FIT - Faculdade de Informática de Taquara
Curso de Sistemas de Informação

- Exercícios de Programação II -

Linguagem de Programação C

Profa. Flávia Pereira de Carvalho

Março de 2008

Sumário

Página

1 Exercícios - Seqüência

1.1. Escreva um programa em C para ler uma temperatura em graus Fahrenheit, calcular e escrever o valor correspondente em graus Celsius (de acordo com a fórmula abaixo).

$$\frac{C}{5} = \frac{F - 32}{9}$$

1.2. Escreva um programa em C para ler uma temperatura em graus Celsius, calcular e escrever o valor correspondente em graus Fahrenheit.

2 Exercícios - Seleção

2.1. Escreva um programa em C para ler 3 valores (considere que não serão informados valores iguais) e escrever a soma dos 2 maiores.

2.2. Escreva um programa em C para ler o número de lados de um polígono regular, e a medida do lado. Calcular e imprimir o seguinte:

Se o número de lados for igual a 3 escrever TRIÂNGULO e o valor do seu perímetro.

Se o número de lados for igual a 4 escrever QUADRADO e o valor da sua área.

Se o número de lados for igual a 5 escrever PENTÁGONO.

Em qualquer outra situação escrever Polígono não identificado.

2.3. Escreva um programa em C que leia as medidas dos lados de um triângulo e escreva se ele é EQUILÁTERO, ISÓSCELES ou ESCALENO.

Observação:

Triângulo equilátero: Possui os 3 lados iguais.

Triângulo isósceles: Possui 2 lados iguais.

Triângulo escaleno: Possui 3 lados diferentes.

2.4. Escreva um programa em C que leia o valor de 3 ângulos de um triângulo e escreva se o triângulo é acutângulo,retângulo ou obtusângulo.

Observação:

Triângulo retângulo: possui um ângulo reto (90 graus).

Triângulo obtusângulo: possui um ângulo obtuso (ângulo maior que 90 graus).

Triângulo acutângulo: possui 3 ângulos agudos (ângulo menor que 90 graus).

2.5. Escreva um programa em C que leia a idade de 2 homens e 2 mulheres (considere que a idade dos homens será sempre diferente, assim como das mulheres). Calcule e escreva a soma das idades do homem mais velho com a mulher mais nova, e o produto das idades do homem mais novo com a mulher mais velha.

2.6. Escreva um programa em C que leia as notas das 2 avaliações normais e a nota da avaliação optativa. Caso o aluno não tenha feito a optativa deve ser fornecido um valor negativo. Calcular a média do semestre considerando que a prova optativa substitui a nota mais baixa entre as 2 primeiras avaliações. Escrever a média e uma mensagem que indique se o aluno foi aprovado, reprovado ou está em exame.

3 Exercícios - Repetição

3.1. Escreva um programa em C para ler o número de alunos existentes em uma turma, ler as notas destes alunos, e calcular a média aritmética destas notas.

3.2. Ler 2 valores inteiros, calcular e escrever a soma dos inteiros existentes entre os 2 valores lidos (incluindo os valores lidos na soma). Considere que o primeiro pode ser menor que o segundo e vice-versa.

3.3. Escreva um programa em C para validar um lote de cheques. O programa deverá inicialmente solicitar a soma do lote e o número de cheques. A seguir deverá ler o valor de cada cheque calculando a soma total. Após a digitação de todos os cheques o programa deverá imprimir as seguintes mensagens:

LOTE Ok se a soma informada for igual a soma calculada.

Diferença negativa se a soma calculada for menor que a informada.

Diferença positiva se a soma calculada for maior que a informada.

Observação: O valor da diferença deve ser impresso (caso exista).

3.4. Escreva um programa em C para ler 2 valores inteiros e imprimir o resultado da divisão do primeiro pelo segundo. Se o segundo valor informado for ZERO, deve ser impressa uma mensagem de VALOR INVÁLIDO e lido um novo valor. Ao final do programa deve ser impressa a seguinte mensagem: VOCÊ DESEJA OUTRO CÁLCULO(S/N)?

Se a resposta for S o programa deverá retornar ao começo, caso contrário deverá encerrar a sua execução imprimindo quantos cálculos foram feitos.

3.5. Escreva um programa em C para ler o saldo inicial de uma conta bancária. A seguir ler um número indeterminado de pares de valores indicando respectivamente o tipo da operação (codificado da seguinte forma: 1.Depósito 2.Retirada 3.Fim) e o valor. Quando for informado para o tipo o código 3, o programa deve ser encerrado e impresso o saldo final da conta com as seguintes mensagens: CONTA ZERADA, CONTA ESTOURADA(se o saldo for negativo) ou CONTA PREFERENCIAL (se o saldo for positivo).

3.6. Escreva um programa em C para ler um número indeterminado de dados, contendo cada um o peso de um indivíduo. O último dado que não entrará nos cálculos, contém um valor negativo. Calcular e imprimir:

- A média aritmética dos pesos das pessoas que possuem mais de 60 Kg.

- O peso do mais pesado.

3.7. Escreva um programa em C para ler um valor A e um valor N. Imprimir a soma dos N números a partir de A (inclusive). Caso N seja negativo ou ZERO, deverá ser lido um novo N (apenas N).

Valores para teste

A	N	SOMA	
3	2	7	(3+4)
4	5	30	(4+5+6+7+8)

3.8. Escreva um programa em C para ler um valor X e um valor Z (se Z for menor que X deve ser lido um novo valor para Z). Contar quantos números inteiros devemos somar em seqüência (a partir do X inclusive) para que a soma ultrapasse a Z o mínimo possível. Escrever o valor final da contagem.

Exemplo:

X	Z	Reposta	
3	20	5	(3+4+5+6+7=25)
2	10	4	(2+3+4+5=14)
30	40	2	(30+31=61)

4 Exercícios - Vetores

4.1. Escreva um programa em C para ler um vetor **X** de 10 elementos inteiros. Logo após copie os elementos do vetor **X** para um vetor **Y** fazendo com que o 1º. elemento de **X** seja copiado para o 10º. de **Y**, o 2º. de **X** para o 9º. de **Y** e assim sucessivamente. Após o término da cópia, imprimir o vetor **Y**.

4.2. Escreva um programa em C para ler um vetor **A** de 10 elementos inteiros e um valor X. A seguir imprimir os índices do vetor **A** em que aparece um valor igual a X.

4.3. Escreva um programa em C para ler um vetor **A** de 10 elementos inteiros e um valor X. A seguir imprimir "ACHEI" se o valor X existir em **A** e "NÃO ACHEI" caso contrário.

4.4. Escreva um programa em C para ler um vetor **A** de 10 elementos e um valor X. Copie para um vetor **S** (sem deixar elementos vazios entre os valores copiados) os elementos de **A** que são maiores que X. Logo após imprimir o vetor **S**.

4.5. Escreva um programa em C para ler o número de elementos (N) que serão armazenados em um vetor (aceitar apenas valores entre 1 e 9). Ler os N elementos armazenando-os no vetor (alocado para 10 elementos). A seguir ler um valor X e inclua-o na 1ª. posição do vetor. Antes da inserção desloque os elementos existentes de 1 posição para o fim do vetor. Imprimir o vetor após a inclusão.

4.6. Escreva um programa em C para ler um vetor de 10 elementos inteiros. Excluir o 1º. elemento do vetor deslocando os elementos subseqüentes de uma posição para o inicio. Imprimir o vetor após a retirada do primeiro elemento.

4.7. Escreva um programa em C para ler um vetor **X** de 10 elementos e um valor P (aceitar apenas valores entre 0 e 9) que representa a posição de um elemento dentro do vetor **X**. Imprimir o valor do elemento que ocupa a posição informada. Logo após excluir esse elemento do vetor fazendo com que os elementos subseqüentes (se houverem) sejam deslocados de 1 posição para o inicio. Imprimir o vetor **X** após a exclusão ter sido executada.

4.8. Escreva um programa em C para ler um vetor **R** (de 5 elementos) e um vetor **S** (de 10 elementos). Gere um vetor **X** que possua os elementos comuns a **R** e a **S**. Considere que pode existir repetição de elementos no mesmo vetor. Nesta situação somente uma ocorrência do elemento comum aos dois deve ser copiada para o vetor **X**. Após o término da cópia, escrever o vetor **X**.

5 Exercícios - Strings

➔ Para os problemas abaixo considere que as palavras informadas possuirão no **máximo 20 letras**.

5.1. Escreva um programa em C para ler uma palavra e escrever:

-A primeira letra da palavra.

-A última letra da palavra.

-O número de letras existente na palavra.

5.2. Ler um valor **n** que representa o número de pares de palavras (2 palavras) que serão lidas. A seguir ler os **n** pares e imprimir para cada par:

-IGUAIS se as palavras informadas (do par) forem iguais.

-ORDEM CRESCENTE se as palavras (do par) foram informadas em ordem crescente.

-ORDEM DECRESCENTE se as palavras (do par) foram informadas em ordem decrescente.

5.3. Escreva um programa em C para ler 3 palavras. A seguir imprimir as 3 palavras em ordem alfabética.

5.4. Escreva um programa em C para ler uma palavra. A seguir copie para outra variável a palavra informada na ordem inversa. Imprimir a palavra copiada.

Exemplo:

Se a palavra informada for: BRASIL, a palavra copiada para a outra variável será LISARB.

➔ Para os problemas abaixo considere que as frases informadas possuirão no **máximo 80 caracteres**.

5.5. Escreva um programa em C para ler uma frase e imprimir o número de caracteres dessa frase (não utilizar a função **strlen**).

5.6. Escreva um programa em C para ler um caractere e logo após um número indeterminado de frases. Para cada frase informada imprimir o número de ocorrências do caractere na frase. O programa deve ser encerrado quando a frase digitada for a palavra "fim".

5.7. Escreva um programa em C para ler uma frase e contar o número de ocorrências de cada uma das 5 primeiras letras do alfabeto. Imprimir as contagens.

5.8. Escreva um programa em C para ler uma frase. A seguir converter todas as letras minúsculas existentes na frase para maiúsculas. Escrever a frase modificada.

5.9. Escreva um programa em C para ler uma frase e uma letra. A seguir retirar da frase, todas as letras iguais a informada. Imprimir a frase modificada.

5.10. Escreva um programa em C para ler uma frase e contar o número de palavras existentes na frase. Considere palavra um conjunto qualquer de caracteres separados por um conjunto qualquer de espaços em branco.

6 Exercícios – Funções 1

6.1. Escreva um programa em C que leia 5 valores inteiros e imprima para cada um o seu correspondente valor absoluto. Para obter o valor absoluto do número utilize a função **Absoluto** especificada abaixo:

Nome: **Absoluto**

Descrição: Retorna o valor absoluto do número fornecido.

Entrada: int n

Saída: (int) O respectivo valor absoluto de n.

Observação:

-O valor absoluto de 10 é 10.

-O valor absoluto de -10 é 10.

6.2. Escreva um programa que leia um número indeterminado de valores que representam raios de círculos. Para cada valor informado imprimir a área e seu perímetro. O programa será encerrado ao ser fornecido para o raio um valor negativo. Para obter o valor da área do círculo o programa deverá chamar a função **AreaCirculo**. Para obter o valor do seu perímetro o programa deverá chamar a função **CompCircunferencia**

Nome: **AreaCirculo**

Descrição: Retorna a área do círculo.

Entrada: float Raio

Saída: (float) A área do círculo

Nome: **CompCircunferencia**

Descrição: Retorna o comprimento da circunferência (perímetro).

Entrada: float Raio

Saída: (float) O comprimento da circunferência

6.3. Escreva um programa em C para ler 5 pares de valores (considere que serão informados apenas valores positivos). Para cada par lido deve ser impresso o valor do maior elemento do par ou a frase "Eles são iguais" se os valores do par forem iguais. Para obter o maior elemento do par utilize a função **MaiorNumero**.

Nome: **MaiorNumero**

Descrição: Retorna o maior elemento entre 2 valores positivos. Se eles forem iguais deve ser retornado o valor -1.

Entrada: (int) Dois valores positivos.

Saída: (int) O maior deles ou -1 se eles forem iguais.

Observação: Considere que os valores de entrada são sempre positivos.

6.4. Escreva um programa para ler 5 números inteiros positivos (utilize a função **LePositivo**). Para cada valor lido escrever a soma dos inteiros de 1 ao número informado. O resultado do cálculo desse somatório deve ser obtido através da função **Somatorio**.

Nome: **LePositivo**

DescriÇÃo: Faz a leitura de um valor. Se ele for negativo ou zero, a leitura deve ser repetida até que o valor lido seja positivo.

Entrada: Nenhuma.

Saída: (int) o valor lido.

Nome: **Somatório**

Descrição: Calcula o somatório dos inteiros de 1 ao número fornecido como entrada.

Entrada: (int) Número limite do somatório.

Saída: (int) O valor do somatório.

6.5. Escreva um programa que leia 5 números inteiros positivos (utilizar **LePositivo**). Para cada número informado escrever a soma de seus divisores (exceto ele mesmo). Utilize a função **SomaDivisores** para obter a soma.

Nome: **SomaDivisores**

Descrição: Calcula a soma dos divisores do número informado (exceto ele mesmo).

Entrada: Um número inteiro e positivo.

Saída: A soma dos divisores.

Exemplo: Para o valor 8: 1+2+4 = 7

6.6. Escreva um programa que imprima na tela os números primos existentes entre 1 e 100. Para verificar se um número é primo utilize a função **EhPrimo**.

Nome: **EhPrimo**

Descrição: Verifica se um número é o ou não primo.

Entrada: (int) um número inteiro.

Saída: (int) 1 se o número de entrada for primo e 0 caso contrário.

6.7. Escreva um programa que leia 5 pares de valores positivos (**LePositivo**). Imprima se os elementos de cada par são números amigos (ou não). Dois números A e B são amigos se a soma dos divisores de A excluindo A é igual a B e a soma dos divisores de B excluindo B é igual a A. Para a verificar se dois números são amigos utilize a função **SaoAmigos**.

Nome: SaoAmigos

Descrição: retorna 1 se os 2 números de entrada forem amigos, 0 caso contrário.

Entrada: (int) Dois números inteiros positivos.

Saída: (int) 1 se os dois números são amigos, 0 caso contrário.

Observação: Utilize a função SomaDividores do exercício anterior.

Exemplo:

220 e 284 são amigos, pois

220: 1+2+4+5+10+11+20+22+44+55+110=284

284: 1+2+4+71+142=220

1184 e 1210 também são amigos.

6.8. Escreva um programa que leia as medidas dos lados de 5 triângulos. Para cada triângulo imprimir a sua classificação (Não é triângulo, Triângulo Equilátero, Isósceles ou Escaleno).O programa deve aceitar apenas valores positivos para as medidas dos lados (utilizar **LePositivo**). Para obter o código da classificação utilize a função **TipoTriangulo**.

Nome: **TipoTriangulo**

Descrição: A partir das medidas dos lados de um triângulo, verifica o tipo do triângulo.

Entrada: (int) 3 valores

Saída: (int) 0 se não formam um triângulo.

1 se for um triângulo equilátero.

2 se for um triângulo isósceles.

3 se for um triângulo escaleno.

Para verificar se as medidas formam um triângulo chamar a função **EhTriangulo**.

Nome: **EhTriangulo**

Descrição: Verifica se as 3 medidas informadas permitem formar um triângulo. Para formar um triângulo é necessário que a medida de cada lado seja menor que a soma dos outros 2.

Entrada: (int) 3 valores.

Saída: (int) 1 se os 3 valores formarem um triângulo e 0 caso contrário.

6.9. Para evitar erros de digitação em números de grande importância, como código de uma conta bancária, geralmente se adiciona ao número um dígito verificador. Por exemplo, o número 1841 é utilizado normalmente como 18414, onde o 4 é o dígito verificador. Ele é calculado da seguinte forma:

a) Cada algarismo do número é multiplicado por um peso começando de 2 da direita para a esquerda. Para cada algarismo o peso é acrescido de 1. Soma-se os produtos obtidos.

1x5 + 8x4 + 4x3 + 1x2 = 51

b) Calcula-se o resto da divisão desta soma por 11:

51 % 11 = 7

c) Subtrai-se de 11 o resto obtido:

11 - 7 = 4

d) Se o valor obtido for 10 ou 11, o dígito verificador será o 0, nos outros casos, o dígito verificador é o próprio valor encontrado.

Escreva um programa que leia um número indeterminado de valores inteiros de 1 a 999. Para cada número imprima o seu correspondente dígito verificador. O programa é encerrado ao ser fornecido um número fora da faixa estabelecida (1 a 999). Para obter o valor do dígito verificador utilize a função **CalculaDigito**.

Nome: **CalculaDigito**

Descrição: Calcula o dígito verificador de um número.

Entrada: (int) Um valor inteiro.

Saída: (int) O dígito verificador do número.

6.10. Escreva um programa que leia um número indeterminado de valores inteiros de 10 a 9999 onde o último algarismo representa o seu dígito verificador e imprima para cada número uma mensagem indicando se ele está correto ou não. O programa é encerrado ao ser fornecido um número fora da faixa estabelecida (10 a 9999). Utilize a função **DigitoCorreto** para verificar se o número está correto.

Nome: **DigitoCorreto**

Descrição: Retorna 1 se o valor de entrada possui o dígito verificador correto e 0 caso contrário.

Entrada: (int) Um número inteiro.

Saída: (int) 1 se o número possui o dígito verificador correto ou 0 caso contrário. Utilizar as funções abaixo: **ObtemDigito** e **ObtemNumero**.

Nome: **ObtemDigito**

Descrição: Separa o último algarismo (a unidade) do número.

Entrada: (int) Um número inteiro

Saída: (int) O valor da unidade do número (o último algarismo)

Ex: Para a entrada 1823 a saída será 3

Nome: **ObtemNumero**

Descrição: Separa o número do dígito verificador.

Entrada: (int) Um número inteiro.

Saída: (int) O número sem o valor da unidade.

Ex: Para a entrada 1823 a saída será 182

Exercícios 7 - Funções 2

7.1.Escreva um programa que leia 10 duplas de valores inteiros. Exibir cada dupla em ordem crescente. A ordem deve ser obtida através da chamada da função **ClassificaDupla** especificada abaixo:

Nome: **ClassificaDupla**
Descrição: Classifica em ordem crescente 2 valores inteiros.
Entrada: 2 inteiros.
Saída: Os 2 inteiros em ordem crescente.

OBS: A função não deve modificar os valores dos parâmetros de entrada.

7.2.Repita o exercício anterior utilizando a função **ClassificaDupla** especificada abaixo:

Nome: **ClassificaDupla**
Descrição: Classifica em ordem crescente 2 valores inteiros.
Entrada/Saida: 2 inteiros.

OBS: Os valores classificados devem ser retornados nas mesmas variáveis de entrada.

7.3.Escreva um programa que leia 10 duplas de valores inteiros armazenando-os em 2 vetores de 10 elementos. Após a leitura de todos os elementos, imprimir as duplas que foram armazenadas nas posições pares em ordem crescente e aquelas armazenadas nas posições ímpares em ordem decrescente. Utilize a função **ImprimeDuplaClassificada** especificada abaixo para escrever os elementos na ordem desejada.

Nome: **ImprimeDuplaClassificada**
Descrição: Imprime os 2 inteiros de entrada na ordem desejada.
Entrada: 2 inteiros e um código que identifica a ordem de classificação (0-ordem crescente 1-ordem decrescente).
Saída: Nenhuma.

7.4.Repita o exercício anterior utilizando a função **ObtemDuplaClassificada** especificada abaixo:

Nome: **ObtemDuplaClassificada**
Entrada: 2 inteiros
Entrada: um código que identifica a ordem de classificação (0-ordem crescente 1-ordem decrescente).
Saída: Os 2 valores de entrada na ordem desejada

7.5.Escreva um programa que leia 10 trincas de valores inteiros. Exibir cada trinca em ordem crescente. Os valores classificados em ordem crescente devem ser obtidos através da chamada da função **ClassificaTrinca** especificada abaixo:

Nome: **ClassificaTrinca**
Descrição: Classifica em ordem crescente 3 valores inteiros.
Entrada/Saída: 3 inteiros.

7.6.Escreva um programa em C para obter o resultado das eleições para o 2o. turno. O programa deve solicitar o código dos dois candidatos. Logo após imprimir um cardápio contendo os códigos dos candidatos mais o item Fim (Cada item do cardápio conterá um número de ordem). A seguir conforme cada escolha dos eleitores o programa deve ir contando os votos que cada um recebe. Ao ser escolhido o item Fim(3) o programa será finalizado e deverá imprimir o código do vencedor com o seu respectivo percentual de votos. O programa deve ser implementado conforme o algoritmo abaixo, e devem ser utilizados os módulos descritos a seguir:

Legenda das variáveis

ct1 : contador de votos do candidato 1
ct2 : contador de votos do candidado 2
codi1 : Código do candidato 1
codi2 : Código do candidato 2
opcao : Item do cardápio escolhido pelo usuário
perc1 : Percentual de votos do candidato 1
perc2 : Percentual de votos do candidato 2

Algoritmo

```
inicio
  ct1 = 0
  ct2 = 0
  Leia codi1,codi2
  faça
    opcao = Chamada da função ObtemEscolha
    se opcao=1 então
       incrementa ct1
    senão
      se opcao=2 então
         incrementa ct2
      fimse
    fimse
  enquanto opcao diferente de 3
  Chama a função ObtemPercentuais
  Chama a função ImprimeVencedor
fim
```

Descrição das funções:

Nome: **ImprimeMenu**
Entrada: Código dos 2 candidatos
Descrição: Imprime um menu com 3 opções
item 1: Código do candidato 1
item 2: Código do candidato 2
item 3: Fim
Sáida: nenhuma

Nome: **ObtemPercentuais**
Entrada: Número de votos de cada um dos 2 candidatos
Saída: O percentual de cada candidato em relação ao total de votos.

Nome: **ObtemEscolha**
Entrada: Código dos 2 candidados, limite inferior e limite superior da escolha.
Saída: O valor da opção escolhida dentro dos limites estabelecidos.
Descrição: Imprime um cardápio utilizando os 2 códigos informados (conforme a descrição do módulo **ImprimeMenu**). Faz a leitura do teclado para obter um valor escolhido pelo usuário (opção). Este valor deve estar dentro dos limites informados. Caso o valor esteja fora dos limites deve ser impressa a mensagem 'Opcao INVALIDA' e repetir a leitura de um novo valor até que este seja válido.

OBS: Este módulo deve utilizar **ImprimeMenu**

Nome: **ImprimeVencedor**
Entrada:código do candidato 1, código do candidato 2, percentual de votos do candidato 1 e percentual de votos do candidato 2.
Descrição: Imprime o código e o percentual de votos do candidato vencedor, ou a palavra empate. Saída: Nenhuma.

7.7.Uma loja que vende CD-ROM deseja automatizar o controle de suas vendas. Escreva um programa em C que controle a venda diária dessa loja da seguinte forma. A cada compra efetuada por um cliente deve ser informado o código do vendedor (codificado da seguinte forma: **1**.Silva **2**.Soares **3**.Fim) que efetuou a venda e a respectiva quantidade de CDs vendidos. Após os dados de entrada os totais devem ser atualizados e retornar para um novo movimento. Se for informado o código 3 (fim) o programa deverá encerrar imediatamente imprimindo um resumo do dia com as seguintes informações: Número de atendimentos do Silva, número de atendimentos do Soares, número total de CD-ROMs vendidos e uma mensagem indicando a situação do dia conforme a tabela abaixo:

Total de CD-ROMs vendidos	Mensagem
abaixo de 10	Dia péssimo
acima de 50	Dia ótimo
qualquer outro valor	Dia normal

O programa deve ser implementado conforme o algoritmo abaixo, e deve ser utilizado os módulos descritos a seguir.

Legenda das variáveis

atsilva : contador do número de atendimentos do Silva
atsoares : contador do número de atendimentos do Soares
codigo : Código do vendedor
qt : quantidade total de CDs vendidos
quant : Quantidade de CDs vendidos em um atendimento
Algoritmo

```
inicio
  atsilva = 0
  atsoares = 0
  qt=0
  codigo = Chamada da função ObtemCodigoVendedor
  Enquanto codigo for diferente de 3 faça
    Leia quant
    Chama a função ContaAtendimentos para atualizar os contadores
    qt = qt + quant
    codigo = Chamada da função ObtemCodigoVendedor
  FimEnquanto
  Escreva atsilva, atsoares, qt
  Chama a função ImprimeMensagem
fim
```

Descrição das funções:

Nome: **ObtemCodigoVendedor**
Entrada: 2 valores inteiros que indicam os limites inferior e superior para validação do código do vendedor.

Saída: O código validado.

OBS: Imprimir a mensagem CODIGO INVALIDO se o código for inválido

Nome: **ContaAtendimentos**
Entrada: Codigo do vendedor
Entrada/Saída: 2 valores que indicam a contagem de vendas de cada vendedor.

OBS: Um dos contadores de entrada deve ser incrementado em função do código do vendedor informado.

Nome: **ImprimeMensagem**
Descrição: Imprime uma mensagem com a situação do dia conforme a tabela acima.
Entrada: Quantidade total vendida
Saída: nenhuma

Exercícios 8 - Funções 3

8.1.Escreva um programa em C que faça o seguinte:
- leia um vetor de 10 elementos e um número inteiro.
- Multiplique cada elemento do vetor pelo número fornecido copiando o resultado para um vetor de saída.
- Imprima na tela o vetor de saída.
O programa deve ser implementado com a utilização das seguintes funções:

Nome: **LeiaVetor10**
Descrição: Faz a leitura de um vetor de 10 inteiros.
Entrada: Nenhuma
Saída: Um vetor com os elementos obtidos pela leitura do teclado.

Nome: **Multiplica**
Descrição: Multiplica cada elemento do vetor fornecido pelo número informado. O resultado deve ser copiado para um vetor de saída.
Entrada: Um vetor de 10 elementos e um número inteiro
Saída: Um vetor com o resultado da multiplicação.

Nome: **EscreveVetor10**
Descrição: Imprime na tela os 10 elementos do vetor
Entrada: Um vetor de 10 elementos
Saída: nenhuma.

8.2.Reescreva o exercício anterior fazendo com que o resultado da multiplicação seja obtido no mesmo vetor de entrada. Utilizar as funções **LeVetor10**, **EscreveVetor10** e a seguinte função multiplica. Nome: **Multiplica**
Descrição: Multiplica cada elemento do vetor fornecido pelo número informado. O resultado deve obtido no mesmo vetor de entrada.
Entrada/Saida: Um vetor de 10 elementos Entrada: Um número inteiro
OBS: O resultado da multiplicação deve ser retornado no mesmo vetor de entrada.

8.3.Escreva um programa em C que leia um vetor de 10 elementos e imprima o maior elemento armazenado no vetor. O programa deve ser implementado utilizando as seguintes funções:

LeVetor10 (do exercício 8.1).

Nome: **ObtemMaior**
Descrição: retorna o maior elemento do vetor de entrada.
Entrada: Um vetor de 10 elementos.

Saída: O maior elemento do vetor de entrada.

Nome: **EscreveResultado**
Descrição: Escreve um número inteiro (o maior do vetor obtido na função anterior.
Entrada: Um valor inteiro.
Saída: Nenhuma.

8.4.Escreva um programa em C para ler um número N (no máximo 10) que representa o número de elementos que serão armazenados em um vetor. Ler os N elementos do vetor. Calcular e imprimir a soma dos N elementos do vetor. O programa deve ser implementado com as seguintes funções:

Nome: **LeiaVetor**
Descrição: Faz a leitura de um vetor de N elementos inteiros.
Entrada: Número de elementos a serem lidos
Saída: Um vetor com os N elementos lidos.

Nome: **CalculaSoma**
Descrição: Retorna a soma dos N elementos de um vetor.
Entrada: Vetor de inteiros e Número de elementos existentes no vetor
Saída: O valor da soma dos N elementos do vetor.

Nome: **EscreveVetor**
Descrição: Imprime na tela os N elementos do vetor
Entrada: Um vetor de N elementos e o número de elementos do vetor
Saída: nenhuma.

8.5.Escreva um programa em C que leia um vetor de 8 elementos e um valor inteiro. Imprimir quantas vezes o valor inteiro aparece no vetor fornecido. O programa deve ser implementado com as seguinte funções:

LeiaVetor (do exercício 8.4).
EscreveResultado (do exercício 8.3)

Nome: **Conta**
Descrição: Retorna o número de vezes que um valor aparece no vetor.
Entrada: Um Vetor com números inteiros.
Número de elementos existentes no vetor.
Valor inteiro que será procurado no vetor.
Saída: Número de ocorrências do valor no vetor.

8.6.Escreva um programa em C que leia um vetor de 5 elementos e um valor inteiro. Imprimir a mensagem 'EXISTE' se o elemento inteiro aparece no vetor e 'NÃO EXISTE' caso contrário. O programa deve ser implementado com as seguintes funções:

LeiaVetor (do exercício 8.4).

Nome: **Existe**
Descrição: Retorna 1 se o valor existe no vetor e 0 caso contrário.
Entrada: Vetor com inteiros.
Número de elementos existentes no vetor.
Valor inteiro que será procurado no vetor.
Saída: 1 se o valor existe no vetor, 0 caso contrário

8.7.Escreva um programa em C que leia um vetor de 10 elementos e um valor inteiro. Imprimir os

elementos do vetor de entrada que são maiores que o valor inteiro informado. O programa deve ser implementado com as seguintes funçãoes:

LeiaVetor10 (do exercício 8.1).
EscreveVetor (do exercício 8.4) para escrever os maiores.

Nome:**ObtemMaiores**
Descrição: Obtem os inteiros existentes no vetor maiores que o valor informado.
Entrada: Um vetor de 10 elementos e um valor inteiro.
Saída: Um vetor contendo os elementos do vetor de entrada que são maiores que o valor informado.
O número de elementos existentes no vetor de saída.

8.8.Ler um valor N (no máximo 9), um vetor de N elementos e um valor inteiro. Incluir o valor inteiro na primeira posição do vetor deslocando os elementos existentes nas N primeiras posições de uma posição em direção ao final do vetor. Escrever o vetor após a inclusão do elemento. O programa deve ser implementado com as seguintes funções:

LeiaVetor (do exercício 8.4)
EscreveVetor(do exercício 8.4)

Nome: **IncluiNoInicio**
Descrição: Inclui um valor inteiro na primeira posição do vetor. Entrada/Saída: Vetor com N elementos.
O número de elementos do vetor.
Valor do elemento a ser incluído.

Saída: Número de elementos resultantes no vetor.

8.9.Ler um valor N (no máximo 9), um vetor de N elementos e um valor inteiro. Ler também um outro valor que representa a posição onde o inteiro será incluído no vetor. Incluir o valor inteiro na posição informada deslocando os elementos existentes (a partir da posição informada)de uma osição em direção ao final do vetor. Escrever o vetor após a inclusão do elemento. O programa deve ser implementado com as seguintes funções:

LeiaVetor (do exercício 8.4)
EscreveVetor(do exercício 8.4)

Nome: **IncluiNoVetor**
Descrição: Inclui um valor inteiro em posiçõ determinada dentro do vetor. Entrada/Saída: Vetor com N elementos.
O número de elementos do vetor.
Valor do elemento a ser incluído.
Índice do vetor onde o elemento será incluído.
Saída: Número de elementos resultantes no vetor.

Exercícios 9 - Funções 4

9.1.Escreva um programa em C que leia uma string e um caractere e imprima quantas vezes o caractere ocorre na string. O programa deve ser implementado com a utilização da seguinte função:

Nome: **ContaLetra**
Descrição: Retorna o número de ocorrências de um caractere em uma string
Entrada: Uma string e uma letra.
Saída: O número de vezes que o caractere aparece na string.

9.2.Escreva um programa em C que leia uma string e imprima quantas vogais existem na string.O

programa deve ser implementado com a utilização da seguinte função:

Nome: **ContaVogais**
Descrição: Retorna o número de vogais existentes na string.
Entrada: Uma string.
Saída: O número de vogais existentes na string.

9.3.Escreva um programa em C que leia uma string e imprima quantas consoantes existem na string.O programa deve ser implementado com a utilização da seguinte função:

Nome: **ContaConsoantes**
Descrição: Retorna o número de consoantes existentes na string.
Entrada: Uma string.
Saída: O número de consoantes existentes na string.

9.4.Escreva um programa em C que leia uma string converta todas as suas letras para maiúsculas. O programa deve ser implementado com a utilização da seguinte função:

Nome: **ConverteParaMaiusculas**
Descrição: Converte a string de entrada para maiúsculas.
Entrada: Uma string.
Saída: A mesma string com todas as suas letras convertidas para maiúsculas.

OBS: Para converter uma letra para maiúscula basta subtrair 32 do seu código ASCII.

9.5.Escreva um programa em C que leia uma string inverta o seu conteúdo fazendo com que o último caractere passe a ser o primeiro e vice-versa. Escrever a string invertida. O programa deve ser implementado com a utilização da seguinte função:

Nome: **InverteString**
Descrição: Inverte o conteúdo de uma string.
Entrada: Uma string.
Saída: A string com seus caracteres na ordem inversa.

9.6.Escreva um programa em C que leia uma string inverta o seu conteúdo fazendo com que o último caractere passe a ser o primeiro e vice-versa. Escrever a string invertida. O programa deve ser implementado com a utilização da seguinte função:

Nome: **InverteString**
Descrição: Inverte o conteúdo de uma string.
Entrada/Saída: Uma string.
Obs: A string invertida deve ser retornada na mesma string de entrada.

9.7.Escreva um programa em C que leia uma string e um caractere. Retirar o caractere informado da string. Escrever a string modificada. O programa deve ser implementado com a utilização da seguinte função:

Nome: **RetiraCaractere**
Descrição: Retira o caractere informado da string.
Entrada/Saída: Uma string e um caractere.
Obs: A string de entrada deve ser retornada sem todas as ocorrências do caractere informado.