Relatório PBL problema 3: Jogo da Disputa

Samuel da Costa Araújo Nunes[1]

[1]Engenharia de Computação – Universidade Estadual de Feira de Santana (UEFS)
Feira de Santana, 14 de setembro de 2019

samuelnunes1920@outlook.com

Resumo. *Este relatório descreve a confecção de um programa que simula um jogo de cartas, embaralhando e distribuindo o baralho de forma fidedigna ao modelo real, envolvendo métodos de sorteio, disputa entre os atributos dos jogadores, mantendo cadastro dos jogadores salvos em um arquivo binário.*

1. Introdução

O Jogo da disputa, criado por Paula e seus amigos, enquanto jovens. É um jogo baseado na altercação dos tributos entre as cartas dos jogadores. Inicialmente, é feito um login dos jogadores, e a escolha do modo de jogo (**Aleatório** ou **Manual**), as cartas do jogo são embaralhadas, e empilhadas. Logo após, são distribuídas 5 cartas para cada a jogador (Sempre retiradas do topo da pilha), o Jogador1 começa a partida, escolhendo o atributo de disputa, podendo ser ele: **Valor**, **Força**, **Energia** ou **Jokempô**. A cada rodada, os jogadores alternam na escolha do tipo de disputa.

O tipo de disputa, define qual será o atributo das cartas que será disputado, assim, após o jogador escolher o tipo. Os decks dos dois jogadores serão organizados, usando o atributo como critério de organização em forma crescente. (Caso, **Jokempô**, seja a opção escolhida, as cartas devem ser embaralhadas de forma aleatória)

Deverá ser apresentado, a cada rodada, o jogador atual, as cartas de ambos e a quantidade de disputas vencidas de cada um.

Se a opção do jogo escolhida for "**Aleatório**", os jogadores deverão sortear um número que determinará a carta escolhida, mas se for, "**Manual**", o jogador que irá escolher sua carta, seguindo a ordem do deck, no caso da escolha "**Jokempô**", seguirá a regra: PEDRA ganha de TESOURA, TESOURA ganha de PAPEL e PAPEL ganha de PEDRA.

O Jogador que perde a disputa, deverá descartar a carta usada e pegar a primeira carta da pilha (Baralho). Em caso de empate, os dois descartaram as cartas e pegaram uma nova carta da pilha (Baralho) e em caso de Vitoria, o jogador apenas descarta a carta usada.

O jogo termina ao passar de 10 rodadas, ou até algum jogador ficar sem cartas. Ganha o jogo, o jogador que ficar sem cartas, ou se ao final das 10 rodadas, a soma dos atributos em comparação ao outro jogador for maior. E o cadastro do jogador deverá ser atualizado. O software proposto foi desenvolvido na IDE *Pycharm*.

2. Desenvolvimento

Ao iniciar o desenvolvimento do jogo, foi analisado os critérios básicos para o seu funcionamento. Logo, surgiu o primeiro problema. Como cadastrar os dados dos jogadores,

usando a linguagem *Python*? Através da leitura do livro *Introdução a algoritmos e programação com Python*, [Wazlawick 2017], foi apresentado funções de manipulação de arquivo binario, como *Dump*, *Load*, entre outros. Através de pesquisas na internet, também foi descoberto a eficiência e praticidade de usar classes, afim de gerar um objeto que representa cada jogador, gerando apenas UMA referência ao jogador.

O primeiro desafio, foi o cadastro dos jogadores, de acordo com o produto proposto pelo problema 3, o cadastro deve ser armazenado em um arquivo binário. Em pesquisa, foi descoberto a biblioteca *Pickle*, a priori, a função de cadastro enviava um dicionário contendo como chaves, o *Nickname* do jogador, e as chaves eram relacionadas a uma lista, que continha os dados do jogador. Porém, quando o arquivo estava vazio, o programa gerava um erro. Assim, usei o *Try* e o *Except*, para quando o arquivo estiver vazio, setar um dicionário com o primeiro jogador[eXcript 2015]. No desenvolvimento do projeto, alterei o dicionário por uma Classe de cadastro.

Ao estudar o conceito de Classes e Objetos, foi desenvolvido as Classes: Cartas, Baralho, Deck e Jogadores.

A Classe Cartas, recebe uma lista contendo o nome, o valor, a força, a energia e o Jokempô, converte o tipo da variável, e retorna em forma de Objeto.

A Classe Baralho, recebe como parâmetro o nome do arquivo que contém as cartas, e gera uma pilha contendo objetos gerados pela Classe Cartas. Como um Baralho de Cartas, é possível misturar, tirar uma carta da pilha ou até mesmo tirar um deck inteiro, tudo isso usando Métodos da Classe, ou seja, funções genéricas que trabalham com os atributos do Objeto. Na função *Misturar*, o programa abre um laço de repetição do tamanho do baralho, e em cada repetição ele sorteia dois valores inteiros entre 0 e o tamanho do baralho, e os usa como índices da lista. Assim, ele troca as posições de cartas por pares aleatórios a cada repetição. As funções de *tirar carta baralho* e *gerar deck*, funcionam praticamente da mesma forma, consideram que o baralho é uma pilha ,então retiram e retornam a última carta dela.

A Classe Deck, recebe inicialmente uma lista contendo 5 cartas do Baralho, ela é responsável pelas cartas na mão dos jogadores, esta classe possui 4 métodos, o *tirar carta deck*, que retirar uma carta do baralho, o *por carta deck*, que adiciona uma carta ao deck, o *mostrar deck*, essa função usa da Biblioteca *prettytable*, primeiro o programa gera 6 listas, que contem em ordem, o índice das cartas, os nomes dos personagens, os valores, as forças, as energias, e os Jokempô. Depois ele, define a tabela e printa no terminal para o usuário, e pôr fim a função ordenador, que recebe a escolha do jogador, e o tipo de jogo. Se o jogo for manual, ele irá gerar uma lista com os nomes dos personagens e uma lista igual de referência, organiza essa lista, usando um método baseado no *Bubble Sort*, primeiro, o programa acha o menor valor da lista, acha o seu índice na lista original, retira da lista e adiciona em uma lista nova, chamada lista ordenada. Em seguida, em um *for*, o programa pega o nome do personagem, acha sua posição na lista de comparação, que mantem a ordem atual das cartas, e adiciona na lista com as cartas ordenadas, chamada decknovo, depois atualiza a ordem das cartas do deck. As opções no modo aleatório, seguem a mesma linha de raciocínio.

E por fim, a Classe Jogadores, guarda o *Nickname*, as *Partidas Jogadas*, e as *Partidas Vencidas*.

No programa principal, ele começa chamando a função Cadastro, que recebe como parâmetro o *Nickname* e o número do jogador (1 ou 2), depois ele tenta ler o arquivo ”Cadastro.dat”, caso esteja vazio, ele vai gerar um erro e cairá no *Except*, vai gerar um dicionário, que possui a chave ”jogador1” e ”jogador2” e associa com o nickname do jogador, ele também adiciona manda pro dicionário o *Nickname* relacionado com o objeto da classe Jogadores.

Na sequência do programa, ele gerar um Objeto chamado Baralho, mistura ele, gera os decks do jogador1 e jogador2, verifica se o tipo do jogo é aleatório ou manual, depois ele entra em um While, no qual as rodadas forem menores que 11, e os jogadores ainda possuem cartas na mão. O programa printa as rodadas e as opções de escolha pros usuários. Depois ele verifica se é a vez do jogador 1 ou 2, vendo se o resto da divisão da rodada é igual ou diferente de 0.

O programa chama a função Escolher, e retorna a escolha da disputa em relação ao atributo da carta, se a opção escolhida for ”Jokempô”, ele apenas printa pro usuário o deck usando o metodo mostrardeck(), já que o deck do jogador já está naturalmente embaralhado, caso outra opção for escolhida, o programa chama o método ordenador, e depois mostra as cartas.

Se o tipo de jogo escolhido for aleatório, o próprio programa ira gera um valor inteiro aleatório que corresponderá com o índice de alguma carta do jogador, caso Manual, o jogador que deverá informar, qual carta ele vai querer jogar.

Nesse momento da rodada, já se sabe o atributo de disputa escolhido pelo jogador, e as cartas que deverão ser jogadas. Assim, o programa verifica se a escolha está entre 1 e 3, se estiver, ele irá verificar se a carta sorteada tem um atributo maior, menor ou igual ao do outro jogador, se for maior, o jogador ganham ele incrementa as vitorias de um jogador, tira uma carta de cada deck e adiciona uma no deck do perdedor, em caso de empate, as duas cartas são retiradas e duas novas são adicionadas. Em caso de Jokempô, ele vai verificar todas as opções em que o jogador1 ganha, os casos de empate e o resto dos casos, que são os casos em que o jogador2 ganha.

Se algum jogador ficar com 0 cartas na mão, ele vai sair do While, e cairá no if, que verá qual o maior deck, quem tiver o menor deck ganha a partida e atualiza o cadastro dos jogadores.

Se o jogo ultrapassar os 10 rounds, ele verifica se a soma do atributo valor do jogador1 é menor que que a soma do jogador2, se for igual, ele passa para o próximo atributo. Caso a soma for maior, o jogador ganha.

Por fim, ele abre um último While, que pergunta se o usuário gostaria de jogar novamente, se sim, a variável Partida continua True, se não, a variável Partida vira False, e sai do loop, encerrando o programa.

3. Conclusão

Com o proposto problema aprendi e pratiquei muito as Classes fornecidas no *Python*, o uso correto da modularização do código, o uso das eficiente das funções. Como um jogo piloto, ele atende a todas as funcionalidades exigidas pelo pedido de Paula e seus amigos, porém é possível melhorar consideravelmente o programa considerando as necessidades reais do jogo.

O foco principal para a construção desse projeto foi facilitar o uso do usuário e apresentar as informações de forma clara, na medida em que o problema proposto possibilitou, deixando a execução do jogo mais fluida e de fácil compreensão.

Nos últimos testes, foi observado problemas na recuperação de dados do arquivo binário, tendo que abrir o arquivo mais de uma vez para enviar ou receber dados.

References

eXcript (2015). Funções dos dicionários no python. Disponível em: http://excript.com/python/funcoes-dicionarios.html. Acesso em: 18 Jul. 2019.

Wazlawick, R. S. (2017). *Introdução a algoritmos e programação com Python*. Elsevier, 2017, 1th edition.