Jogo da Disputa

Eduardo Fernandes de Brito Pereia[1]

[1]UEFS – Universidade Estadual de Feira de Santana Av. Transnordestina, s/n, Novo Horizonte Feira de Santana – BA, Brasil – 44036-900

duarduz@gmail.com

Resumo. *Este relatório descreve o problema proposto pelo componente obrigatório, do primeiro semestre, MI Algoritmo I, o qual retrata a implementação de um jogo criado por um grupo de amigos. Trata-se de um jogo de cartas disputado entre dois jogadores: Jogador 1 e Jogador 2. Cada jogador começa o jogo com 5 cartas. Para implementação do jogo, foi necessário desenvolver um programa na linguagem Python, sendo baseado em percorrer arquivos e utilização de estruturas de dados. Assim, funciona o Jogo da Disputa.*

1. Introdução

Paula e seus amigos, quando crianças, eram muito criativos e viviam inventando novos jogos e brincadeiras para se divertirem. Eles cresceram, mas nunca perderam o contato e sempre se lembram de um jogo de cartas muito divertido que inventaram. Eles tiveram uma ideia para eternizar o jogo que inventaram, porém Paula e seus amigos não entendem nada de Programação e precisam de ajuda para transformar essa ideia em um jogo eletrônico. Abraçando a causa, os tutores do MI Algoritmos trouxeram para os alunos do MI o desafio de desenvolver o jogo de cartas inventado por Paula e seus amigos: O Jogo da Disputa.

Este relatório objetiva descrever um sistema desenvolvido na linguagem de programação Python, como solução para o problema anterior.

Para resolver este problema, algumas questões precisaram ser estudadas: 0) como trabalhar com arquivo texto, 1) como trabalhar com arquivo binário, 2) como trabalhar com classes, 3) como ordenar um vetor, 4) como comparar atributos de um objeto.

A solução é organizada da seguinte forma:

1. As classes necessárias para resolução do problema são importadas;
2. Os arquivos necessários para resolução do problema são abertos e transformados em um objeto do tipo carta;
3. Um menu é apresentado aos jogadores para que se inicie a partida;
4. Com o início da partida, comparações de acordo com a escolha de rodada são feitas;
5. Por fim, cadastra-se os jogadores num arquivo chamado “Jogadores_Cadastrados.dat”;

3. Desenvolvimento

Nesta seção, procura-se apresentar como as questões anteriores elencadas na introdução foram tratadas na solução do problema.

Como trabalhar com arquivo texto? Durante a execução de um programa, seus dados ficam na memória. Quando o programa termina, ou o computador é desligado, os dados na memória desaparecem. Para armazenar os dados permanentemente, você tem que colocá-los em um **arquivo**. Arquivos usualmente são guardados em um disco rígido (HD), num disquete ou em um CD-ROM. [aprendacompany]

Para abrir um arquivo, o Python possui a função open(). Ela recebe dois parâmetros: o primeiro é o nome do arquivo a ser aberto, e o segundo parâmetro é o modo que queremos trabalhar com esse arquivo - se queremos ler ou escrever. O modo é passado através de uma *string*: "w" para escrita e "r" para leitura.

No software desenvolvido, utiliza-se o modo de leitura e de escrita, o modo de leitura para ler o arquivo "Cartas.txt" para trabalhar com ele posteriormente e o modo de escrita para criar um arquivo "Jogando.txt" para salvar o nome dos jogadores daquela partida, a fim de cadastra-los posteriormente.

Como trabalhar com arquivo binário? Arquivo binário é um tipo de arquivo que preserva a estrutura de dado em que foi escrita nele.

Para abrir um arquivo binário, o possui uma biblioteca chamada pickle que utiliza métodos para preservar o tipo de dado escrito no arquivo.

Para abrir um arquivo binário, o Python possui a função open(). Ela recebe dois parâmetros: o primeiro é o nome do arquivo a ser aberto, e o segundo parâmetro é o modo que queremos trabalhar com esse arquivo - se queremos ler ou escrever. O modo é passado através de uma *string*: "wb" com auxílio do método "write" do pickle para escrita e "rb" com auxílio do método "load" do pickle para leitura.

No software desenvolvido, utiliza-se o modo de leitura e de escrita, o modo de leitura para ler o arquivo o arquivo "Jogadores_Cadastrados.dat" para verificar os jogadores já cadastrados e o modo de escrita para cadastrar novos jogadores.

Como trabalhar com classes? Nós já vimos classes como str, int, float e Turtle. Estas foram definidos pelo Python e disponibilizadas para nós usarmos. No entanto, em muitos casos quando estamos resolvendo problemas, precisamos criar objetos de dados relacionados ao problema que estamos tentando resolver. Precisamos criar nossas próprias classes.

As definições de classes podem aparecer em qualquer lugar em um programa, mas são geralmente perto do início (após os comandos import). As regras de sintaxe para a definição de uma classe são as mesmas de outros comandos compostas. Há um cabeçalho que começa com a palavra-chave class, seguido pelo nome da classe e terminando com dois pontos.

Toda classe deve ter um método com o nome especial __init__. Este método de inicialização, muitas vezes referido como o construtor, é chamado automaticamente sempre que uma nova instância de Point é criada. Ela dá ao programador a oportunidade

de configurar os atributos necessários dentro da nova instância, dando-lhes seus estaduais valores iniciais. O parâmetro self (você poderia escolher qualquer outro nome, mas ninguém nunca faz!) é definido automaticamente para referenciar o objeto recém-criado que precisa ser inicializado. [Como Pensar Como um Cientista da Computação]

No software desenvolvido, utiliza-se 2 classes, a classe "Carta" e a classe "Jogador", a classe "Carta" recebe como parâmetro uma lista e tem os atributos: nome, valor, força, energia e jokenpô e a classe "Jogador" tem os atributos: partidas, vitorias e porcentagem de vitorias.

Como ordenar um vetor? No jogo, as cartas devem ser ordenadas crescentemente de acordo com a escolha do jogador (Valor, Força e Energia) para isso faz-se necessário a utilização de um algoritmo de ordenação

Algoritmo de ordenação, em ciência da computação, é um algoritmo que coloca os elementos de uma dada sequência em uma certa ordem. Em outras palavras efetua sua ordenação completa ou parcial. O objetivo da ordenação é facilitar a recuperação dos dados de uma lista. [Devmedia]

No software desenvolvido utiliza-se o algoritmo Selection Sort, este algoritmo é baseado em se passar sempre o menor valor do vetor para a primeira posição (ou o maior dependendo da ordem requerida), depois o segundo menor valor para a segunda posição e assim sucessivamente, até os últimos dois elementos.

Como comparar atributos de um objeto? No jogo da disputa, cada carta possui um nome, um valor inteiro representando o valor da carta, um valor real representando a força do personagem, um valor real representando a energia do personagem e uma das três opções de pedra, papel ou tesoura.

Dessa forma o software possui uma função que compara o atributo escolhido pelo usuário e o menor atributo vence, assim o jogo vai tomando forma, por fim o vencedor da partida é cadastrado no arquivo "Jogadores_cadastrados.dat".

2.1. Organização do código fonte

O código é dividido em várias funções com finalidades especificas numa tentativa de manter o código com alta coesão e baixo acoplamento.

4. Conclusões

Este relatório apresentou uma solução de software para criação do jogo da disputa. O simulador foi implementado em um programa na linguagem Python. A solução resolve o problema proposto, fazendo uso de várias construções de comandos, como os de entrada e saída, estruturas de repetição e estruturas de dados.

Vários subproblemas foram explicitados neste relatório e suas soluções foram mostradas na seção de desenvolvimento.

O software resolve o problema proposto, atendendo as especificações solicitadas pelo cliente. Versões futuras podem adicionar mais flexibilidade pelo uso de conexão online para que os jogadores possam jogar de computadores diferentes.

Os pontos fortes deste relatório são a organização detalhada do problema em subproblemas e de suas soluções. Por outro lado, pode haver problemas de redação que reduzam a clareza e a concisão.

5. Referências

CURSOS, Caelum. (2004). **Python e Orientação a Objetos.** Disponível em: <https://www.caelum.com.br/apostila-python-orientacao-objetos/orientacao-a-objetos/>. Acesso em: 12 set. 2019.

BRUNO. (2013). **Algoritmos de ordenação: análise e comparação**. Disponível em: <https://www.devmedia.com.br/algoritmos-de-ordenacao-analise-e-comparacao/28261>. Acesso em: 14 set. 2019.

DOWNEY, Allen; ELKNER, Jeff; MEYERS, Chris. (2009). **Aprenda Computação com Python 2.** Disponível em: <https://aprendacompy.readthedocs.io/pt/latest/capitulo_12.html>. Acesso em: 12 set. 2019.