

Jogo optado: Jogo da Velha 3x3

Nome dos integrantes:

Matheus Farias Porto Dos Anjos

RGM: 31259821

Ryan Do Nascimento Bezerra

RGM: 31268684

Fábio Harley Silva Filho

RGM: 30290759

Gabriel Henriques Cassiano

RGM: 32096143

Introdução:

O jogo da velha é um jogo para dois jogadores, a estrutura do jogo é apenas um tabuleiro 3x3 onde os jogadores marcam 'X' ou 'O' em cada campo do tabuleiro. Para vencer o jogo, um dos jogadores deve preencher uma linha, uma coluna ou uma diagonal inteira com 'X', ou 'O', o primeiro que fizer isso vence, e o jogo termina, caso o tabuleiro fique totalmente preenchido e ninguém ganhar, teremos um empate.

Descrição geral do jogo:

O jogo da velha desenvolvido possui um menu inicial onde o jogador pode escolher:

- **Jogar:**

Abre mais duas opções de escolha, uma para o modo jogador vs jogador, e outra para o modo jogador vs computador;

- **Ver Ranking Geral:**

Log de todas as partidas jogadas após a primeira execução do script;

- **Créditos:**

Nome dos desenvolvedores do algoritmo;

- **Sair:**

Opção de terminar a execução do script;

Para jogar, a cada jogada precisamos fornecer dois números que representam a posição do tabuleiro em que queremos jogar: a coluna e a linha, nessa ordem. As linhas, bem como as colunas, variam de 0 até 2 e somente os locais vazios podem ser escolhidos como jogada.

Demonstração do código:

1. Interação com o Jogador em jogo

```
do{
    printf("\n %s - Coloque a coluna: ", dados[i].jogador);
    retorno = scanf("%d",&c);
    printf(" %s - Coloque a linha: ", dados[i].jogador);
    retorno = scanf("%d",&l);
do{
    letra = getchar();
    printf("%c", letra);
}while(letra != '\n');
}while(retorno == 0 || c < 0 || c >= 3 || l < 0 || l >= 3);

if((matriz[l][c] == 'O') || (matriz[l][c] == 'X')){
    printf("\nPosicao Ocupada");
    sleep(1000);
    jogadas--;
    break;
}else if(jogadas %2 == 1){
    matriz[l][c] = 'X';
    sair = 1;
}else if(jogadas %2 == 0){
    matriz[l][c] = 'O';
    sair = 1;
}
```

Figura 1 - Entrada de Dados, Linha e Coluna

Tivemos que limpar o buffer do teclado para evitar o loop infinito no caso do jogador digitar sem querer alguma letra e isso pode ser resolvido com a função `getchar()`. O último dado no buffer do teclado sempre será um “\n”, que representa a tecla ENTER. Assim, basta ler caractere por caractere enquanto o caractere lido for diferente de “\n”, como apresentado no trecho de código a seguir, limpando o buffer e dando oportunidade para o usuário digitar novamente. O `while` de `retorno == 0` foi modificado conforme os

limites impostos por nós, como visto acima, caso o jogador digite qualquer número diferente de 0, 1, 2, haverá o retorno de colocar a linha e a coluna novamente

Uma área de armazenamento temporário é chamada de **buffer**. Todos os dispositivos de entrada e saída padrão contêm um **buffer** de entrada e saída.

2. Dados de Vitórias em arquivo

```
void rank(){
    FILE *p;
    char str[80] = "test0.txt";
    char f;

    p = fopen(str,"a+");

    if(dados[0].vitorias >0 || dados[1].vitorias >0 || test == 1){
        fprintf(p,"%s ganhou %d vezes\n", dados[0].jogador, dados[0].vitorias);
        fprintf(p,"%s ganhou %d vezes\n", dados[1].jogador, dados[1].vitorias);
        fclose(p);
    }else{
        p = fopen(str,"a+");
        while (!feof(p)){
            fscanf(p,"%c",&f);
            printf("%c",f);
        }
        fclose(p);
    }
}
```

Figura 2 – Área de Ficheiros

O “**void rank()**” foi definido como uma função que salva os dados de vitórias dos jogadores/computador na partida armazenados em uma struct, esses dados são salvos no arquivo “ranking” toda vez que uma nova partida é concluída, se encerrada pela opção “Sair (4)” (Partidas de uma execução anterior da atual também ficam salvas com o intuito de ser um ranking geral de todas as partidas já jogadas)

3. Jogador VS Computador

Um dos problemas que tivemos foi relacionado ao “BOT” que jogaria contra o usuário caso não houvesse segundo jogador. Inicialmente, pensamos em apenas fazer com que esse bot gerasse números aleatórios de 0 a 2, tanto para linha quanto para coluna, assim, definindo o local da jogada do computador. Segue a imagem:

```

while(sair == 0){
    if(bot == 1 && jogadas % 2 == 0){

        srand (time(NULL)); //BOT
        l = rand() % 3 + 0;
        c = rand() % 3 + 0;

        bot_tentativa++;

        if(bot_tentativa > 5){
            for(l = 0; l < 3; l++){
                for(c = 0; c < 3; c++){
                    if(matriz[l][c] == ' '){
                        matriz[l][c] = 'O';
                        l = 3;
                        c = 3;
                        bot_tentativa = 0;
                        sair = 1;
                    }
                }
            }

        }else if(matriz[l][c] == 'O' || matriz[l][c] == 'X'){
            jogadas--;
            break;
        }else if(jogadas % 2 == 0){
            matriz[l][c] = 'O';
            sair = 1;
        }
    }
}

```

Figura 3 - Área do BOT

Para que isso fosse possível, utilizamos a função *srand()* para geração de números aleatórios, mas após alguns testes, notamos que, na verdade, gerar números aleatórios não é tão simples assim: o maior dos problemas era que a própria função *srand()* não gerava números aleatórios. Em todos os testes, os números gerados eram os mesmos, e no fim, o computador nunca jogava, pois o local da jogada já estava ocupado.

Mais tarde, descobrimos ser necessário definir uma “**semente**” para a função *srand()*, o pontapé inicial para que a função começasse a gerar números **diferentes**. Precisávamos que essa semente mudasse a todo momento, para isso, usamos uma nova função: *time(NULL)* dentro da função *srand()*. Essa função retorna os segundos desde 1 de Janeiro de 1970 até o horário local da máquina que está rodando o código, dessa forma, a **semente** muda a cada segundo que é passado.

Mesmo depois disso, ainda existia um limitador que impedia que o código rodasse corretamente: a probabilidade que a função gerava números diferentes era bem baixa, acreditamos ser devido à pouca liberdade de escolha de números pela função, na qual definimos de 0 até 2. A solução implementada foi fazer com que, caso o computador tenha 6 ou mais tentativas de jogadas, e ainda assim, não foi possível encontrar um local

vazio, forçamos o BOT a percorrer toda a matriz da tabela 3x3 utilizando 2 “**for()**” até que ele encontre o primeiro espaço vazio no jogo da velha.

Código Fonte:

```
1  #include <stdio.h>
2  #include <string.h>
3  #include <stdlib.h>
4  #include <windows.h>
5  #include <time.h>
6  #include <locale.h>
7
8  int **matriz;
9  int l, c, id;
10 int jogadas, empate, sair, rodando;
11 int escolha, esc, resposta, bot ,bot_tentativa;
12 char letra;
13 int retorno = -1;
14
15 struct t_dados{
16     char jogador[30];
17     int vitorias;
18 };
19     t_dados dados[2];
20 void rank(){
21     FILE *p;
22     char str[80] = "ranking";
23     char f;
```

```

24
25     p = fopen(str,"a+");
26
27     if(dados[0].vitorias >0 || dados[1].vitorias >0 || empate == 1){
28         fprintf(p,"%s ganhou %d vezes\n", dados[0].jogador,
29 dados[0].vitorias);
30         fprintf(p,"%s ganhou %d vezes\n", dados[1].jogador,
31 dados[1].vitorias);
32         fclose(p);
33     }else{
34         p = fopen(str,"a+");
35         while (!feof(p)){
36             fscanf(p,"%c",&f);
37             printf("%c",f);
38         }
39         fclose(p);
40     }
41 }
42 void tabela(){
43     printf("\n");
44     printf("\tJogo da velha\n");
45     printf("\n\n\t 0   1   2\n\n");
46     for(l = 0; l < 3; l++){
47         for(c = 0; c < 3; c++){
48             if(c == 0)
49                 printf("\t");
50                 printf(" %c ", matriz[l][c]);
51             if(c < 2)
52                 printf("|");

```



```
81         do{
82             letra = getchar();
83             printf("%c", letra);
84             system("cls");
85             }while(letra != '\n');
86     }while(retorno == 0 || bot != 1 && bot != 2);
87     system("cls");
88     if(bot == 1){
89         fflush(stdin);
90         printf("Nome de Usuario: ");
91         gets(dados[0].jogador);
92         strcpy(dados[1].jogador, "Computador");
93         system("cls");
94     }else
95     if(bot == 2){
96         fflush(stdin);
97         printf("Nome do primeiro Usuario: ");
98         gets(dados[0].jogador);
99         fflush(stdin);
100        printf("Nome do segundo Usuario: ");
101        gets(dados[1].jogador);
102        fflush(stdin);
103        system("cls");
104    }
105    }else
106    if (escolha == 2){
107        system("cls");
108        do{
```



```

109             printf("Ranking Geral:\n");
110
111             rank();
112
113             printf("Digite (3) para retorna ou (4) para sair:
114 ");
115
116             retorno = scanf("%d", &esc);
117
118             do{
119
120                 letra = getchar();
121
122                 printf("%c", letra);
123
124                 system("cls");
125
126             }while(letra != '\n');
127
128             }while(retorno == 0 || esc != 3 && esc != 4);
129
130
131
132
133             if(esc == 4){
134
135                 exit(0);
136
137             }else
138
139             if(esc == 3){
140
141                 system("cls");
142
143                 }
144
145             }else
146
147             if (escolha == 3){
148
149                 system("cls");
150
151                 do{
152
153                     printf("\tCréditos\n");
154
155                     printf("Ryan do Nascimento Bezerra\n");
156
157                     printf("Matheus      Farias      Porto      Dos
158 Anjos\n");
159
160                     printf("Fábio Harley Silva Filho\n");
161
162                     printf("Gabriel Henriques Cassiano\n");

```

```

138         printf("Digite (3) para retorna ou (4) para sair:
139     ");
140
141         retorno = scanf("%d", &esc);
142
143         do{
144
145             letra = getchar();
146
147             printf("%c", letra);
148
149             system("cls");
150
151             }while(letra != '\n');
152
153         }while(retorno == 0 || esc != 3 && esc != 4);
154
155         if(esc == 4){
156
157             exit(0);
158
159         }
160
161         }else
162
163         if (escolha == 4){
164
165             exit(0);
166
167         }
168
169         }while(letra != '\n');
170
171     }while(retorno == 0 || escolha != 1);
172 }
173
174 void propriedade(){
175
176     matriz = (int **) malloc(sizeof(int*)*3);
177
178
179
180     for(l=0;l<3;l++){
181
182         matriz[l] = (int *) malloc(sizeof(int*)*3);
183
184         for(c=0;c<3;c++){
185
186             matriz[l][c] = ' ';
187
188         }
189
190     }
191 }

```

```

166 void interacao(){
167
168     while(sair == 0){
169         if(bot == 1 && jogadas % 2 == 0){
170             srand (time(NULL));
171             l = rand() % 3 + 0;
172             c = rand() % 3 + 0;
173
174             bot_tentativa++;
175
176             if(bot_tentativa > 5){
177                 for(l = 0; l < 3; l++){
178                     for(c = 0; c < 3; c++){
179                         if(matriz[l][c] == ' '){
180                             matriz[l][c] = 'O';
181                             l = 3;
182                             c = 3;
183                             bot_tentativa = 0;
184                             sair = 1;
185                         }
186                     }
187                 }else
188                 if(matriz[l][c] == 'O' || matriz[l][c] == 'X'){
189                     jogadas--;
190                     break;
191                 }else
192                 if(jogadas % 2 == 0){
193                     matriz[l][c] = 'O';

```

```

194                 sair = 1;
195             }
196         }else{
197             do{
198                 printf("\n  %s  -  Coloque  a  coluna:  ",
199 dados[id].jogador);
200
201                 retorno = scanf("%d",&c);
202
203                 printf("  %s  -  Coloque  a  linha:  ",
204 dados[id].jogador);
205
206                 retorno = scanf("%d",&l);
207
208                 do{
209                     letra = getchar();
210
211                     printf("%c", letra);
212
213                 }while(letra != '\n');
214
215             }while(retorno == 0 || c < 0 || c >= 3 || l < 0 || l >= 3);
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999

```

```

223                 sair = 1;
224             }
225         }
226     }
227 }
228 void VerificaVitoria(){
229
230     if(matriz[0][0] == 'X' && matriz[1][1] == 'X' && matriz[2][2] ==
231 'X' ||
232
233     matriz[0][2] == 'X' && matriz[1][1] == 'X' && matriz[2][0] ==
234 'X'){
235         printf("\n\t%s (X), Venceu!", dados[0].jogador);
236
237         dados[0].vitorias++;
238
239         rodando=0;
240     }else
241
242     if(matriz[0][0] == 'X' && matriz[1][0] == 'X' && matriz[2][0] ==
243 'X' ||
244
245     matriz[0][1] == 'X' && matriz[1][1] == 'X' && matriz[2][1]
246 == 'X' ||
247
248     matriz[0][2] == 'X' && matriz[1][2] == 'X' && matriz[2][2]
249 == 'X'){
250         printf("\n\t%s (X), Venceu!", dados[0].jogador);
251
252         dados[0].vitorias++;
253
254         rodando=0;
255     }else
256
257     if(matriz[0][0] == 'X' && matriz[0][1] == 'X' && matriz[0][2] ==
258 'X' ||
259
260     matriz[1][0] == 'X' && matriz[1][1] == 'X' && matriz[1][2]
261 == 'X' ||
262
263     matriz[2][0] == 'X' && matriz[2][1] == 'X' && matriz[2][2]
264 == 'X'){

```

```

254         printf("\n\t%s (X), Venceu!", dados[0].jogador);
255         dados[0].vitorias++;
256         rodando=0;
257     }else
258         if(matriz[0][0] == 'O' && matriz[1][1] == 'O' && matriz[2][2] ==
259 'O' ||
260         matriz[0][2] == 'O' && matriz[1][1] == 'O' && matriz[2][0]
261 == 'O'){
262         printf("\n\t%s (O), Venceu!", dados[1].jogador);
263         dados[1].vitorias++;
264         rodando=0;
265     }else
266         if(matriz[0][0] == 'O' && matriz[1][0] == 'O' && matriz[2][0] ==
267 'O' ||
268         matriz[0][1] == 'O' && matriz[1][1] == 'O' && matriz[2][1]
269 == 'O' ||
270         matriz[0][2] == 'O' && matriz[1][2] == 'O' && matriz[2][2]
271 == 'O'){
272         printf("\n\t%s (O), Venceu!", dados[1].jogador);
273         dados[1].vitorias++;
274         rodando=0;
275     }else
276         if(matriz[0][0] == 'O' && matriz[0][1] == 'O' && matriz[0][2] ==
277 'O' ||
278         matriz[1][0] == 'O' && matriz[1][1] == 'O' && matriz[1][2]
279 == 'O' ||
280         matriz[2][0] == 'O' && matriz[2][1] == 'O' && matriz[2][2]
281 == 'O'){
282         printf("\n\t%s (O), Venceu!", dados[1].jogador);
283         dados[1].vitorias++;
284         rodando=0;

```

```
285         }else
286         {
287             if(jogadas == 9){
288                 printf("\nEmpate");
289                 rodando=0;
290                 empate = 1;
291             }
292         }
293     }
294     int main(){
295
296         setlocale(LC_ALL, "Portuguese");
297
298         rodando = 1;
299
300         menu();
301
302         do{
303
304             jogadas = 0;
305
306             propriedade();
307
308             while(rodando=1){
309
310                 system("cls");
311
312                 tabela();
313
314                 sair = 0;
315
316
317
318                 VerificaVitoria();
319
320                 if(rodando == 0){
321
322                     jogadas--;
323
324                     break;
325
326                 }
327
328                 jogadas++;
329             }
330         }
331     }
332 }
```

```

313
314         if(jogadas %2 == 1){
315             id=0;
316         }else
317         if(jogadas %2 == 0){
318             id=1;
319         }
320         interacao();
321     }
322     do{
323         Sleep(1000);
324         system("cls");
325         printf("\nDigite (3) para jogar novamente ou (4) para sair:
326     ");
327         retorno = scanf("%d", &esc);
328         do{
329             letra = getchar();
330             printf("%c", letra);
331             system("cls");
332         }while(letra != '\n');
333     }while(retorno == 0 || esc != 3 && esc != 4);
334         if(esc == 4){
335             rank();
336             exit(0);
337         }
338     }while(esc == 3);
339
340 //Libera memória da Matriz

```



```
341     if (matriz!=NULL){
342         for (l=0 ; l<3 ; l++){
343             if (matriz[l] != NULL)
344                 free(matriz[l]);
345         }
346         free(matriz);
347     }
348 }
```