Fundação CECIERJ - Vice-Presidência de Educação Superior a Distância

Curso de Tecnologia em Sistemas de Computação
Disciplina Fundamentos de Programação

APX2 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 pela atividade específica na Plataforma antes da data final de entrega estabelecida. Não serão aceitas entregas tardias ou substituição de respostas após término do prazo.
- As APXs 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.

1ª Questão (4,0 pontos)

Faça um programa contendo subprogramas que:

(1) Leia da entrada padrão o nome de um arquivo texto, onde cada linha possua zero ou mais palavras, separadas por espaços em branco.

(2) Identifique e escreva ordenadamente, na saída padrão, todos os caracteres e todas as palavras contidas no texto. Veja o exemplo a seguir.

Restrição

Faça e utilize seu método de ordenação. Não é permitido o uso de métodos pré-implementados em Python, tais como sort, sorted, etc.

Exemplo

Conteúdo do arquivo Drummond (contendo o Poema “No Meio do Caminho”)
No meio do caminho tinha uma pedra Tinha uma pedra no meio do caminho Tinha uma pedra No meio do caminho tinha uma pedra Nunca me esquecerei desse acontecimento Na vida de minhas retinas tão fatigadas Nunca me esquecerei que no meio do caminho Tinha uma pedra Tinha uma pedra no meio do caminho No meio do caminho tinha uma pedra

Entrada
Drummond
Saída
Caracteres encontrados no arquivo: N T a c d e f g h i m n o p q r s t u v ã Palavras encontradas no arquivo: Na No Nunca Tinha acontecimento caminho

```
de
desse
do
esquecerei
fatigadas
me
meio
minhas
no
pedra
que
retinas
tinha
tão
uma
vida
```

Distribuição de Pontos

Entrada – 0,4 pontos; Identificação dos caracteres – 0,75 pontos; Identificação das palavras – 0,75 pontos; Ordenação dos caracteres – 1,0 ponto; Ordenação das palavras – 1,0 ponto; Saída – 0,1 pontos.

2ª Questão (2,5 pontos)

Faça um programa contendo subprogramas que:

(1) Leia da entrada padrão o nome do arquivo texto, contendo em cada linha o resultado de uma medalha nas Olimpíadas de Tóquio, no seguinte formato:

País#Medalha#Modalidade

(2) Produza e escreva na saída padrão uma tabela de medalhas das olimpíadas. A tabela deve estar ordenada decrescentemente pelo número de medalhas, onde uma medalha de ouro valha mais que todas as medalhas de prata; uma medalha de prata valha mais que todas as medalhas de bronze. Caso haja empate entre dois países, escreva o alfabeticamente maior antes. Veja exemplo a seguir.

Restrição

Faça e utilize seu método de ordenação. Não é permitido o uso de métodos pré-implementados em Python, tais como sort, sorted, etc.

Exemplo

Conteúdo do arquivo tokio
Brasil#ouro#futebol feminino Argentina#bronze#basquete masculino Brasil#prata#futebol masculino Argentina#prata#tênis masculino Brasil#prata#volei feminino China#ouro#tênis de mesa masculino Japão#ouro#judô feminino -50 kg

Entrada	Saída
tokio	Tabela: (1, 2, 0, 'Brasil') (1, 0, 0, 'Japão') (1, 0, 0, 'China') (0, 1, 1, 'Argentina')

Distribuição de Pontos

Entrada – 0,4 pontos; Organização do quadro de medalhas – 2,0 pontos; Saída – 0,1 pontos.

3ª Questão (3,5 pontos)

Com a chegada das festas de final de ano, chegam também às mesas os doces carregados de uvas-passas. O problema é que seu amigo Noel não gosta de uva-passa, mas adora um dos bolos que a mãe dele faz nessa época do ano. Noel quer sua ajuda para encontrar a parte do bolo que possui menos uvas-passas.

O bolo é assado em um tabuleiro retangular e cortado em pedaços quadrados, formando uma grade com L linhas e C colunas de pedaços. Cada pedaço pode ter entre zero e dez uvas-passas. Sua tarefa é escrever um programa que encontre a porção consecutiva composta por M×N pedaços com a menor quantidade total de uvas-passas, onde M é a quantidade de linhas e N a quantidade de colunas de pedaços que formam a porção.

Descrição da Entrada

A entrada é fornecida como um arquivo binário de nome “bolo.bin” composto por 4 + L × C valores inteiros. Os quatro primeiros valores inteiros correspondem, respectivamente, aos valores de L, C, M e N. Os próximos L × C valores inteiros indicam a quantidade de uvas-passas contidas em cada um dos pedaços do bolo. Os pedaços são informados linha a linha.

Descrição da Saída

Seu programa deve imprimir na saída padrão um único valor indicando a quantidade mínima de uvas-passas que uma porção M×N de pedaços pode ter.

Exemplos

Os exemplos abaixo exibem valores numéricos separados por espaços em branco apenas para fins de ilustração. Na prática os arquivos de entrada são arquivos binários e devem ser tratados como tais.

Arquivo “bolo.bin”	Saída Padrão
3 3 1 1 1 2 3 1 3 3 1 10 1	10

Arquivo “bolo.bin”	Saída Padrão
4 4 2 1 1 2 3 4 5 6 7 8 1 10 5 2 1 5 9 10	16

Arquivo “bolo.bin”	Saída Padrão
5 5 2 2 1 1 1 3 1 1 2 1 1 1 1 1 1 2 1 1 1 2 1 1 1 3 1 1 3	7

Observação

Se a questão for resolvida considerando arquivos “bolo.bin” textuais então a nota atribuída para a mesma será 0 (zero), mesmo que a solução esteja correta no contexto de arquivos texto.

Dicas

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

Entrada – 0,4 pontos; Processamento – 3,0 pontos; Saída – 0,1 pontos.

Boa Avaliação!