Fundação CECIERJ - **Vice-Presidência de Educação Superior a Distância**

Curso de Tecnologia em Sistemas de Computação
Disciplina Fundamentos de Programação

AD2 – 2° semestre de 2020

IMPORTANTE

- As respostas (programas) deverão ser entregues pela plataforma em um arquivo ZIP contendo todos os arquivos de código fonte (extensão ".py") necessários para que os programas sejam testados. Respostas entregues fora do formato especificado, por exemplo, em arquivos com extensão ".pdf", ".doc" ou outras, não serão corrigidas.
- Serão aceitos apenas soluções escritas na linguagem Python 3. Programas com erro de intepretação não serão corrigidos. Evite problemas utilizando tanto a versão da linguagem de programação (Python 3.X) quanto a IDE (PyCharm) indicadas na Aula 1.
- Quando o enunciado de uma questão inclui especificação de formato de entrada e saída, tal especificação deve ser seguida à risca pelo programa entregue. Atender ao enunciado faz parte da avaliação e da composição da nota final.
- Os exemplos fornecidos nos enunciados das questões correspondem a casos específicos apontados para fins de ilustração e não correspondem ao universo completo de entradas possíveis especificado no enunciado. Os programas entregues devem ser elaborados considerando qualquer caso que siga a especificação e não apenas os exemplos dados. Essa é a prática adotada tanto na elaboração das listas exercícios desta disciplina quanto no mercado de trabalho.
- Faça uso de boas práticas de programação, em especial, na escolha de identificadores de variáveis, subprogramas e comentários no código.
- As respostas deverão ser entregues via atividade específica na Plataforma antes da data final de entrega estabelecida no calendário de entrega de ADs. Não serão aceitas entregas tardias ou substituição de respostas após término do prazo.
- As ADs são um mecanismo de avaliação individual. As soluções podem ser buscadas por grupos de alunos, mas a redação final de cada prova tem que ser individual. Respostas plagiadas não serão corrigidas.

Boa Avaliação!

1ª Questão (2,0 pontos)

Faça um programa, contendo subprograma(s), que leia da entrada padrão o nome de um arquivo texto e escreva qual a palavra de maior comprimento no arquivo, qual seu comprimento e qual linha ela ocorre primeiro. Caso haja empate, escreva uma delas.

Definição: Um número inteiro é primo se e somente se ele é maior que 1 e só é divisível por 1 e por ele mesmo.

Exemplo

Entrada	Saída

`samba.txt`	`Palavra mais comprida contida no arquivo: aguentar`
	`Comprimento: 8 caracter(es)`
	`Localizada na linha 7 do arquivo`

Conteúdo do arquivo `samba.txt`, que contém a letra da música “Não deixe o samba morrer” dos autores Edson Conceição e Aloísio Silva:

```
Não deixe o samba morrer
Não deixe o samba acabar
O morro foi feito de samba
De samba pra gente sambar

Quando eu não puder pisar mais na avenida
Quando as minhas pernas não puderem aguentar
Levar meu corpo junto com meu samba
O meu anel de bamba entrego a quem mereça usar
Eu vou ficar no meio do povo espiando
Minha escola perdendo ou ganhando
Mais um carnaval
Antes de me despedir
Deixo ao sambista mais novo
O meu pedido final
```

2ª Questão (2,0 pontos)

Faça um programa, contendo subprograma(s), que leia da entrada padrão o código de um produto e o nome do arquivo do tipo texto existente, a ser processado. O arquivo contém todos os produtos de um supermercado, zero ou mais produtos, estando um produto por linha, onde cada linha contém um código, uma quantidade e um preço, separados por um "#". Localize o código no arquivo e escreva o preço e a quantidade disponível do respectivo produto. Caso ele não seja encontrado, escreva a mensagem: "código inexistente!!!".

Exemplo

Entrada	Saída

`wX65k`
`ForaCovid.txt`
	`Produto Localizado: wX65k`
	`Preço Unitário: R$ 7.25`
	`Quantidade Disponível: 128`

Conteúdo do arquivo `ForaCovid.txt`:

```
absfg#1000#1.99
axsds#4723#3.50
csddd#843#23.99
wX65k#128#7.25
c3974#423#0.99
akjsd#10#8.99
```

3ª Questão (2,0 pontos)

Faça um programa contendo subprograma(s) que leia da entrada padrão resultados de partidas de futebol, obedecendo ao formato apresentado no teste a seguir. Produza um dicionário da tabela do campeonato, ordenado pelos nomes dos times. Suponha que uma derrota valha zero pontos, um empate valha um ponto e uma vitória valha 3 pontos. A cada inserção de um jogo, mostre o conteúdo do dicionário. O término das entradas de resultados é dado por uma linha vazia.

Exemplo

Entrada	Saída
Corinthians#5#Flamendo#1 São Paulo#0#Botafogo#2 Vasco da Gama#4#Fluminense#4 *<linha em branco>*	Tabela após 1 partida(s): Corinthians 3 Flamengo 0 Tabela após 2 partida(s): Botafogo 3 pontos Corinthians 3 pontos Flamengo 0 pontos São Paulo 0 pontos Tabela após 3 partida(s): Botafogo 3 ponto(s) Corinthians 3 ponto(s) Flamengo 0 ponto(s) Fluminense 1 ponto(s) São Paulo 0 ponto(s) Vasco da Gama 1 ponto(s)

4ª e 5ª Questões (2,0 pontos cada)

Escreva um programa que:

a) Solicite ao usuário o nome de um arquivo binário que mantém uma sequência de valores inteiros não negativos. Cada valor é armazenado em um inteiro primitivo de 4 bytes. A mensagem a ser emitida é:

"Informe o nome do arquivo binário de números: "

b) Em seguida, abra o arquivo, mas não leia seu conteúdo de uma só vez para a memória principal. Inclusive, nessa questão é proibido fazer a carga completa do arquivo, pois deve-se assumir que o arquivo é tão grande que a leitura completa levaria à falta de memória e ao término prematuro de seu programa.

c) Ordene o conteúdo do arquivo utilizando um dos métodos de ordenação vistos em aula. Porém, lembre-se que não é permitido ler o arquivo todo para a memória de uma só vez.

d) Após a ordenação do arquivo, seu programa deverá entrar em uma repetição que, a cada iteração, solicita ao usuário que informe um número a ser localizado no arquivo. O programa deverá emitir a seguinte mensagem na solicitação:

"Favor informar o valor a ser encontrado (-1 para sair): "

Caso o valor informado seja -1, a repetição termina e seu programa é encerrado com sucesso. Caso contrário, seu programa utilizará a função "buscar", para retornar a posição do número solicitado, caso esse exista no arquivo, ou o valor -1 caso esse não exista. O valor retornado pela função deverá ser impresso para o usuário na saída padrão com a mensagem:

"O número está na posição %d\n"

ou

"O número não foi encontrado.\n"

Abaixo é apresentada a especificação da função "buscar":

```
def buscar(arq, num):
      # Utiliza deslocamento de cursos (Aula 12) e busca binária
      # para localizar no arquivo binário arq o número num
      # informado. A função retorna a posição do número no
      # arquivo, sendo que esta vai de zero à quantidade de
      # números armazenados menos 1, ou -1 caso o número não
      # conste no arquivo.
```

e) No programa principal, feche o arquivo após sair da repetição.

Dica: Para fins de depuração de código, escreva um programa auxiliar que crie arquivos binários contendo nada mais do que valores inteiros armazenados sequencialmente conforme a especificação do enunciado, como inteiros primitivos de 4 bytes. O programa auxiliar não deve ser entregue junto com a solução da questão.

Os tamanhos (quantidade de bytes) assumidos para formatos nativos de valores inteiros e de valores em ponto flutuante lidos ou escritos de arquivos binários podem variar de plataforma para plataforma. Ou seja, podem ocorrer problemas de compatibilidade entre programas que rodam perfeitamente em computadores que assumem determinados tamanhos para tipos primitivos, mas que não rodam corretamente em computadores que assumem outros tamanhos para o mesmo tipo. Para forçar a leitura e escrita assumindo os tamanhos padrão (standard)

que são indicados na Aula 12 e ficar livre de problemas de compatibilidade, inclua o símbolo "=" na frente do formato indicado nas funções `.pack` e `.unpack` de `struct`. Por exemplo, `struct.unpack("i", bloco)` converte o bloco de bytes em um valor inteiro, mas o tamanho do bloco é dependente da plataforma (não é necessariamente de 4 bytes), enquanto que `struct.unpack("=i", bloco)` converte blocos de 4 bytes em valores inteiros, independentemente da plataforma.

Distribuição de Pontos: A Questão 4 corresponde à ordenação do arquivo e vale 2,0 pontos e a Questão 5 corresponde à busca binária e também vale 2,0 pontos.

Atenção: Se a questão for resolvida considerando arquivos texto, a nota atribuída para a mesma será 0 (zero), mesmo que a solução esteja correta no contexto de arquivos texto.