UNIVERSIDADE FEDERAL DO AMAZONAS - UFAM
FACULDADE DE TECNOLOGIA
BACHARELADO EM ENGENHARIA DA COMPUTAÇÃO

Disciplina: IEC011 - Introdução a Computação
Projeto do jogo de Dominó

Luís Henrique Raheem Simões

Sumário

Parte 1

O jogo de Dominó

A Lista de problemas que enunciaremos a seguir é baseada em uma das várias formas de se jogar "dominó". As explicações que serão apresentadas não se destinam a ensinar o jogo mas simplesmente dar contexto para os problemas.

O "dominó", é um conjunto formado por 28 "peças", cada uma delas com duas "pontas". Cada "ponta" representa um valor de 0 (zero) a seis (6), perfazendo portanto 7 valores diferentes. Cada valor possui um nome próprio: o zero chama-se "branco", o um chama-se "ás", o dois é o "duque", o três chama-se "terno", o quatro é a "quadra", o cinco é a "quina" e o seis denomina-se "sena". O nome de uma "pedra" é dado pelo nome de suas "pontas", por exemplo, "quina e terno", é o nome da "pedra" que possui em uma ponta o valor 5 e na outra o valor 3. As pedras que possuem o mesmo valor nas duas pontas são denominadas de "carroça". Para cada tipo de valor existem 7 pedras, por exemplo, para o "terno" teremos: terno e branco, terno e ás, terno e duque, carroça de terno, quadra e terno, quina e terno, sena e terno.

Para jogar, as "pedras" são embaralhadas e escolhidas pelos jogadores. A cada jogador cabem 7 pedras. Com o desenrolar do jogo a quantidade de pedras vai sendo decrescida, até que, eventualmente chegue em zero. A "mão" de um jogador pode ser representada por uma lista de pares, como por exemplo: [(2,3),(3,4),(4,4)].

Parte 1

Bloco 1

P01: Escreva a função **pedrap** que associe um par a True se e somente se (sss) o par é uma representação válida para uma "pedra" e False caso contrário.
pedrap (2, 7) ==> False
pedrap ((-3), 4) ==> False
pedrap (3,4) ==> True
P02: Escreva a função **maop** que associe uma lista de pares de inteiros a True sss a lista é uma representação válida para a "mão" de um jogador e False caso contrário.
P03: Escreva a função **carrocap** que associe um par a True sss o par é uma "carroça" e False caso contrário.
P04: Escreva a função **tem_carroca_p** que associe uma "mão" a True sss a mão possuir pelo menos uma carroça e False caso contrário.
P05: Escreva a **função tem_carrocas** que associe a uma "mão" a lista das "carroças" nela contida.

Parte 1

Bloco 2

Em vários momentos do jogo faz-se necessário saber a quantidade de pontos associado à uma coleção de pedras. Em particular, no final do jogo, quem "sentou" a sua última pedra faz jus à "garagem" que é determinada a partir dos pontos que restaram na(s) mão(s) dos adversários.

P06: Escreva a função **pontos** que associe uma lista de "pedras" a soma dos pontos das pedras nela contidos. Onde os pontos de uma pedra é a soma de suas pontas.

P07: Escreva a função **garagem** que associe uma lista de "pedras" ao maior múltiplo de 5 (cinco), menor ou igual à soma dos pontos nela contidos.

P08: Escreva a função **pedra_igual_p** que associe dois pares de inteiros a True sss representam a mesma pedra e False caso contrário. É bom lembrar que a ordem das pontas é irrelevante, assim (2,4) e (4,2) representam a mesma pedra.

P09: Escreva a função **ocorre_pedra_p** que associe uma "pedra" e uma "mão" a Truesss a "pedra" ocorre na "mão" e False caso contrário.

P10: Escreva a função **ocorre_valor_p** que associe um valor válido para "ponta" e uma "mão" e produza True sss o valor ocorre em alguma pedra da mão e False caso contrário.

P11: Escreva a função **ocorre_pedra** que associe a um valor e uma "mão", uma lista contendo as pedras da "mão" que possuem o valor dado.

P12: Escreva a função **pedra_maior** que associe uma "mão" a pedra de maior valor na "mão" dada. Uma pedra p1 é maior que uma outra p2 sss a soma das pontas de p1 for maior que a soma das pontas de p2.

P13: Escreva a função **ocorre_valor_q** que associe um valor e uma "mão" e produza o número de pedras na mão que possuem o valor dado.

P14: Escreva a função **ocorre_carroca_q** que associe uma mão à quantidade de carroças nela existentes.

P15: Escreva a função **tira_maior** que associe uma mão a uma lista similar à "mão" de onde foi extraída a pedra de maior ponto.

P16: Escreva a função **tira_maior_v** que associe um valor e uma "mão" à lista similar à "mão" de onde se extraiu a pedra de maior pontos de um determinado valor para ponta.

Parte 1

Bloco 3

O jogo se desenvolve pela colocação, pelo jogador da vez, de uma pedra que combine com alguma das "pontas" da "mesa". Num momento genérico do jogo temos quatro pontas disponíveis para execução de uma jogada. Uma ponta pode ser simples ou uma carroça. As carroças são dispostas de tal forma que todos os seus pontos estejam para "fora".

Chamaremos "mesa" à lista de pontas disponíveis para jogada. Pontas simples serão representadas por listas de um elemento e carroças por uma lista com dois elementos idênticos. Por exemplo, a "mesa" ilustrada na Figura 1 é representada pela quadrupla ([5],[5,5],[0],[5]).

Uma ponta ainda não aberta é representada por lista vazia. Dizemos que há marcação de pontos em uma mesa quando a soma das pontas é um múltiplo de 5. Os pontos a serem marcados é a soma das pontas, com as carroças contando em dobro.

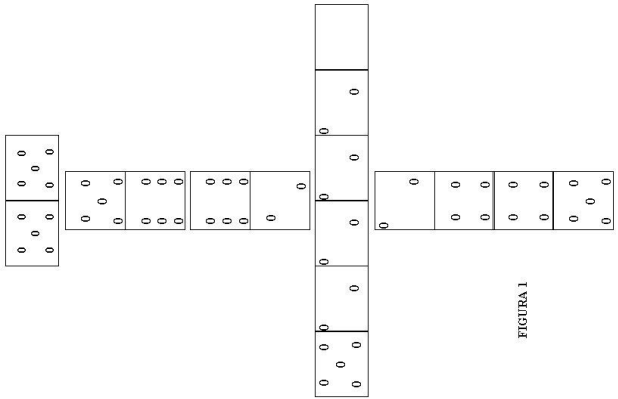

FIGURA 1

Parte 1

Bloco 3

P17: Escreva a função **mesap** que associe uma quadrupla de listas a True sss a quadrupla for uma descrição válida de "mesa".
P18: Escreva a função **carroca_m_p** que associe uma mesa a True sss pelo menos uma das pontas for carroça.
P19: Escreva a função **pontos_marcados** que associe uma mesa ao o número de pontos a serem marcados se a soma das pontas for múltiplo de cinco e zero em caso contrário.
P20: Escreva a função **pode_jogar_p** que associe uma "pedra" e uma "mesa" a Truesss a pedra possui uma ponta que combina com pelo menos uma das pontas da mesa.
P21: Escreva a função **marca_ponto_p** que tenha como entrada uma "pedra" e uma "mesa" e produza True sss a pedra pode ser jogada fazendo pontos em uma das pontas da mesa. Lembre-se que as carroças devem ser contadas pelas duas pontas da pedra.
P22: Escreva a função **maior_ponto** que associa uma pedra e uma mesa ao número da "ponta" da mesa onde pode ser marcado o maior valor de ponto que será marcado pela pedra. Considere que a em uma "mesa" as pontas são numeradas a partir de zero, da esquerda para a direita.
P23: Escreva a função **joga_pedra** que associe uma "pedra", uma "mesa" e um número de "ponta" da mesa a uma nova mesa obtida ao se jogar a "pedra" na "ponta" indicada.
P24: Escreva a função **jogap** que associe uma "mão" e uma "mesa" e produza True sssexiste pelo menos uma pedra na mão que possa ser jogada em pelo menos uma ponta da mesa. Caso contrário produza False.
P25: Escreva a função **jogada** que associe uma "mão" e uma "mesa" ao número da pedra na mão e número da ponta na mesa onde pode ser feita a jogada que marque mais ponto. Considere inclusive jogada onde não há marcação de ponto.
P26: Escreva a função **faz_jogada** que associe uma "mão" e uma "mesa" e produza uma nova "mesa" obtida por se jogar marcando o maior número de pontos possível

Parte 1

Bloco 4

Considere agora uma nova representação para mesa, da seguinte maneira: Uma lista formada por uma lista unitária e quatro listas, onde a lista unitária representa a "pedra" usada como saída e cada uma das outras lista representa a sequencia de pedras jogadas em uma das pontas. As pontas são numeradas conforme o esquema abaixo (Figura 2):

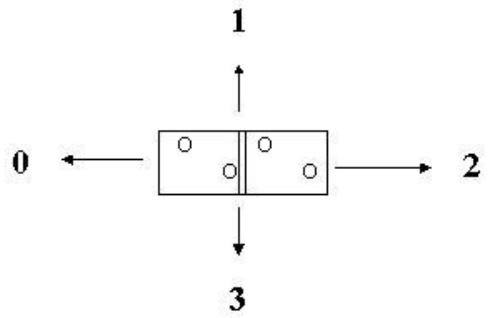

Figura 2

Parte 1

Bloco 4

A mesa representada na Figura 1 teria a seguinte representação:
[[(2,2)], [(5,2)], [(5,5),(5,6),(6,2)], [(0,2)], [(5,4), (4,2)]]
Observe que em cada ponta a pedra mais externa aparece por primeiro e que há uma maneira correta de representar a lista de jogadas, ou seja, a segunda ponta de uma pedra é igual à primeira da pedra seguinte. Pontas ainda não abertas são representadas por uma lista vazia. Observe ainda que não é permitido que as pontas 1 ou 3 estejam vazias quando uma das outras duas (0 e 2) não estejam também.
P27: Escreva a função **lista_de_jogadas** que associa uma lista de pedras com True, sss ela representa corretamentamente uma sequência de jogadas.
P28: Escreva a função **mesa2p** que associa um valor com True, sss ele representacorretamentamente a descrição de uma mesa no formato 2 com sua representação no formato 1.
P29: Escreva a função **marca_ponto_2** que associa uma mesa no formato 2 com o número de pontos marcados.
P30: Escreva a função **faz_jogada_2** que associa uma pedra, uma mesa e um número de ponta na mesa, com a mesa obtida após jogar a pedra na ponta indicada.
P31: Escreva a função **pedra_de_ponto** que associa uma mesa no formato 1 com uma pedra que pode marcar ponto.
P32: Escreva a função **pedras_de_ponto** que associa uma mesa no formato 1 com a lista de pedras que podem marcar ponto.
P33: Escreva a função **pedra_de_maior_ponto** que associa uma mesa no formato 1 com a pedra que marcar mais pontos.
P34: Escreva a função **pedras_fora_p** que associa uma mesa no formato 2 e uma pedra com True sss ela ainda não foi jogada.

Parte 2

Tuplas e Listas em Geral

P35: Defina a função **somavet** que determine a soma de dois vetores x e y, de origem no ponto (0,0) e situados no primeiro quadrante do plano cartesiano.
P36: Defina a função **sumdo** que dado um número inteiro positivo n, construa uma lista com todos os pares cuja soma de elementos é igual a n.
P37: Dada uma lista L, contendo um número igual de números inteiros pares e ímpares (em qualquer ordem), defina a função **alterna** que, quando avaliada, produz uma lista na qual esses números pares e ímpares encontram-se alternados.
P38: Defina a função **intersec** que a partir de duas listas de elementos distintos, determina a interseção entre elas.
P39: Defina a função **uni** que dadas duas listas de elementos distintos, determina a união entre elas.

Parte 3

Processamento de Cadeia de Caracteres

Escrever funções para descrever cada uma das situações abaixo descritas (o nome da função é indicado entre parêntesis):
P40: Verificar se uma cadeia é uma palavra (apenas letras); (e_palav)
P41: Verificar se uma cadeia é um número inteiro positivo; (e_int)
P42: Dado um verbo regular e um tempo do modo indicativo, produzir as conjugações; (conjuga)
conjuga "jogar" "presente" ==> ["eu jogo", "tu jogas", "ele joga", "nos jogamos", "vos jogais", "eles jogam"]
P43: Verificar se uma cadeia representa um número real (escrito na notação com ponto decimal); (e_float)
e_float "3" ==> False
e_float "3." ==> False
e_float "ab" ==> False
e_float "3.5" ==> True
P44: Determinar a cadeia formada pela parte inteira e a cadeia formada pela parte fracionária da representação de um número através de cadeia de caracteres. Se o número for inteiro, a parte fracionária será zero; (int_frac)
int_frac "324.8765" ==> ("324","8765")
int_frac "4586" ==> ("4586","0")
P45: Dado um número de dois algarismo, produzir a cadeia de caracteres que seja a sua representação literal; (traduz)
traduz 35 ==> "trinta e cinco"
traduz 3 ==> "tres"
traduz 15 ==> "quinze"