

# Algoritmos e Estruturas de Dados I (DCC003 TA1)

## Lista de Exercícios 3

Professora: Camila Laranjeira  
camilalaranjeira@dcc.ufmg.br

Data da entrega: 27/11/2018

- Os executáveis não devem ser enviados, apenas os arquivos de extensão ".c". Todos devem ser zipados em um arquivo chamado "**lista3.zip**".

### 1 Tipos definidos pelo usuário

- 1.1 Escreva um programa que possua a estrutura *carro*, com dois atributos: uma string *modelo* (Fusca, Gol, etc.) e *consumo*, sendo o consumo referente a quantos km esse carro faz por litro. Leia do usuário os modelos e os seus respectivos consumos e imprima qual dos carros é o mais econômico. Seu programa deve alocar dinamicamente o espaço necessário para a estrutura.

**Entrada:** A primeira linha da entrada é a quantidade  $N$  de instâncias que o usuário vai digitar. As  $2N$  linhas seguintes contém respectivamente o modelo e o consumo de cada um dos  $N$  carros.








**Saída:** A saída possui uma única linha com o nome do modelo do carro mais econômico.

**Exemplo:**

Entrada	Saída
3 Fusca 8 Prius 18 Fiesta 13	Prius

[salve o seu código com o nome: exercicio1-1.c]

**1.2** Numa partitura, as notas são representadas por símbolos diferentes que indicam a sua duração. Uma música é dividida em uma sequência de compassos, e cada compasso possui um conjunto de notas. Como Pedro é um iniciante, seu professor de música lhe ensinou que a duração das notas de um compasso deve sempre somar 1. No nosso software, cada nota vai ser representada por um caracter, de acordo com a tabela a seguir:

Notes							
Identifier	W	H	Q	E	S	T	X
Duration	1	1/2	1/4	1/8	1/16	1/32	1/64

Por exemplo, Pedro compôs uma música com cinco compassos que obedecem a regra estabelecida por seu professor:

/HH/QQQQ/XXXTXTEQH/W/HH/

Sabendo disso, **escreva um programa que armazene em um vetor de estruturas as características das notas musicais** (a estrutura *nota* deve conter dois atributos: o caracter correspondente e a duração da nota). E informe quantos compassos da entrada obedecem a regra.

**Entrada:** Cada entrada é composta por uma sequência de caracteres identificando as notas, de modo que o limite de cada compasso é separado por uma barra.

**Saída:** A saída deve ser um único inteiro correspondendo à quantidade de compassos que obedecem a regra.

**Exemplo:**

Entrada	Saída
/HH/QQQQ/XXXTXTEQH/W/HW/	4
/W/W/SQHES/	3
/WE/TEX/THES/	0

[salve o seu código com o nome: **exercicio1-2.c**]

**1.3** Vamos criar uma agenda de compromissos! Para isso serão necessárias os seguintes tipos de dados:

- Data: composto por dia, mês, ano (inteiros)
- Hora: composto por horas, minutos, segundos (inteiros)
- Compromisso: composto por data, hora e descrição (sendo a descrição uma string).

**Faça um menu para o usuário com as seguintes opções:**

1. Registrar compromisso
2. Listar todos os compromissos
3. Listar compromissos de um mês
4. Sair

A funcionalidade listar por mês, deve solicitar ao usuário o mês (de 01 a 12) e listar os compromissos daquele mês (do ano corrente). **O programa deve sempre retornar ao menu principal até que o usuário escolha a opção Sair.**

**Entrada e Saída:** Cada entrada é iniciada com a opção do menu que o usuário deseja. Cada opção requer uma quantidade diferente de parâmetros.

1. Essa opção é seguida de três linhas contendo respectivamente a data, hora e descrição do compromisso. Note que os elementos da estrutura data e hora devem ser do tipo inteiro. A saída deve ser a string "**Compromisso registrado com sucesso**". Considere que os dados de entrada estarão sempre corretos (data e hora válidos).
2. Essa opção imprime todos os compromissos registrados na agenda. Cada compromisso em uma única linha contendo respectivamente: descrição, data e hora.
3. Essa opção é seguida de uma linha indicando o mês que o usuário deseja consultar. A saída segue a mesma formatação da opção 2.
4. Encerra o programa.

**Exemplo:**

Entrada	Saída
1	Compromisso registrado com sucesso
25/11/2018	Compromisso registrado com sucesso
10:30	Reuniao 25/11/2018 10:30
Reuniao	Compras de natal 10/12/2018 09:00
1	Reuniao 25/11/2018 10:30
10/12/2018	
09:00	
Compras de natal	
2	
3	
11	
4	

[salve o seu código com o nome: exercicio1-3.c]

## 2 Arquivos

### 2.1 Escreva um programa que leia dois arquivos:

- "alunos.txt" contendo o nome e matrícula dos alunos
- "notas.txt" contendo a matrícula dos alunos e 4 notas de cada aluno

Seu programa deve escrever um novo arquivo "medias.txt" com os nomes dos alunos e as médias.

<b>alunos.txt</b>	<b>notas.txt</b>
Alberto Mota 201899	201827 92 73 84 59
Bernardo Souza 201890	201871 100 100 90 95
Bianca Portela 201871	201888 65 60 76 79
Juliana Moreira 201888	201890 70 78 92 87
Roberto Alves 201827	201899 85 70 91 80

<b>medias.txt</b>
Alberto Mota 81
Bernardo Souza 84
Bianca Portela 96
Juliana Moreira 70
Roberto Alves 77

## 2.2 Escreva um código que recebe como entrada o caminho para um arquivo e crie as seguintes funções:

- *conta\_linhas* Retorna a quantidade de linhas do arquivo
- *maior\_linha*: Retorna o número da linha com maior quantidade de caracteres (começando de 1).
- *menor\_linha*: Retorna o número da linha com menor quantidade de caracteres (começando de 1).
- *conta\_vogais*: Retorna um vetor com o número de ocorrências de cada vogal

Seu programa principal deve rodar as 4 funções em sequência e imprimir seus resultados.

### Exemplo:

<b>exemplo.txt</b>
Ae, ae, ae, ae
Ei, ei, ei, ei
Oo, oo, oo, oo, oo, oo, oo
 Quando voce chegar
Numa nova estacao
Te espero no verao

<b>Entrada</b>	<b>Saída</b>
exemplo.txt	7
	3
	4
	a: 11 e: 15 i: 4 o: 21 u: 2

[salve o seu código com o nome: **exercicio2-2.c**]

## 2.3 Incremente a implementação do exercício 1.3, salvando a agenda de compromissos do usuário em um **arquivo binário**. Para isso, seu menu deve ter as seguintes opções:

1. Registrar compromisso
2. Listar todos os compromissos

3. Listar compromissos de um mês
4. Salvar agenda
5. Carregar agenda salva
6. Sair

O programa deve sempre retornar ao menu principal até que o usuário escolha a opção Sair. As novas alternativas terão os seguintes comportamentos:

- **Salvar agenda:** o usuário informará também o caminho do arquivo onde ele deseja salvar seus registros.
- **Carregar agenda salva:** o usuário informará também o caminho do arquivo onde seus registros estão salvos. **Não imprima a agenda nessa opção**, caso o usuário deseje, ele deve escolher a opção de listar compromissos.

**Exemplo:**

Entrada	Saída
1	Compromisso registrado com sucesso
25/11/2018	Compromisso registrado com sucesso
10:30	Reuniao 25/11/2018 10:30
Reuniao	Compras de natal 10/12/2018 09:00
1	Reuniao 25/11/2018 10:30
10/12/2018	Agenda salva com sucesso
09:00	
Compras de natal	
2	
3	
11	
4	
agenda.txt	
6	

Entrada	Saída
5	Agenda carregada com sucesso
agenda.txt	Reuniao 25/11/2018 10:30
2	Compras de natal 10/12/2018 09:00
6	

[salve o seu código com o nome: exercicio2-3.c]

### 3 Recursividade

- 3.1 Faça um procedimento recursivo que receba dois valores inteiros  $a$  e  $b$  e imprima o intervalo fechado entre eles. Se  $a > b$  imprima uma mensagem de erro.

Entrada	Saída
1 10	1 2 3 4 5 6 7 8 9 10
-5 1	-5 -4 -3 -2 -1
5 -5	Valores invalidos
17 20	17 18 19 20

[salve o seu código com o nome: exercicio3-1.c]

- 3.2 Escreva uma função recursiva para determinar o número de dígitos de um inteiro  $N$ .

Entrada	Saída
578100	6
234	3
11000290	8

[salve o seu código com o nome: exercicio3-2.c]

- 3.3 Escreva uma função recursiva que calcule o valor da série  $S$  descrita a seguir para um valor  $n > 0$  fornecido como parâmetro.

$$S = 2 + \frac{5}{2} + \frac{10}{3} + \dots + \frac{1 + n^2}{n}$$

Entrada	Saída
1	2
2	4.5
3	7.83
30	468.99
57	1657.63

[salve o seu código com o nome: exercicio3-3.c]