1 - Implemente um TAD Jogador de Futebol. Cada jogador possui os campos nome, jogos, gols e assistências.

- Atribui: atribui valores para os campos.
- Imprime: imprime os dados do jogador.
- Soma: soma estatísticas de dois jogadores.
- EhBom: testa se o jogador é bom (defina seu próprio critério de bom jogador).
- Libera: libera a memória alocada para o jogador de futebol.

Obs1.: Utilize alocação dinâmica.
Obs2.: Crie o main para testar seu TAD.

```
typedef struct {
   char nome[50];
   int jogos;
   int gols;
   int assistencias;
} JogadorFutebol;
```

2 - Crie um TAD chamado Aluno com os seguintes campos: nome, matrícula, nota1, nota2 e nota3. Implemente as seguintes operações (cada uma vale 1 ponto):

- Atribuir: Crie uma função que atribui valores para os campos do aluno.
- Imprimir: Implemente uma função que imprime os dados do aluno.
- Média: Crie uma função para calcular a média das notas do aluno.
- Aprovação: Implemente uma função que verifica se o aluno foi aprovado ou reprovado. O critério de aprovação é que a média das notas seja maior ou igual a 6.0.
- Liberação de Memória: Crie uma função para liberar a memória alocada para o aluno.

Utilize alocação dinâmica para criar e manipular os alunos. Além disso, crie uma função main para testar seu TAD, onde você deve criar pelo menos dois alunos, atribuir valores, imprimir seus dados, calcular a média e verificar se foram aprovados ou reprovados.

Obs1: Certifique-se de que o código seja organizado e bem comentado para facilitar a compreensão.

Obs2: Defina um critério de aprovação de acordo com as médias das notas e utilize a seguinte estrutura para o TAD Aluno:

```
typedef struct {
   char nome[50];
```

```
   int matricula;
   float nota1;
   float nota2;
   float nota3;
} Aluno;
```

3 - Você foi contratado para criar um TAD chamado Livro para representar informações de livros em uma biblioteca virtual. Cada livro possui os seguintes campos: título, autor, ano de publicação e uma lista de palavras-chave. O TAD deve oferecer as seguintes operações (cada uma vale 2 pontos):

- Atribuir: Implemente uma função que atribui valores para os campos do livro, incluindo as palavras-chave. A lista de palavras-chave deve ser implementada como uma estrutura de dados adequada.

- Imprimir: Crie uma função que imprime os dados do livro, incluindo título, autor, ano de publicação e lista de palavras-chave.

- Adicionar Palavra-chave: Implemente uma função que permite adicionar palavras-chave à lista do livro.

- Buscar Palavra-chave: Crie uma função que verifica se uma determinada palavra-chave está presente na lista do livro.

- Liberação de Memória: Crie uma função para liberar a memória alocada para o livro, incluindo a lista de palavras-chave.

- Lista de Livros por Autor: Implemente uma função que recebe uma lista de livros e o nome de um autor como entrada e retorna uma lista contendo todos os livros escritos por esse autor.

- Remoção de Palavra-chave: Crie uma função que permite remover uma palavra-chave da lista do livro.

Utilize alocação dinâmica para criar e manipular os livros e suas listas de palavras-chave. Além disso, crie uma função main para testar seu TAD, onde você deve criar uma lista de livros, adicionar, buscar e remover palavras-chave, listar os livros de um autor específico e, finalmente, liberar a memória alocada.

Obs1: Certifique-se de que o código seja organizado e bem comentado para facilitar a compreensão.

Obs2: Implemente uma estrutura adequada para a lista de palavras-chave de cada livro.

4 - Você foi designado para criar um TAD chamado Contato para representar informações de contatos em uma agenda. Cada contato possui os seguintes campos: nome, número de telefone e um conjunto de etiquetas que o descrevem. O TAD deve oferecer as seguintes operações (cada uma vale 3 pontos):

- Atribuir: Implemente uma função que atribui valores para os campos do contato, incluindo as etiquetas. As etiquetas devem ser representadas como uma estrutura de dados adequada.

- Imprimir: Crie uma função que imprime os dados do contato, incluindo nome, número de telefone e etiquetas.

- Adicionar Etiqueta: Implemente uma função que permite adicionar etiquetas ao conjunto do contato.

- Buscar Contatos por Etiqueta: Crie uma função que recebe uma etiqueta como entrada e retorna uma lista de contatos que contenham essa etiqueta.

- Remover Contato: Implemente uma função que permite remover um contato da agenda com base no nome.

- Listar Todos os Contatos: Crie uma função que lista todos os contatos da agenda.

- Atualizar Número de Telefone: Implemente uma função que permite atualizar o número de telefone de um contato existente.

- Liberação de Memória: Crie uma função para liberar a memória alocada para o contato, incluindo as etiquetas.

Utilize alocação dinâmica para criar e manipular os contatos e suas etiquetas. Além disso, crie uma função main para testar seu TAD, onde você deve criar uma agenda de contatos, adicionar, buscar, remover e atualizar contatos, listar todos os contatos e, finalmente, liberar a memória alocada.

Obs1: Certifique-se de que o código seja organizado e bem comentado para facilitar a compreensão.

Obs2: Implemente uma estrutura adequada para representar o conjunto de etiquetas de cada contato.

5 - Você foi encarregado de criar um TAD chamado Projeto para gerenciar informações sobre projetos em uma empresa. Cada projeto possui os seguintes campos: nome do projeto, data de início, data de conclusão, uma lista de tarefas associadas e uma lista de membros da equipe que trabalham no projeto. O TAD deve oferecer as seguintes operações (cada uma vale 4 pontos):

Atribuir: Implemente uma função que atribui valores para os campos do projeto, incluindo tarefas e membros da equipe. As tarefas e membros da equipe devem ser representados como estruturas de dados adequadas.

Imprimir: Crie uma função que imprime os dados do projeto, incluindo nome, datas, tarefas e membros da equipe.

Adicionar Tarefa: Implemente uma função que permite adicionar tarefas ao projeto. Cada tarefa deve conter uma descrição e um status (concluída ou em andamento).

Adicionar Membro da Equipe: Crie uma função que permite adicionar membros à equipe do projeto.

Remover Tarefa: Implemente uma função que permite remover uma tarefa do projeto com base na descrição.

Conclusão de Tarefa: Crie uma função que marca uma tarefa como concluída com base em sua descrição.

Listar Projetos por Membro da Equipe: Implemente uma função que recebe o nome de um membro da equipe e retorna uma lista de projetos em que esse membro está envolvido.

Liberação de Memória: Crie uma função para liberar a memória alocada para o projeto, incluindo tarefas e membros da equipe.

Utilize alocação dinâmica para criar e manipular os projetos, tarefas e membros da equipe. Além disso, crie uma função main para testar seu TAD, onde você deve criar e gerenciar vários projetos, adicionar tarefas, membros da equipe, listar projetos por membros da equipe, marcar tarefas como concluídas e, finalmente, liberar a memória alocada.

Obs1: Certifique-se de que o código seja organizado e bem comentado para facilitar a compreensão.

Obs2: Implemente estruturas adequadas para representar tarefas e membros da equipe.