Prática 3 – Structs

1. Um ponto é definido por duas coordenadas (X, Y) no plano cartesiano. Codifique um programa em C qu e utilize a estrutura **Ponto**, contendo as coordenadas. O programa deve ainda possuir as seguintes funções:
   a. **imprime_coord**: imprime na tela as duas coordenadas de um ponto;
   b. **le_coord**: lê do teclado os valores das coordenadas para um ponto;
   c. **distancia**: calcula a distância entre dois pontos, através da equação $d = \sqrt{(x_{p2} - x_{p1})^2 - (y_{p2} - y_{p1})^2}$
   d. a função **main** deverá ler valores para dois pontos, imprimir suas coordenadas e calcular a distância entre eles. Todas as variáveis devem ser locais, e as funções devem receber, ou retornar, as estruturas dos pontos através de ponteiros, somente.
2. Um sistema de segurança militar, usado num submarino nuclear, controla o acesso de usuários a três sub-sistemas (armamento, navegação e comunicações) através da digitação do *login* do usuário e de sua senha. Cada usuário tem um certo privilégio de acesso a cada um dos três subsistemas; o privilégio definirá o que o usuário poderá fazer no sub-sistema específico. Você deve implementar, em linguagem de programação C, algumas funções deste sistema de segurança, atendendo completamente as especificações. *Leia todas as especificações do programa antes de começar a implementar!*

   a. **Especificações gerais:** o controle é representado através de uma tabela conforme o exemplo (Tabela 1), denominada Tabela de Controle de Acessos. Nenhuma variável global deve ser usada, tampouco macros ou constantes com *#define*. A tabela será representada no programa através de matrizes de estruturas com os tipos apropriados. Estas estruturas serão locais à função ***main***, e as outras funções deverão receber, ou retornar, ponteiros para estas estruturas, quando necessário.

Tabela 1: Cadastro de usuários, senhas e privilégios (exemplo).

Login	Senha	Privilégio Armamento	Privilégio Navegação	Privilégio Comunicações
lcalrissian	88aagh	1	0	2
organa	7ghav0	3	2	5
ackbar	4sithlo	8	8	5
…	…	…	…	…

   b. **Função *main*:** definir nesta função as matrizes de estruturas que armazenarão a tabela de controle de acessos, prevendo o campo *login* com tamanho máximo de 12 e o campo *senha* com tamanho máximo de 6 (decida quanto à quantidade máxima de usuários). O conteúdo das matrizes deve estar zerado. Esta função proverá um menu para o usuário, onde ele escolherá a operação que deseja fazer: cadastrar usuários, senhas e privilégios; listar usuários, senhas e privilégios; acessar sub-sistema armamento; acessar sub-sistema navegação; acessar sub-sistema comunicações; sair do programa. Cada opção do menu, obviamente, chamará a(s) função(ões) apropriadas e executará os comandos extras porventura necessários.

I. Nas três opções para acessar sub-sistemas, o programa deverá ler do teclado o *login* e senha (usando a(s) função(ões) especificadas) e verificar se o usuário tem privilégio de acesso ao sub-sistema solicitado. Para que o acesso seja liberado, o privilégio do usuário tem de ser igual ou maior ao privilégio mínimo requerido por sub-sistema, de acordo com a Tabela 2. Se o acesso for liberado, imprimir na tela a mensagem "*acesso liberado*". Se o usuário não tiver privilégio suficiente, imprimir na tela a mensagem "*acesso negado*". Se a senha do usuário for inválida, imprimir "*senha inválida ou usuário inexistente*".

Tabela 2: Privilégios de acesso por sub-sistema.

Sub-Sistema	Privilégio
Armamento	8
Navegação	3
Comunicações	4

c. **Função *cadastra_usuario*:** Esta função criará a tabela de controle de acessos, com dados digitados por um administrador de sistema. A função deve solicitar a digitação do *login*, da senha e dos privilégios de acesso para cada usuário, continuamente, até que o administrador decida encerrar a entrada de dados. Gravar estes dados nas estruturas já definidas dentro da função *main*. A função deve prever um campo *login* de tamanho limite 12, o campo *senha* com tamanho máximo de 6, mas esta função deve ser capaz de trabalhar com tabelas com qualquer número de linhas (e este número *não* será digitado pelo usuário).

d. **Função *lista_usuarios*:** imprime na tela o conteúdo da tabela de controle de acessos (que foi criada em outra função do programa), com um cabeçalho apropriado. A função deve prever campo *login* com tamanho máximo 12, campo *senha* com tamanho máximo 6 e qualquer número de linhas.

e. **Função *pedido_acesso*:** ler, do teclado, o *login* e a senha do usuário. Esta função determinará se a senha digitada coincide com a senha armazenada na Tabela de Controle de Acessos. Caso positivo, a função deve retornar o valor do privilégio de acesso que o usuário tem para o sub-sistema; este sub-sistema em questão será passado para a função como parâmetro. Se a senha não for válida ou o *login* não for encontrado na Tabela, a função deve retornar o valor -1.

f. **Função *mensagem*:** recebe dois valores, um representando o privilégio mínimo de acesso a um sub-sistema, e outro representando o privilégio do usuário. Compara os dois valores. Se o privilégio do usuário for maior ou igual ao mínimo do sub-sistema, imprimir na tela a mensagem "*acesso liberado*". Se o usuário não tiver privilégio suficiente, imprimir na tela a mensagem "*acesso negado*". Se o privilégio do usuário for igual a -1, imprimir "*senha inválida ou usuário inexistente*".