

Printout

Wednesday, February 20, 2019 3:10 AM

```

1: # Drew Pesall - 1129022
2: # CS 370
3: # 2/11/2019
4:
5:
6: .data
7: gameBoard: .ascii  "\n\n\n\n| . . . . | . . . . | . . . . | . . . . |  a b c d\t\t"
8:           .ascii      "\n| . . . . | . . . . | . . . . | . . . . |  e f g h\t\t"
9:           .ascii      "\n| . . . . | . . . . | . . . . | . . . . |  i j k l\t\t"
10:          .ascii      "\n| . . . . | . . . . | . . . . | . . . . |  m n o p\t\t"
11:          .asciiz      "\n|  (0)  |  (1)  |  (2)  |  (3)  |  (index)\n"
12:
13: introText: .asciiz  "Start a One-Player, 4x4x4x4, 3D Tic-Tac-Toe Game.\n"
14:
15: offset: .half  6,  8, 10, 12, 16, 18, 20, 22, 26, 28, 30, 32, 36, 38,
          40, 42
16:          .half 60, 62, 64, 66, 70, 72, 74, 76, 80, 82, 84, 86, 90, 92,
          94, 96
17:          .half 114, 116, 118, 120, 124, 126, 128, 130, 134, 136, 138, 140, 144, 146,
          148, 150
18:          .half 168, 170, 172, 174, 178, 180, 182, 184, 188, 190, 192, 194, 198, 200,
          202, 204
19:
20: gridMessage: .asciiz  "\nSelect a grid(0-3): "
21: gridError: .asciiz  "\nPlease select a valid grid."
22:
23: indexMessage: .asciiz  "\nSelect an index(a-p): "
24: indexError: .asciiz  "\nPlease select a valid index."
25:
26: continueMessage: .asciiz  "\nContinue playing?(y/n): "
27: newGameMessage: .asciiz  "\nStart a new game?(y/n): "
28: validMessage: .asciiz  "\nPlease select a valid option."
29:
30: selectPieceMessage: .asciiz  "\nWhat would you like to play as?(x/o): "
31: selectPieceError: .asciiz  "\nPlease select a valid piece."
32:
33: occupiedPosition: .asciiz  "\nThere is already a piece there."
34:
35: pieceX: .byte 'x'
36: pieceO: .byte 'o'
37: grid0: .byte '0'
38: grid1: .byte '1'
39: grid2: .byte '2'
40: grid3: .byte '3'
41: yes: .byte 'y'
42: no: .byte 'n'
43: zero: .half 0
44: occupiedPositionMessage: .asciiz  "\nThere is already a piece there."
45:

```

```

46: userInput:      .space 4
47: userInputGrid:  .space 4
48: userInputCell:  .space 4
49:
50: index1:          .byte 'a'
51: index2:          .byte 'p'
52:
53: .text
54:
55: main:
56:
57: jal initialDisplay
58: jal selectPiece
59:
60: li $v0, 10
61:     syscall
62:
63: initialDisplay: # Displays Initial Game Board and Greeting
64:
65: # Prints out greeting
66: li     $v0, 4
67: la     $a0, introText
68: syscall
69:
70: jr     $ra
71:
72: resetGameBoard:
73: lh     $t0, zero
74: add    $t0, $t4, $t0
75: mul    $t0, $t0, 2
76: lh     $t1, offset($t0)
77: li     $t2, '.'
78: sb     $t2, gameBoard($t1)
79: add    $t4, $t4, 1
80: ble    $t4, 64, resetGameBoard
81:
82: j main
83:
84: selectPiece:
85: #Display prompt
86: li     $v0, 4
87: la     $a0, selectPieceMessage
88: syscall
89:
90: #Enter your desired piece
91: move   $a0,$t2
92: li     $v0, 8
93: la     $a0, userInput
94: li     $a1, 10
95: syscall
96:

```

```

97: #Compare
98: la      $s2, pieceX
99: lb      $t2, ($s2)
100: la     $s3, userInput
101: lb     $t3, ($s3)
102: beq     $t2,$t3,gameLoopX
103: la     $s4, piece0
104: lb     $t2, ($s4)
105: beq     $t2,$t3,gameLoop0
106:
107: li     $v0, 4
108: la     $a0, selectPieceError
109: syscall
110:
111: j       selectPiece
112:
113: jr      $ra
114:
115: selectGrid:
116:
117: #Display prompt
118: li     $v0, 4
119: la     $a0, gridMessage
120: syscall
121:
122: #Enter your desired grid
123: move   $a0,$t2
124: li     $v0, 8
125: la     $a0, userInputGrid
126: li     $a1, 10
127: syscall
128:
129: #Compare
130: la     $s2, grid0
131: lb     $t2, ($s2)
132: la     $s6, userInputGrid
133: lb     $t3, ($s6)
134: beq     $t2,$t3,test
135: la     $s4, grid1
136: lb     $t2, ($s4)
137: beq     $t2,$t3,test
138: la     $s4, grid2
139: lb     $t2, ($s4)
140: beq     $t2,$t3,test
141: la     $s4, grid3
142: lb     $t2, ($s4)
143: beq     $t2,$t3,test
144:
145: li     $v0, 4
146: la     $a0, gridError
147: syscall

```

```

148:
149: j      selectGrid
150:
151: jr      $ra
152:
153: test:
154:
155: jr      $ra
156:
157: continueGameX:
158: # Continue?
159: li      $v0, 4
160: la      $a0, continueMessage
161: syscall
162:
163: #Enter your desired choice
164: move    $a0,$t2
165: li      $v0, 8
166: la      $a0, userInput
167: li      $a1, 10
168: syscall
169:
170: #Compare
171: la      $s2, yes
172: lb      $t2, ($s2)
173: la      $s3, userInput
174: lb      $t3, ($s3)
175: beq     $t2,$t3,gameLoopX
176: la      $s4, no
177: lb      $t2, ($s4)
178: beq     $t2,$t3,newGame
179:
180: li      $v0, 4
181: la      $a0, validMessage
182: syscall
183:
184: j      continueGameX
185:
186: jr      $ra
187:
188: continueGameO:
189: # Continue?
190: li      $v0, 4
191: la      $a0, continueMessage
192: syscall
193:
194: #Enter your desired choice
195: move    $a0,$t2
196: li      $v0, 8
197: la      $a0, userInput
198: li      $a1, 10

```

```

199: syscall
200:
201: #Compare
202: la      $s2, yes
203: lb      $t2, ($s2)
204: la      $s3, userInput
205: lb      $t3, ($s3)
206: beq     $t2,$t3,gameLoop0
207: la      $s4, no
208: lb      $t2, ($s4)
209: beq     $t2,$t3,newGame
210:
211: li      $v0, 4
212: la      $a0, validMessage
213: syscall
214:
215: j       continueGame0
216:
217: jr      $ra
218:
219: exit:
220: li      $v0, 10
221: syscall
222:
223: newGame:
224: # New Game?
225: li      $v0, 4
226: la      $a0, newGameMessage
227: syscall
228:
229: #Enter your desired choice
230: move    $a0,$t2
231: li      $v0, 8
232: la      $a0, userInput
233: li      $a1, 10
234: syscall
235:
236: #Compare
237: la      $s2, yes
238: lb      $t2, ($s2)
239: la      $s3, userInput
240: lb      $t3, ($s3)
241: lh      $t4, zero
242: beq     $t2,$t3,resetGameBoard
243: la      $s4, no
244: lb      $t2, ($s4)
245: beq     $t2,$t3,exit
246:
247: li      $v0, 4
248: la      $a0, validMessage
249: syscall

```

```

250:
251: j      newGame
252:
253: jr      $ra
254:
255: selectIndex:
256: # Select Index
257: li      $v0, 4
258: la      $a0, indexMessage
259: syscall
260:
261: #Enter your desired choice
262: move     $a0,$t2
263: li      $v0, 8
264: la      $a0, userInput
265: li      $a1, 10
266: syscall
267:
268: #Compare
269: la      $s2, index1
270: lb      $t2, ($s2)
271: la      $s7, userInput
272: lb      $t3, ($s7)
273: blt     $t3,$t2,indexRetry
274: la      $s4, index2
275: lb      $t2, ($s4)
276: bgt     $t3,$t2,indexRetry
277:
278: jr      $ra
279:
280: indexRetry:
281:
282: li      $v0, 4
283: la      $a0, indexError
284: syscall
285:
286: j      selectIndex
287:
288: jr      $ra
289:
290: gameLoopX:
291: # Printing Board
292: li      $v0, 4
293: la      $a0, gameBoard
294: syscall
295:
296: # Selecting Grid
297: jal     selectGrid
298:
299:
300: # Selecting Index

```

```

301: jal      selectIndex
302:
303: # Printing Board
304: lh        $t0, zero
305: lh        $t1, zero
306: lh        $t2, zero
307: lh        $t3, zero
308: lh        $t4, zero
309:
310: lb        $t1, ($s6)
311: lb        $t2, ($s7)
312: sub       $t1, $t1, 48 # Grid
313: sub       $t2, $t2, 'a' # Index
314:
315: div       $t0, $t2, 4
316: mul       $t0, $t0, 16
317:
318: mul       $t1, $t1, 4
319:
320:
321: add       $t0, $t0, $t1
322: div       $t2, $t2, 4
323: mfhi     $t2
324: add       $t0, $t2, $t0
325:
326: mul       $t0, $t0, 2
327: lh        $t1, offset($t0)
328: li        $t2, 'X'
329:
330: # Checks to see if there is already a piece there
331: lb        $t9, gameBoard($t1)
332: bne       $t9, '.', occupiedX
333:
334: sb        $t2, gameBoard($t1)
335:
336: li        $v0, 4
337: la        $a0, gameBoard
338: syscall
339:
340: # Continue?
341: jal      continueGameX
342:
343: jr        $ra
344:
345: occupiedX:
346: li        $v0, 4
347: la        $a0, occupiedPosition
348: syscall
349: j        gameLoopX
350:
351: jr        $ra

```



```

352:
353: occupied0:
354: li      $v0, 4
355: la      $a0, occupiedPosition
356: syscall
357: j       gameLoop0
358:
359: jr      $ra
360:
361: gameLoop0:
362: # Printing Board
363: li      $v0, 4
364: la      $a0, gameBoard
365: syscall
366:
367: # Selecting Grid
368: jal     selectGrid
369:
370:
371: # Selecting Index
372: jal     selectIndex
373:
374: # Printing Board
375: lh      $t0, zero
376: lh      $t1, zero
377: lh      $t2, zero
378: lh      $t3, zero
379: lh      $t4, zero
380:
381: lb      $t1, ($s6)
382: lb      $t2, ($s7)
383: sub     $t1, $t1, 48 # Grid
384: sub     $t2, $t2, 'a' # Index
385:
386: div     $t0, $t2, 4
387: mul     $t0, $t0, 16
388:
389: mul     $t1, $t1, 4
390:
391:
392: add     $t0, $t0, $t1
393: div     $t2, $t2, 4
394: mfhi    $t2
395: add     $t0, $t2, $t0
396:
397: mul     $t0, $t0, 2
398: lh      $t1, offset($t0)
399: li      $t2, 'O'
400:
401: # Checks to see if there is already a piece there.
402: lb      $t9, gameBoard($t1)

```

```
403: bne      $t9, '.', occupied0
404:
405: sb        $t2, gameBoard($t1)
406:
407: li        $v0, 4
408: la        $a0, gameBoard
409: syscall
410:
411: # Continue?
412: jal        continueGame0
413:
414: jr         $ra
```