Exercício 1

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 com os tipos apropriados; um máximo de três matrizes pode ser usado. As matrizes são definidas na função *main* (que não precisa ser implementada).

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 *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 matrizes 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).

c. **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.

d. **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.

Solução:

```
/*  SISTEMA DE SEGURANÇA DE SUBMARINO NUCLEAR
*
*                                          Privilegio
*           Login       Senha       Sub 1  Sub 2  Sub 3
*  +------------+-------------+--------+--------+-------+
*  |            |             |        |        |       |
*  +------------+-------------+--------+--------+-------+
*  |            |             |        |        |       |
*  +------------+-------------+--------+--------+-------+
*  |            |             |        |        |       |
*  +------------+-------------+--------+--------+-------+
*
*  O privilégio mínimo para acesso aos subsistemas é:
*
*     Sub-Sistema Privilégio
*     Armamento         8
*     Navegação         3
*     Comunicações      4
*
*/

#include <stdio.h>
#include <stdlib.h>
#include <string.h>

//Protótipos
int pedido_acesso(char login[][13], char senha[][7], int privilegio[][3], int
linha, int subsis);
void cadastra_usuario(char login[][13], char senha[][7], int privilegio[][3],
int linha);
void lista_usuarios(char login[][13], char senha[][7], int privilegio[][3], int
linha);
void mensagem(int privSubsis, int priv);

int main() {
      char login[3][13]={0}, senha[3][7]={0};
      int privilegio[3][3]={0};
      int escolha, priv;

      while (1) {
            printf("\n\nSistema de Controle de Acesso - Submarino Nuclear\n\n");
            printf("1 - Cadastrar usuarios, senhas, privilegios");
            printf("\n2 - Listar usuarios, senhas, privilegios");
            printf("\n3 - Acessar Sub-sistema Armamentos");
            printf("\n4 - Acessar Sub-sistema Navegacao");
            printf("\n5 - Acessar Sub-sistema Comunicacoes");
            printf("\n0 - Sair do programa");
            printf("\nopcao:  ");
            scanf("%d", &escolha);

            switch (escolha) {
                  case 1:
                        cadastra_usuario(login, senha, privilegio, 3);
                        break;
                  case 2:
                        lista_usuarios(login, senha, privilegio, 3);
                        break;
                  case 3:
                        priv=pedido_acesso(login, senha, privilegio, 3, 0);
```

1: definição matriz
1: chamar funções opção 1, 2, saída.
1: chamar funções de acesso sub-sistema

```
                        mensagem(8, priv);
                        break;
                  case 4:
                        priv=pedido_acesso(login, senha, privilegio, 3, 1);
                        mensagem(3, priv);
                        break;
                  case 5:
                        priv=pedido_acesso(login, senha, privilegio, 3, 2);
                        mensagem(4, priv);
                        break;
                  case 0:
                        return 0;
                        break;
                  default:
                        printf("\nOpcao invalida.");
                        break;
            }
      }
}

/* CADASTRA USUARIOS, SENHAS E PRIVILEGIOS
*/
void cadastra_usuario(char login[][13], char senha[][7], int privilegio[][3],
int linha) {
      int i;
      char *p;
      char cont;

      for (i=0; i<linha; i++) {
            printf("\nUsuario %d:", i);
            printf("\n\tlogin:  ");
            flushall();
            fgets(login[i], 13, stdin); //limita  entrada a 12 letras, pois
última posição (coluna) tem de ser reservado para \0
            flushall(); //limpa resquícios deixados no buffer
            printf("\tsenha:  ");
            fgets(senha[i], 7, stdin);
            flushall();
            /*É preciso remover o caractere newline do string digitado.
            strchr busca o char '\n' (newline) no string digitado.  O ponteiro p
é igualado ao retorno desta função dentro do if.
            Se o ponteiro não for NULL (isto é, o newline foi encontrado), então
o conteúdo da memória apontado por p é substituído por \0.
            Ou seja, o newline é substituído por \o no string nome[i].*/
            if ((p=strchr(login[i], '\n')) != NULL)
                  *p = '\0';
            if ((p=strchr(senha[i], '\n')) != NULL)
                  *p = '\0';
            printf("\tPrivilegio Armamentos:  ");
            scanf("%d", &privilegio[i][0]);
            printf("\tPrivilegio Navegacao:  ");
            scanf("%d", &privilegio[i][1]);
            printf("\tPrivilegio Comunicacoes:  ");
            scanf("%d", &privilegio[i][2]);
            printf("\nDigite 'x' para sair, qualquer outra para continuar a
entrada:  ");
            flushall();
            scanf("%c", &cont);
            if (cont == 'x')
                  return;
      }
```

1: passagem parâmetros
1: loop controlado pelo usuário, mas com limite para **evitar estouro** das matrizes
1: ler login e senha
1: ler privilégios

```
}

/* LISTA USUARIOS, SENHAS E PRIVILEGIOS
*/
void lista_usuarios(char login[][13], char senha[][7], int privilegio[][3], int
linha) {
      int i, j;

      printf("\n\nTabela de Controle de Acessos\n\n");
      printf("\t\t\t\tPrivilegios\n");
      printf("Login\t\tSenha\tSubSist 1   SubSist 2   SubSist 3\n");
      for (i=0; i<linha; i++) {
            printf("\n%12s\t%6s\t", login[i], senha[i]);
            for (j=0; j<3; j++) {
                  printf("\t%d", privilegio[i][j]);
            }
      }
}

/* TESTA LOGIN E SENHA, DEVOLVE PRIVILEGIO ACESSO
*/
int pedido_acesso(char login[][13], char senha[][7], int privilegio[][3], int
linha, int subsis) {
      char loginl[13], senhal[7];
      char *p;
      int i=0;

      printf("\nLogin:  ");
      flushall();
      fgets(loginl, 13, stdin);
      flushall();
      printf("\nSenha:  ");
      fgets(senhal, 7, stdin);
      flushall();
      //remove newline \n se houver
      if ((p=strchr(loginl, '\n')) != NULL)
                  *p = '\0';
      if ((p=strchr(senhal, '\n')) != NULL)
                  *p = '\0';

      //faz busca na lista de usuarios e compara senha
      while (i<linha) {
            if (strcmp(loginl, login[i])==0) { //login encontrado
                  if (strcmp(senhal, senha[i])==0)
                        return privilegio[i][subsis]; //senha confere; retorne
privilégio
                  else
                        return -1; //login encontrado, mas senha não confere;
retorne imediatamente com -1
            }
            i++; //login ainda não encontrado; continue a busca
      }
      return -1; //login não encontrado na lista; retorne com -1
}

/* COMPARA PRIVILEGIOS E MOSTRA MENSAGEM
*/
void mensagem(int privSubsis, int priv) {
      if (privSubsis < priv)
            printf("\nAcesso liberado.");
      else if (priv == -1)
```

1: funcionamento requerido

1: passagem parâmetros
1: ler login e senha
1: busca por login (strcmp)
1: teste se senha confere (strcmp)
1: retorno de privilégio ou -1

1: funcionamento requerido

```
            printf("\nSenha invalida ou usuario inexistente.");
      else printf("\nAcesso negado.");
}
```