Printout

Tuesday, April 30, 2019 1:57 PM

```
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, 4×4×4x4, 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: comb:    .ascii  "0b0c0d 0f0k0p 0e0i0m 1a2a3a 1b2c3d 1e2i3m 1f2k3p"      # 0a
21:          .ascii  "0a0c0d 0f0j0n 1b2b3b 1f2j3n                    "      # 0b
22:          .ascii  "0a0b0d 0g0k0o 1c2c3c 1g2k3o                    "      # 0c
23:          .ascii  "0a0b0c 0h0l0p 0g0j0m 1d2d3d 1g2j3m 1c2b3a 1h2l3p"      # 0d
24:          .ascii  "0a0i0m 0f0g0h 1e2e3e 1f2g3h                    "      # 0e
25:          .ascii  "0e0g0h 0b0j0n 0a0k0p 1f2f3f                    "      # 0f
26:          .ascii  "0e0f0h 0c0k0o 0d0j0m 1g2g3g                    "      # 0g
27:          .ascii  "0e0f0g 0d0l0p 1h2h3h 1g2f3e                    "      # 0h
28:          .ascii  "0a0e0m 0j0k0l 1i2i3i 1j2k3l                    "      # 0i
29:          .ascii  "0i0k0l 0b0f0n 0m0g0d 1j2j3j                    "      # 0j
30:          .ascii  "0i0j0l 0c0g0o 0a0f0p 1k2k3k                    "      # 0k
31:          .ascii  "0i0j0k 0d0h0p 1l2l3l 1k2j3i                    "      # 0l
32:          .ascii  "0a0e0i 0n0o0p 0j0g0d 1m2m3m 1i2e3a 1n2o3p 1j2g3d"      # 0m
33:          .ascii  "0m0o0p 0b0f0j 1n2n3n 1j2f3b                    "      # 0n
34:          .ascii  "0m0n0p 0c0g0k 1o2o3o 1k2g3c                    "      # 0o
35:          .ascii  "0m0n0o 0d0h0l 0k0f0a 1p2p3p 1l2h3d 1k2f3a 1o2n2m"      # 0p
36:          .ascii  "1b1c1d 1f1k1p 1e1i1m 0a2a3a                    "      # 1a
37:          .ascii  "1a1c1d 1f1j1n 0b2b3b 0a2c3d                    "      # 1b
38:          .ascii  "1a1b1d 1g1k1o 0c2c3c 0d2b3a                    "      # 1c
39:          .ascii  "1a1b1c 1h1l1p 1g1j1m 0d2d3d                    "      # 1d
40:          .ascii  "1a1i1m 1f1g1h 0e2e3e 0a2i3m                    "      # 1e
41:          .ascii  "1e1g1h 1b1j1n 1a1k1p 0f2f3f 0a2k3p 0b2j3n 0e2g3h"      # 1f
42:          .ascii  "1e1f1h 1c1k1o 1d1j1m 0g2g3g 0d2j3m 0c2k3o 0h2f3e"      # 1g
```

```
43:          .ascii  "1e1f1g 1d1l1p 0h2h3h 0d2l3p                    "      # 1h
44:          .ascii  "1a1e1m 1j1k1l 0i2i3i 0m2e3a                    "      # 1i
45:          .ascii  "1i1k1l 1b1f1n 1d1g1m 0j2j3j 0m2g3d 0n2f3b 0i2k3l"      # 1j
46:          .ascii  "1i1j1l 1c1g1o 1a1f1p 0k2k3k 0p2f3a 0o2g3c 0l2j3i"      # 1k
47:          .ascii  "1i1j1k 1d1h1p 0l2l3l 0p2h3d                    "      # 1l
48:          .ascii  "1a1e1i 1n1o1p 1j1g1d 0m2m3m                    "      # 1m
49:          .ascii  "1m1o1p 1b1f1j 0n2n3n 0m2o3p                    "      # 1n
50:          .ascii  "1m1n1p 1c1g1k 0o2o3o 0p2n3m                    "      # 1o
51:          .ascii  "1m1n1o 1d1h1l 1k1f1a 0p2p3p                    "      # 1p
52:          .ascii  "2b2c2d 2f2k2p 2e2i2m 0a1a3a                    "      # 2a
53:          .ascii  "2a2c2d 2f2j2n 0b1b3b 3a1c0d                    "      # 2b
54:          .ascii  "2a2b2d 2g2k2o 0c1c3c 3d1b0a                    "      # 2c
55:          .ascii  "2a2b2c 2h2l2p 2g2j2m 0d1d3d                    "      # 2d
56:          .ascii  "2a2i2m 2f2g2h 0e1e3e 3a1i0m                    "      # 2e
57:          .ascii  "2e2g2h 2b2j2n 2a2k2p 0f1f3f 0h1g3e 0n1j3b 0p1k3a"      # 2f
58:          .ascii  "2e2f2h 2c2k2o 2d2j2m 0g1g3g 0o1k3c 0e1f3h 0m1j3d"      # 2g
59:          .ascii  "2e2f2g 2d2l2p 0h1h3h 3d1l0p                    "      # 2h
60:          .ascii  "2a2e2m 2j2k2l 0i1i3i 3m1e0a                    "      # 2i
61:          .ascii  "2i2k2l 2b2f2n 2d2g2m 0j1j3j 0l1k3i 0b1f3n 0d1g3m"      # 2j
62:          .ascii  "2i2j2l 2c2g2o 2a2f2p 0k1k3k 0c1g3o 0i1j3l 0a1f3p"      # 2k
63:          .ascii  "2i2j2k 2d2h2p 0l1l3l 0d1h3p                    "      # 2l
64:          .ascii  "2a2e2i 2n2o2p 2d2g2j 0m1m3m                    "      # 2m
65:          .ascii  "2m2o2p 2b2f2j 0n1n3n 0p1o3m                    "      # 2n
66:          .ascii  "2m2n2p 2c2g2k 0o1o3o 0m1n3p                    "      # 2o
67:          .ascii  "2m2n2o 2d2h2l 2a2f2k 0p1p3p                    "      # 2p
68:          .ascii  "3b3c3d 3f3k3p 3e3i3m 0a1a2a 0d1c2b 0p1k2f 0m1i2e"      # 3a
69:          .ascii  "3a3c3d 3f3j3n 0b1b2b 0n1j2f                    "      # 3b
70:          .ascii  "3a3b3d 3g3k3o 0c1c2c 0o1k2g                    "      # 3c
71:          .ascii  "3a3b3c 3h3l3p 3g3j3m 0d1d2d 0a1b2c 0m1j2g 0p1l2h"      # 3d
72:          .ascii  "3a3i3m 3f3g3h 0e1e2e 0h1g2f                    "      # 3e
73:          .ascii  "3e3g3h 3b3j3n 3a3k3p 0f1f2f                    "      # 3f
74:          .ascii  "3e3f3h 3c3k3o 3d3j3m 0g1g2g                    "      # 3g
75:          .ascii  "3e3f3g 3d3l3p 0h1h2h 0e1f2g                    "      # 3h
76:          .ascii  "3a3e3m 3j3k3l 0i1i2i 0l1k2j                    "      # 3i
77:          .ascii  "3i3k3l 3b3f3n 3d3g3m 0j1j2j                    "      # 3j
78:          .ascii  "3i3j3l 3c3g3o 3a3f3p 0k1k2k                    "      # 3k
79:          .ascii  "3i3j3k 3d3h3p 0l1l2l 0i1j2k                    "      # 3l
80:          .ascii  "3a3e3i 3n3o3p 3d3g3j 0m1m2m 0a1e2i 0d1g2j 0p1o2n"      # 3m
81:          .ascii  "3m3o3p 3b3f3j 0n1n2n 0b1f2j                    "      # 3n
82:          .ascii  "3m3n3p 3c3g3k 0o1o2o 0c1g2k                    "      # 3o
83:          .ascii  "3m3n3o 3d3h3l 3a3f3k 0p1p2p 0a1f2k 0d1h2l 0m1n2o"      # 3p
84:
85: gridMessage:        .asciiz     "\nSelect a grid(0-3): "
86: gridError:          .asciiz     "\nPlease select a valid grid."
87:
88: testGrid:           .word   0
89: testIndex:          .byte   'a'
90:
91: indexMessage:       .asciiz     "\nSelect an index(a-p): "
92: indexError:         .asciiz     "\nPlease select a valid index."
93:
```

```
94: continueMessage:    .asciiz     "\nContinue playing?(y/n): "
95: newGameMessage:     .asciiz     "\nStart a new game?(y/n): "
96: validMessage:       .asciiz     "\nPlease select a valid option."
97:
98: madeIt:             .asciiz     "\nMade it"
99:
100: playerSelectPieceMessage:  .asciiz     "\nWhat would you like to play as?(x/o): "
101: playerSelectPieceError:        .asciiz     "\nPlease select a valid piece."
102:
103: occupiedPosition:  .asciiz     "\nThere is already a piece there."
104:
105: computerFirstO:        .asciiz     "\nThe computer went first and selected 'O'. You a
re playing as 'X'"
106: computerFirstX:        .asciiz     "\nThe computer went first and selected 'X'. You a
re playing as 'O'"
107:
108: pieceX:            .byte   'x'
109: pieceO:            .byte   'o'
110: grid0:             .byte    '0'
111: grid1:             .byte    '1'
112: grid2:             .byte    '2'
113: grid3:             .byte    '3'
114:
115: spota:             .byte    'a'
116: spotb:             .byte    'b'
117: spotc:             .byte    'c'
118: spotd:             .byte    'd'
119: spote:             .byte    'e'
120: spotf:             .byte    'f'
121: spotg:             .byte    'g'
122: spoti:             .byte    'h'
123: spoth:             .byte    'i'
124: spotj:             .byte    'j'
125: spotk:             .byte    'k'
126: spotl:             .byte    'l'
127: spotm:             .byte    'm'
128: spotn:             .byte    'n'
129: spoto:             .byte    'o'
130: spotp:             .byte    'p'
131:
132: playerPiece:       .word   0
133: computerPiece:     .word   0
134:
135: playerWon:         .asciiz     "Congratulations, you won! "
136: computerWon:       .asciiz     "Uh oh, looks like the computer won! "
137: xWon:              .asciiz     "X got four in a a row. "
138: oWon:              .asciiz     "O got four in a row. "
139: gameTie:           .asciiz     "All cells filled: Game results in a tie."
140:
141: yes:               .byte   'y'
142: no:                    .byte   'n'
```

```
143: zero:              .half   0
144: occupiedPositionMessage: .asciiz "\nThere is already a piece there."
145:
146: userInput:         .space  4
147: userInputGrid:     .space  4
148: userInputIndex:        .space  4
149:
150: userPreviousGrid:  .word   0
151: userPreviousIndex: .word   0
152:
153: computerPreviousGrid:  .word   0
154: computerPreviousIndex: .word   0
155:
156: index1:                .byte   'a'
157: index2:                .byte   'p'
158:
159: .text
160:
161: main:
162:
163: jal initialDisplay
164:
165: li $a1, 2  #Here you set $a1 to the max bound.
166: li $v0, 42  #generates the random number.
167: syscall
168:
169: beqz   $a0, computerSelectPiece
170:
171: jal    playerSelectPiece
172:
173: li $v0, 10
174:    syscall
175:
176: initialDisplay: # Displays Initial Game Board and Greeting
177:
178: # Prints out greeting
179: li          $v0, 4
180: la          $a0, introText
181: syscall
182:
183: jr      $ra
184:
185: resetGameBoard:
186: lh      $t0, zero
187: add         $t0, $t4, $t0
188: mul         $t0, $t0, 2
189: lh          $t1, offset($t0)
190: li      $t2, '.'
191: sb      $t2, gameBoard($t1)
192: add         $t4, $t4, 1
193: ble         $t4, 64, resetGameBoard
```

```
194:
195: j main
196:
197: computerSelectPiece:
198:
199: li     $v0, 4
200: la     $a0, computerFirstO
201: syscall
202:
203: jal        playerSetX
204:
205: j      computerTurnO
206:
207:
208: computerTurnO:
209:
210: # Selecting Grid
211: jal    selectGridComputer
212:
213:
214: # Selecting Index
215: jal        selectIndexComputer
216:
217: # Printing Board
218: lh         $t0, zero
219: lh         $t1, zero
220: lh     $t2, zero
221: lh         $t3, zero
222: lh     $t4, zero
223:
224: lb     $t1, ($s6)
225: lb         $t2, ($s7)
226: sub        $t1, $t1, 48  # Grid
227: sub        $t2, $t2, 'a' # Index
228: sb     $t2, computerPreviousIndex
229:
230: div        $t0, $t2, 4
231: mul        $t0, $t0, 16
232:
233: mul        $t1, $t1, 4
234:
235:
236: add        $t0, $t0, $t1
237: div        $t2, $t2, 4
238: mfhi   $t2
239: add        $t0, $t2, $t0
240:
241: mul        $t0, $t0, 2
242: lh         $t1, offset($t0)
243: li     $t2, 'O'
244:
```

```
245: # Checks to see if there is already a piece there.
246: lb     $t9, gameBoard($t1)
247: bne        $t9, '.', computerTurnO
248:
249: sb     $t2, gameBoard($t1)
250:
251: jal        readCombComputerO
252:
253: jal        gameLoopX
254:
255: computerTurnX:
256:
257: # Selecting Grid
258: jal    selectGridComputer
259:
260:
261: # Selecting Index
262: jal        selectIndexComputer
263:
264: # Printing Board
265: lh         $t0, zero
266: lh         $t1, zero
267: lh     $t2, zero
268: lh         $t3, zero
269: lh     $t4, zero
270:
271: lb     $t1, ($s6)
272: lb         $t2, ($s7)
273: sub        $t1, $t1, 48  # Grid
274: sub        $t2, $t2, 'a' # Index
275: sb     $t2, computerPreviousIndex
276:
277: div        $t0, $t2, 4
278: mul        $t0, $t0, 16
279:
280: mul        $t1, $t1, 4
281:
282:
283: add        $t0, $t0, $t1
284: div        $t2, $t2, 4
285: mfhi   $t2
286: add        $t0, $t2, $t0
287:
288: mul        $t0, $t0, 2
289: lh         $t1, offset($t0)
290: li     $t2, 'X'
291:
292: # Checks to see if there is already a piece there.
293: lb     $t9, gameBoard($t1)
294: bne        $t9, '.', computerTurnX
295:
```

```
296: sb     $t2, gameBoard($t1)
297:
298: jal        readCombComputerX
299:
300: jal        gameLoopO
301:
302:
303: playerSelectPiece:
304: #Display prompt
305: li      $v0, 4
306: la      $a0, playerSelectPieceMessage
307: syscall
308:
309: #Enter your desired piece
310: move    $a0,$t2
311: li     $v0, 8
312: la     $a0, userInput
313: li     $a1, 10
314: syscall
315:
316: j start
317:
318: playerSetO:
319: li         $v0, 'O'
320: sb         $v0, playerPiece
321: li         $v0, 'X'
322: sb         $v0, computerPiece
323: jr     $ra
324:
325: playerSetX:
326: li         $v0, 'X'
327: sb         $v0, playerPiece
328: li         $v0, 'O'
329: sb         $v0, computerPiece
330: jr     $ra
331:
332: start:
333:
334: #Compare
335: la         $s2, pieceX
336: lb         $t2, ($s2)
337: la         $s3, userInput
338: lb         $t3, ($s3)
339: beq        $t2,$t3,gameLoopX
340: la     $s4, pieceO
341: lb     $t2, ($s4)
342: beq        $t2,$t3,gameLoopO
343:
344: li     $v0, 4
345: la     $a0, playerSelectPieceError
346: syscall
```

```
347:
348: j      playerSelectPiece
349:
350: jr         $ra
351:
352: pgrid0:
353: la $s6, grid0
354: jr         $ra
355: pgrid1:
356: la $s6, grid1
357: jr         $ra
358: pgrid2:
359: la $s6, grid2
360: jr         $ra
361: pgrid3:
362: la $s6, grid3
363: jr         $ra
364:
365: selectGridComputer:
366:
367: xor  $a0, $a0, $a0     # Set a seed number.
368: li   $a1, 4           # random number 0 to 15
369: li   $v0, 42           # random number generator
370: syscall
371:
372: sb     $a0, computerPreviousGrid
373:
374: sb     $a0, userInputGrid
375:
376: beqz   $a0, pgrid0
377: beq        $a0, 1, pgrid1
378: beq        $a0, 2, pgrid2
379: beq        $a0, 3, pgrid3
380:
381: #Compare
382: la         $s2, grid0
383: lb         $t2, ($s2)
384: lb         $t3, ($s6)
385: beq        $t2,$t3,test
386: la     $s4, grid1
387: lb     $t2, ($s4)
388: beq        $t2,$t3,test
389: la     $s4, grid2
390: lb     $t2, ($s4)
391: beq        $t2,$t3,test
392: la     $s4, grid3
393: lb     $t2, ($s4)
394: beq        $t2,$t3,test
395:
396: li     $v0, 4
397: la     $a0, gridError
```

```
398: syscall
399:
400: j      selectGridComputer
401:
402: jr         $ra
403:
404: selectGrid:
405:
406: #Display prompt
407: li      $v0, 4
408: la      $a0, gridMessage
409: syscall
410:
411: #Enter your desired grid
412: move    $a0,$t2
413: li     $v0, 8
414: la     $a0, userInputGrid
415: li     $a1, 10
416: syscall
417:
418: #Compare
419: la         $s2, grid0
420: lb         $t2, ($s2)
421: la         $s6, userInputGrid
422: lb         $t3, ($s6)
423: beq        $t2,$t3,test
424: la     $s4, grid1
425: lb     $t2, ($s4)
426: beq        $t2,$t3,test
427: la     $s4, grid2
428: lb     $t2, ($s4)
429: beq        $t2,$t3,test
430: la     $s4, grid3
431: lb     $t2, ($s4)
432: beq        $t2,$t3,test
433:
434: li     $v0, 4
435: la     $a0, gridError
436: syscall
437:
438: j      selectGrid
439:
440: jr         $ra
441:
442: test:
443:
444: jr     $ra
445:
446: continueGameX:
447: # Continue?
448: li      $v0, 4
```

```
449: la      $a0, continueMessage
450: syscall
451:
452: #Enter your desired choice
453: move    $a0,$t2
454: li     $v0, 8
455: la     $a0, userInput
456: li     $a1, 10
457: syscall
458:
459: #Compare
460: la         $s2, yes
461: lb         $t2, ($s2)
462: la         $s3, userInput
463: lb         $t3, ($s3)
464: beq        $t2,$t3,computerTurnX
465: la     $s4, no
466: lb     $t2, ($s4)
467: beq        $t2,$t3,newGame
468:
469: li     $v0, 4
470: la     $a0, validMessage
471: syscall
472:
473: j      continueGameX
474:
475: jr         $ra
476:
477: continueGameO:
478: # Continue?
479: li      $v0, 4
480: la      $a0, continueMessage
481: syscall
482:
483: #Enter your desired choice
484: move    $a0,$t2
485: li     $v0, 8
486: la     $a0, userInput
487: li     $a1, 10
488: syscall
489:
490: #Compare
491: la         $s2, yes
492: lb         $t2, ($s2)
493: la         $s3, userInput
494: lb         $t3, ($s3)
495: beq        $t2,$t3,computerTurnO
496: la     $s4, no
497: lb     $t2, ($s4)
498: beq        $t2,$t3,newGame
499:
```

```
500: li     $v0, 4
501: la     $a0, validMessage
502: syscall
503:
504: j      continueGameO
505:
506: jr         $ra
507:
508: exit:
509: li     $v0, 10
510: syscall
511:
512: newGame:
513: # New Game?
514: li      $v0, 4
515: la      $a0, newGameMessage
516: syscall
517:
518: #Enter your desired choice
519: move    $a0,$t2
520: li     $v0, 8
521: la     $a0, userInput
522: li     $a1, 10
523: syscall
524:
525: #Compare
526: la         $s2, yes
527: lb         $t2, ($s2)
528: la         $s3, userInput
529: lb         $t3, ($s3)
530: lh     $t4, zero
531: beq        $t2,$t3,resetGameBoard
532: la     $s4, no
533: lb     $t2, ($s4)
534: beq        $t2,$t3,exit
535:
536: li     $v0, 4
537: la     $a0, validMessage
538: syscall
539:
540: j      newGame
541:
542: jr         $ra
543:
544: sindexa:
545: la     $s7, spota
546: sb     $s7, computerPreviousIndex
547: jr     $ra
548: sindexb:
549: la     $s7, spotb
550: sb     $s7, computerPreviousIndex
```

```
551: jr     $ra
552: sindexc:
553: la     $s7, spotc
554: sb     $s7, computerPreviousIndex
555: jr     $ra
556: sindexd:
557: la     $s7, spotd
558: sb     $s7, computerPreviousIndex
559: jr     $ra
560: sindexe:
561: la     $s7, spote
562: sb     $s7, computerPreviousIndex
563: jr     $ra
564: sindexf:
565: la     $s7, spotf
566: sb     $s7, computerPreviousIndex
567: jr     $ra
568: sindexg:
569: la     $s7, spotg
570: sb     $s7, computerPreviousIndex
571: jr     $ra
572: sindexh:
573: la     $s7, spoth
574: sb     $s7, computerPreviousIndex
575: jr     $ra
576: sindexi:
577: la     $s7, spoti
578: sb     $s7, computerPreviousIndex
579: jr     $ra
580: sindexj:
581: la     $s7, spotj
582: sb     $s7, computerPreviousIndex
583: jr     $ra
584: sindexk:
585: la     $s7, spotk
586: sb     $s7, computerPreviousIndex
587: jr     $ra
588: sindexl:
589: la     $s7, spotl
590: sb     $s7, computerPreviousIndex
591: jr     $ra
592: sindexm:
593: la     $s7, spotm
594: sb     $s7, computerPreviousIndex
595: jr     $ra
596: sindexn:
597: la     $s7, spotn
598: sb     $s7, computerPreviousIndex
599: jr     $ra
600: sindexo:
601: la     $s7, spoto
```

```
602: sb     $s7, computerPreviousIndex
603: jr     $ra
604: sindexp:
605: la     $s7, spotp
606: sb     $s7, computerPreviousIndex
607: jr     $ra
608:
609:
610: playerWinScreen:
611:
612: li     $v0, 4
613: la     $a0, playerWon
614: syscall
615:
616: jr     $ra
617:
618:
619: selectIndexComputer:
620:
621: #Enter your desired choice
622: xor  $a0, $a0, $a0     # Set a seed number.
623: li   $a1, 16           # random number 0 to 15
624: li   $v0, 42           # random number generator
625: syscall
626:
627: beqz   $a0, sindexa
628: beq        $a0, 1, sindexb
629: beq        $a0, 2, sindexc
630: beq        $a0, 3, sindexd
631: beq        $a0, 4, sindexe
632: beq        $a0, 5, sindexf
633: beq        $a0, 6, sindexg
634: beq        $a0, 7, sindexh
635: beq        $a0, 8, sindexi
636: beq        $a0, 9, sindexj
637: beq        $a0, 10, sindexk
638: beq        $a0, 11, sindexl
639: beq        $a0, 12, sindexm
640: beq        $a0, 13, sindexn
641: beq        $a0, 14, sindexo
642: beq        $a0, 15, sindexp
643:
644: #Compare
645: la         $s2, index1
646: lb         $t2, ($s2)
647: lb         $t3, ($s7)
648: blt        $t3,$t2,indexRetry
649: la     $s4, index2
650: lb     $t2, ($s4)
651: bgt        $t3,$t2,indexRetry
652:
```

```
653: jr         $ra
654:
655: selectIndex:
656: # Select Index
657: li      $v0, 4
658: la      $a0, indexMessage
659: syscall
660:
661: #Enter your desired choice
662: move    $a0,$t2
663: li     $v0, 8
664: la     $a0, userInputIndex
665: li     $a1, 10
666: syscall
667:
668: #Compare
669: la         $s2, index1
670: lb         $t2, ($s2)
671: la         $s7, userInputIndex
672: lb         $t3, ($s7)
673: blt        $t3,$t2,indexRetry
674: la     $s4, index2
675: lb     $t2, ($s4)
676: bgt        $t3,$t2,indexRetry
677:
678: jr         $ra
679:
680: indexRetry:
681:
682: li     $v0, 4
683: la     $a0, indexError
684: syscall
685:
686: j      selectIndex
687:
688: jr     $ra
689:
690: gameLoopX:
691: jal        playerSetX
692:
693: # Printing Board
694: li         $v0, 4
695: la         $a0, gameBoard
696: syscall
697:
698: # Selecting Grid
699: jal    selectGrid
700:
701:
702: # Selecting Index
703: jal        selectIndex
```

```
704:
705: # Printing Board
706: lh          $t0, zero
707: lh          $t1, zero
708: lh      $t2, zero
709: lh          $t3, zero
710: lh      $t4, zero
711:
712: lb      $t1, ($s6)
713: lb          $t2, ($s7)
714: sub         $t1, $t1, 48  # Grid
715: sub         $t2, $t2, 'a' # Index
716:
717: div         $t0, $t2, 4
718: mul         $t0, $t0, 16
719:
720: mul         $t1, $t1, 4
721:
722:
723: add         $t0, $t0, $t1
724: div         $t2, $t2, 4
725: mfhi    $t2
726: add         $t0, $t2, $t0
727:
728: mul         $t0, $t0, 2
729: lh          $t1, offset($t0)
730: li      $t2, 'X'
731:
732: # Checks to see if there is already a piece there
733: lb      $t9, gameBoard($t1)
734: bne         $t9, '.', occupiedX
735:
736: sb      $t2, gameBoard($t1)
737:
738: li      $v0, 4
739: la      $a0, gameBoard
740: syscall
741:
742: jal         readCombX ##########################################
743:
744: # Continue?
745: jal         continueGameO
746:
747: jr          $ra
748:
749: occupiedX:
750: li      $v0, 4
751: la      $a0, occupiedPosition
752: syscall
753: j       gameLoopX
754:
```

```
755: jr     $ra
756:
757: occupiedO:
758: li     $v0, 4
759: la     $a0, occupiedPosition
760: syscall
761: j      gameLoopO
762:
763: jr     $ra
764:
765: gameLoopO:
766: jal        playerSetO
767:
768: # Printing Board
769: li         $v0, 4
770: la         $a0, gameBoard
771: syscall
772:
773: # Selecting Grid
774: jal    selectGrid
775:
776:
777: # Selecting Index
778: jal        selectIndex
779:
780: # Printing Board
781: lh         $t0, zero
782: lh         $t1, zero
783: lh     $t2, zero
784: lh         $t3, zero
785: lh     $t4, zero
786:
787: # $s6 = grid
788: # $s7 = cell
789: lb     $t1, ($s6)
790: lb         $t2, ($s7)
791: sub        $t1, $t1, 48  # Grid
792: sub        $t2, $t2, 'a' # Index
793:
794: # Equation: $t0 = (cell÷4)×16 + grid×4 + cell%4
795: div        $t0, $t2, 4
796: mul        $t0, $t0, 16
797:
798: mul        $t1, $t1, 4
799:
800:
801: add        $t0, $t0, $t1
802: div        $t2, $t2, 4
803: mfhi   $t2
804: add        $t0, $t2, $t0
805:
```

```
806: mul         $t0, $t0, 2
807: lh          $t1, offset($t0)
808: li      $t2, 'O'
809:
810: # Checks to see if there is already a piece there.
811: lb      $t9, gameBoard($t1)
812: bne         $t9, '.', occupiedO
813:
814: sb      $t2, gameBoard($t1)
815:
816: li      $v0, 4
817: la      $a0, gameBoard
818: syscall
819:
820: jal         readCombO
821:
822: # Continue?
823: jal         continueGameX
824:
825: jr          $ra
826:
827:
828: winScreenX:
829: li      $v0, 4
830: la      $a0, xWon
831: syscall
832:
833: li      $v0, 4
834: la      $a0, playerWon
835: syscall
836:
837: j       newGame
838:
839: winScreenO:
840: li      $v0, 4
841: la      $a0, oWon
842: syscall
843:
844: li      $v0, 4
845: la      $a0, computerWon
846: syscall
847:
848: j       newGame
849:
850: readCombX: # $t0 = grid        $t1 = index
851: # Equation: (Grid×16+Index)×48
852: lb          $t0, userInputGrid
853: sub         $t0, $t0, 48
854:
855: lb      $t1, userInputIndex
856: sub         $t1, $t1, 'a'
```

```
857:
858: mul         $t2, $t0, 16
859: add         $t2, $t2, $t1
860: mul         $t2, $t2, 48
861:
862: lb          $t3, comb($t2)
863:
864: add         $t2, $t2, 1
865: lb          $t4, comb($t2)
866:
867: # Saving the $t2 offset in # $s5 and $s0
868: add         $s5, $t2, $zero
869: add         $s0, $t2, $zero
870:
871: # $t3 = grid
872: # $t4 = index
873:
874: sb      $t3, userPreviousGrid
875: sb      $t4, userPreviousIndex
876:
877: j       winConditionX1
878:
879: winConditionX1:
880:
881: lb      $t1, userPreviousGrid
882: lb          $t2, userPreviousIndex
883:
884: sub         $t1, $t1, 48  # Grid
885: sub         $t2, $t2, 'a' # Index
886:
887: div         $t0, $t2, 4
888: mul         $t0, $t0, 16
889:
890: mul         $t1, $t1, 4
891:
892: add         $t0, $t0, $t1
893: div         $t2, $t2, 4
894: mfhi    $t2
895: add         $t0, $t2, $t0
896:
897: mul         $t0, $t0, 2
898: lh          $t1, offset($t0)
899:
900: # Checks to see if there is already a piece there
901: lb      $t9, gameBoard($t1)
902: beq         $t9, 'X', winConditionX2
903:
904: ###############################################
905:
906: add         $s5, $s5, 6
907: lb          $t1, comb($s5)
```

```
908: add        $s5, $s5, 1
909: lb         $t2, comb($s5)
910:
911: sub        $t1, $t1, 48  # Grid
912: sub        $t2, $t2, 'a' # Index
913:
914: div        $t0, $t2, 4
915: mul        $t0, $t0, 16
916:
917: mul        $t1, $t1, 4
918:
919: add        $t0, $t0, $t1
920: div        $t2, $t2, 4
921: mfhi   $t2
922: add        $t0, $t2, $t0
923:
924: mul        $t0, $t0, 2
925: lh         $t1, offset($t0)
926:
927: # Checks to see if there is already a piece there
928: lb     $t9, gameBoard($t1)
929: beq        $t9, 'X', winConditionX2
930:
931: ############################################################
932:
933: add        $s5, $s5, 6
934: lb         $t1, comb($s5)
935: add        $s5, $s5, 1
936: lb         $t2, comb($s5)
937:
938: sub        $t1, $t1, 48  # Grid
939: sub        $t2, $t2, 'a' # Index
940:
941: div        $t0, $t2, 4
942: mul        $t0, $t0, 16
943:
944: mul        $t1, $t1, 4
945:
946: add        $t0, $t0, $t1
947: div        $t2, $t2, 4
948: mfhi   $t2
949: add        $t0, $t2, $t0
950:
951: mul        $t0, $t0, 2
952: lh         $t1, offset($t0)
953:
954: # Checks to see if there is already a piece there
955: lb     $t9, gameBoard($t1)
956: beq        $t9, 'X', winConditionX2
957:
958: ############################################################
```

```
959:
960: add         $s5, $s5, 6
961: lb          $t1, comb($s5)
962: add         $s5, $s5, 1
963: lb          $t2, comb($s5)
964:
965: sub         $t1, $t1, 48  # Grid
966: sub         $t2, $t2, 'a' # Index
967:
968: div         $t0, $t2, 4
969: mul         $t0, $t0, 16
970:
971: mul         $t1, $t1, 4
972:
973: add         $t0, $t0, $t1
974: div         $t2, $t2, 4
975: mfhi   $t2
976: add         $t0, $t2, $t0
977:
978: mul         $t0, $t0, 2
979: lh          $t1, offset($t0)
980:
981: # Checks to see if there is already a piece there
982: lb     $t9, gameBoard($t1)
983: beq         $t9, 'X', winConditionX2
984:
985: ############################################################
986:
987: add         $s5, $s5, 6
988: lb          $t1, comb($s5)
989: add         $s5, $s5, 1
990: lb          $t2, comb($s5)
991:
992: sub         $t1, $t1, 48  # Grid
993: sub         $t2, $t2, 'a' # Index
994:
995: div         $t0, $t2, 4
996: mul         $t0, $t0, 16
997:
998: mul         $t1, $t1, 4
999:
1000: add        $t0, $t0, $t1
1001: div        $t2, $t2, 4
1002: mfhi  $t2
1003: add        $t0, $t2, $t0
1004:
1005: mul        $t0, $t0, 2
1006: lh         $t1, offset($t0)
1007:
1008: # Checks to see if there is already a piece there
1009: lb         $t9, gameBoard($t1)
```

```
1010: beq       $t9, 'X', winConditionX2
1011:
1012: ############################################################
1013:
1014: add       $s5, $s5, 6
1015: lb        $t1, comb($s5)
1016: add       $s5, $s5, 1
1017: lb        $t2, comb($s5)
1018:
1019: sub       $t1, $t1, 48  # Grid
1020: sub       $t2, $t2, 'a' # Index
1021:
1022: div       $t0, $t2, 4
1023: mul       $t0, $t0, 16
1024:
1025: mul       $t1, $t1, 4
1026:
1027: add       $t0, $t0, $t1
1028: div       $t2, $t2, 4
1029: mfhi  $t2
1030: add       $t0, $t2, $t0
1031:
1032: mul       $t0, $t0, 2
1033: lh        $t1, offset($t0)
1034:
1035: # Checks to see if there is already a piece there
1036: lb        $t9, gameBoard($t1)
1037: beq       $t9, 'X', winConditionX2
1038:
1039: ############################################################
1040:
1041: add       $s5, $s5, 6
1042: lb        $t1, comb($s5)
1043: add       $s5, $s5, 1
1044: lb        $t2, comb($s5)
1045:
1046: sub       $t1, $t1, 48  # Grid
1047: sub       $t2, $t2, 'a' # Index
1048:
1049: div       $t0, $t2, 4
1050: mul       $t0, $t0, 16
1051:
1052: mul       $t1, $t1, 4
1053:
1054: add       $t0, $t0, $t1
1055: div       $t2, $t2, 4
1056: mfhi  $t2
1057: add       $t0, $t2, $t0
1058:
1059: mul       $t0, $t0, 2
1060: lh        $t1, offset($t0)
```

```
1061:
1062: # Checks to see if there is already a piece there
1063: lb        $t9, gameBoard($t1)
1064: beq       $t9, 'X', winConditionX2
1065:
1066: j     continueGameO
1067:
1068: winConditionX2:
1069:
1070: add       $s5, $s5, 1
1071: lb        $t1, comb($s5)
1072: add       $s5, $s5, 1
1073: lb        $t2, comb($s5)
1074:
1075: sub       $t1, $t1, 48  # Grid
1076: sub       $t2, $t2, 'a' # Index
1077:
1078: div       $t0, $t2, 4
1079: mul       $t0, $t0, 16
1080:
1081: mul       $t1, $t1, 4
1082:
1083: add       $t0, $t0, $t1
1084: div       $t2, $t2, 4
1085: mfhi  $t2
1086: add       $t0, $t2, $t0
1087:
1088: mul       $t0, $t0, 2
1089: lh        $t1, offset($t0)
1090:
1091: # Checks to see if there is already a piece there
1092: lb        $t9, gameBoard($t1)
1093: beq       $t9, 'X', winConditionX3
1094:
1095: j     continueGameO
1096:
1097: winConditionX3:
1098:
1099: add       $s5, $s5, 1
1100: lb        $t1, comb($s5)
1101: add       $s5, $s5, 1
1102: lb        $t2, comb($s5)
1103:
1104: sub       $t1, $t1, 48  # Grid
1105: sub       $t2, $t2, 'a' # Index
1106:
1107: div       $t0, $t2, 4
1108: mul       $t0, $t0, 16
1109:
1110: mul       $t1, $t1, 4
1111:
```

```
1112: add       $t0, $t0, $t1
1113: div       $t2, $t2, 4
1114: mfhi  $t2
1115: add       $t0, $t2, $t0
1116:
1117: mul       $t0, $t0, 2
1118: lh        $t1, offset($t0)
1119:
1120: # Checks to see if there is already a piece there
1121: lb        $t9, gameBoard($t1)
1122: beq       $t9, 'X', winScreenX
1123:
1124: j     continueGameO
1125:
1126: ##################
1127:
1128: readCombO: # $t0 = grid       $t1 = index
1129: # Equation: (Grid×16+Index)×48
1130: lb        $t0, userInputGrid
1131: sub       $t0, $t0, 48
1132:
1133: lb        $t1, userInputIndex
1134: sub       $t1, $t1, 'a'
1135:
1136: mul       $t2, $t0, 16
1137: add       $t2, $t2, $t1
1138: mul       $t2, $t2, 48
1139:
1140: lb        $t3, comb($t2)
1141:
1142: add       $t2, $t2, 1
1143: lb        $t4, comb($t2)
1144:
1145: # Saving the $t2 offset in # $s5 and $s0
1146: add       $s5, $t2, $zero
1147: add       $s0, $t2, $zero
1148:
1149: # $t3 = grid
1150: # $t4 = index
1151:
1152: sb        $t3, userPreviousGrid
1153: sb        $t4, userPreviousIndex
1154:
1155: j     winConditionO1
1156:
1157: winConditionO1:
1158:
1159: lb        $t1, userPreviousGrid
1160: lb        $t2, userPreviousIndex
1161:
1162: sub       $t1, $t1, 48  # Grid
```

```
1163: sub       $t2, $t2, 'a' # Index
1164:
1165: div       $t0, $t2, 4
1166: mul       $t0, $t0, 16
1167:
1168: mul       $t1, $t1, 4
1169:
1170: add       $t0, $t0, $t1
1171: div       $t2, $t2, 4
1172: mfhi  $t2
1173: add       $t0, $t2, $t0
1174:
1175: mul       $t0, $t0, 2
1176: lh        $t1, offset($t0)
1177:
1178: # Checks to see if there is already a piece there
1179: lb        $t9, gameBoard($t1)
1180: beq       $t9, 'O', winCondition02
1181:
1182: ################################################
1183:
1184: add       $s5, $s5, 6
1185: lb        $t1, comb($s5)
1186: add       $s5, $s5, 1
1187: lb        $t2, comb($s5)
1188:
1189: sub       $t1, $t1, 48  # Grid
1190: sub       $t2, $t2, 'a' # Index
1191:
1192: div       $t0, $t2, 4
1193: mul       $t0, $t0, 16
1194:
1195: mul       $t1, $t1, 4
1196:
1197: add       $t0, $t0, $t1
1198: div       $t2, $t2, 4
1199: mfhi  $t2
1200: add       $t0, $t2, $t0
1201:
1202: mul       $t0, $t0, 2
1203: lh        $t1, offset($t0)
1204:
1205: # Checks to see if there is already a piece there
1206: lb        $t9, gameBoard($t1)
1207: beq       $t9, 'O', winCondition02
1208:
1209: ############################################################
1210:
1211: add       $s5, $s5, 6
1212: lb        $t1, comb($s5)
1213: add       $s5, $s5, 1
```

```
1214: lb        $t2, comb($s5)
1215:
1216: sub       $t1, $t1, 48  # Grid
1217: sub       $t2, $t2, 'a' # Index
1218:
1219: div       $t0, $t2, 4
1220: mul       $t0, $t0, 16
1221:
1222: mul       $t1, $t1, 4
1223:
1224: add       $t0, $t0, $t1
1225: div       $t2, $t2, 4
1226: mfhi  $t2
1227: add       $t0, $t2, $t0
1228:
1229: mul       $t0, $t0, 2
1230: lh        $t1, offset($t0)
1231:
1232: # Checks to see if there is already a piece there
1233: lb        $t9, gameBoard($t1)
1234: beq       $t9, 'O', winCondition02
1235:
1236: ##########################################################
1237:
1238: add       $s5, $s5, 6
1239: lb        $t1, comb($s5)
1240: add       $s5, $s5, 1
1241: lb        $t2, comb($s5)
1242:
1243: sub       $t1, $t1, 48  # Grid
1244: sub       $t2, $t2, 'a' # Index
1245:
1246: div       $t0, $t2, 4
1247: mul       $t0, $t0, 16
1248:
1249: mul       $t1, $t1, 4
1250:
1251: add       $t0, $t0, $t1
1252: div       $t2, $t2, 4
1253: mfhi  $t2
1254: add       $t0, $t2, $t0
1255:
1256: mul       $t0, $t0, 2
1257: lh        $t1, offset($t0)
1258:
1259: # Checks to see if there is already a piece there
1260: lb        $t9, gameBoard($t1)
1261: beq       $t9, 'O', winCondition02
1262:
1263: ##########################################################
1264:
```

```
1265: add       $s5, $s5, 6
1266: lb        $t1, comb($s5)
1267: add       $s5, $s5, 1
1268: lb        $t2, comb($s5)
1269:
1270: sub       $t1, $t1, 48  # Grid
1271: sub       $t2, $t2, 'a' # Index
1272:
1273: div       $t0, $t2, 4
1274: mul       $t0, $t0, 16
1275:
1276: mul       $t1, $t1, 4
1277:
1278: add       $t0, $t0, $t1
1279: div       $t2, $t2, 4
1280: mfhi  $t2
1281: add       $t0, $t2, $t0
1282:
1283: mul       $t0, $t0, 2
1284: lh        $t1, offset($t0)
1285:
1286: # Checks to see if there is already a piece there
1287: lb        $t9, gameBoard($t1)
1288: beq       $t9, 'O', winCondition02
1289:
1290: ############################################################
1291:
1292: add       $s5, $s5, 6
1293: lb        $t1, comb($s5)
1294: add       $s5, $s5, 1
1295: lb        $t2, comb($s5)
1296:
1297: sub       $t1, $t1, 48  # Grid
1298: sub       $t2, $t2, 'a' # Index
1299:
1300: div       $t0, $t2, 4
1301: mul       $t0, $t0, 16
1302:
1303: mul       $t1, $t1, 4
1304:
1305: add       $t0, $t0, $t1
1306: div       $t2, $t2, 4
1307: mfhi  $t2
1308: add       $t0, $t2, $t0
1309:
1310: mul       $t0, $t0, 2
1311: lh        $t1, offset($t0)
1312:
1313: # Checks to see if there is already a piece there
1314: lb        $t9, gameBoard($t1)
1315: beq       $t9, 'O', winCondition02
```

```
1316:
1317: ############################################################
1318:
1319: add       $s5, $s5, 6
1320: lb        $t1, comb($s5)
1321: add       $s5, $s5, 1
1322: lb        $t2, comb($s5)
1323:
1324: sub       $t1, $t1, 48  # Grid
1325: sub       $t2, $t2, 'a' # Index
1326:
1327: div       $t0, $t2, 4
1328: mul       $t0, $t0, 16
1329:
1330: mul       $t1, $t1, 4
1331:
1332: add       $t0, $t0, $t1
1333: div       $t2, $t2, 4
1334: mfhi  $t2
1335: add       $t0, $t2, $t0
1336:
1337: mul       $t0, $t0, 2
1338: lh        $t1, offset($t0)
1339:
1340: # Checks to see if there is already a piece there
1341: lb        $t9, gameBoard($t1)
1342: beq       $t9, 'O', winCondition02
1343:
1344: j     continueGameX
1345:
1346: winCondition02:
1347:
1348: add       $s5, $s5, 1
1349: lb        $t1, comb($s5)
1350: add       $s5, $s5, 1
1351: lb        $t2, comb($s5)
1352:
1353: sub       $t1, $t1, 48  # Grid
1354: sub       $t2, $t2, 'a' # Index
1355:
1356: div       $t0, $t2, 4
1357: mul       $t0, $t0, 16
1358:
1359: mul       $t1, $t1, 4
1360:
1361: add       $t0, $t0, $t1
1362: div       $t2, $t2, 4
1363: mfhi  $t2
1364: add       $t0, $t2, $t0
1365:
1366: mul       $t0, $t0, 2
```

```
1367: lh        $t1, offset($t0)
1368:
1369: # Checks to see if there is already a piece there
1370: lb        $t9, gameBoard($t1)
1371: beq       $t9, 'O', winConditionO3
1372:
1373: j     continueGameX
1374:
1375: winConditionO3:
1376:
1377: add       $s5, $s5, 1
1378: lb        $t1, comb($s5)
1379: add       $s5, $s5, 1
1380: lb        $t2, comb($s5)
1381:
1382: sub       $t1, $t1, 48  # Grid
1383: sub       $t2, $t2, 'a' # Index
1384:
1385: div       $t0, $t2, 4
1386: mul       $t0, $t0, 16
1387:
1388: mul       $t1, $t1, 4
1389:
1390: add       $t0, $t0, $t1
1391: div       $t2, $t2, 4
1392: mfhi  $t2
1393: add       $t0, $t2, $t0
1394:
1395: mul       $t0, $t0, 2
1396: lh        $t1, offset($t0)
1397:
1398: # Checks to see if there is already a piece there
1399: lb        $t9, gameBoard($t1)
1400: beq       $t9, 'O', winScreenO
1401:
1402: j     continueGameX
1403:
1404: ##############################################
1405: ##################    Computer    ##############################
1406: ##############################################
1407:
1408: readCombComputerX: # $t0 = grid       $t1 = index
1409: # Equation: (Grid×16+Index)×48
1410: lb        $t0, computerPreviousGrid
1411:
1412: lb        $t1, computerPreviousIndex
1413:
1414: mul       $t2, $t0, 16
1415: add       $t2, $t2, $t1
1416: mul       $t2, $t2, 48
1417:
```

```
1418: lb        $t3, comb($t2)
1419:
1420: add       $t2, $t2, 1
1421: lb        $t4, comb($t2)
1422:
1423: # Saving the $t2 offset in # $s5 and $s0
1424: add       $s5, $t2, $zero
1425: add       $s0, $t2, $zero
1426:
1427: # $t3 = grid
1428: # $t4 = index
1429:
1430: sb        $t3, computerPreviousGrid
1431: sb        $t4, computerPreviousIndex
1432:
1433: j     winConditionComputerX1
1434:
1435: winConditionComputerX1:
1436:
1437: lb        $t1, ($s6)
1438: lb        $t2, ($s7)
1439:
1440: sub       $t1, $t1, 48  # Grid
1441: sub       $t2, $t2, 'a' # Index
1442:
1443: div       $t0, $t2, 4
1444: mul       $t0, $t0, 16
1445:
1446: mul       $t1, $t1, 4
1447:
1448: add       $t0, $t0, $t1
1449: div       $t2, $t2, 4
1450: mfhi  $t2
1451: add       $t0, $t2, $t0
1452:
1453: mul       $t0, $t0, 2
1454: lh        $t1, offset($t0)
1455:
1456: # Checks to see if there is already a piece there
1457: lb        $t9, gameBoard($t1)
1458: beq       $t9, 'X', winConditionComputerX2
1459:
1460: ###############################################
1461:
1462: add       $s5, $s5, 6
1463: lb        $t1, comb($s5)
1464: add       $s5, $s5, 1
1465: lb        $t2, comb($s5)
1466:
1467: sub       $t1, $t1, 48  # Grid
1468: sub       $t2, $t2, 'a' # Index
```

```
1469:
1470: div       $t0, $t2, 4
1471: mul       $t0, $t0, 16
1472:
1473: mul       $t1, $t1, 4
1474:
1475: add       $t0, $t0, $t1
1476: div       $t2, $t2, 4
1477: mfhi  $t2
1478: add       $t0, $t2, $t0
1479:
1480: mul       $t0, $t0, 2
1481: lh        $t1, offset($t0)
1482:
1483: # Checks to see if there is already a piece there
1484: lb        $t9, gameBoard($t1)
1485: beq       $t9, 'X', winConditionComputerX2
1486:
1487: ############################################################
1488:
1489: add       $s5, $s5, 6
1490: lb        $t1, comb($s5)
1491: add       $s5, $s5, 1
1492: lb        $t2, comb($s5)
1493:
1494: sub       $t1, $t1, 48  # Grid
1495: sub       $t2, $t2, 'a' # Index
1496:
1497: div       $t0, $t2, 4
1498: mul       $t0, $t0, 16
1499:
1500: mul       $t1, $t1, 4
1501:
1502: add       $t0, $t0, $t1
1503: div       $t2, $t2, 4
1504: mfhi  $t2
1505: add       $t0, $t2, $t0
1506:
1507: mul       $t0, $t0, 2
1508: lh        $t1, offset($t0)
1509:
1510: # Checks to see if there is already a piece there
1511: lb        $t9, gameBoard($t1)
1512: beq       $t9, 'X', winConditionComputerX2
1513:
1514: ############################################################
1515:
1516: add       $s5, $s5, 6
1517: lb        $t1, comb($s5)
1518: add       $s5, $s5, 1
1519: lb        $t2, comb($s5)
```

```
1520:
1521: sub       $t1, $t1, 48  # Grid
1522: sub       $t2, $t2, 'a' # Index
1523:
1524: div       $t0, $t2, 4
1525: mul       $t0, $t0, 16
1526:
1527: mul       $t1, $t1, 4
1528:
1529: add       $t0, $t0, $t1
1530: div       $t2, $t2, 4
1531: mfhi  $t2
1532: add       $t0, $t2, $t0
1533:
1534: mul       $t0, $t0, 2
1535: lh        $t1, offset($t0)
1536:
1537: # Checks to see if there is already a piece there
1538: lb        $t9, gameBoard($t1)
1539: beq       $t9, 'X', winConditionComputerX2
1540:
1541: ############################################################
1542:
1543: add       $s5, $s5, 6
1544: lb        $t1, comb($s5)
1545: add       $s5, $s5, 1
1546: lb        $t2, comb($s5)
1547:
1548: sub       $t1, $t1, 48  # Grid
1549: sub       $t2, $t2, 'a' # Index
1550:
1551: div       $t0, $t2, 4
1552: mul       $t0, $t0, 16
1553:
1554: mul       $t1, $t1, 4
1555:
1556: add       $t0, $t0, $t1
1557: div       $t2, $t2, 4
1558: mfhi  $t2
1559: add       $t0, $t2, $t0
1560:
1561: mul       $t0, $t0, 2
1562: lh        $t1, offset($t0)
1563:
1564: # Checks to see if there is already a piece there
1565: lb        $t9, gameBoard($t1)
1566: beq       $t9, 'X', winConditionComputerX2
1567:
1568: ############################################################
1569:
1570: add       $s5, $s5, 6
```

```
1571: lb        $t1, comb($s5)
1572: add       $s5, $s5, 1
1573: lb        $t2, comb($s5)
1574:
1575: sub       $t1, $t1, 48  # Grid
1576: sub       $t2, $t2, 'a' # Index
1577:
1578: div       $t0, $t2, 4
1579: mul       $t0, $t0, 16
1580:
1581: mul       $t1, $t1, 4
1582:
1583: add       $t0, $t0, $t1
1584: div       $t2, $t2, 4
1585: mfhi  $t2
1586: add       $t0, $t2, $t0
1587:
1588: mul       $t0, $t0, 2
1589: lh        $t1, offset($t0)
1590:
1591: # Checks to see if there is already a piece there
1592: lb        $t9, gameBoard($t1)
1593: beq       $t9, 'X', winConditionComputerX2
1594:
1595: ############################################################
1596:
1597: add       $s5, $s5, 6
1598: lb        $t1, comb($s5)
1599: add       $s5, $s5, 1
1600: lb        $t2, comb($s5)
1601:
1602: sub       $t1, $t1, 48  # Grid
1603: sub       $t2, $t2, 'a' # Index
1604:
1605: div       $t0, $t2, 4
1606: mul       $t0, $t0, 16
1607:
1608: mul       $t1, $t1, 4
1609:
1610: add       $t0, $t0, $t1
1611: div       $t2, $t2, 4
1612: mfhi  $t2
1613: add       $t0, $t2, $t0
1614:
1615: mul       $t0, $t0, 2
1616: lh        $t1, offset($t0)
1617:
1618: # Checks to see if there is already a piece there
1619: lb        $t9, gameBoard($t1)
1620: beq       $t9, 'X', winConditionComputerX2
1621:
```

```
1622: j     gameLoopO
1623:
1624: winConditionComputerX2:
1625:
1626: add       $s5, $s5, 1
1627: lb        $t1, comb($s5)
1628: add       $s5, $s5, 1
1629: lb        $t2, comb($s5)
1630:
1631: sub       $t1, $t1, 48  # Grid
1632: sub       $t2, $t2, 'a' # Index
1633:
1634: div       $t0, $t2, 4
1635: mul       $t0, $t0, 16
1636:
1637: mul       $t1, $t1, 4
1638:
1639: add       $t0, $t0, $t1
1640: div       $t2, $t2, 4
1641: mfhi  $t2
1642: add       $t0, $t2, $t0
1643:
1644: mul       $t0, $t0, 2
1645: lh        $t1, offset($t0)
1646:
1647: # Checks to see if there is already a piece there
1648: lb        $t9, gameBoard($t1)
1649: beq       $t9, 'X', winConditionComputerX3
1650:
1651: j     gameLoopO
1652:
1653: winConditionComputerX3:
1654:
1655: add       $s5, $s5, 1
1656: lb        $t1, comb($s5)
1657: add       $s5, $s5, 1
1658: lb        $t2, comb($s5)
1659:
1660: sub       $t1, $t1, 48  # Grid
1661: sub       $t2, $t2, 'a' # Index
1662:
1663: div       $t0, $t2, 4
1664: mul       $t0, $t0, 16
1665:
1666: mul       $t1, $t1, 4
1667:
1668: add       $t0, $t0, $t1
1669: div       $t2, $t2, 4
1670: mfhi  $t2
1671: add       $t0, $t2, $t0
1672:
```

```
1673: mul       $t0, $t0, 2
1674: lh        $t1, offset($t0)
1675:
1676: # Checks to see if there is already a piece there
1677: lb        $t9, gameBoard($t1)
1678: beq       $t9, 'X', winScreenComputerX
1679:
1680: j     gameLoopO
1681:
1682: winScreenComputerX:
1683: li        $v0, 4
1684: la        $a0, gameBoard
1685: syscall
1686:
1687: li        $v0, 4
1688: la        $a0, xWon
1689: syscall
1690:
1691: li        $v0, 4
1692: la        $a0, computerWon
1693: syscall
1694:
1695: j     newGame
1696:
1697:
1698: ##################
1699:
1700: readCombComputerO: # $t0 = grid       $t1 = index
1701: # Equation: (Grid×16+Index)×48
1702: lb        $t0, computerPreviousGrid
1703: lb        $t1, computerPreviousIndex
1704:
1705: mul       $t2, $t0, 16
1706: add       $t2, $t2, $t1
1707: mul       $t2, $t2, 48
1708:
1709: lb        $t3, comb($t2)
1710:
1711: add       $t2, $t2, 1
1712: lb        $t4, comb($t2)
1713:
1714:
1715: # Saving the $t2 offset in # $s5 and $s0
1716: add       $s5, $t2, $zero
1717: add       $s0, $t2, $zero
1718:
1719:
1720:
1721: # $t3 = grid
1722: # $t4 = index
1723:
```

```
1724: sb        $t3, computerPreviousGrid
1725: sb        $t4, computerPreviousIndex
1726:
1727: j     winConditionComputerO1
1728:
1729: winConditionComputerO1:
1730:
1731: lb        $t1, ($s6)
1732: lb        $t2, ($s7)
1733:
1734: sub       $t1, $t1, 48  # Grid
1735: sub       $t2, $t2, 'a' # Index
1736:
1737: div       $t0, $t2, 4
1738: mul       $t0, $t0, 16
1739:
1740: mul       $t1, $t1, 4
1741:
1742: add       $t0, $t0, $t1
1743: div       $t2, $t2, 4
1744: mfhi  $t2
1745: add       $t0, $t2, $t0
1746:
1747: mul       $t0, $t0, 2
1748: lh        $t1, offset($t0)
1749:
1750: # Checks to see if there is already a piece there
1751: lb        $t9, gameBoard($t1)
1752: beq       $t9, 'O', winConditionComputerO2
1753:
1754: ################################################
1755:
1756: add       $s5, $s5, 6
1757: lb        $t1, comb($s5)
1758: add       $s5, $s5, 1
1759: lb        $t2, comb($s5)
1760:
1761: sub       $t1, $t1, 48  # Grid
1762: sub       $t2, $t2, 'a' # Index
1763:
1764: div       $t0, $t2, 4
1765: mul       $t0, $t0, 16
1766:
1767: mul       $t1, $t1, 4
1768:
1769: add       $t0, $t0, $t1
1770: div       $t2, $t2, 4
1771: mfhi  $t2
1772: add       $t0, $t2, $t0
1773:
1774: mul       $t0, $t0, 2
```

```
1775: lh        $t1, offset($t0)
1776:
1777: # Checks to see if there is already a piece there
1778: lb        $t9, gameBoard($t1)
1779: beq       $t9, 'O', winConditionComputerO2
1780:
1781: ############################################################
1782:
1783: add       $s5, $s5, 6
1784: lb        $t1, comb($s5)
1785: add       $s5, $s5, 1
1786: lb        $t2, comb($s5)
1787:
1788: sub       $t1, $t1, 48  # Grid
1789: sub       $t2, $t2, 'a' # Index
1790:
1791: div       $t0, $t2, 4
1792: mul       $t0, $t0, 16
1793:
1794: mul       $t1, $t1, 4
1795:
1796: add       $t0, $t0, $t1
1797: div       $t2, $t2, 4
1798: mfhi  $t2
1799: add       $t0, $t2, $t0
1800:
1801: mul       $t0, $t0, 2
1802: lh        $t1, offset($t0)
1803:
1804: # Checks to see if there is already a piece there
1805: lb        $t9, gameBoard($t1)
1806: beq       $t9, 'O', winConditionComputerO2
1807:
1808: ############################################################
1809:
1810: add       $s5, $s5, 6
1811: lb        $t1, comb($s5)
1812: add       $s5, $s5, 1
1813: lb        $t2, comb($s5)
1814:
1815: sub       $t1, $t1, 48  # Grid
1816: sub       $t2, $t2, 'a' # Index
1817:
1818: div       $t0, $t2, 4
1819: mul       $t0, $t0, 16
1820:
1821: mul       $t1, $t1, 4
1822:
1823: add       $t0, $t0, $t1
1824: div       $t2, $t2, 4
1825: mfhi  $t2
```

```
1826: add       $t0, $t2, $t0
1827:
1828: mul       $t0, $t0, 2
1829: lh        $t1, offset($t0)
1830:
1831: # Checks to see if there is already a piece there
1832: lb        $t9, gameBoard($t1)
1833: beq       $t9, 'O', winConditionComputerO2
1834:
1835: ############################################################
1836:
1837: add       $s5, $s5, 6
1838: lb        $t1, comb($s5)
1839: add       $s5, $s5, 1
1840: lb        $t2, comb($s5)
1841:
1842: sub       $t1, $t1, 48  # Grid
1843: sub       $t2, $t2, 'a' # Index
1844:
1845: div       $t0, $t2, 4
1846: mul       $t0, $t0, 16
1847:
1848: mul       $t1, $t1, 4
1849:
1850: add       $t0, $t0, $t1
1851: div       $t2, $t2, 4
1852: mfhi  $t2
1853: add       $t0, $t2, $t0
1854:
1855: mul       $t0, $t0, 2
1856: lh        $t1, offset($t0)
1857:
1858: # Checks to see if there is already a piece there
1859: lb        $t9, gameBoard($t1)
1860: beq       $t9, 'O', winConditionComputerO2
1861:
1862: ############################################################
1863:
1864: add       $s5, $s5, 6
1865: lb        $t1, comb($s5)
1866: add       $s5, $s5, 1
1867: lb        $t2, comb($s5)
1868:
1869: sub       $t1, $t1, 48  # Grid
1870: sub       $t2, $t2, 'a' # Index
1871:
1872: div       $t0, $t2, 4
1873: mul       $t0, $t0, 16
1874:
1875: mul       $t1, $t1, 4
1876:
```

```
1877: add       $t0, $t0, $t1
1878: div       $t2, $t2, 4
1879: mfhi  $t2
1880: add       $t0, $t2, $t0
1881:
1882: mul       $t0, $t0, 2
1883: lh        $t1, offset($t0)
1884:
1885: # Checks to see if there is already a piece there
1886: lb        $t9, gameBoard($t1)
1887: beq       $t9, 'O', winConditionComputerO2
1888:
1889: ############################################################
1890:
1891: add       $s5, $s5, 6
1892: lb        $t1, comb($s5)
1893: add       $s5, $s5, 1
1894: lb        $t2, comb($s5)
1895:
1896: sub       $t1, $t1, 48  # Grid
1897: sub       $t2, $t2, 'a' # Index
1898:
1899: div       $t0, $t2, 4
1900: mul       $t0, $t0, 16
1901:
1902: mul       $t1, $t1, 4
1903:
1904: add       $t0, $t0, $t1
1905: div       $t2, $t2, 4
1906: mfhi  $t2
1907: add       $t0, $t2, $t0
1908:
1909: mul       $t0, $t0, 2
1910: lh        $t1, offset($t0)
1911:
1912: # Checks to see if there is already a piece there
1913: lb        $t9, gameBoard($t1)
1914: beq       $t9, 'O', winConditionComputerO2
1915:
1916: j     gameLoopX
1917:
1918: winConditionComputerO2:
1919:
1920: add       $s5, $s5, 1
1921: lb        $t1, comb($s5)
1922: add       $s5, $s5, 1
1923: lb        $t2, comb($s5)
1924:
1925: sub       $t1, $t1, 48  # Grid
1926: sub       $t2, $t2, 'a' # Index
1927:
```

```
1928: div       $t0, $t2, 4
1929: mul       $t0, $t0, 16
1930:
1931: mul       $t1, $t1, 4
1932:
1933: add       $t0, $t0, $t1
1934: div       $t2, $t2, 4
1935: mfhi  $t2
1936: add       $t0, $t2, $t0
1937:
1938: mul       $t0, $t0, 2
1939: lh        $t1, offset($t0)
1940:
1941: # Checks to see if there is already a piece there
1942: lb        $t9, gameBoard($t1)
1943: beq       $t9, 'O', winConditionComputerO3
1944:
1945: j     gameLoopX
1946:
1947: winConditionComputerO3:
1948:
1949: add       $s5, $s5, 1
1950: lb        $t1, comb($s5)
1951: add       $s5, $s5, 1
1952: lb        $t2, comb($s5)
1953:
1954: sub       $t1, $t1, 48  # Grid
1955: sub       $t2, $t2, 'a' # Index
1956:
1957: div       $t0, $t2, 4
1958: mul       $t0, $t0, 16
1959:
1960: mul       $t1, $t1, 4
1961:
1962: add       $t0, $t0, $t1
1963: div       $t2, $t2, 4
1964: mfhi  $t2
1965: add       $t0, $t2, $t0
1966:
1967: mul       $t0, $t0, 2
1968: lh        $t1, offset($t0)
1969:
1970: # Checks to see if there is already a piece there
1971: lb        $t9, gameBoard($t1)
1972: beq       $t9, 'O', winScreenComputerO
1973:
1974: j     gameLoopX
1975:
1976: winScreenComputerO:
1977: li        $v0, 4
1978: la        $a0, gameBoard
```

```
1979: syscall
1980:
1981: li        $v0, 4
1982: la        $a0, oWon
1983: syscall
1984:
1985: li        $v0, 4
1986: la        $a0, computerWon
1987: syscall
1988:
1989: j     newGame
```