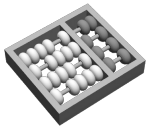

Universidade Estadual de Campinas
INSTITUTO DE COMPUTAÇÃO

MC 102 – Algoritmos e Programação de Computadores

Terceira Lista de Exercícios

1 – Sudoku é jogado numa malha de 9x9 quadrados, dividida em sub-malhas de 3x3 quadrados, chamada "quadrantes". O objetivo do jogo é preencher os quadrados com números entre 1 e 9 de acordo com as seguintes regras:

- Cada número pode aparecer apenas uma vez em cada linha.
- Cada número pode aparacer apenas uma vez em cada coluna.
- Cada número pode aparecer apenas uma vez em cada quadrante.

Exemplo:

9	5	3	4	8	6	2	7	1
1	2	7	9	3	5	8	4	6
6	8	4	7	1	2	9	3	5
5	6	8	3	9	1	4	2	7
4	9	1	2	6	7	3	5	8
3	7	2	8	5	4	1	6	9
7	4	9	5	2	8	6	1	3
2	3	6	1	7	9	5	8	4
8	1	5	6	4	3	7	9	2

Escreva uma *<u>função</u>* e um programa que utilize essa função, que lê um jogo de Sodoku (matriz 9x9, toda preenchida com números de 1 a 9) e verifica se é um jogo válido ou não. Um jogo válido respeita as três regras acima.

2 – O objetivo do jogo "Batalha Naval" é destruir as embarcações do adversário. A frota é composta por 3 tipos de embarcações, que serão posicionadas no tabuleiro:

f - fragata (tamanho 2) c - cruzador (tamanho 3) s - submarino (tamanho 4)

Considera-se que uma certa quantidade de navios foi distribuída na área de combate, de forma horizontal, vertical ou diagonal, mas sem se sobreporem. Além disso, navios do mesmo tipo não se tocam (ou seja, existe pelo menos um quadrado entre navios do mesmo tipo, sendo que neste quadrado pode haver parte de outro tipo de navio ou água).

Seu programa deverá ler o tamanho do tabuleiro (quadrilátero com l linhas e c colunas), o tabuleiro (de caracteres) montado pelo adversário, a quantidade n de jogadas, e as jogadas realizadas pelo jogador (coordenadas x e y no tabuleiro). Como saída, espera-se o número de tiros acertados (se uma parte da embarcação já foi acertada, outros tiros nessa parte não são considerados) e o número de navios de cada tipo que foram destruídos.

Exemplo de entrada:

```
5 10
-ff-ssssff
c-s-------
c-s--ccc--
c-s-----ff
--s-------
14
0 1
0 2
0 3
0 4
0 5
0 6
0 7
1 0
1 1
1 4
2 9
3 8
3 9
4 2
```

Exemplo de saída:

```
Tiros Acertados: 10
Fragata: 2
Cruzador: 0
Submarino: 1
```

3 – Historicamente César foi o primeiro codificar mensagens. Ele reorganizava o texto de suas mensagens de maneira que o texto parecia não ter sentido. Cada mensagem sempre possuía uma contagem de letras cujo total equivalia a um quadrado perfeito, dependendo de quanto César tivesse que escrever. Assim, uma mensagem com 16 caracteres usava um quadrado de quatro por quatro; se fossem 25 caracteres, seria cinco por cinco; 100 caracteres requeriam um quadrado de dez por dez, etc. Seus oficiais sabiam que deviam transcrever o texto preenchendo as casas do quadrado sempre que uma mensagem aleatória chegasse. Ao fazerem isso, podiam ler a mensagem na vertical e seu sentido se tornaria claro.

Escreva um programa que lê o tamanho de uma string e a string de um **arquivo de texto** "entrada.txt" e escreve a mensagem decifrada em outro **arquivo de texto**. "saida.txt". Você deve programar o tratamento de erro caso os arquivos não estejam disponíveis. Exemplo:

36
MEEUMOCSHMSC1T*AGU0A***L2****T*****A

Esta mensagem pode ser trancrita em um quadrado perfeito 6x6. Exemplo:

M	E	E	U	M	O
C	S	H	M	S	C
1	T	*	A	G	U
0	A	*	*	*	L
2	*	*	*	*	T
*	*	*	*	*	A

Lendo cada coluna da matriz (desconsiderando o char '*'), o arquivo de saída irá conter:

MC102 ESTA EH UMA MSG OCULTA.

4 – Escreva um programa que contenha a seguinte estrutura:

```
typedef struct duracao{  int horas;
                         int minutos;
                         int segundos;
                       } tpduracao;
```

O programa também deverá conter uma função com a seguinte assinatura:

```
int calcula(tpduracao a, tpduracao b, tpduracao *total, tpduracao *diferenca);
```

A função deverá receber por valor a duração dos experimentos *a* e *b*, e deverá calcular a soma da duração dos dois experimentos e a diferença de tempo dos mesmos, através dos parâmetros *total* e *diferenca*, recebidos por referência. O programa deverá retornar 1, por meio do uso do comando return, se o experimento *a* é mais curto ou de igual duração a *b* e 0 caso contrário.

O programa deverá receber os valores das durações dos testes e informar ao usuário as duas medidas calculadas. O programa também deverá informar qual é o teste mais curto.

5 – Você foi encarregado de desenvolver um programa para uma grande companhia de petróleo. O programa deve fornecer alguns dados sobre as bacias da companhia após um série de operações envolvendo o estoque das mesmas. Para fins de testes a companhia utilizará o programa em 5 bacias, identificadas por números de 0 a 4.

Um operador poderá entrar com as seguintes operações:

d - descoberta (adiciona o petróleo descoberto ao estoque da bacia),
c - consumo (utiliza o petróleo do estoque da bacia para fins internos),
v - venda (vende o petróleo do estoque da bacia a um certo preço),
f - fim das operações.

A companhia trabalha com dois tipo de petróleo: Pesado e Leve. Assuma que o preço da unidade de petróleo é R$100, tanto para o tipo Pesado quanto para o Leve. Assuma também que o estoque inicial de cada bacia é de 10 unidades de cada tipo de petróleo. Se ocorrer tentativa de venda ou consumo de mais petróleo do que o disponível, uma mensagem de erro deve ser exibida.

Após indicação do final das operações seu programa deverá retornar, para cada uma das bacias, o estoque total, a quantidade de petróleo Pesado, a quantidade de petróleo Leve e a renda (quantidade de dinheiro ganho com a venda do petróleo).

Além disso, a companhia também quer saber qual é a bacia com o maior estoque (se houver empate as bacias com os maiores estoques) e qual a bacia com a maior renda (se houver empate as bacias com as maiores rendas). Preços e quantidades são determinados por números inteiros.

É obrigatório o uso de vetores, registros e funções! Utilize também o comando switch.

Entrada	**Saída**
d 0 p 50 (descoberta 50 unid de petróleo pesado na bacia 0) c 1 l 10 v 2 p 5 v 3 p 10 d 4 l 30 f	0 70 60 10 0 (bacia total pesado leve renda) 1 10 10 0 0 2 15 5 10 500 3 10 0 10 1000 4 50 10 40 0 0 (bacia com maior estoque) 3 (bacia com maior renda)

6 – O programa deve ler um número inteiro n que indicará a quantidade de alunos na turma. Após isso, deve ler as notas das 3 provas de cada aluno e calcular a média de cada aluno. A média geral da turma em cada prova e o desvio padrão de cada prova também devem ser calculados e impressos. A função main é dada. Implemente as demais funções.

```
#include <stdio.h>
#define MAX_ALUNOS 10

struct prova { float p[3];
               float media;
             };
typedef struct prova tp_prova;

struct aluno { int ra[6];
               tp_prova nota;
             };
typedef struct aluno tp_aluno;

// protótipos da funções

int main(void)
{
    tp_aluno alunos[MAX_ALUNOS];
    int n;
    float medias_turma[3] = {0.0, 0.0, 0.0};   // média da turma em cada prova
    float desvio_turma[3] = {0.0, 0.0, 0.0};  // desvio da média da turma em cada prova

    leitura(alunos, &n);   // lê número de alunos e dados de cada aluno
    calcula_media_aluno(alunos, n);  // cálculo da média de cada aluno
    calcula_media_desvio_turma(alunos, n, medias_turma, desvio_turma);
    imprime_dados(alunos, n, medias_turma, desvio_turma);
    return 0;
}
```

Exemplo de entrada:

```
3
030034
4.0 5.0 6.0
123456
0.0 5.0 10.0
987654
6.0 9.0 3.0
```

Exemplo de saída:

```
RA: 030034
Prova 0: 4.0
Prova 1: 5.0
Prova 2: 6.0
Média: 5.0
----------
RA: 123456
Prova 0: 0.0
Prova 1: 5.0
Prova 2: 10.0
Média: 5.0
----------
RA: 987654
Prova 0: 6.0
Prova 1: 9.0
Prova 2: 3.0
Média: 6.0
----------
Média da turma P0: 3.3
Desvio padrão P0: 2.4
Média da turma P1: 6.3
Desvio padrão P1: 1.8
Média da turma P2: 6.3
Desvio padrão P2: 2.8
```