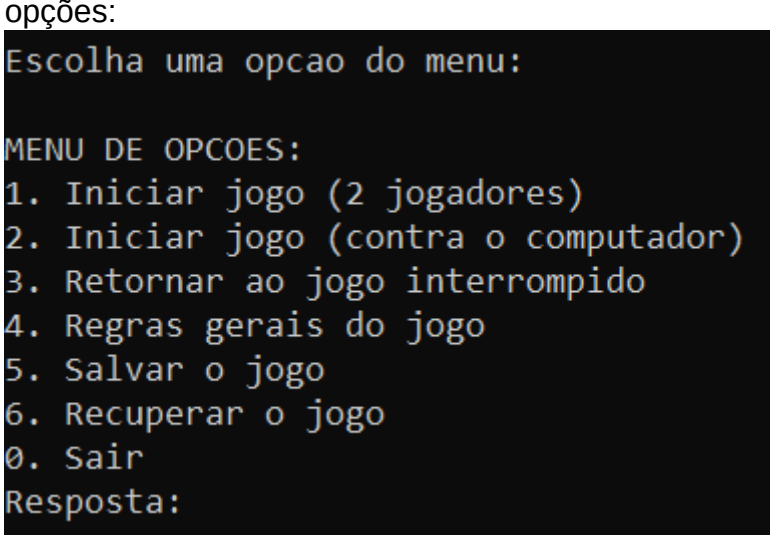
RELATÓRIO FINAL – PROJETO DOMINÓ

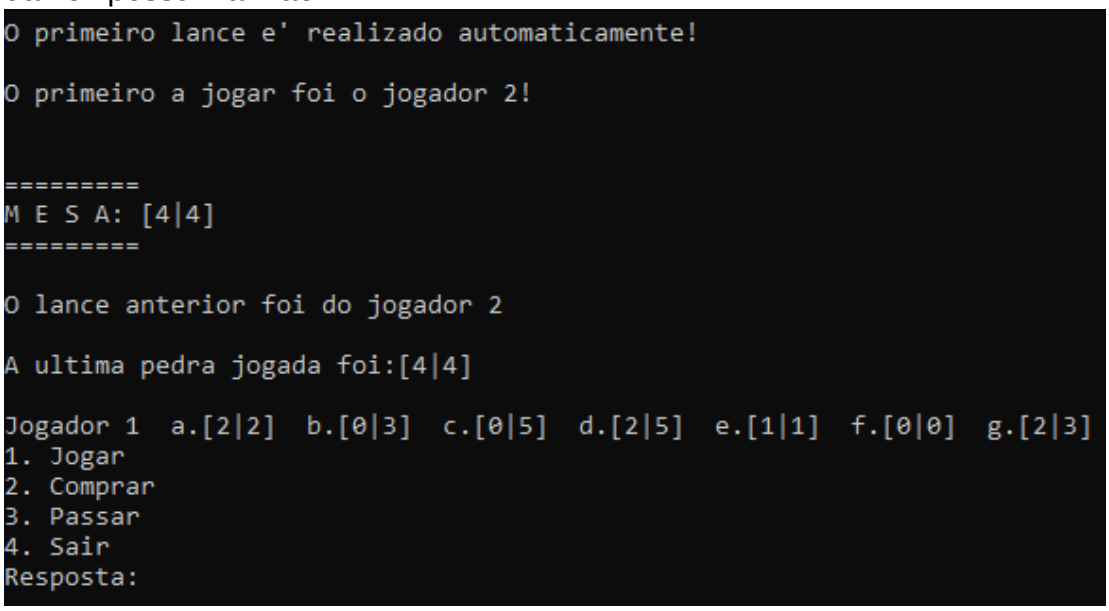
Manual de operação do jogo Dominó

Abaixo, a visualização do menu principal do jogo de Dominó, que oferece 7 opções:

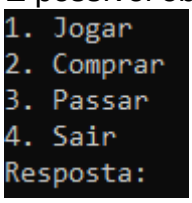


Em que a opção '**1**' inicia um jogo com 2 jogadores, a '**2**' inicia o jogo com um jogador contra o computador, a '**3**' retorna a um jogo interrompido que **não** havia sido salvo, a '**4**' mostra as regras gerais do jogo, '**5**' salva o jogo, '**6**' recupera o jogo que havia sido salvo anteriormente e '**0**' sai do programa. Caso seja inserido um número diferente dos acima o menu principal continuará aparecendo até uma opção correta ser inserida.

Caso a opção escolhida for “**1. Iniciar Jogo (2 Jogadores)**”, como é possível ver abaixo, aparecerão quatro mensagens, “**O primeiro lance é realizado automaticamente!**”; “**O primeiro a jogar foi o Jogador X!**”; “**O lance anterior foi do Jogador X**”; “**A última pedra jogada foi:** “. Aparecerá também a mesa com a primeira pedra que foi gerada automaticamente e as pedras que o jogador da vez possui na mão.

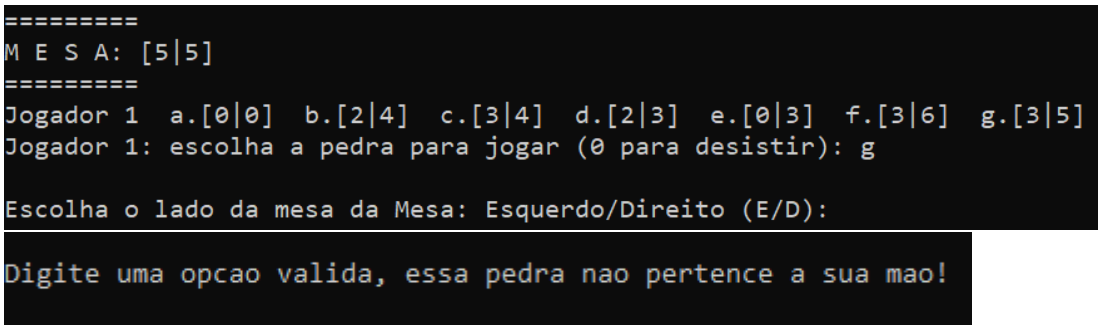


É possível observar que também é gerado um submenu que contém 4 opções:



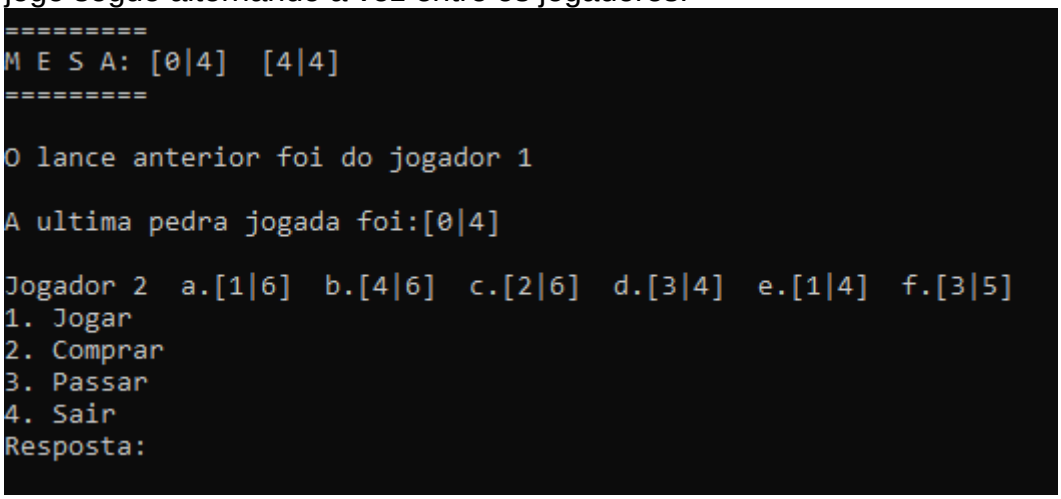
Em que a opção ‘**1**’ é quando o jogador escolhe jogar uma pedra que tem em sua mão, a ‘**2**’ é quando o jogador resolve comprar pedras que estão no monte, ‘**3**’ é para passar a vez, somente quando não há pedras válidas a serem jogadas naquele momento e quando não há mais nenhuma pedra no monte, e ‘**4**’ para sair do jogo e retornar ao menu principal. Caso seja inserido um número diferente dos acima o submenu continuará aparecendo até uma opção correta ser inserida.

Caso a opção escolhida do submenu for “**1. Jogar**”, como podemos ver abaixo, serão mostradas as pedras que estão atualmente na mesa e as pedras que estão na mão do jogador, que nesse caso se trata do Jogador 1, e serão pedidas duas informações, qual pedra o jogador quer jogar, e caso as duas extremidades (direita e esquerda) da mesa sejam iguais, em qual lado ele deseja jogar a pedra escolhida. Caso a letra inserida não corresponda as opções que aparecem como “**Jogador X:**” continuará sendo pedida a pedra que o jogador deseja jogar e aparecerá a mensagem “**Digite uma opção válida, essa pedra não pertence a sua mão!**”.

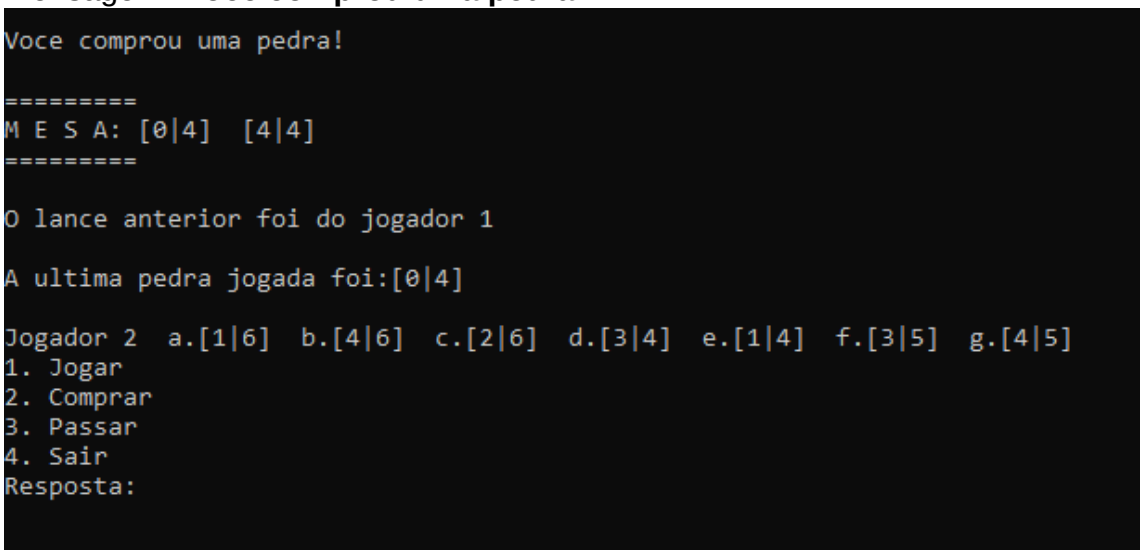


Observa-se também que é dada a opção de desistir do jogo apertando ‘**0**’, fazendo com que volte ao submenu, o que possibilita que o mesmo jogador escolha outra opção.

Depois do Jogador 1 jogar e escolher a pedra e o lado, a vez passa para o Jogador 2, e aparecerá novamente a mesa, atualizada, as mensagens “**O lance anterior foi do jogador X**” e “**A última pedra jogada foi:** “, e as pedras do jogador da vez atual em que será feita a escolha do submenu, dessa forma o jogo segue alternando a vez entre os jogadores.

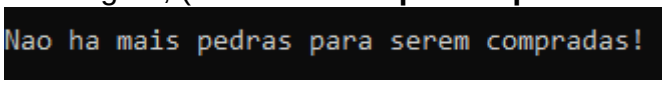


Nesse caso, o Jogador 2 resolveu comprar, percebe-se que na imagem acima a última pedra que havia em sua mão era a pedra "**f. [3|5]**", mas depois de escolher a opção "**2. Comprar**" foi acrescentada a pedra "**g. [4|5]**" e apareceu a seguinte mensagem "**Você comprou uma pedra!**".

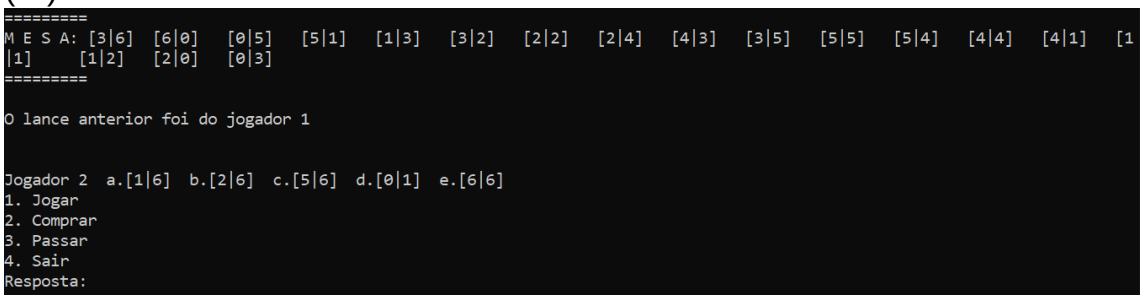


OBS: Um jogador pode comprar quantas pedras quiser até acabarem as pedras no monte, não há nenhuma condição necessária para que ele possa comprar.

Quando não há mais pedras no monte a serem compradas aparecerá a seguinte mensagem, ("**Não há mais pedras para serem compradas!**").



Aqui, o Jogador 2 não possui pedras válidas para jogar e não tem mais pedras a serem compradas, dessa forma a única opção é passar ('**3**') ou sair do jogo ('**4**').



Caso a opção escolhida seja “**4. Sair**”, aparecerá a mensagem (“**SAINDO DO JOGO...**“) e voltará para o menu principal.

```
1. Jogar
2. Comprar
3. Passar
4. Sair
Resposta: 4
SAINDO DO JOGO...

Escolha uma opcao do menu:

MENU DE OPCOES:
1. Iniciar jogo (2 jogadores)
2. Iniciar jogo (contra o computador)
3. Retornar ao jogo interrompido
4. Regras gerais do jogo
5. Salvar o jogo
6. Recuperar o jogo
0. Sair
Resposta:
```

Caso a opção escolhida seja “**3. Passar**”, vai passar para o Jogador 1 e aparecerá a seguinte mensagem (“**O jogador X está sem pedras validas e pulou a jogada!**”).

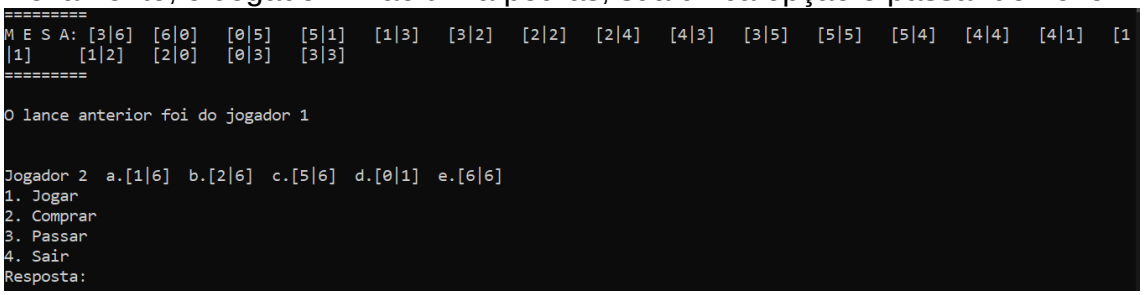
```
O jogador 2 esta sem pedras validas e pulou a jogada!

=========
M E S A: [3|6]  [6|0]   [0|5]   [5|1]   [1|3]   [3|2]   [2|2]   [2|4]   [4|3]   [3|5]   [5|5]   [5|4]   [4|4]   [4|1]   [1
|1]     [1|2]   [2|0]   [0|3]
=========
Jogador 1  a.[0|0]  b.[2|5]  c.[0|4]  d.[4|6]  e.[3|3]
1. Jogar
2. Comprar
3. Passar
4. Sair
Resposta:
```

OBS: O jogador só pode passar caso não tenha nenhuma pedra válida para jogar naquela rodada e não tenha mais pedras disponíveis no monte a serem compradas. Caso o jogador tente passar sem essas condições serem verdadeiras aparecerá a seguinte mensagem (“**Digite uma opção válida! Você não pode passar essa jogada.**”) e o submenu aparecerá novamente.

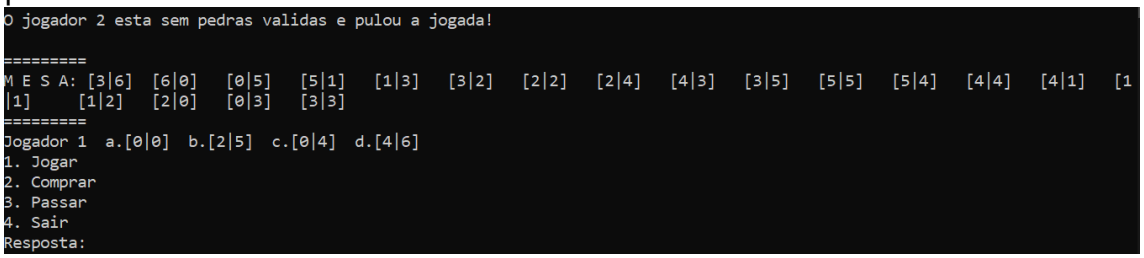
```
Digite uma opcao valida! Voce nao pode passar essa jogada.
```

Caso o próximo jogador tenha pedra para jogar, o jogo segue normalmente.

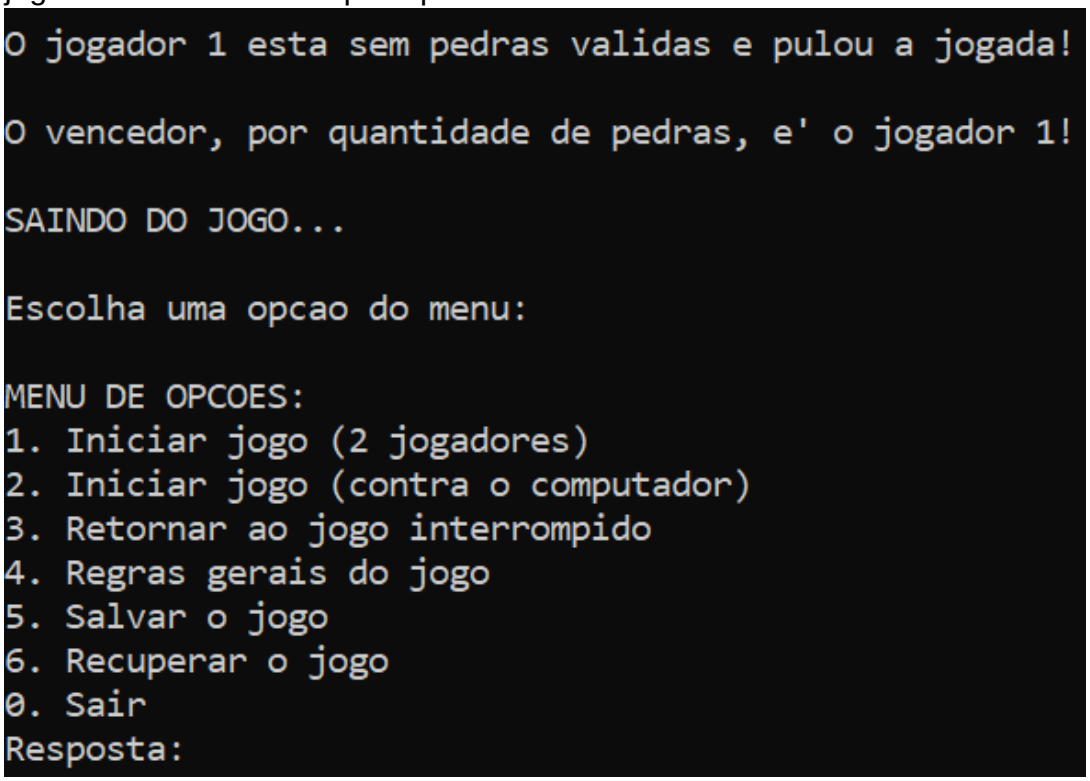
Novamente, o Jogador 2 não tinha pedras, sua única opção é passar de novo.



O Jogador seguinte (1) não tem pedras válidas para jogar, então precisará passar.

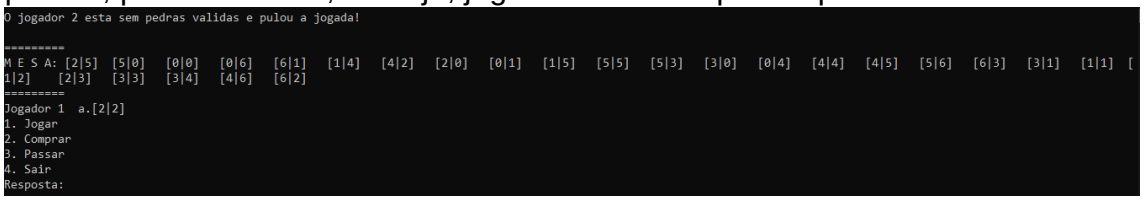


Por conta de os 2 jogadores terem pulado a vez consecutivamente o jogo é finalizado, nesse caso as pedras são contabilizadas e por isso o Jogador 1 venceu por ter a menor quantidade de pedras, assim aparece a seguinte mensagem ("**O vencedor, por quantidade de pedras, é o jogador X!**"), sai do jogo e retorna ao menu principal.

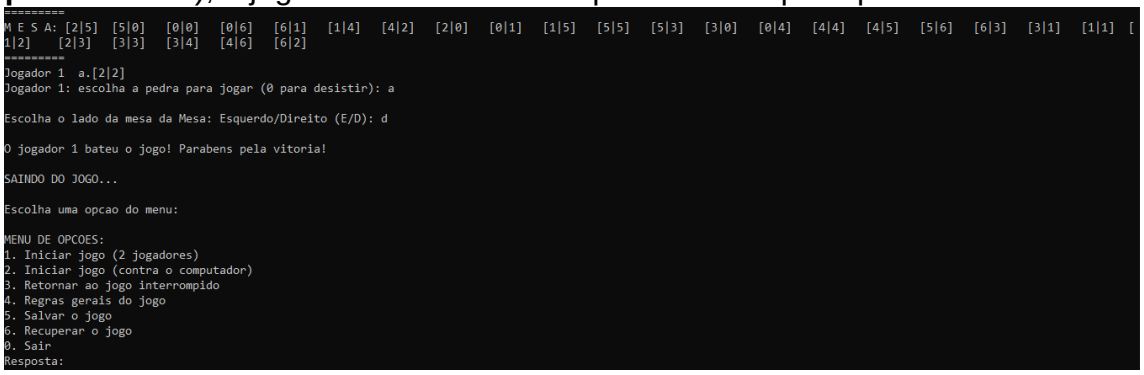


Outra maneira de vencer é “batendo” o jogo, veja o exemplo abaixo:

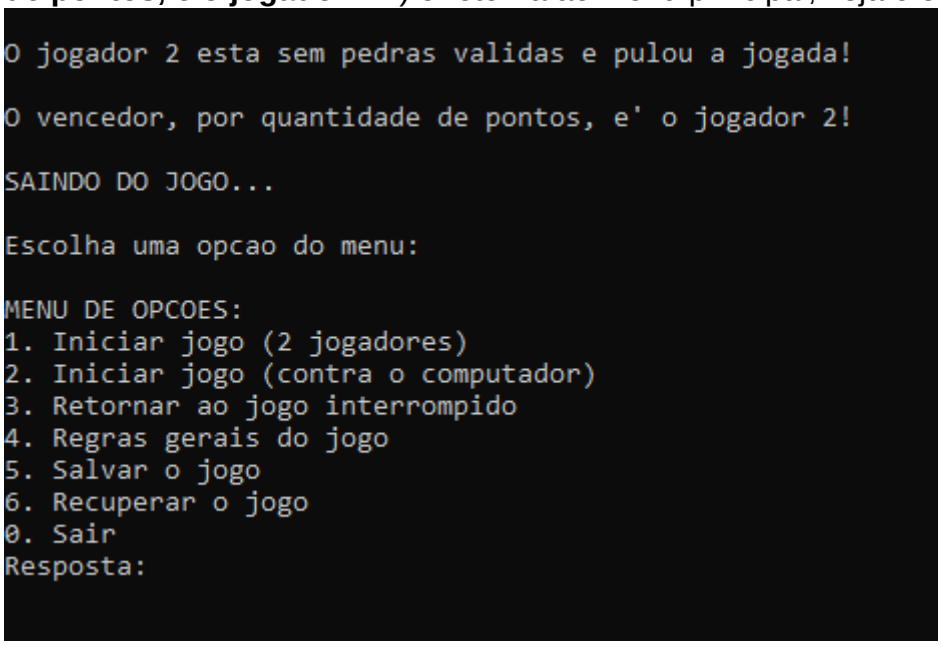
O Jogador 2 não tinha pedras válidas para jogar nem pedras para comprar e passou a vez. Como o Jogador 1 tem pedra válida para jogar, vai vencer a partida, pois irá “bater”, ou seja, jogará sua última pedra que tinha na mão.



Assim, aparece a seguinte mensagem (“**O Jogador X bateu o jogo! Parabéns pela vitória!**”), o jogo é finalizado e volta para o menu principal

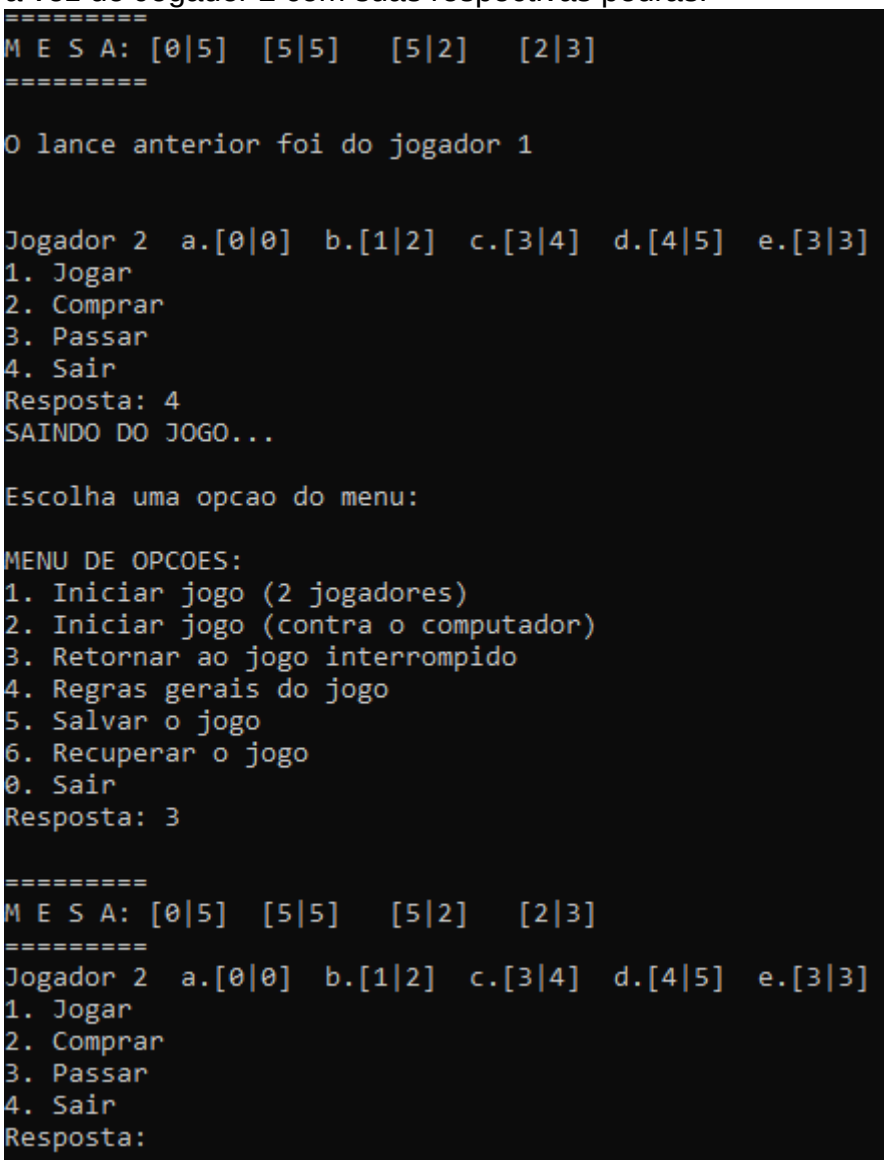


Outro modo de vencer é pela “quantidade de pontos“, que acontece quando ambos os jogadores passam a vez consecutivamente, e ambos possuem a mesma quantidade de pedras. Dessa forma, são somados os números em cada pedra que restar na mão de cada jogador, e vence aquele que tem a menor quantidade de pontos, aparecendo a mensagem (“**O vencedor, por quantidade de pontos, é o jogador X!**”) e retorna ao menu principal, veja o exemplo abaixo.

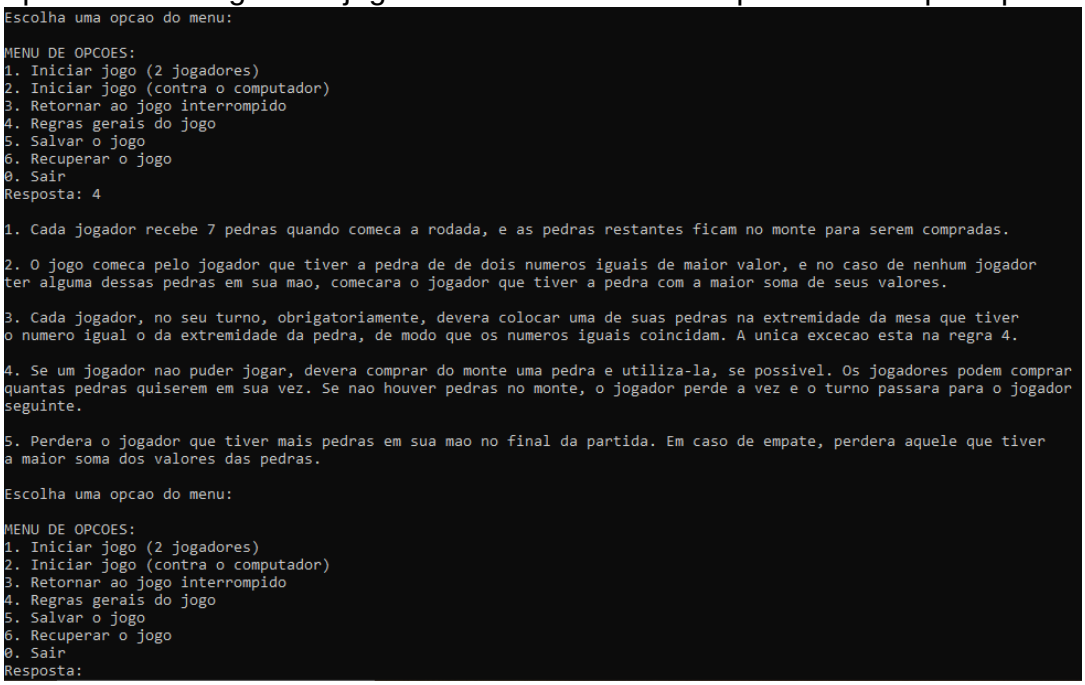


Caso a opção escolhida no menu principal for “**2. Iniciar jogo (contra o computador)**”, um jogo contra o computador será iniciado e funcionará praticamente igual o jogo entre dois jogadores, a única diferença será que o computador jogará automaticamente como Jogador 2 e o Jogador 1 não conseguirá ver suas pedras, mas será sinalizado na tela a pedra que o computador jogou.

Caso a opção escolhida no menu principal for “**3. Retornar ao jogo Interrompido**”, acontecerá como é possível ver neste exemplo, em que o Jogador 2, na sua vez, opta pela opção ‘**4**’ do submenu, e então no menu geral escolhe retornar ao jogo interrompido que não havia sido salvo, voltando assim, a vez do Jogador 2 com suas respectivas pedras.



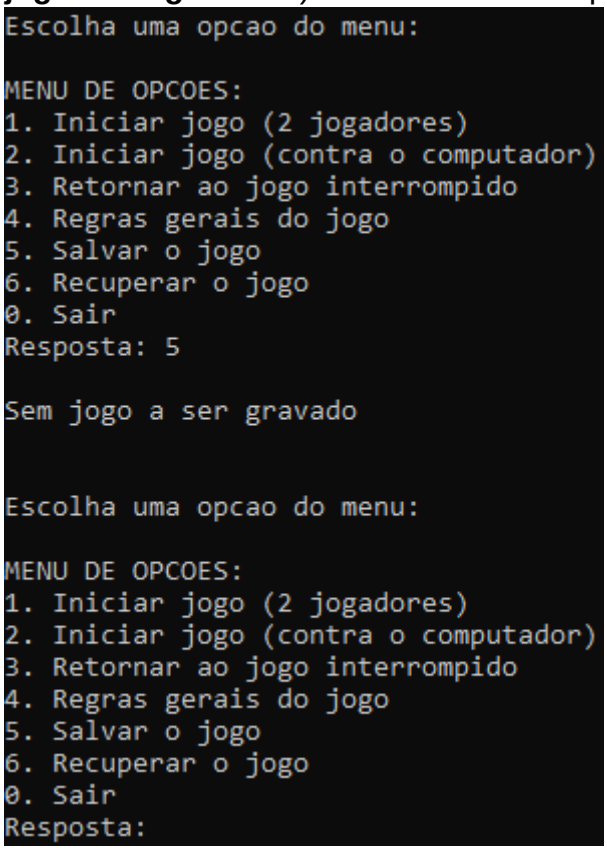
Caso a opção escolhida no menu principal seja “**4. Regras Gerais do jogo**”, aparecerá as regras do jogo de dominó e retornará para o menu principal.



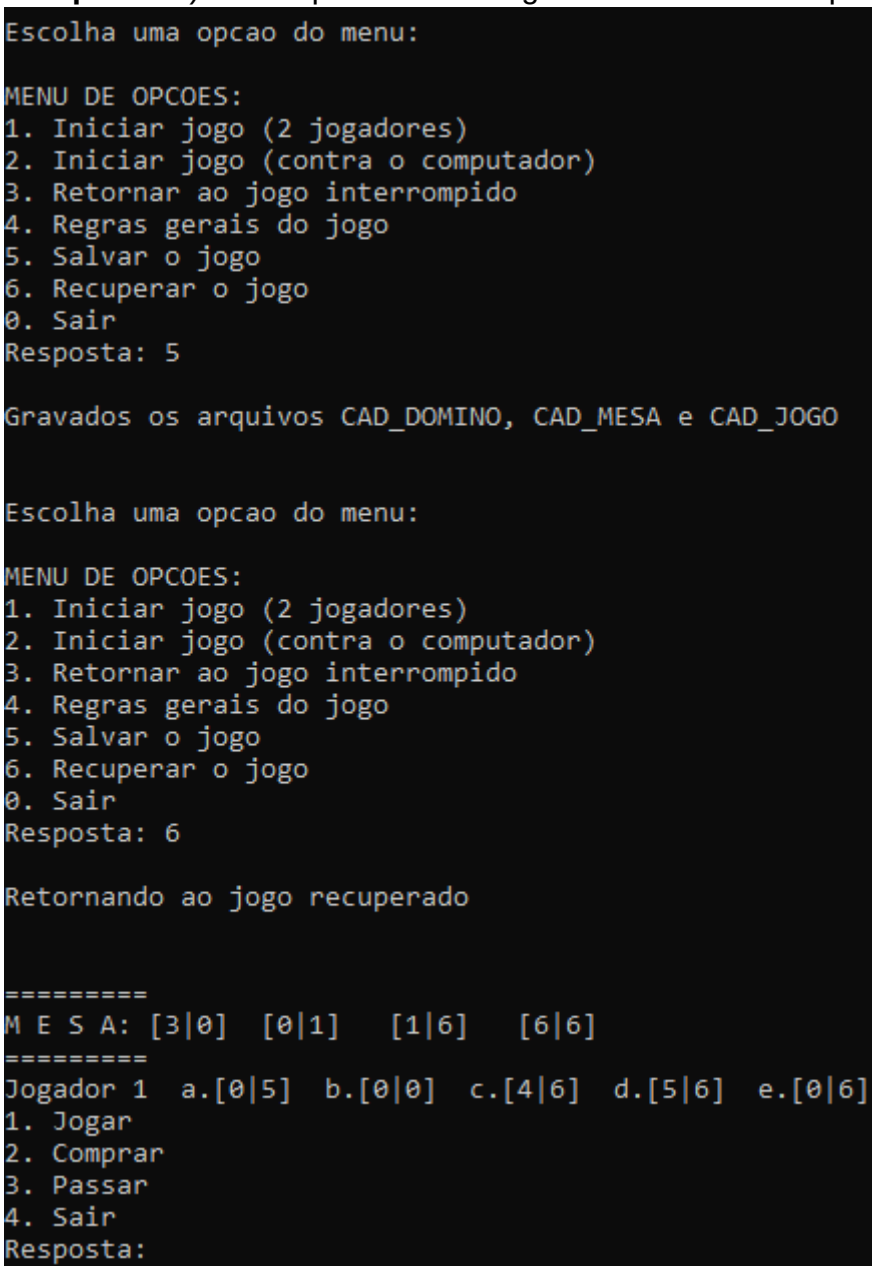
Caso a opção escolhida no menu principal seja "**5. Salvar o jogo**", podem acontecer duas situações, a primeira seria como no exemplo abaixo, em que o Jogador 1 na sua vez escolhe a opção '**4**' do submenu e ao voltar para o menu principal escolhe a opção '**5**', dessa forma aparece a mensagem ("**Gravados os arquivos CAD_DOMINO, CAD_MESA E CAD_JOGO**"), o que significa que o jogo foi salvo, e assim ele retorna ao menu principal.

```
=========
M E S A: [3|0]  [0|1]   [1|6]   [6|6]
=========

O lance anterior foi do jogador 2


Jogador 1  a.[0|5]  b.[0|0]  c.[4|6]  d.[5|6]  e.[0|6]
1. Jogar
2. Comprar
3. Passar
4. Sair
Resposta: 4
SAINDO DO JOGO...

Escolha uma opcao do menu:

MENU DE OPCOES:
1. Iniciar jogo (2 jogadores)
2. Iniciar jogo (contra o computador)
3. Retornar ao jogo interrompido
4. Regras gerais do jogo
5. Salvar o jogo
6. Recuperar o jogo
0. Sair
Resposta: 5

Gravados os arquivos CAD_DOMINO, CAD_MESA e CAD_JOGO


Escolha uma opcao do menu:

MENU DE OPCOES:
1. Iniciar jogo (2 jogadores)
2. Iniciar jogo (contra o computador)
3. Retornar ao jogo interrompido
4. Regras gerais do jogo
5. Salvar o jogo
6. Recuperar o jogo
0. Sair
Resposta:
```

A segunda situação é quando há uma tentativa de salvar o jogo quando não há nenhum jogo a ser salvo, nessa ocasião aparecerá a seguinte mensagem ("**Sem jogo a ser gravado**") e retorna ao menu principal.



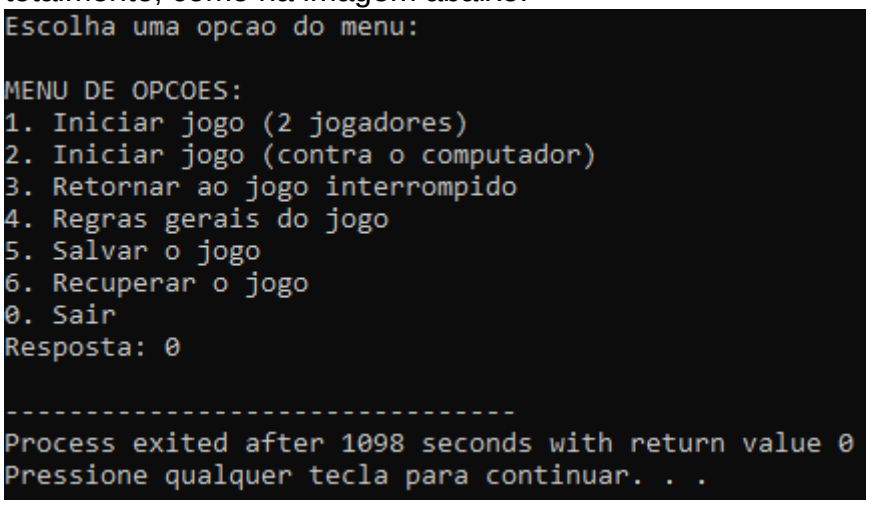
Caso a opção escolhida no menu principal seja “**6. Recuperar o jogo**”, irá retornar ao jogo que foi salvo anteriormente, quando foi escolhido a opção ‘**5**’ do menu principal, como é possível observar abaixo, usando o mesmo exemplo que foi usado na opção ‘**5**’, é mostrada a mensagem (“**Retornando ao jogo recuperado**”) e volta para vez do Jogador 1 com suas respectivas pedras.



Caso o jogo não tenha sido salvo e é inserida a opção '**6**' para recuperá-lo, não irá voltar ao jogo, mas voltará para um que havia sido salvo anteriormente. Como é possível observar abaixo, foi criado um jogo novo em que a mesa consistia em **[5|5] [5|4]** no entanto o usuário não salvou o jogo, dessa forma ao tentar recuperá-lo, voltou ao último jogo salvo que é o mesmo do exemplo acima.

```
=========
M E S A: [5|5]  [5|4]
=========

O lance anterior foi do jogador 1


Jogador 2  a.[0|6]  b.[3|6]  c.[2|6]  d.[2|4]  e.[3|3]  f.[2|5]
1. Jogar
2. Comprar
3. Passar
4. Sair
Resposta: 4
SAINDO DO JOGO...

Escolha uma opcao do menu:

MENU DE OPCOES:
1. Iniciar jogo (2 jogadores)
2. Iniciar jogo (contra o computador)
3. Retornar ao jogo interrompido
4. Regras gerais do jogo
5. Salvar o jogo
6. Recuperar o jogo
0. Sair
Resposta: 6

Retornando ao jogo recuperado


=========
M E S A: [3|0]  [0|1]  [1|6]  [6|6]
=========
Jogador 49  a.[0|5]  b.[0|0]  c.[4|6]  d.[5|6]  e.[0|6]
1. Jogar
2. Comprar
3. Passar
4. Sair
Resposta:
```

Caso a opção escolhida no menu principal seja “**0. Sair**”, o jogo será finalizado totalmente, como na imagem abaixo.



Dados Técnicos

O jogo de Dominó está divido em 7 arquivos: o Main, Controller (cpp e h), Model (cpp e h) e o View (cpp e h).
O Main é responsável por compilar o programa com as duas funções principais (“IniciarPedras” e “executarMenu”), também chama o Model.cpp e o Controller.cpp.
O Model.h contém as variáveis globais e o Model.cpp, chama o Model.h e possui as estruturas dos dados, (“typedef struct stpedra”, “struct Mesa” e “struct Jogo”).
O View.h declara as funções que serão utilizadas no View.cpp, e o cpp chama o View.h e desenvolve as funções, anteriormente declaradas, que compõem a interface do usuário e a integração homem-máquina.
Por último, o Controller, que se trata de um mediador entre o model e o View, então o Controller.h declara as funções que serão desenvolvidas no Controller.cpp, esse, que chama o Controller.h e o View.cpp.
Dessa forma, o jogo é desenvolvido e executado de maneira fracionada em que cada arquivo tem sua respectiva função.
OBS: Todas as funções, variáveis e estruturas mencionadas acima serão explicadas posteriormente.

Abaixo estão as variáveis globais que foram utilizadas no programa (estão contidas no Model.h):

“**char alfabeto[] = "abcdefghijklmnopqrstuvwxyz"**;” - Se trata de um vetor em que cada posição é uma letra do alfabeto, esse vetor será usado para o jogador escolher a pedra que irá jogar, por meio de uma letra do alfabeto.

“**int jvez;**” - Armazena o jogador da vez de cada rodada.

“**int mesaE;**” – É uma variável referente a extremidade esquerda da mesa.

“**int mesaD;**” - É uma variável referente a extremidade direita da mesa.

“**int jogcomp;**” – Indica se é um jogo entre dois jogadores (jogcomp = 0) ou se é um jogo com um jogador contra o computador (jogcomp = 1).

“**int qtmesa;**” - Armazena a quantidade de pedras na mesa.

“**int passou;**” - Armazena a quantidade de vezes que passou consecutivamente.

“**int win;**” - Define a vitória.

“**int playerAuto;**” - “player automático”, indica qual dos jogadores é o computador em um jogo contra o computador.

“**int last_L1;**” - Última pedra que foi jogada lado 1.
“**int last_L2;**” - Última pedra que foi jogada lado 2.

Foram utilizadas as seguintes estruturas de dados no programa (contidas Model.cpp):

```
typedef struct stpedra[1]
{
        int L1, L2;[2]
        int status;[3]
} tipoPedra;[4]

tipoPedra dom[28];[5]
tipoPedra aux;[6]

struct Mesa[7] {
        int ladoE;[8]
        int ladoD;[9]
} mesa[28];[10]


struct Jogo[11] {
 int qtmesaJogo;[12]
 int jvezJogo;[13]
 int jogadorComp;[14]
 int mesaDJogo;[15]
 int mesaEJogo;[16]
} sitJogo;[17]
```

1 – Estrutura das pedras.

2 – Variáveis do Lado 1 e Lado 2 da pedra.
3 – Armazena o status da pedra.
OBS: Existem 4 status da pedra que são:

‘**0**’ - Pedra não pertence a nenhum jogador, está no monte para ser comprada.
‘**1**’ - Pedra pertence ao Jogador 1.
‘**2**’ - Pedra pertence ao Jogador 2.
‘**9**’ - Pedra já foi jogada e está na mesa.

4 – Abrevia o struct stpedra
5 – Array de pedras.
6 – Variável auxiliar utilizada para embaralhar.
7 – Estrutura da mesa.
8 – Variável do Lado Esquerdo da mesa.
9 – Variável do Lado Direito da mesa.
10 – Array com no máximo 28 pedras.
11 – Estrutura do jogo.
12 – Quantidade de pedras na mesa.
13 – Jogador atual.
14 – Armazena qual dos dois jogadores irá ser o computador.
15 – Extremidade direita da mesa.
16 – Extremidade Esquerda da mesa.
17 – Nome do tipo que grava os dados necessários da situação de jogo para que possam ser recuperados.

Abaixo segue a lista e explicação das funções e procedimentos utilizados no decorrer do programa:

- Contido no View:

“**void funcMostraDomino()**” – Mostra as pedras.

“**int menugeral()**” – Exibe na tela as opções do menu principal que está no Controller e armazena a opção escolhida.

“**void apresentaPedra(int jvez)**” – Mostra as pedras do jogador da vez.

“**int submenuV()**” – Exibe na tela as opções do submenu que está no Controller e capta a opção escolhida.

“**char escolhePedra()**” – Capta a opção de pedra que o usuário escolheu para jogar.

“**char escolheLado()**” – Capta a opção do lado da mesa que o usuário escolheu para jogar a pedra.

“**void apresentaMensagem(char mens[100])**” – Apresenta uma determinada mensagem na tela.

“**void flush_in()**” – Limpa o buffer de entrada.

“**void mostraRegras()**” – Mostra as regras.

“**void mostraUltimaPedra();**” – Mostra a última pedra que foi colocada na mesa.

- Contido no Controller:

“**void IniciarPedras()**” – Cria as pedras e coloca seu status como ‘**0**’ que indica que está no monte.

“**void executarMenu()**” – Executa o menu principal com base na opção escolhida pelo usuário, dessa forma cada opção do jogador leva a um case diferente do menu, que pode iniciar um novo jogo entre dois jogadores, iniciar um jogo contra o computador, retornar ao jogo, mostrar as regras do jogo, salvar o jogo, retornar ao jogo salvo ou sair.

“**void funcEmbaralhaPedras()**” – Embaralha as pedras.

“**void iniciarJogo()**” – Chama a função que embaralha as pedras, depois distribui, e faz o primeiro lance automático (chamando a função do primeiro lance).

“**void Jogo()**” – Loop em que as rodadas acontecem, é encerrada caso um vencedor seja encontrado ou o jogador escolha sair do jogo por meio do submenu.

“**void primeiroLance()**” – Define o primeiro jogador, por meio da verificação de quem tem a maior pedra com os dois lados iguais, caso não esteja na mão de nenhum jogador, verifica a pedra de maior soma, realiza o primeiro lance automaticamente, verifica quem foi o jogador que realizou esse primeiro lance, carrega a mesa com a primeira peça na posição 0, muda o status da pedra para ‘**9**’ que indica que está na mesa e por última chama a função que troca o jogador.

“**int jogar(int jvez)**” – Chama o submenu que fica em loop até o usuário sair do jogo.

“**void submenuC()**” – Submenu de opções que o usuário pode escolher durante a partida, apresenta mensagens a respeito de quem foi a jogada anterior, e com base na opção escolhida pelo usuário vai entrar em um case dentro do menu, que pode realizar a jogada, comprar pedra, passar a vez e sair, esse menu

funciona enquanto o jogador não pede para sair e enquanto não há nenhum vencedor.

“**void executaJogada()**” – Descobre onde está a pedra que o jogador escolheu, chama a função que verifica se a pedra é válida, verifica se o computador precisa passar, executa a jogada, e por último verifica se tem algum vencedor.

“**void carregaMesaE(int k)**” – Desloca toda a mesa para o lado esquerdo para abrir a posição 0, verifica se é necessário inverter a pedra, executa a jogada no lado esquerdo da mesa na posição 0, atualiza a variável global mesaE, aumenta no contador a quantidade de pedras na mesa, atualiza o status da pedra jogada, verifica se tem algum vencedor e por último troca de jogador.

“**void carregaMesaD(int k)**” – Verifica se é necessário inverter a pedra, executa a jogada no lado direito da mesa, atualiza a variável global mesaD, aumenta no contador a quantidade de pedras na mesa, atualiza o status da pedra jogada, verifica se tem algum vencedor e por último troca de jogador.
Executa a jogada no lado direito da mesa.

“**void comprarPedra(int jvez)**” – Verifica se tem pedras disponíveis para serem compradas, apresenta mensagem de acordo com a verificação, compra as pedras e coloca na mão do jogador.

“**int VerificaPedra(int k)**” – Verifica se a pedra é válida para ser jogada.

“**void trocaJogador()**” – Troca de jogador.

“**int pedraIgualMesa(int k)**” – Verifica se as duas extremidades da pedra são iguais as extremidades da mesa ou se os dois lados da mesa são iguais a apenas um lado da pedra.

“**int descobreLadoMesa(int k)**” – Verifica em qual lado a pedra válida deverá ser colocada.

“**int passarVez()**” – Verifica se há pedras para serem compradas e se há pedras válidas na mão do jogador, se não tiver pedras no monte nem pedras válidas na mão do jogador, apresenta mensagem sobre qual jogador passou a vez e retorna 1. É falsa quando não é possível passar a vez.

“**int vencedor()**” – Descobre o vencedor por meio da verificação se há pedras no monte, contando as pedras e os pontos dos jogadores, e no fim apresenta mensagem sobre quem ganhou e como.

“**void resetar()**” – Reseta o jogo.

“**void gravaCadastro()**” – Verifica se há pedras na mesa ou vencedores, define os documentos onde os dados serão armazenados, verifica se os documentos

estão vazios, se os dados nos documentos coincidem com struct original, salva os dados do jogo e fecha os documentos.

“**void recuperarCadastro()**” – Verifica se os documentos estão vazios, se os dados nos documentos coincidem com struct original, recupera os dados do jogo.

“**int CompEscolhePedra()**” – Escolhe uma pedra quando o jogador é o computador, retorna o índice dessa pedra e caso não tenha retorna -1, que será usado na função void executaJogada para o computador passar a vez.