Correção: a questão está dividida em casos visíveis e ocultos. O aluno ganhará os pontos conforme a tabela abaixo. Para ganhar os pontos referentes à modularização (implementação dos tipos complexos e suas funções conforme pedido), o aluno deverá acertar pelo menos os casos visíveis.

Casos	**Pontos**
Visíveis	0,5
Ocultos	0,5
Modularização	1,5

- (BOCA: P2_Q3) Problema: Faça um programa que leia da entrada padrão uma lista de pastas com seus respectivos arquivos e uma lista de pastas com filtros. Em seguida, imprima na saída padrão os arquivos das pastas que tenham as extensões informadas.

  O código do programa deve ser modularizado, utilizando os tipos tPasta e tArquivo.

  O tPasta deve armazenar informações gerais sobre uma pasta: nome da pasta, quantidade de arquivos e os arquivos. A utilização e o acesso ao tPasta devem ser feitos através de funções do próprio tipo:

  - **tPasta tPasta_le();** Lê as informações de uma pasta da entrada padrão e retorna um tPasta com essas informações. Assumir que a função será chamada logo antes do nome da pasta ser fornecido na entrada padrão. A função deverá adicionar arquivos através da função *tPasta_adicionaArquivo*;

  - **void tPasta_imprime(tPasta pasta, char *filtro);** Recebe uma pasta e uma string de filtro como parâmetros. Imprimindo o nome da pasta seguido por dois pontos. Nas linhas seguintes, será impresso os arquivos da pasta que têm a extensão igual ao filtro, com dois espaços antes do nome do arquivo. Caso o filtro seja igual a *string* "-" (sinal de negativo), deve-se imprimir todos os arquivos da pasta;

  - **tPasta tPasta_adicionaArquivo(tPasta pasta, tArquivo arquivo);** Recebe uma pasta e um arquivo como parâmetros. Retornando a pasta atualizada com o arquivo adicionado;

- **int tPasta_ehPasta(tPasta pasta, char *nome);** Recebe uma pasta e um nome de pasta como parâmetros. Retorna verdadeiro se a pasta tiver o mesmo nome fornecido, caso contrário, retorna falso.

O tArquivo deve armazenar informações sobre um arquivo: nome e extensão. A utilização e o acesso ao tArquivo devem ser feitos através de funções do próprio tipo:

- **tArquivo tArquivo_le();** Lê as informações de um arquivo da entrada padrão e retorna um tArquivo com essas informações. Assumir que a função será chamada logo antes da extensão ser fornecida na entrada padrão;

- **void tArquivo_imprime(tArquivo arquivo);** Recebe um arquivo como parâmetro. Imprimindo na saída padrão a extensão e o nome do arquivo, ambos na mesma linha. O formato deve seguir o padrão "(X) Y", onde X é o nome da extensão do arquivo e Y o nome do arquivo;

- **int tArquivo_ehExtensao(tArquivo arquivo, char *extensao);** Recebe um arquivo e uma extensão. Retorna verdadeiro se o arquivo tiver a mesma extensão fornecida, caso contrário, retorna falso;

Definição dos formatos de entrada e saída:

- Entrada: A entrada começa com uma linha contendo um número inteiro N (1 ≤ N ≤ 10) representando a quantidade de pastas, seguido das N pastas (uma após a outra), e finaliza com Q nomes de pastas e extensões. Cada pasta será representada por um nome e um inteiro M, separados por um espaço. O inteiro M (M ≤ 100) representa a quantidade de arquivos da pasta. As M linhas seguintes contêm, cada uma, a extensão e o nome do arquivo. Todas as strings terão no máximo 100 caracteres, sem espaços ou caracteres especiais. Após todas as pastas e seus respectivos arquivos, haverá uma linha contendo apenas um número inteiro Q representando a quantidade de pastas requisitadas a terem seus arquivos exibidos na tela. Cada uma das Q linhas seguintes conterá apenas o nome da pasta e, separado por um espaço, uma extensão que será utilizada como filtro. Assuma que os nomes das pastas requisitadas estarão sempre entre as pastas fornecidas anteriormente.

- Saída: A saída deve exibir os arquivos das Q pastas requisitadas que tenham o mesmo filtro informado junto à pasta em questão. O formato de impressão deve seguir as funções tPasta_imprime e tArquivo_imprime. Note que, conforme definido em tPasta_imprime, o filtro "-" deve imprimir todos os arquivos da pasta. A ordem dos arquivos da pasta deve ser a mesma ordem fornecida na entrada padrão.

Ver exemplos de formato de entrada e saída nos arquivos fornecidos com a questão.

Observação: Recomenda-se utilizar a função *strcmp* da biblioteca *string.h*. A função strcmp compara duas strings retornando 0 quando elas são iguais. Para maiores informações, consultar o manual com *man strcmp*.