```cpp
/*****************************************************************************
 * PROGRAMMER : Ali Eshghi
 * STUDENT ID : 1112261
 * CLASS      : CS1B
 * SECTION    : MW 7:30pm
 * Assign #2  : tic-tac-toe game (multi-dimensional arrays)
 * DUE DATE   : 19 September 2019
 *****************************************************************************/
#include "MyHeader.h"

/*****************************************************************************
 * GetAndCheckInp
 *  This functions gets each player's play and checks if the inputed numbers are
 *  in the domain of the row and column of the game. also it checks if the
 *  row and column in the board is empty or no
 *
 * RETURNS: nothing
 *          it puts the players token in the boardAr
 *****************************************************************************/
void GetAndCheckInp(char   boardAr[][3],
                    char   token,
                    string playerX,
                    string playerO,
                    int    option,
                    char   tokenChoice,
                    char   compToken)
{
    /*************
     * VARIABLES *
     *************/

    bool    valid;          //PROCESS - to exit the loop
    boolxPlayersTurn;   //PROCESS - if player token is X, player's turn
    booloPlayersTurn;   //PROCESS - if player token is O, player's turn

    int     row;        //IN & PROCESS - row input for the token
    int     col;        //IN & PROCESS - col input for the token


    /*****************************************************************************
    * PROCESS - determines whose turn is it and if the input is valid or no
    *****************************************************************************/
    valid = false;

    xPlayersTurn = ((token == toupper(tokenChoice))
                 &&  (toupper(tokenChoice) == 'X'));

    oPlayersTurn = ((token == toupper(tokenChoice))
                 &&  (toupper(tokenChoice) == 'O'));



    if (option == 1) //single player
    {
        do
```

```
56          {
57              /*************************************************************************
58              * INPUT & PROCESS - gets the input for the row and the col from the
59              *                   player and checks if it is valid
60              *************************************************************************/
61              if (xPlayersTurn)
62              {
63                  cout << playerX << "\'s turn! What is your play?: ";
64                  cin  >> row   >> col;
65
66                  row--;
67                  col--;
68
69                  if(row > ROW_SIZE - 1 || row < 0)
70                  {
71                      cout << "Invalid row - Please try again! \n";
72                  }
73                  else if(col > COL_SIZE - 1 || col < 0)
74                  {
75                      cout << "Invalid column - please try again!\n";
76                  }
77                  else if(!isspace(boardAr[row][col]))
78                  {
79                      cout << "That spot is already taken - try again\n";
80                  }
81                  else
82                  {
83                      valid = true;
84
85
86                      boardAr[row][col] = token;
87                  }
88
89
90              }
91
92              else if (oPlayersTurn)
93              {
94                  cout << playerO << "\'s turn! What is your play?: ";
95                  cin  >> row   >> col;
96
97                  row--;
98                  col--;
99
100                 if(row > ROW_SIZE - 1 || row < 0)
101                 {
102                     cout << "Invalid row - Please try again! \n";
103                 }
104                 else if(col > COL_SIZE - 1 || col < 0)
105                 {
106                     cout << "Invalid column - please try again!\n";
107                 }
108                 else if(!isspace(boardAr[row][col]))
109                 {
110                     cout << "That spot is already taken - try again\n";
```

```
111                 }
112                 else
113                 {
114                     valid = true;
115                     boardAr[row][col] = token;
116                 }
117 
118             }
119 
120 
121             /*****************************************************************
122             * PROCESS - if the player plays before computer, the computer puts
123             *           its token based on the position of the last play by the
124             *           player, and if it is the computer's turn to play and
125             *           there is no threat for the computer, the computer tries
126             *           to put the token for winning and if the computer sees
127             *           any threat from the player, it blocks the player.
128             *****************************************************************/
129             else if(!xPlayersTurn && !oPlayersTurn)
130             {
131                 cout << "Master's turn..." << endl;
132 
133 
134 
135                 /*****************************************************************
136                 * PROCESS - there are 24 ways that the computer could win and if
137                 *           the computer sees any of those 24 ways available and
138                 *           to win the game. the computer puts its token to the
139                 *           board.
140                 *****************************************************************/
141 
142 
143                 if((boardAr[0][0] == compToken)
144                 && (boardAr[2][0] == compToken)
145                 &&  isspace(boardAr[1][0])) //1
146                 {
147                     boardAr[1][0] = compToken;
148 
149 
150                 }
151                 else if((boardAr[0][1] == compToken)
152                      && (boardAr[2][1] == compToken)
153                      &&  isspace(boardAr[1][1])) //2
154                 {
155                     boardAr[1][1] = compToken;
156 
157 
158                 }
159                 else if((boardAr[0][2] == compToken)
160                      && (boardAr[2][2] == compToken)
161                      &&  isspace(boardAr[1][2])) //3
162                 {
163                     boardAr[1][2] = compToken;
164 
165 
```

```
166                 }
167                 else if((boardAr[0][0] == compToken)
168                      && (boardAr[0][2] == compToken)
169                      && isspace(boardAr[0][2])) //4
170                 {
171                     boardAr[0][1] = compToken;
172
173
174                 }
175                 else if((boardAr[1][0] == compToken)
176                      && (boardAr[1][2] == compToken)
177                      && isspace(boardAr[1][1])) //5
178                 {
179                     boardAr[1][1] = compToken;
180
181
182                 }
183                 else if((boardAr[2][0] == compToken)
184                      && (boardAr[2][2] == compToken)
185                      && isspace(boardAr[2][1])) //6
186                 {
187                     boardAr[2][1] = compToken;
188
189
190                 }
191                 else if((boardAr[0][0] == compToken)
192                      && (boardAr[2][2] == compToken)
193                      && isspace(boardAr[1][1])) //7
194                 {
195                     boardAr[1][1] = compToken;
196
197
198                 }
199                 else if((boardAr[0][2] == compToken)
200                      && (boardAr[2][0] == compToken)
201                      && isspace(boardAr[1][1])) //8
202                 {
203                     boardAr[1][1] = compToken;
204
205
206                 }
207                 else if((boardAr[1][0] == compToken)
208                      && (boardAr[2][0] == compToken)
209                      && isspace(boardAr[0][0])) //9
210                 {
211                     boardAr[0][0] = compToken;
212
213
214                 }
215                 else if((boardAr[1][1] == compToken)
216                      && (boardAr[2][1] == compToken)
217                      && isspace(boardAr[0][1])) //10
218                 {
219                     boardAr[0][1] = compToken;
220
```

```
221
222                 }
223                 else if((boardAr[1][2] == compToken)
224                      && (boardAr[2][2] == compToken)
225                      && isspace(boardAr[0][2])) //11
226                 {
227                     boardAr[0][2] = compToken;
228
229
230                 }
231                 else if((boardAr[0][1] == compToken)
232                      && (boardAr[0][2] == compToken)
233                      && isspace(boardAr[0][0])) //12
234                 {
235                     boardAr[0][0] = compToken;
236
237
238                 }
239                 else if((boardAr[1][1] == compToken)
240                      && (boardAr[1][2] == compToken)
241                      && isspace(boardAr[1][0])) //13
242                 {
243                     boardAr[1][0] = compToken;
244
245
246                 }
247                 else if((boardAr[2][1] == compToken)
248                      && (boardAr[2][2] == compToken)
249                      && isspace(boardAr[2][0])) //14
250                 {
251                     boardAr[2][0] = compToken;
252
253
254                 }
255                 else if((boardAr[1][1] == compToken)
256                      && (boardAr[2][2] == compToken)
257                      && isspace(boardAr[0][0])) //15
258                 {
259                     boardAr[0][0] = compToken;
260
261
262                 }
263                 else if((boardAr[1][1] == compToken)
264                      && (boardAr[2][0] == compToken)
265                      && isspace(boardAr[0][2])) //16
266                 {
267                     boardAr[0][2] = compToken;
268
269
270                 }
271                 else if((boardAr[0][0] == compToken)
272                      && (boardAr[1][0] == compToken)
273                      && isspace(boardAr[2][0])) //17
274                 {
275                     boardAr[2][0] = compToken;
```

```
276
277
278                 }
279                 else if((boardAr[0][1] == compToken)
280                      && (boardAr[1][1] == compToken)
281                      && isspace(boardAr[2][1])) //18
282                 {
283                     boardAr[2][1] = compToken;
284
285
286                 }
287                 else if((boardAr[0][2] == compToken)
288                      && (boardAr[1][2] == compToken)
289                      && isspace(boardAr[2][2])) //19
290                 {
291                     boardAr[2][2] = compToken;
292
293
294                 }
295                 else if((boardAr[0][0] == compToken)
296                      && (boardAr[0][1] == compToken)
297                      && isspace(boardAr[0][2])) //20
298                 {
299                     boardAr[0][2] = compToken;
300
301
302                 }
303                 else if((boardAr[1][0] == compToken)
304                      && (boardAr[1][1] == compToken)
305                      && isspace(boardAr[1][2])) //21
306                 {
307                     boardAr[1][2] = compToken;
308
309
310                 }
311                 else if((boardAr[2][0] == compToken)
312                      && (boardAr[2][1] == compToken)
313                      &&  isspace(boardAr[2][2])) //22
314                 {
315                     boardAr[2][2] = compToken;
316
317
318                 }
319                 else if((boardAr[0][0] == compToken)
320                      && (boardAr[1][1] == compToken)
321                      &&  isspace(boardAr[2][2])) //23
322                 {
323                     boardAr[2][2] = compToken;
324
325
326                 }
327                 else if((boardAr[0][2] == compToken)
328                      && (boardAr[1][1] == compToken)
329                      &&  isspace(boardAr[2][0])) //24
330                 {
```

```
331                     boardAr[2][0] = compToken;
332
333
334                 }
335                 /*****************************************************************
336                 * PROCESS - there are 24 ways that the computer can feel the
337                 *           threat from the player and if there were no way to
338                 *           win before the player could put the token, the
339                 *           computer puts the token in the place to block the
340                 *           player from the winning
341                 *****************************************************************/
342                 else if((boardAr[0][0] == toupper(tokenChoice))
343                      && (boardAr[2][0] == toupper(tokenChoice))
344                      &&  isspace(boardAr[1][0]))
345                 {
346                     boardAr[1][0] = compToken;
347
348                 }
349                 else if((boardAr[0][1] == toupper(tokenChoice))
350                      && (boardAr[2][1] == toupper(tokenChoice))
351                      && isspace(boardAr[1][1]))
352                 {
353                     boardAr[1][1] = compToken;
354
355                 }
356                 else if((boardAr[0][2] == toupper(tokenChoice))
357                      && (boardAr[2][2] == toupper(tokenChoice))
358                      && isspace(boardAr[1][2]))
359                 {
360                     boardAr[1][2] = compToken;
361
362                 }
363                 else if((boardAr[0][0] == toupper(tokenChoice))
364                      && (boardAr[0][2] == toupper(tokenChoice))
365                      && isspace(boardAr[0][2]))
366                 {
367                     boardAr[0][1] = compToken;
368
369                 }
370                 else if((boardAr[1][0] == toupper(tokenChoice))
371                      && (boardAr[1][2] == toupper(tokenChoice))
372                      && isspace(boardAr[1][1]))
373                 {
374                     boardAr[1][1] = compToken;
375
376                 }
377                 else if((boardAr[2][0] == toupper(tokenChoice))
378                      && (boardAr[2][2] == toupper(tokenChoice))
379                      && isspace(boardAr[2][1]))
380                 {
381                     boardAr[2][1] = compToken;
382
383                 }
384                 else if((boardAr[0][0] == toupper(tokenChoice))
385                      && (boardAr[2][2] == toupper(tokenChoice))
```

```
386                      && isspace(boardAr[1][1]))
387                 {
388                     boardAr[1][1] = compToken;
389 
390                 }
391                 else if((boardAr[0][2] == toupper(tokenChoice))
392                      && (boardAr[2][0] == toupper(tokenChoice))
393                      && isspace(boardAr[1][1]))
394                 {
395                     boardAr[1][1] = compToken;
396 
397                 }
398                 else if((boardAr[1][0] == toupper(tokenChoice))
399                      && (boardAr[2][0] == toupper(tokenChoice))
400                      && isspace(boardAr[0][0]))
401                 {
402                     boardAr[0][0] = compToken;
403 
404                 }
405                 else if((boardAr[1][1] == toupper(tokenChoice))
406                      && (boardAr[2][1] == toupper(tokenChoice))
407                      && isspace(boardAr[0][1]))
408                 {
409                     boardAr[0][1] = compToken;
410 
411                 }
412                 else if((boardAr[1][2] == toupper(tokenChoice))
413                      && (boardAr[2][2] == toupper(tokenChoice))
414                      && isspace(boardAr[0][2]))
415                 {
416                     boardAr[0][2] = compToken;
417 
418                 }
419                 else if((boardAr[0][1] == toupper(tokenChoice))
420                      && (boardAr[0][2] == toupper(tokenChoice))
421                      && isspace(boardAr[0][0]))
422                 {
423                     boardAr[0][0] = compToken;
424 
425                 }
426                 else if((boardAr[1][1] == toupper(tokenChoice))
427                      && (boardAr[1][2] == toupper(tokenChoice))
428                      && isspace(boardAr[1][0]))
429                 {
430                     boardAr[1][0] = compToken;
431 
432                 }
433                 else if((boardAr[2][1] == toupper(tokenChoice))
434                      && (boardAr[2][2] == toupper(tokenChoice))
435                      && isspace(boardAr[2][0]))
436                 {
437                     boardAr[2][0] = compToken;
438 
439                 }
440                 else if((boardAr[1][1] == toupper(tokenChoice))
```

```
441                      && (boardAr[2][2] == toupper(tokenChoice))
442                      && isspace(boardAr[0][0]))
443                 {
444                     boardAr[0][0] = compToken;
445
446                 }
447                 else if((boardAr[1][1] == toupper(tokenChoice))
448                      && (boardAr[2][0] == toupper(tokenChoice))
449                      && isspace(boardAr[0][2]))
450                 {
451                     boardAr[0][2] = compToken;
452
453                 }
454                 else if((boardAr[0][0] == toupper(tokenChoice))
455                      && (boardAr[1][0] == toupper(tokenChoice))
456                      &&  isspace(boardAr[2][0]))
457                 {
458                     boardAr[2][0] = compToken;
459
460                 }
461                 else if((boardAr[0][1] == toupper(tokenChoice))
462                      && (boardAr[1][1] == toupper(tokenChoice))
463                      && isspace(boardAr[2][1]))
464                 {
465                     boardAr[2][1] = compToken;
466
467                 }
468                 else if((boardAr[0][2] == toupper(tokenChoice))
469                      && (boardAr[1][2] == toupper(tokenChoice))
470                      && isspace(boardAr[2][2]))
471                 {
472                     boardAr[2][2] = compToken;
473
474                 }
475                 else if((boardAr[0][0] == toupper(tokenChoice))
476                      && (boardAr[0][1] == toupper(tokenChoice))
477                      && isspace(boardAr[0][2]))
478                 {
479                     boardAr[0][2] = compToken;
480
481                 }
482                 else if((boardAr[1][0] == toupper(tokenChoice))
483                      && (boardAr[1][1] == toupper(tokenChoice))
484                      && isspace(boardAr[1][2]))
485                 {
486                     boardAr[1][2] = compToken;
487
488                 }
489                 else if((boardAr[2][0] == toupper(tokenChoice))
490                      && (boardAr[2][1] == toupper(tokenChoice))
491                      && isspace(boardAr[2][2]))
492                 {
493                     boardAr[2][2] = compToken;
494
495                 }
```

```
496                 else if((boardAr[0][0] == toupper(tokenChoice))
497                      && (boardAr[1][1] == toupper(tokenChoice))
498                      && isspace(boardAr[2][2]))
499                 {
500                     boardAr[2][2] = compToken;
501 
502                 }
503                 else if((boardAr[0][2] == toupper(tokenChoice))
504                      && (boardAr[1][1] == toupper(tokenChoice))
505                      && isspace(boardAr[2][0]))
506                 {
507                     boardAr[2][0] = compToken;
508 
509                 }
510 
511                 /*****************************************************************
512                 * PROCESS - if the computer determines no chance to win or
513                 *           determines no threat from the player, it randomly
514                 *           puts a token on the board.
515                 *****************************************************************/
516                 else
517                 {
518 
519                     srand(time(NULL));
520 
521                     row = rand() % 3 + 1;
522                     col = rand() % 3 + 1;
523 
524 
525                     row--;
526                     col--;
527 
528                     while(!isspace(boardAr[row][col]))
529                     {
530                         srand(time(NULL));
531 
532                         row = rand() % 3 + 1;
533                         col = rand() % 3 + 1;
534 
535 
536                         row--;
537                         col--;
538 
539                     }
540 
541                     if(isspace(boardAr[row][col]))
542                     {
543 
544 
545                         boardAr[row][col] = compToken;
546                         valid = true;
547 
548                     }
549 
550 
```

```
551
552                     else if(!isspace(boardAr[row - 1][col - 1]))
553                     {
554                         while(!isspace(boardAr[row - 1][col - 1]))
555                         {
556                             if(row == 1)
557                             {
558                                 srand(time(NULL));
559
560                                 row = rand() % 2 + 1;
561
562                                 if(row == 1)
563                                 {
564                                     row = 2;
565                                 }
566                                 if(row == 2)
567                                 {
568                                     row = 3;
569                                 }
570
571                                 row--;
572                                 col--;
573
574                                 boardAr[row][col] = compToken;
575                                 valid = true;
576                             }
577
578                             else if(row == 2)
579                             {
580                                 srand(time(NULL));
581
582                                 row = rand() % 2 + 1;
583
584                                 if(row == 1)
585                                 {
586                                     row = 1;
587                                 }
588                                 if(row == 2)
589                                 {
590                                     row = 3;
591                                 }
592
593                                 row--;
594                                 col--;
595
596                                 boardAr[row][col] = compToken;
597                                 valid = true;
598                             }
599
600                             else if(row == 3)
601                             {
602                                 srand(time(NULL));
603
604                                 row = rand() % 2 + 1;
605
```

```
606                         if(row == 1)
607                         {
608                             row = 1;
609                         }
610                         if(row == 2)
611                         {
612                             row = 2;
613                         }
614
615                         row--;
616                         col--;
617
618                         boardAr[row][col] = compToken;
619                         valid = true;
620                     }
621
622
623                     else if(col == 1)
624                     {
625                         srand(time(NULL));
626
627                         row = rand() % 2 + 1;
628
629                         if(col == 1)
630                         {
631                             col = 2;
632                         }
633                         if(row == 2)
634                         {
635                             col = 3;
636                         }
637
638                         row--;
639                         col--;
640
641                         boardAr[row][col] = compToken;
642                         valid = true;
643                     }
644
645                     else if(col == 2)
646                     {
647                         srand(time(NULL));
648
649                         col = rand() % 2 + 1;
650
651                         if(col == 1)
652                         {
653                             col = 1;
654                         }
655                         if(col == 2)
656                         {
657                             col = 3;
658                         }
659
660                         row--;
```

```
661                             col--;
662 
663                             boardAr[row][col] = compToken;
664                             valid = true;
665                         }
666 
667                         else if(col == 3)
668                         {
669                             srand(time(NULL));
670 
671                             col = rand() % 2 + 1;
672 
673                             if(col == 1)
674                             {
675                                 col = 1;
676                             }
677                             if(col == 2)
678                             {
679                                 col = 2;
680                             }
681 
682                             row--;
683                             col--;
684 
685                             boardAr[row][col] = compToken;
686                             valid = true;
687                         }
688                     }
689                 }
690             }
691             valid = true;
692         }
693     }while(!valid);
694 }
695 
696 
697 
698 
699 else if(option == 2) // multiplayer
700 {
701 
702     do
703     {
704         /*************************************************************************
705         * INPUT & PROCESS - gets the input for the row and the col from the
706         *                   players and checks if it is valid
707         *************************************************************************/
708         if(token == 'X')
709         {
710             cout << playerX;
711         }
712         else if(token == 'O')
713         {
714             cout << playerO;
715         }
```

```
716
717             cout << "\'s turn! What is your play?: ";
718             cin  >> row   >> col;
719
720             row--;
721             col--;
722
723             if(row > ROW_SIZE - 1 || row < 0)
724             {
725                 cout << "Invalid row - Please try again! \n";
726             }
727             else if(col > COL_SIZE - 1 || col < 0)
728             {
729                 cout << "Invalid column - please try again!\n";
730             }
731             else if(!isspace(boardAr[row][col]))
732             {
733                 cout << "That spot is already taken - try again\n";
734             }
735             else
736             {
737                 valid = true;
738             }
739         }while(!valid);
740
741         boardAr[row][col] = token;
742         cin.ignore(10000,'\n');
743
744     }
745     //clear();
746 }
747
```