**Pontifícia Universidade Católica de São Paulo**

**Faculdade de Ciências Exatas e Tecnologia**

**Departamento de Ciência da Computação**

# Projeto Dominó LP – Grupo JJLM

Projeto desenvolvido por alunos do 2° semestre de Ciência da Computação.

# Manual do Usuário

## Sumário

## 1) Menu Principal

Selecione a opção desejada no menu digitando os dígitos ao lado de cada operação:

**(1) – Iniciar novo jogo (2 jogadores).**

Começa um jogo contra outro jogador.

**(2) – Iniciar novo jogo (Vs. Computador).**

Começa um jogo contra o computador (neste caso as jogadas do jogador 2 são automáticas).

**(3) - Retornar ao jogo atual.**

Retorna ao jogo que havia sido começado antes da saída ao menu.

**(4) - Recuperar jogo salvo.**

Recupera um jogo salvo anteriormente.

**(5) - Salvar jogo.**

Salva o jogo atual.

**(6) - Regras do jogo.**

Mostra as regras do jogo.

**(0) – Sair do jogo.**

Fecha o programa.

Ex: Ao digitar-se 6 e pressionar a tecla ENTER as regras do jogo aparecerão na tela.

## 2) Menu do jogo

Após iniciar um jogo cada jogador será apresentado com o menu e a mesa para as jogadas:

Selecione a opção desejada no menu digitando os dígitos ao lado de cada operação:

**(1) – Jogar**

Mostra as peças do jogador atual e permite que ele jogue uma delas.

**(2) – Comprar**

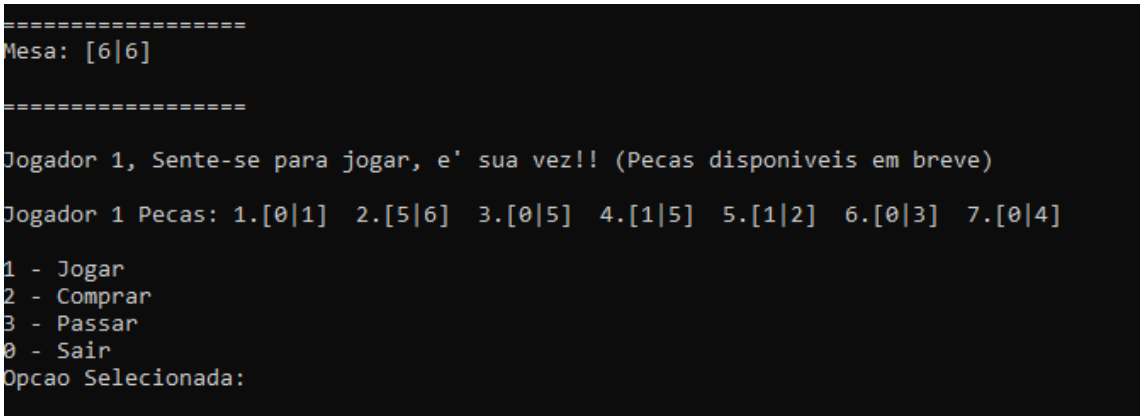
Permite que o jogador compre uma peça.

**(3) – Passar**

Passa a vez do jogador caso não possua peças a jogar e comprar.

**(0) – Sair**

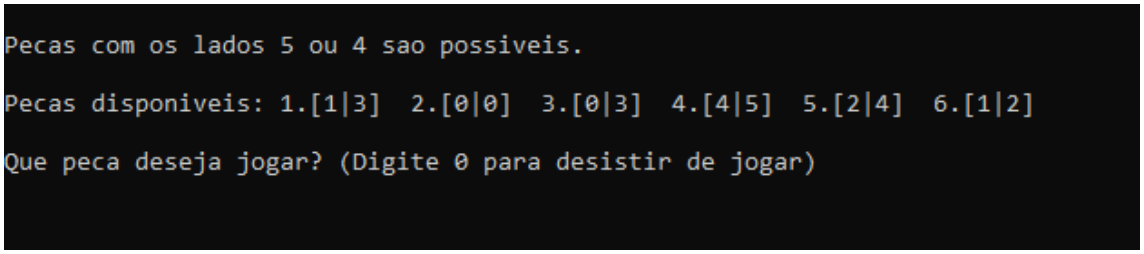
Retorna ao menu principal onde o usuário poderá salvar o jogo e sair, retornar ao jogo atual ou iniciar uma nova partida.



## 3) Interface de jogo

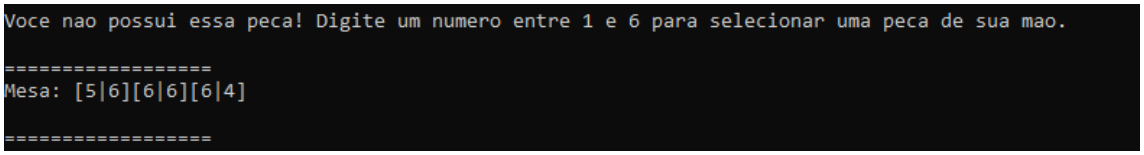
### Ao selecionar a opção 1:

Cada peça adicionada a mesa é colocada apenas horizontalmente, limitando as opções de jogada ao lado esquerdo ou direito. Cada jogador pode selecionar a peça que vai jogar digitando o número indexado ao seu lado ou 0 para desistir da jogada:




Os lados extremos da mesa que possuem as possíveis jogadas estão sempre logo acima da mão do jogador.

A parte superior da mesa é destinada as mensagens de erro que podem aparecer a cada usuário durante o jogo, além das mensagens de acontecimentos.


No modo contra o computador o jogador efetuará sua jogada normalmente e as ações do computador são automáticas.

### Ao selecionar a opção 2:

O jogador pode comprar quantas peças quiser até que o depósito esvazie. Cada vez que um jogador compra uma peça ela será automaticamente adicionada a sua mão e o menu de opções será mostrado para que ele decida seu novo movimento.

**Ao selecionar a opção 3:**

O jogo será passado automaticamente, a não ser que o jogador atual tenha peças a jogar ou possa comprar.

**Fim do jogo:**

Caso o jogo termine, a partida será interrompida automaticamente e será anunciado o jogador vencedor.

# Informações técnicas

## Estruturas

**typedef int booleano:**

Define o tipo de dado booleano que retorna TRUE ou FALSE.

**typedef struct Peca:**

Define o tipo de dado estrutura peca que armazena cada peça, reservando o lado esquerdo, direito e o seu status.

**struct Mesa:**

Cria o vetor mesa[28] que armazenará as peças colocadas na mesa, a estrutura reserva o lado esquerdo e direito das peças colocadas na mesa.

**struct Jogo:**

Armazena dados importantes do jogo para o salvamento posterior.

**tipoPeca pecas[28]:**

Cria o vetor estrutura que armazena as peças do jogo com lado esquerdo e direito, além do seu status (pertence a que jogador).

**tipoPeca aux:**

Peça auxiliar com lado esquerdo e direito usada para reservar uma peça e embaralhar o vetor pecas[28].

## Funções

**No controller:**

**void criarPecas():**

Cria as peças do jogo e preenche o vetor estrutura pecas[28] com os lados das peças por meio de comandos for.

**void embaralhaPecas():**

Embaralha a ordem do vetor estrutura pecas[28] trocando peça a peça de lugar com a ajuda de uma variável auxiliar e um índice aleatório obtido pela função srand.

**void executaJogo():**

Inicia o menu principal e embaralha as peças.

**void iniciarjogo():**

Atribui o status de cada peça a um jogador ou ao deposito, seleciona a primeira peça a ser jogada e apresenta a mensagem dizendo que foi o jogador que a lançou.

**void PrimeiraPeca():**

Seleciona a primeira peça procurando se encontra a maior peça com lados iguais ou a peça com maior soma entre os lados por meio de comandos if.

**int qtdPecasJg(char jogador):**

Retorna quantas peças possui um jogador por meio de um for que verifica se uma peça é desse jogador e soma a um contador.

**char PecaValida():**

Sequência de if que verificam se há possibilidade de se inserir uma peça na mesa em algum lado e se for possível retorna este.

**void JogadaEsq(int pj)** e **void JogadaDir(int pj):**

Efetuam a jogada no lado esquerdo ou direito da mesa, acrescentam +1 a quantidade de peças na mesa e atribuem a peça jogada ao vetor estrutura mesa[28].

**booleano comprar(char jogador):**

Verifica se é possível comprar uma peça e se for a adiciona a mão do jogador atribuindo o seu status ao jogador atual. Retorna TRUE se for possível e FALSE se não for.

**booleano depositoVazio():**

Verifica se o deposito de peças está ou não vazio por meio de um comando for e retorna TRUE se estiver vazio e FALSE se estiver com peças.

**booleano passar():**

Verifica se é possível passar a vez do jogador por um comando if que verifica se é possível jogar ou comprar. Retorna TRUE e troca a vez do jogador se for possível passar e FALSE se não for.

**booleano PossivelJogar():**

Verifica se é possível jogar testando as peças da mão do jogador com os lados da mesa.

**void ftempo(int seg):**

Adiciona um atraso no prompt de comando.

**void MenuPrin():**

Aciona o menu principal que vai coletar a opção do operador e substituir no comando switch.

**void MenuDeJogo():**

Aciona o menu de jogo que coleta a opção do operador e substituir no comando switch.

**void JogadaComp():**

Aciona a jogada do computador que irá sequencialmente verificar se é possível jogar (e jogar se sim), se é possível comprar (e comprar se sim) ou passar. Entra em loop por meio do menu de jogo caso o computador não jogue ou passe a vez.

**booleano Fimdejogo(char jogador, int passou2vez):**

Verifica se o jogo terminou utilizando comandos if para verificar como ele acabou (fim das peças de um jogador, 2 vezes passadas consecutivamente etc.) e apresenta a mensagem de fim de jogo apresentando o jogador.

**void fclear():**

Limpa o buffer do teclado.

**void gravaCadastro():**

Salva as condições do jogo atual: O vetor estrutura Peças, o vetor estrutura Mesa e a quantidade de peças na mesa, o jogador atual, o lado extremo esquerdo e direito, a quantidade de vezes que foi passado o jogo em sequência, a sentinela que indica que o computador está jogando e a quantidade de peças compradas.

**void recuperaCadastro():**

Recupera os dados salvos de um jogo anterior pela função gravaCadastro();

**No view:**

**void apresentaMensagem(char msg[100]):**

Função utilizada para apresentar mensagens ao usuário fora do view.

**int Txtmenuprin():**

Apresenta o texto e recolhe a opção do operador do menu principal.

**void Mesa():**

Imprime a mesa utilizando o vetor mesa[28] para imprimir as peças em ordem.

**void apresentaPeca(char jogador):**

Imprime as peças de um jogador acompanhadas de um índice por meio de um for que roda o vetor estrutura pecas[28].

**int Txtmenudojogo(char jogador):**

Apresenta o texto e recolhe a opção do operador do menu de jogo.

**int Jogar(char jogador):**

Se for o computador esta função recolhe o índice da primeira peça jogável em sua mão e o retorna como valor.

Se for um jogador, a função recolherá o índice que está ao lado da peça impressa, associado a ordem da mão do jogador e transformará esse índice no real da peça no vetor pecas[28] por meio de um contador que descobrirá a posição da peça na mão do jogador. Após coletar o índice verifica se ela é válida em algum lugar da mesa e retorna como valor o índice.

**char escolherLado():**

Recolhe o lado que o jogador escolhe para efetuar sua jogada.

**void MSGdelay(char jogador):**

Apresenta um texto que indica atraso para a troca de usuário.

**void regrasdojogo():**

Apresenta um texto com as regras do jogo.

**void pecajogada(int d):**

Apresenta a peça jogada pelo jogador anterior.

**void qtpecacomprada():**

Apresenta quantas peças foram compradas pelo jogador anterior por meio da variável global qtcompra.

**void Passou():**

Apresenta mensagem indicando que o jogador anterior passou a vez.

## Variáveis globais

**#define FALSE 0**

**#define TRUE 1**

**int mesaE;**

Inicia a extremidade esquerda da mesa.

**int mesaD;**

Inicia a extremidade direita da mesa.

**int qtmesa;**

Inicia a quantidade de peças na mesa que permite que seja impressa a mesa.

**char jogador;**

Reserva qual o jogador da vez.

**char vez;**

Reserva qual o jogador da vez.

**int passou2vez;**

Armazena a quantidade de vezes que o jogo foi passado de vez, se houver sido passado 2 vezes consecutivas o jogo termina.

**int jogcomp;**

Sentinela que indica se o computador está jogando, se for igual a 2, é o computador, se não então é um jogador.

**int qtcompra;**

Armazena quantas peças foram compradas para apresentar a mensagem de compra.