MAC0113 - Introdução à Computação para Ciências Humanas

Prof. Roberto Hirata Jr.

Segundo Exercício Programa (EP1) – Data de entrega: 11/7/2021

1 Introdução

A disciplina de Introdução à Programação para Ciências Humanas do curso de Bacharelado em Administração é a primeira disciplina da área de programação do curso. Examinando a página da FEA para a graduação em Administração, encontramos o seguinte texto (os grifos são meus): "O Curso de Administração da FEA-USP é um curso multidisciplinar que qualifica o aluno para **trabalhar em todas as áreas**, com capacidades de **gerenciar** cada uma, bem como, o que é característica única do curso de Administração, **integrar o conhecimento de todas as áreas**. Dessa forma pode-se dizer que o foco do Curso de Administração é **desenvolver a capacidade do aluno para ter uma visão sistêmica na atividade de gestão possibilitando a implementação de técnicas e instrumentos de forma sustentável nos níveis ambiental, social e econômico-financeiro.**"

A disciplina de programação ajuda a alcançar os objetivos acima de várias formas, principalmente no que tange a desenvolver a capacidade do estudante em organizar seu pensamento de forma lógica, exercitando os raciocínios abdutivo, indutivo e dedutivo. Através da programação, o estudante aprende a pensar num problema e a encontrar uma forma de resolvê-lo de forma algorítmica, resolvendo-o por partes, verificando que cada parte está correta, que parte consome mais recursos etc.

Na disciplina, usualmente o estudante precisa aprender a fazer um algoritmo a partir de uma especificação matemática de um problema. Além disso, o estudante vai aprender e treinar a a escrever um código claro e bem documentado. E a eficiência? Bem, isso é um dos objetivos de outra disciplina, por exemplo, MAC0121, o que não significa que não vamos abordar o assunto. Porém, neste primeiro momento, não vamos cobrar fortemente que as soluções sejam eficientes.

A disciplina tem uma parte teórica e uma prática e é essa que nos interessa neste documento. Parte da prática é cobrada a partir de exercícios programa (*EPs*) que são feitos pelo estudante **individualmente** no seu computador pessoal, ou em algum computador a que tenha acesso. A especificação do exercício será sempre divulgada no *e-disciplinas*, assim como a data de entrega e o "link" para a entrega.

Para esta disciplina, para efeitos de avaliação, serão considerados:

- Funcionamento do código. Este item é de fundamental importância para um exercício programa ser considerado entregue. Por funcionamento, entenda-se: o código apresentado implementa o que foi especificado no enunciado? Existem várias formas de avaliar este quesito. Uma delas é executar o programa com uma bateria de testes e verificar se o programa passou nos testes, isto é, verificar se os resultados são iguais aos esperados.

- Organização e clareza do código. O código é fácil de ler e entender?

- Documentação do código. As passagens mais difíceis do algoritmo tem frases que ajudem o seu entendimento? As variáveis e constantes estão associadas a frases que dizem para que elas servem?

2 O segundo exercício programa (EP)

Especificação matemática: dadas várias avaliações de filmes que foram coletados pelo "GroupLens Research Project" da Universidade de Minnesota, calcule várias estatísticas e apresente alguns gráficos.

Os dados consistem de 100.000 avaliações com notas de 1 a 5 de 943 usuários sobre 1682 filmes. Cada usuário avaliou no mínimo 20 filmes e no conjunto de dados há informações dos usuários como: idade, gênero, ocupação e cep.

As tarefas propostas para este exercício programa são:

E1. leia o arquivo com os dados de nome "u.data" que contém uma tabela com quatro colunas: a identificação (user id) de cada usuário (de 1 a 943); a identificação do filme (item id); a nota que o usuário deu para aquele item; e um carimbo de tempo que não vamos usar. Desta vez, você terá que ler a documentação do conjunto de dados (README) para saber a formatação do arquivo.

E2. transforme os dados lidos em uma matriz tal que: as avaliações são os valores que serão colocadas na matriz formada pelos user ids nas linhas e item ids nas colunas. Por exemplo, se chamarmos de `Recomendacao` a matriz, a recomendação do user id = 100 e item id = 258 é 4:

$$> print(Recomendacao[100, 258]) > [1]4$$

Quando não houver recomendação, o conteúdo da célula deverá ser `NA`.

Dica para o exercício: (1) crie uma matriz de 943 linhas e 1682 colunas de valores `NA`. (2) os índices da matriz são respectivamente o user id e o item id.

E3. Faça uma função `contaLinha` que recebe uma matriz como parâmetro e retorna um vetor com o número de avaliações que cada usuário deu, ignorando as entradas que são `NA`.

E4. Faça uma função `contaColuna` que recebe uma matriz como parâmetro e retorna um vetor com o número de avaliações que cada filme recebeu, ignorando as entradas que são `NA`.

E5. Faça uma função `mediaColuna` que recebe uma matriz como parâmetro e retorna um vetor com a média de cada coluna da matriz, ignorando as entradas que são `NA`.

E6. Usando a função `mediaColuna`, calcule a média das notas de cada filme e armazene o resultado em um novo vetor chamado mediaFilmes.

E7. Leia o arquivo com os dados de nome "u.item" que contém uma tabela com 24 colunas cujas duas primeiras são: a identificação do filme (movie id) (de 1 a 1682); o nome do filme (movie title); etc. Se você quiser saber dos outros campos, leia a documentação do conjunto de dados (README). Usando o vetor mediaFilmes, crie um vetor com o nome de todos os filmes com média acima de 4.3 (inclusive).

E8. Usando a função `contaLinha`, conte quantas avaliações cada usuário deu e armazene em um novo vetor, contaUsers.

E9. Usando a função `contaColuna`, conte quantas avaliações cada filme recebeu e armazene em um novo vetor, contaFilmes.

E10. Faça os gráficos (com a função plot) do histograma dos vetores contaUsers e contaFilmes. Para isso, você pode usar a função `hist` (leia o help da função para entender como ela funciona). Experimente mudar o parâmetro `breaks` e faça mais dois gráficos de cada vetor mudando esse parâmetro.

A entrega do EP consistirá no envio ("upload"), via e-disciplinas, do código em R para resolver o problema. Não esqueça de colocar seu nome e número USP no EP, pois trabalhos sem essas informações não serão corrigidos.

3 Plágio

Plágio é a cópia/modificação não autorizada e/ou sem o conhecimento do autor original. O plágio é um problema grave que pode levar até a expulsão do aluno da universidade.

4 Observações:

1. A endentação correta do programa será considerada.
2. O exercício é **individual**. Exercícios que indiquem o contrário serão tratados como quebra do código de ética discente da USP, receberão nota 0 (zero) e o caso será levado ao conhecimento da Câmara de Graduação da FEA.
3. O cabeçalho do programa deve ser **obrigatoriamente** como abaixo. Não esqueça de colocar o seu nome completo, o seu número USP, e a sua turma (21 ou 22).

```
#################################################################
##  AO PREENCHER ESSE CABEÇALHO COM O MEU NOME E O MEU NÚMERO USP,
##  DECLARO QUE SOU O ÚNICO AUTOR E RESPONSÁVEL POR ESSE PROGRAMA.
##  TODAS AS PARTES ORIGINAIS DESSE EXERCÍCIO PROGRAMA (EP) FORAM
##  DESENVOLVIDAS E IMPLEMENTADAS POR MIM SEGUINDO AS INSTRUÇÕES
##  DESSE EP E QUE PORTANTO NÃO CONSTITUEM DESONESTIDADE ACADÊMICA
##  OU PLÁGIO.
##  DECLARO TAMBÉM QUE SOU RESPONSÁVEL POR TODAS AS CÓPIAS
##  DESSE PROGRAMA E QUE EU NÃO DISTRIBUI OU FACILITEI A
##  SUA DISTRIBUIÇÃO. ESTOU CIENTE QUE OS CASOS DE PLÁGIO E
##  DESONESTIDADE ACADÊMICA SERÃO TRATADOS SEGUNDO OS CRITÉRIOS
##  DIVULGADOS NA PÁGINA DA DISCIPLINA.
##  ENTENDO QUE EPS SEM ASSINATURA NÃO SERÃO CORRIGIDOS E,
##  AINDA ASSIM, PODERÃO SER PUNIDOS POR DESONESTIDADE ACADÊMICA.

##  Nome :
##  NUSP :
##  Turma:
##  Prof.: Roberto Hirata Jr.

##  Referências: Com exceção das rotinas fornecidas no enunciado
##  e em sala de aula, caso você tenha utilizado alguma refência,
##  liste-as abaixo para que o seu programa não seja considerado
##  plágio ou irregular.

##  Exemplo:
##  - O algoritmo Quicksort foi baseado em
##  http://wiki.python.org.br/QuickSort
#################################################################

##  [Seu programa]
```