INF101 – Introdução à Programação II
Roteiro de Prática: 6 de maio de 2021

Introdução

Vamos considerar, na prática de hoje, o problema de elaboração de um *horário escolar.* Vamos usar a solução projetada pela heurística simples que foi apresentada em aula teórica. O que mais importa é o exercício de utilização da estrutura de dados *conjunto,* dentre outras também utilizadas na solução. Conjuntos permitem simplificar bastante a implementação da solução. Não há necessidade de usar arranjos (vetores e matrizes) como seria feito se usássemos linguagens mais antigas tais como C/C++, Algol ou Fortran. O que faremos aqui é organizar melhor a implementação e entrar com os dados de maneira mais genérica, por meio de arquivos, e não simplesmente engessando dados particulares na estrutura do programa. Para tanto, siga as instruções abaixo.

Instruções

1. Faça o *download* do esqueleto do programa que se encontra no Sistema de Entrega do LBI. Abra o IDLE e mude o nome do arquivo-fonte para `p13.py`. Não se esqueça de salvá-lo de tempos em tempos, porque pode ocorrer falha de energia elétrica ou do próprio sistema computacional durante a aula prática.
2. Complete os comentários obrigatórios (nome, matrícula, data de criação, data de atualização e uma breve descrição sobre o que o programa faz).
3. O programa está estruturado em quatro funções: `main()`, `leia_arq_disciplinas(arqnome)`, `leia_arq_matriculas(arqnome)`, `faz_horario_escolar(disciplinas, matriculas)`.
4. Nesta aula, você completará o código-fonte que já veio pronto no esqueleto preenchendo as seguintes três funções ainda não implementadas: `leia_arq_disciplinas`, `leia_arq_matriculas`, `faz_horario_escolar`.
5. A função `main()` faz o seguinte: primeiro, decide quais serão os nomes externos dos arquivos de entrada, se serão usados os nomes *default* ou os nomes passados como argumentos pela linha de comando. Depois, chama as funções de leitura do arquivo contendo as disciplinas a ser oferecidas e o arquivo das matrículas feitas pelos alunos. Em seguida, chama a função que elabora o horário de acordo com os argumentos `discs` e `matrics`, contendo, respectivamente, as disciplinas e as matrículas. Para finalizar, ela imprime as sessões possíveis produzidas pelo horário. A `main` já está pronta no esqueleto. Favor não a modificar.
6. A função `leia_arq_disciplinas(nomearq)` lê o arquivo de entrada cujo nome externo é passado pelo parâmetro `nomearq`. Esta função deve usar os blocos `try-except` para efetuar uma manipulação segura do arquivo. Além disso, ela deve armazenar, em uma lista, os dados lidos por linha do arquivo de entrada. Tal lista de disciplinas deve ser retornada depois pela função.
7. A função `leia_arq_matriculas(nomearq)` é semelhante à anterior (item 6 acima). Ela recebe o argumento com o nome externo do arquivo de entrada por meio do parâmetro `nomearq`. Lê o arquivo de forma segura dentro de um bloco `try-except`. Cada linha do arquivo é lida e dela são extraídos o nome do aluno e as respectivas disciplinas em que ele se matriculou. Em cada linha, os dados estão separados por vírgula, `','`. Os dados são armazenados em um dicionário em que as chaves são os nomes dos alunos e os valores associados, o conjunto das disciplinas matriculadas pelo respectivo aluno. No final da execução da função, ela retorna o dicionário produzido durante a leitura do arquivo. (**Obs.:** Não é necessário fazer aqui, agora, a validação das disciplinas matriculadas face à lista das disciplinas oferecidas, porém, em uma situação de fato prática, seria necessário validar os dados das matrículas).

8. A função `faz_horario_escolar(disciplinas, matriculas)` elabora o horário de acordo com a heurística dada em aula teórica. Implemente a função passando o código em Python dado na aula teórica para cá, sob a forma de função. O seu retorno é a lista de sessões produzidas, ou seja, a lista `horario`.
9. Teste seu programa com os dados dos arquivos fornecidos, `disciplinas.txt` e `matriculas.txt`. Veja como deve ser a saída impressa nos Exemplo 1 de Teste do Programa dado abaixo.
10. Se seu programa entrar em *laço infinito* ou travar por alguma razão, digite `CTRL-C` na janela do *Python Shell* para interromper a execução do programa.
11. Teste o seu programa também com os arquivos extras fornecidos, `disciplinas2.txt` e `matriculas2.txt`. Passe esses nomes de arquivos como argumentos da linha de comando. O IDLE permite fazer isso selecionando, para a execução do arquivo-fonte, a opção `'Run... Customized'` no menu `'Run'`, ou por meio do atalho `Shift-F5`.

☞ Não se esqueça de preencher o cabeçalho do código fonte com seus dados, as datas e uma breve descrição do programa.

Após certificar-se de que seu programa esteja correto, envie o arquivo com o código fonte (`p13.py`) através do Sistema de Entrega do LBI.

Exemplo 1 de Teste do Programa (com os arquivos *default*)

```
Sessões:
  0:  ['INF100', 'INF101']
  1:  ['FIS203', 'INF103', 'MAT140']
  2:  ['MAT141', 'MAT144']
  3:  ['LET100']
```

Exemplo 2 de Teste do Programa (com os arquivos `disciplinas2.txt` e `matriculas2.txt`)

```
Sessões:
  0:  ['INF100', 'INF101']
  1:  ['INF103']
  2:  ['MAT140']
```