MAB225 – Computação II – Aula Prática 7

Objetivo: manipulação de arquivos

Para todos os exercícios: teste as classes, crie instâncias, modifique os atributos e utilize os métodos.

1. Crie uma classe em Python para implementar um jogo de forca. Para simplificar a solução, um zip com três arquivos deve ser baixado de www.pads.ufrj.br/~fernanda/compii . O código forcaIncompleto.py possui parte do método jogar pertencente à classe. Os arquivos "forcaPadrao.txt" e "pontPadrao.txt" devem estar na mesma pasta do arquivo com o código em Python. Essa classe deve possuir os seguintes atributos e métodos:
    a. Método construtor – recebe dois argumentos com os nomes dos arquivos que serão utilizados para (1) sortear uma palavra e (2) guardar a pontuação. O primeiro argumento, que deve ser o nome do arquivo de palavras, possui valor default igual a "forcaPadrao.txt". O segundo argumento, com o nome do arquivo de pontuação, possui valor default igual a "pontPadrao.txt". O método construtor deve utilizar o método verificarArquivo para saber se o nome do arquivo de palavras passado por argumento é um arquivo existe. Se não for, crie o atributo pseudo-privado com nome do arquivo de palavras igual a "forcaPadrao.txt". Caso contrário, crie o atributo pseudo-privado com nome do arquivo de palavras igual ao valor passado por argumento. Realize o mesmo procedimento para o arquivo de pontuação (também deve ser pseudo-privado). Em seguida, crie um atributo com o número de linhas do arquivo de palavras (abra-o para leitura, leia linha a linha, e conte a quantidade de linhas). Crie também um atributo pseudo-privado para guardar a palavra certa e outro para guardar a palavra do jogador (que é modificada a cada iteração do jogo).
    b. Método verificarArquivo – recebe o nome de um arquivo como argumento e utiliza tratamento de exceção para tentar abrir o arquivo para leitura. Caso o arquivo não exista a exceção FileNotFoundError será gerada. Se FileNotFoundError for gerada, imprima uma mensagem de erro e retorne o booleano False. Se a mensagem não for gerada, retorne o booleano True.
    c. Método adicionar – interage com o usuário para pedir a palavra que deseja adicionar e a escreve no final do arquivo com as palavras, cujo nome está salvo em um dos atributos pseudo-privados. Garanta que a palavra será escrita em uma nova linha e que o conteúdo que já existia no arquivo não será apagado.
    d. Método criar – interage com o usuário para perguntar o nome do arquivo que quer criar. Cria o novo arquivo (adicione '.txt' ao final do nome passado pelo usuário) e interage em loop para pedir cada palavra que deseja incluir. Escreva a palavra e pergunte se o usuário deseja sair. Caso o usuário digite 's', saia do loop (encerrando o método). Se não, volte a perguntar uma palavra até que a resposta à segunda pergunta seja 's'. Garanta que cada palavra será escrita em uma nova linha.
    e. Método escolherPalavra – utiliza a função randint para sortear uma linha do arquivo de palavras (entre 1 e o número de linhas guardado no atributo). Lê o arquivo linha a linha até chegar na linha selecionada. Guarda o valor lido referente à linha sorteada no atributo com a palavra certa.

f. Método jogar – parte desse método está implementado em forcaIncompleto.py. Complete as linhas que estão faltando: no primeiro caso, leia o arquivo de pontuação completo e guarde tudo o que foi lido na variável conteúdo. No segundo caso, escreva a variável conteúdo em um novo arquivo de pontuação com o mesmo nome do anterior. O método tem como objetivo interagir com o usuário para perguntar letra a letra que o usuário deseja tentar. A cada iteração, mostra o conteúdo do atributo com a palavra do jogador, começando com uma string composta de asteriscos e tamanho igual à palavra certa. A medida que o usuário acerta, a letra é incluída nas posições corretas. A cada iteração pergunta se o usuário já possui um palpite da palavra correta. Se o usuário acertar antes de terminar a quantidade máxima de tentativas, ele ganha uma pontuação igual ao número de letras distintas da palavra mais um e menos o número de tentativas. Ao terminar, pergunte o nome do jogador e inclua sua pontuação no arquivo de pontuação. O código implementado vê se o nome passado já existe. Se o nome do usuário já existir, a pontuação deve ser somada. Se não, a pontuação do usuário e seu nome devem ser acrescentados no final do arquivo.