,205,203,205,205,205,203,205,205,205,185,13,10,0
data002 DB 186,32,32,32,186,32,32,32,186,32,32,32,186,13,10,0
data003 DB 204,205,205,205,206,205,205,205,206,205,205,205,185,13,10,0
data004 DB 200,205,205,205,202,205,205
        DB 205,202,205,205,205,188,13,10,0

;Variable
val001 DB 0 ;Zaehler fuer Auswertung
val002 DB 0 ;Zeichen fuer Auswertung
val003 DB 0 ;Aktueller Spieler
val004 DB 0 ;Anzahl der gesetzten Zeichen

.CODE

load:
;Datensegment in DS laden
MOV AX,@DATA
MOV DS,AX

;Bildschirm leeren
start: MOV AX,03h
        INT 10h

;Titel ausgeben
MOV SI,OFFSET msg001
CALL WriteStr
```

TIC- TAC - TOE

;Tabelle Zeichnen
Call WriteTable

;Status
Call WriteCRLF
MOV SI,OFFSET msg002
CALL WriteStr

;X Faengt an
MOV [val003],'X'
Call RefreshStatus

;keine Zuege
MOV [val004],00h

tast: MOV AH,02h ;Cursor ans Ende der Statusmessage
MOV DX,0A0Fh ;(Aktueller Spieler) setzen
INT 10h

MOV AH,00h ;Zeichen von Tastatur abfragen
INT 16h

CMP AL,1Bh ;ESC --> Beenden
JE fin

CMP AL,31h ;Nur Zahlen von 1 - 9 (ASCII: 31h - 39h)
JB tast ;zulassen, bei Fehler nochmal Abfragen

CMP AL,39h
JA tast
SUB AL,30h ;ASCII Code in Zahl umwandeln (30h abziehen)

;Zeichenkoordinaten ausrechnen

CMP AL,04h
JB r1 ;Zahl < 4 --> erste Reihe

CMP AL,07h
JB r2 ;Zahl < 7 --> zewite Reihe
JMP r3 ;Zahl >= 7 --> dritte Reihe

;ASCII Reihe in DH speichern

r1: MOV DH,03h ;Erste Reihe: 3
JMP rf

r2: MOV DH,05h ;Zweite Reihe: 5
SUB AL,03h ;3 Von der Ausgangszahl abziehen um
JMP rf ;Werte von 1 - 3 für die Spalten zu erhalten

r3: MOV DH,07h ;Dritte Reihe: 7
SUB AL,06h ;6 abziehen (erkl. s. o.)

;AL hat nun Werte von 1 - 3 welche die Spalten angeben
rf: CMP AL,01h
JE c1
CMP AL,02h
JE c2
JMP c3

;ASCII Spalte in DL speichern

TIC- TAC - TOE

```

c1:  MOV DL,02h          ;Erste Spalte: 2
      JMP cf
c2:  MOV DL,06h          ;Zweite Spalte: 6
      JMP cf
c3:  MOV DL,0Ah          ;Dritte Spalte: 10

cf:  MOV AH,02h          ;Cursor an die ausgerechnete Position setzen
      INT 10h

      MOV AH,08h          ;Cursor lesen und pruefen ob frei ist
      INT 10h
      CMP AL,'X'          ;X gesetzt - unguelteig - neue Abfrage
      JE tast
      CMP AL,'O'          ;O gesetzt - unguelteig - neue Abfrage
      JE tast

      ;Feld ist frei
      MOV AH,0Eh          ;Das zeichen des aktuellen Spielers in das
      MOV AL,[val003]      ;Feld schreiben
      INT 10h

      ;Zeichen in val003 umtauschen
      CMP [val003],'X'
      JE po
px:  MOV [val003],'X'
      JMP pf
po:  MOV [val003],'O'

pf:  ADD [val004],01h      ;val004 erhoeuen (ein Zug mehr)

      Call RefreshStatus   ;Statusmessage aktualisieren
      Call Analyze         ;Feststellen ob schon jemand gewonnen hat

      CMP [val004],09h     ;Wenn bereits 9 Zeichen gesetzt wurden (Feld voll)
      JB tast             ;keinen Durchlauf mehr starten

      ;Spielende, fragen ob neue Runde beginnen soll
ask: MOV AH,02h           ;Cursor in den Nachrichtenbereich (unterm Spielfeld)
      MOV DX,0B00h         ;setzen
      INT 10h

      MOV SI,OFFSET msg003;Fragemessage ausgeben
      Call WriteStr

      ;Tasten abfragen für neue Runde (J/N) oder (j/n)
askkey: MOV AH,00h        ;Zeichen einlesen
      INT 16h
      CMP AL,4Ah           ;Eing: J (ASCII: 4Ah)
      JE start             ;an Programmanfang springen
      CMP AL,6Ah           ;Eing: j (ASCII: 6Ah)
      JE start             ;an Programmanfang springen
      CMP AL,4Eh           ;Eing: N (ASCII: 4Eh)
      JE fin              ;Ende
      CMP AL,6Eh           ;Eing: n (ASCII: 6Eh)
      JE fin              ;Ende

```

TIC- TAC - TOE

JMP askkey ;Falsche Eingabe, nochmal abfragen

fin: ;Programmende
MOV AH,02h ;Cursor ganz nach unten setzen, da DOS sonst
MOV DX,0C00h ;mit seinen Ausgaben nicht hinter den Ausgaben
INT 10h ;vom Programm fortfaehrt

MOV AH,4Ch ;Programm beenden
INT 21h

;Baut die Tabelle zeilenweise auf
WriteTable:

MOV SI,OFFSET data001
CALL WriteStr
MOV SI,OFFSET data002
Call WriteStr
MOV SI,OFFSET data003
Call WriteStr
MOV SI,OFFSET data002
Call WriteStr
MOV SI,OFFSET data003
Call WriteStr
MOV SI,OFFSET data002
Call WriteStr
MOV SI,OFFSET data004
Call WriteStr
RET

RefreshStatus:

MOV AL,[val003] ;Spielerzeichen das gesetzt werden soll

MOV AH,03h ;Cursorposition lesen und Pushen
INT 10h
PUSH DX

MOV AH,02h ;Cursor an letzte Stelle der Statusmessage setzen
MOV DX,0A0Eh
INT 10h

MOV AH,0Eh ;Zeichen schreiben (Altes wird ersetzt)
INT 10h

POP DX ;Cursorposition wiederherstellen
MOV AH,02h
INT 10h
RET

Analyze:

MOV AH,03h ;Cursorposition lesen und Pushen
INT 10h
PUSH DX

;Erst waagerechte Kombinationen pruefen

TIC- TAC - TOE

	MOV [val001],00h	;Zaehler zuruecksetzen
row:	MOV AH,02h	;Cursor an Stelle des ersten Zeichens setzen
	MOV DX,0302h	
	ADD DH,[val001]	;DH erhoehen fuer zweite und dritte Zeile
	INT 10h	
	MOV AH,08h	;Zeichen von Cursorposition lesen und in val002 legen
	INT 10h	
	MOV [val002],AL	
	CMP [val002],'X'	;Pruefen ob 'X' oder 'O' gesetzt ist
	JE rcont	
	CMP [val002],'O'	
	JNE nextrow	;Nichts gesetzt? Naechste Reihe
rcont:	MOV AH,02h	;Cursor an Stelle des zweiten Zeichens in der Reihe setzen
	MOV DL,06h	;DL wird erhoeht, DH bleibt
	INT 10h	
	MOV AH,08h	;Zeichen von Cursorposition lesen
	INT 10h	
	CMP AL,[val002]	;Pruefen ob erstes und zweites Zeichen gleich sind
	JNE nextrow	;Wenn nein: naechste Reihe
	MOV AH,02h	;Cursor an Stelle des dritten Zeichens in der Reihe setzen
	MOV DL,0Ah	;DL wird erhoeht, DH bleibt
	INT 10h	
	MOV AH,08h	;Zeichen von Cursorposition lesen
	INT 10h	
	CMP AL,[val002]	;Wenn alle drei Zeichen gleich sind Gewinner ausgeben
	JE win	
nextrow:	ADD [val001],02h	;Naechste Reihe, 2 zum Zaehler addieren, da dieser fuer ;die Errechnung der ASCII Reihen gebraucht wird
	CMP [val001],0Bh	;Wenn alle Reihen abgearbeitet sind mit Spalten fortfahren
	JB row	;ansonsten nach oben springen und mit der naechsten Reihe ;fortfahren
		;Senkrechte Kombinationen pruefen
	MOV [val001],00h	;Zaehler zuruecksetzen
col:	MOV AH,02h	;Cursor an Stelle des ersten Zeichens setzen
	MOV DX,0302h	
	ADD DL,[val001]	;DL erhoehen fuer zweite und dritte Spalte
	INT 10h	
	MOV AH,08h	;Zeichen von Cursorposition lesen und in val002 legen
	INT 10h	
	MOV [val002],AL	

TIC- TAC - TOE

	CMP [val002],'X'	;Pruefen ob 'X' oder 'O' gesetzt ist
	JE ccont	
	CMP [val002],'O'	
	JNE nextcol	;Nichts gesetzt? Naechste Spalte
ccont:	MOV AH,02h	;Cursor an Stelle des zewiten Zeichens in der Spalte setzen
	MOV DH,05h	;DH wird erhoeht, DL bleibt
	INT 10h	
	MOV AH,08h	;Zeichen von Cursorposition lesen
	INT 10h	
	CMP AL,[val002]	;Pruefen ob erstes und zweites Zeichen gleich sind
	JNE nextcol	;Wenn nein: naechste Spalte
	MOV AH,02h	;Cursor an Stelle des dritten Zeichens in der Reihe setzen
	MOV DH,07h	;DH wird erhoeht, DL bleibt
	INT 10h	
	MOV AH,08h	;Zeichen von Cursorposition lesen
	INT 10h	
	CMP AL,[val002]	;Wenn alle drei Zeichen gleich sind Gewinner ausgeben
	JE win	
nextcol:	ADD [val001],04h	;Naechste Spalte, 4 zum Zaehler addieren, da dieser fuer
		;die Errechnung der ASCII Spalten gebraucht wird
	CMP [val001],08h	;Wenn alle Reihen abgearbeitet sind mit Spalten fortfahren
	JB col	;ansonsten nach oben springen und mit der naechsten Reihe
		;fortfahren
		;Diagonale Kombinationen pruefen
diag:	MOV AH,02h	;Cursor an Stelle des mittleren Zeichens setzen
	MOV DX,0506h	
	INT 10h	
	MOV AH,08h	;Zeichen von Cursorposition lesen und in val002 legen
	INT 10h	
	MOV [val002],AL	
	CMP [val002],'X'	;Pruefen ob 'X' oder 'O' gesetzt ist
	JE dcont	
	CMP [val002],'O'	
	JNE nwin	;Nichts gesetzt? Noch kein Gewinner
dcont:	MOV AH,02h	;Cursor an Stelle des ersten Zeichens (oben links) setzen
	MOV DX,0302h	
	INT 10h	
	MOV AH,08h	;Zeichen von Cursorposition lesen
	INT 10h	

TIC- TAC - TOE

CMP AL,[val002]	;Pruefen ob erstes und zweites Zeichen gleich sind
JNE diag2	;Wenn nein: zweite Diagonale pruefen
MOV AH,02h	;Cursor an Stelle des neunten Zeichens (unten rechts) setzen
MOV DX,070Ah	
INT 10h	
MOV AH,08h	;Zeichen von Cursorposition lesen
INT 10h	
CMP AL,[val002]	;Wenn alle drei Zeichen gleich sind Gewinner ausgeben
JE win	
;Zweite diagonale:	
diag2: MOV AH,02h	;Cursor an Stelle des dritten Zeichens (oben rechts) setzen
MOV DX,030Ah	
INT 10h	
MOV AH,08h	;Zeichen von Cursorposition lesen
INT 10h	
CMP AL,[val002]	;Pruefen ob erstes und zweites Zeichen gleich sind
JNE diag2	;Wenn nein: zweite Diagonale pruefen
MOV AH,02h	;Cursor an Stelle des neunten Zeichens (unten rechts) setzen
MOV DX,070Ah	
INT 10h	
MOV AH,08h	;Zeichen von Cursorposition lesen
INT 10h	
CMP AL,[val002]	;Wenn alle drei Zeichen gleich sind Gewinner ausgeben
JE win	
;Zweite diagonale:	
diag2: MOV AH,02h	;Cursor an Stelle des dritten Zeichens (oben rechts) setzen
MOV DX,030Ah	
INT 10h	
MOV AH,08h	;Zeichen von Cursorposition lesen
INT 10h	
CMP AL,[val002]	;Pruefen ob erstes und zweites Zeichen gleich sind
JNE nwin	;Wenn nein: noch kein Gewinner
MOV AH,02h	;Cursor an Stelle des ersten Zeichens (unten links) setzen
MOV DX,0702h	
INT 10h	
MOV AH,08h	;Zeichen von Cursorposition lesen
INT 10h	
CMP AL,[val002]	;Wenn alle drei Zeichen gleich sind Gewinner ausgeben
JE win	

TIC- TAC - TOE

```
        JMP nwin                ;Sonst: kein Gewinner

win:    MOV AH,02h              ;Cursor in Nachrichtenbereich setzen
        MOV DX,0A00h
        INT 10h

        MOV AH,0Eh             ;Zeichen des Gewinners ausgeben
        MOV AL,[val002]
        INT 10h

        MOV SI,OFFSET msg004    ;Message ausgeben
        Call WriteStr

        JMP ask                 ;Nach neuer Runde fragen

nwin:   CMP [val004],09h        ;Wenn keiner Gewonnen hat und alle Felder voll
                                   ;sind: Unentschieden
        JNE contplay

        MOV AH,02h              ;Cursor in Nachrichtenbereich setzen
        MOV DX,0A00h
        INT 10h

        MOV SI,OFFSET msg005;Message ausgeben
        Call WriteStr

        JMP ask                 ;Nach neuer Runde fragen

contplay:
        POP DX                  ;Urspruengliche Cursorposition wiederherstellen
        MOV AH,02h
        INT 10h
        RET                    ;Mit aktueller Runde fortfahren

WriteStr:                                ;Prozedur zur Ausgabe einer Zeichenkette
        LODSB
        OR AL,AL                  ;Ende ist durch Nullbyte markiert
        JZ wsret
        MOV AH,0Eh
        MOV BX,0007h
        INT 10h
        JMP WriteStr

wsret:  RET

WriteCRLF:                            ;Schreibt eine neue Zeile auf den Bildschirm
        MOV AX,0E0Dh
        INT 10h
        MOV AX,0E0Ah
        INT 10h
        RET

END    load

END
```

2. Erklärung des Spiels

2.0 Spielregeln



Die beiden Spieler setzen abwechselnd ihre Zeichen „X“ und „O“ in ein freies Feld in der Tabelle. Ein Spiel gilt als Gewonnen, sobald der erste Spieler drei seiner Zeichen in einer Reihe platziert hat. Die Reihen zählen sowohl waagerecht, senkrecht und diagonal. Sind alle 9 Felder belegt, aber es wurde keine 3er Position erreicht so gilt das Spiel als unentschieden.

2.1 Bedienung des Programms

Das Setzen eines Zeichens in ein Feld erfolgt durch Eingabe einer Zahl. Wichtig hierbei ist, dass die Durchnummerierung der Felder nicht mit der Anordnung der Tasten auf dem Zahlenblock einer Computer Tastatur übereinstimmt. Das Feld mit der Nummer 1 befindet sich also nicht unten links sondern oben links. Die Felder werden durchlaufend Nummeriert, also entspricht die erste Reihe den Zahlen 1, 2 und 3, die mittlere 4, 5 und 6 und die untere dementsprechend 7, 8 und 9.

Der Spieler, besser gesagt das Zeichen des Spielers wird unter dem Spielfeld im Statusbereich angezeigt.

Im oben zu sehenden Screenshot ist also Spieler „X“ an der Reihe. Wird nun die Zahl 5 gedrückt so erscheint im mittleren Feld ein „X“ und Spieler „O“ ist an der Reihe.

Das Spiel kann jederzeit durch einen Druck auf Escape abgebrochen werden.

Ist eine Runde durchgespielt, egal mit welchem Ergebnis, so fragt das Programm nach, ob eine neue Runde beginnen soll. Diese Frage ist mit den Tasten J für Ja und N für Nein zu bestätigen. Groß und Kleinschreibung der Bestätigung spielt keine Rolle.

1.Realisierung



3.0 Setzen der Zeichen

Nach der Eingabe einer Zahl wird abgefragt ob diese gültig ist, also ob eine Zahlentaste von 1 – 9 gedrückt wurde. Da die Zahlentasten einen ASCII-Code von 31h – 39h haben reicht es also vom ASCII-Code 30h abzuziehen um die Eingabe als normalen Integer wert vorliegen zu haben.

Aus dieser Zahl (1 – 9) werden nun die Koordinaten des ASCII Zeichens errechnet, welches in dem jeweiligen Feld liegt. Dafür wird zuerst verglichen ob die Zahl kleiner als 4 oder kleiner als 7 ist um die Zeile zu erhalten. Bei dieser Berechnung wird die Zahl auch noch herunter gesetzt, das von ihr nur noch Werte von 1 – 3 übrig bleiben, welche die Spalten nummerieren.

Um die Zahlen 1 – 3 für die Spalten zu erhalten wird von der Ausgangszahl bei der ersten Reihe 0, bei der zweiten Reihe 3 und bei der dritten Reihe 6 abgezogen.

Wird also 4 eingegeben, so liegt diese nicht kleiner als 4 aber unter 7, was dem Programm sagt es braucht die Y-Koordinate 5, welche der zweiten Zeile entspricht (siehe Bild). Von der Zahl wird noch 3 Abgezogen. Übrig bleibt also 1 für die erste Spalte, welche der X-Koordinate 2 entspricht.

Darauf folgt die Überprüfung ob das gewünschte Feld leer ist. Ist dies der Fall so wird das Zeichen des Aktuellen Spielers in das gewünschte Feld gesetzt und der andere Spieler ist an der Reihe.

3.1 Auswerten des Spiels

TIC- TAC - TOE

Die Auswertung erfolgt jedes mal wenn ein Zeichen gesetzt wurde. Hier werden hintereinander alle Möglichkeiten überprüft die es für 3er-Kombinationen gibt. Ist eine 3er-Kombination gefunden worden werden die restlichen Prüfungen übersprungen.

Es Beginnt mit der Prüfung der Waagerechten Reihen. Hier werden Reihe für Reihe, Zeichen für Zeichen verglichen. Ist bereits das erste Feld nicht gesetzt oder stimmt das zweite schon nicht mit dem ersten überein, so wird gleich zur nächsten Reihe Gesprungen.

Die Überprüfung der senkrechten Reihen erfolgt auf die gleiche Weise.

Bei den diagonalen Reihen wird zuerst das Mittlere Feld geprüft, dann oben links und unten rechts, und anschließend oben rechts und unten links.

Diese Abfragen hören sich kompliziert und Aufwändig an, und sind auch der größte Teil des Quellcodes, jedoch läuft diese Prozedur so schnell durch, dass selbst keine nennenswerte Langsamkeit auf einem 286er CPU festzustellen ist.

3.2 Ende des Spiels

Beim Abfragen der Zahlen wird zusätzlich abgefragt ob die Escapetaste gedrückt wurde. Tritt dies ein, so wird das Spiel ohne Rückfrage Beendet. Ist eine Runde beendet, so wird gefragt, ob eine Weitere gespielt werden will. Diese Frage wird mit J für Ja und N für nein beantwortet und beendet durch die Eingabe von N ebenfalls ohne Rückfrage das Spiel.

Vor dem eigentlichen Ende des Programms muss der Cursor jedoch ans Ende der Ausgabe gesetzt werden, da DOS sonst mit seinen Ausgaben dort fortfährt wo der Cursor zuletzt gestanden ist, und dies folgt zu recht unschönen Überlagerungen der Ausgaben.

TIC- TAC - TOE

1.Autor

Rouven Maier
Schafweide 25
71364 Winnenden

Email: mrpc@mrpc.de

Homepage: <http://www.mrpc.de>