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

Curso de Tecnologia em Sistemas de Computação
Disciplina Fundamentos de Programação

AD2 – 2° semestre de 2019

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 "Entrega de AD2" 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 (1,5 pontos)

Faça um programa que leia da entrada padrão o nome de um arquivo texto a ser processado. Este arquivo pode ter várias palavras por linha. Escreva na saída padrão todas as palavras que sejam palíndromos. Para isto, utilize em seu código um subprograma recursivo que verifica se uma string é ou não palíndomo. Utilize o cabeçalho a seguir:

```
def palindromo(palavra):
```

que retorna um valor booleano (`True` ou `False`).

Definição

Uma string é palíndromo se e somente se o primeiro caractere é igual ao último, o segundo é igual ao penúltimo, e assim sucessivamente.

Exemplo

Entrada (um arquivo texto)
Hoje quem mais ama a anna é o chico Amanhã de asa delta vai descer o Augusto socorrammesubinoonibusemmarrocos bob esponja

Saída
ama a anna é o asa socorrammesubinoonibusemmarrocos bob

2ª Questão (1,5 pontos)

Faça um programa que leia da entrada padrão dois nomes de arquivos texto. Um já existente e outro a ser criado. Suponha que arquivo já existente possua apenas números inteiros, podendo ter vários números em cada linha, separados por espaço em branco. Leia cada linha do arquivo de entrada e escreva apenas no arquivo de saída, a ser criado, os números primos. Caso em uma linha não ocorra nenhum primo escreva uma linha vazia no arquivo de saída. Veja exemplo a seguir.

Definição

Um número inteiro é primo se e somente se for maior que um e for apenas divisível por ele e por um.

Exemplo

Entrada (um arquivo texto contendo vários inteiros por linha)
1 2 3 4 5 6 7 8 9 10 50 -13 13 11 1 2 3

Saída (arquivo com respostas resultados correspondentes)
2 3 5 7 13 11 2 3

3ª Questão (2,0 pontos)

Faça um programa que leia da entrada padrão o nome de um arquivo texto, contendo pontos no espaço bidimensional, cada linha possui um ponto, definido por dois números de ponto flutuante, representando coordenadas x e y. Processe todo o arquivo e escreva o centro geométrico, ou centroide, dado pela média de todos os x's e y's. Processe novamente o arquivo e escreva qual é o ponto mais próximo do centroide.

Definição

A distância entre dois pontos (xA, yA) e (xB, yB) é dada pela raiz quadrada do quadrado das diferenças entre seus x's e y's, ou seja: (xA-xB) e (yA-yB).

Exemplo

Entrada (um arquivo texto contendo vários pontos, um por linha)
9.3 20.4 9.1 4.2 4.5 5.3 1.9 2.5

Saída (resultados correspondentes)
Centroide: (6.2, 8.1) Ponto Mais Próximo: (4.5, 5.3)

4ª Questão (2,0 pontos)

Um amigo seu é vendedor e viaja por todo o estado oferecendo produtos a pequenos mercados e padarias. Ao chegar em casa depois de uma semana cansativa, seu amigo percebeu que estava sem seu talão de pedidos. Logo deduziu que havia esquecido o talão em um dos clientes visitados por ele. Por conta própria, seu amigo vendedor revisitou os clientes onde esteve no último dia. Mas, infelizmente, o talão de pedidos não foi encontrado. Foi então que seu amigo pediu sua ajuda para procurar pelo material. Para isso, ele lhe informará alguns

lugares onde ele esteve na última semana. Ajude seu amigo a encontrar o bloco de pedidos informando em quais clientes é possível que ele tenha esquecido o material.

Entrada

A entrada é um arquivo binário de nome "`entrada.bin`". O arquivo é sequencial e composto por dois valores inteiros (ocupando 4 bytes cada), Q ($1 \leq Q \leq 1000$) e E ($1 \leq E \leq Q$) representado, respectivamente, a quantidade de clientes onde seu amigo esteve na última semana e a quantidade de clientes onde esteve no último dia.

Em seguida são apresentados E inteiros S_i ($1 \leq S_i \leq 1000$) contendo o número de identificação de cada um dos clientes em que ele esteve no último dia (cada inteiro S_i ocupa 4 bytes). Seguem Q inteiros C_i ($1 \leq C_i \leq 1000$) contendo o número de identificação de cada um dos clientes em que seu amigo vendedor esteve durante a última semana (cada inteiro C_i ocupa 4 bytes).

É garantido que Q, E, S_i e C_i atenderão os limites indicados. Logo, não é preciso fazer a verificação.

Saída

Seu programa deverá gerar um arquivo binário de nome "`saida.bin`" contendo o resultado. Para cada cliente em que seu amigo esteve na última semana seu programa deverá coloca sequencialmente no arquivo o valor inteiro 0 (ocupando 4 bytes) caso ele já tenha visitado esse cliente ao procurar pelo talão de pedidos, ou o valor inteiro 1 (ocupando 4 bytes) caso ele não tenha visitado esse cliente ainda enquanto procurava pelo material.

Exemplo

O exemplo que segue montra o conteúdo dos arquivos binários de entrada e saída de forma textual apenas para facilitar a leitura dos dados. Você deve estar ciente que esses arquivos são na prática binários, tendo cada valor armazenado como um valor numérico inteiro com 4 bytes.

Entrada (`entrada.bin`)
`10 5 1 15 5 998 27 1 88 15 88 99 5 100 7 27 998`

Saída (`saida.bin`)
`0 1 0 0 1 0 1 1 0 0`

Dicas

Para fins de depuração de código, escreva um programa auxiliar que crie arquivos binários contendo nada mais do que valores armazenados sequencialmente conforme a especificação do enunciado. Escreva, também, programas auxiliares que ajudem a verificar se o conteúdo do arquivo de saída está correto. Os programas auxiliares não devem ser entregues junto com a solução da questão. Caso sejam entregues, os mesmos não serão considerados na correção, independentemente de estarem corretos ou errados.

O uso de conjuntos (`set`) pode ajudar na elaboração de um programa mais simples.

Restrição

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

5ª Questão (1,5 pontos)

Implemente o subprograma `insereOrdenado`, cujo protótipo é:

```
def insereOrdenado(nomeDoArquivo, novoValor):
    # Coloque seu código aqui
```

Suponha que o arquivo texto recebido no primeiro parâmetro, isto é, em `nomeDoArquivo`, tem um número de ponto flutuante em cada linha. Além disso, suponha que o conteúdo do arquivo já está ordenado de forma crescentemente. O que o subprograma `insereOrdenado` deve fazer é inserir o `novoValor`, recebido no segundo parâmetro, no arquivo texto, de forma que o arquivo preserve sua ordenação.

Exemplo

Para o arquivo informado, de nome "`teste.txt`" e conteúdo:

Arquivo de exemplo "`teste.txt`" antes da inserção
4.6 5.12 5.12 6.8 8.5 8.67

O efeito da chamada `insereOrdenado("teste.txt", 6.0)` atualiza o arquivo para:

Arquivo de exemplo "`teste.txt`" após a inserção
4.6 5.12 5.12 6.0 6.8 8.5 8.67

Restrição

Não é permitido manter todo o conteúdo do arquivo em memória principal, sendo atribuída nota 0,0 (zero) para soluções que violem tal restrição.

Sugestão

Utilize um arquivo auxiliar.

6ª Questão (1,5 pontos)

Leia de um arquivo binário uma coleção de N pares de valores inteiros e calcule o máximo divisor comum (MDC) desses valores pela ativação da função recursiva `mdc`, de protótipo:

```
def mdc(a, b):
    # Coloque seu código aqui
```

e imprima na saída padrão os MDCs calculados. Por definição, o MDC de dois ou mais números inteiros é o maior número inteiro que é fator de tais números.

Mais especificamente, seu programa deverá:

a) Abrir e ler o conteúdo do arquivo binário "`valores.bin`", composto por um valor inteiro N que indica a quantidade de pares de valores inteiros não negativos (A e B) e os valores A e B propriamente ditos.

b) Implementar a função recursiva `mdc`, que é definida conforme o Algoritmo Euclidiano por:

`mdc(a, b) == mdc(b, a % b)`, para `b > 0` e

`mdc(a, 0) == a`, caso contrário.

c) Ativar a função `mdc` para cada um dos N pares de valores A e B lidos do arquivo e imprimir o resultado na saída padrão, colocando um resultado por linha.

Exemplo

Assumindo, neste exemplo, que o arquivo "`valores.bin`" é composto pelos valores:

Conteúdo do arquivo "valores.bin"				
2	12	18	25	26

temos que `N = 2`. Para o primeiro par temos `A = 12` e `B = 18`, enquanto que para o segundo par temos `A = 25` e `B = 26`. Os resultados a serem impressos na saída padrão são:

Saída Padrão
6 1

Dicas

1) Para o primeiro par do exemplo acima, a sequência de chamas recursivas se desenrola da seguinte forma: `mdc(12,18) == mdc(18,12) == mdc(12,6) == mdc(6,0) == 6`.

2) Lembre-se que, por padrão, cada valor inteiro ocupa 4 bytes.

Restrição

Será atribuída nota 0,0 (zero) para o item (a) de soluções que tratarem o arquivo binário como arquivo texto.