Projeto de Tutoria do DECSI

Apostila de CSI030 – Programação de Computadores I

João Monlevade, MG

2016/02

Projeto de Tutoria do DECSI

Apostila de CSI030 – Programação de Computadores I

Apostila de exercícios para a Tutoria da Disciplina CSI030 – Programação de Computadores I

Universidade Federal de Ouro Preto (UFOP)

João Monlevade, MG

2016/02

Agradecimentos

Agradecemos às Pró-reitorias de Planejamento (PROPLAD) e de Graduação (PROGRAD) pelas bolsas do projeto e, também, aos alunos que colaboraram durante a tutoria com o desenvolvimento da apostila e aos professores que ministraram as disciplinas e colaboraram na revisão do conteúdo da apostila.

Sumário

1 Variáveis, Constantes, Tipos de Dados e Entrada e Saída de Dados

1.1 Exercícios Resolvidos

1. Supondo as declarações int `A` = 3, int `B` = 7 e int `C` = 4, calcule o valor da expressão: `B` >= (`A` + 2)

$$
\begin{aligned}
& B >= (A + 2) \\
& = 7 >= (3 + 2) \\
& = 7 >= 5 \\
& = true
\end{aligned}
$$

2. Faça um programa em C para ler 3 números inteiros e imprimir a média deles.

   Resposta:

```c
#include <stdio.h>
int main()
{
    int num1, num2, num3;
    float media;
    printf("\nDigite o primeiro número: ");
    scanf("%d", &num1);
    printf("\nDigite o segundo número: ");
    scanf("%d", &num2);
    printf("\nDigite o terceiro número: ");
    scanf("%d", &num3);
    media = (num1 + num2 + num3) / 3.0;
    printf("A média é: $f", media);
    return 0;
}
```

Código 1.1: codigos_CEA030/Lista1/mediaInteiros.c

1.2 Exercícios Propostos

1.2.1 Variáveis, Tipos de Dados

1. Você, como programador, é responsável por modelar os problemas reais no computador. Como parte disto, você deve escolher os tipos das variáveis que armazenarão os dados no seu programa. Para cada problema abaixo, defina o tipo de variável que deve ser usada.

a) O número de portas de uma casa.

b) A idade dos alunos ingressantes.

c) O conceito para avaliação de desempenho de um aluno, medido em A, B, C, D e E.

d) O conceito para avaliação de desempenho de um aluno, medido em um intervalo de 0 (inclusive) a 10 (inclusive), com incrementos de $0,5$.

e) O salário de um funcionário.

f) A resposta para uma pergunta cujas respostas possíveis são Verdadeiro ou Falso.

g) A resposta para uma pergunta cujas respostas possíveis são a, b, c, d e e.

2. Sobre o funcionamento do tipo de dados char, responda o que está armazenado na variável abaixo em cada linha:

```
char numero;
numero = '9';
numero = 9;
```

1.2.2 Expressões Lógicas e Relacionais

3. Supondo as declarações int A = 3, int B = 7 e int C = 4, calcule o valor das expressões abaixo:

a) (A + C) > B

b) C == (B − A)

c) (B + A) <= C

d) (C + A) > B

4. Supondo as declarações int A = 5, int B = 4, int C = 3 e int D = 6, calcule o valor as expressões abaixo:

a) (A > C) && (C <= D)

b) (A + B) > 10 || (A + B) == (C + D)

c) (A >= C)&& (D >= C)

5. Supondo as declarações int A = 5, int B = 4, int C = 3, char C1 = 'A', char ↩ C2 = 'a' e int L = 0, calcule o valor as expressões abaixo:

a) B == A * C && L

b) `C1` == `C2` || 'F' != 'Q'

c) `A` + `C` < 5

d) `A` * `C` / `B` > `A` * `B` * `C`

e) ! `L`

6. Supondo as declarações int `A` = 3, int `B` = 5, int `C` = 8, int `D` = 7 e int `X` = 1, calcule o valor as expressões abaixo:

a) !(`X` > 3)

b) (`X` < 1) && (!(`B` > `D`))

c) !(`D` < 0) && (`C` > 5)

d) !((`X` > 3) || (`C` < 7))

e) (`A` > `B`) || (`C` > `B`)

f) `X` >= 2

g) (`X` < 1) && (`B` >= `D`)

h) (`D` < 0) || (`C` > 5)

i) !(`D` > 3) || !(`B` < 7)

7. Supondo as declarações int `A` = 3, int `B` = 5, int `C` = 8, int `D` = 7 e int `X` = 1, calcule o valor as expressões abaixo:

a) !(`X` > 3)

b) (`X` < 1) && (!(`B` > `D`))

c) !(`D` < 0) && (`C` > 5)

d) !((`X` > 3) || (`C` < 7))

e) (`A` > `B`) || (`C` > `B`)

f) `X` >= 2

g) (`X` < 1) && (`B` >= `D`)

h) (`D` < 0) || (`C` > 5)

i) !(`D` > 3) || !(`B` < 7)

1.2.3 Entrada e Saída

8. Este é o nosso primeiro programa em C, o famoso hello_world.

9. Codifique um programa que leia dois valores inteiros nas variáveis a e b e troque os valores contidos nas variáveis. Para verificar a troca, imprima o conteúdo das variáveis após a leitura e após a troca dos valores.

10. Crie um programa que receba três números inteiros e imprima a média dos três valores.

11. Crie um programa para calcular a área de um triângulo.

12. Crie um programa para converter graus Celsius para Fahrenheit. O usuário deve digitar a temperatura em Celsius e imprimir a temperatura convertida.

13. Codifique um programa que pergunte ao usuário a altura e a base de um retângulo e imprima a área e o perímetro deste retângulo.

14. Codifique um programa que pergunte ao usuário o raio de um círculo e imprima a área e o perímetro deste círculo.

15. Codifique um programa que leia os três lados de um triângulo e imprima a área e o perímetro deste triângulo. Para o cálculo da área, deve-se utilizar a fórmula de Heron:

$$Area = \sqrt{aux(aux - lado_1)(aux - lado_2)(aux - lado_3)}$$

em que

$$aux = \frac{lado_1 + lado_2 + lado_3}{2}$$

16. Escreva um algoritmo que dado um raio calcule o diâmetro,o perímetro e a área de um circulo.

17. Identifique e corrija os erros em cada uma das instruções a seguir (Nota: pode haver mais de um erro por instrução):

a) `scanf("d", valor);`

b) `printf("O produto de \%d e \%d e \%d"\n, x, y);`

c) `primeiroNumero + segundoNumero = somaDosNumeros`

d) `if (numero => maior)maior == numero;`

e) `*/ Programa para determinar o maior de três inteiros /*`

f) `Scanf ("\%d", umlnteiro);`

g) `printf("0 resto de \%d dividido por \%d e\n", x, y, x % y);`

h) if (x = y); printf(\%d e igual a \%d\n", x, y);

i) print("A soma e \%d\n, "x + y);

j) Printf("O valor fornecido e: \%d\n, &valor);

2 Controle do fluxo de execução

2.1 Exercícios Resolvidos

1. Codifique um programa que, dado um número, o classifique como par ou ímpar.

   Resposta:

```c
#include <stdio.h>

int main(int argc, char const *argv[])
{
    int n;
    printf("Digite o número: \n");
    scanf("%d",&n);

    if (n % 2 == 0){
        printf("%d é par.\n", n);
    } else {
        printf("%d é ímpar.\n", n);
    }

    return 0;
}
```

Código 2.1: codigos_CEA030/Lista1/parimpar.c

2. Codifique um programa que leia um número de 1 a 12, indicativo um mês, e imprima o nome do mês correspondente, por extenso.

   Resposta:

```c
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char const *argv[])
{
    int mes;
    printf("Digite o número do mês:\n");
    scanf("%d",&mes);

    switch(mes){
        case 1:
            printf("O número %d equivale ao mês de ↩
               Janeiro.\n",mes);
            break;
        case 2:
            printf("O número %d equivale ao mês de ↩
               Fevereiro.\n",mes);
            break;
        case 3:
            printf("O número %d equivale ao mês de Março↩
               .\n",mes);
```

```
            break;
        case 4:
            printf("O número %d equivale ao mês de Abril.\n",mes);
            break;
        case 5:
            printf("O número %d equivale ao mês de Maio.\n",mes);
            break;
        case 6:
            printf("O número %d equivale ao mês de Junho.\n",mes);
            break;
        case 7:
            printf("O número %d equivale ao mês de Julho.\n",mes);
            break;
        case 8:
            printf("O número %d equivale ao mês de Agosto.\n",mes);
            break;
        case 9:
            printf("O número %d equivale ao mês de Setembro.\n",mes);
            break;
        case 10:
            printf("O número %d equivale ao mês de Outubro.\n",mes);
            break;
        case 11:
            printf("O número %d equivale ao mês de Novembro.\n",mes);
            break;
        case 12:
            printf("O número %d equivale ao mês de Dezembro.\n",mes);
            break;
        default:
            printf("Número inválido para mês!\n");
            exit(1);
    }
    return 0;
}
```

Código 2.2: codigos_CEA030/Lista1/numeromes.c

3. Codifique um programa que leia 10 valores, conte quantos destes valores são negativos e imprima esta informação.

Resposta:

```
#include<stdio.h>

int main()
{
    float numero;
    int count = 0;
```

```
    int i;
    for(i = 0; i < 10; i++)
    {
        printf("Digite o numero %d: ", i);
        scanf("%f", &numero);
        if(numero < 0)
            count++;
    }
    printf("Quantidade de numero negativos: %d ", count);

    return 0;

}
```

Código 2.3: codigos_CEA030/Lista1/dez_negativos.c

4. Construa um programa que imprima um triângulo retângulo invertido abaixo, com o tamanho máximo da base sendo indicado pelo usuário.

```
      *
    * *
  * * *
* * * *
...
```

Resposta:

```
#include <stdio.h>

int main (int argc, char const *argv[])
{
  int linha, coluna, col_brancos, base;

  printf("Qual o tamanho máximo da base?\n");
  scanf("%d", &base);

  for(linha = 1; linha <= base; ++linha)  {
    for(col_brancos = 1; col_brancos <= base - linha; ++↩
       col_brancos) {
      printf("  ");
    }

    for(coluna = 1; coluna <= linha; ++coluna)  {
      printf(" * ");
    }
    printf("\n");
  }
  return 0;
}
```

Código 2.4: codigos_CEA030/Lista2/triangulo_star_direita.c

2.2 Exercícios Propostos

2.2.1 Desvio condicional simples: if-then e if-then-else

1. Codifique um programa que leia dois números e os imprima em ordem decrescente.
2. Codifique um programa que leia três números e os imprima em ordem crescente.
3. Escreva um programa que receba um número e diga se este é maior do que vinte.
4. Crie um programa que lê um inteiro e informa se este número é par ou ímpar.
5. Codifique um programa que leia três números e imprima o maior deles.
6. Codifique um programa que leia um intervalo (deve-se ler o valor inferior e o valor superior do intervalo) e um número. O programa deve dizer se o número lido está dentro ou fora do intervalo informado.
7. Codifique um programa que, dado um número, o classifique como positivo, negativo ou neutro (0).
8. Escreva uma algoritmo que receba um número inteiro e verifique se o mesmo é ou não um palíndromo. Dica: utilize os operadores de resto e divisão inteira para decompor seu número em 5 algarismos.
9. Para doar sangue é necessário que a pessoa tenha entre 18 e 67 anos e possua mais de 50Kg. Faça um programa que receba o nome da pessoa, sua idade e seu peso e verifique se ela pode ou não doar sangue. Faça uma calculadora de 4 operações. O programa deve exibir um menu para que o usuário escolha qual operação a ser realizada e exibir uma mensagem de erro caso seja escolhida uma opção errada.
10. Escreva um programa que receba 3 números inteiros referentes aos lados de um triângulo, verifique se os mesmos satisfazem a condição de existência do triângulo e caso satisfaçam indique se o triângulo é equilátero, isósceles ou escaleno.
11. Codifique um programa que leia um par ordenado (x, y) e informe a qual quadrante ele pertence.
12. Codifique um programa que, dados dois valores inteiros entre 1 e 10, calcule e imprima:
    - a média dos números, caso a soma deles seja menor que 8;
    - o produto entre os números, caso a soma deles seja igual a 8;
    - a divisão do maior número pelo menor, caso a soma deles seja maior que 8.

13. Crie um programa que recebe o peso e a altura de um indivíduo e calcula o seu IMC e para cada valor do IMC exibe uma informação sobre o peso do indivíduo. Para IMC maior que 20 e menor que 25 exibir a mensagem de peso ideal, para IMC entre 25 e 30 exibir uma mensagem de sobre peso, para IMC entre 30 e 40 exibir uma mensagem de obesidade e para IMC maior que 40 de obesidade mórbida.

14. Considerando o salário mínimo como R$750,00 crie um programa que recebe o salário de um funcionário e calcula o novo salário deste funcionário. Para que o funcionário tenha aumento seu salário deve ser de até 1,5 vezes o salário mínimo. O aumento dado será de 30%. Funcionários que recebam mais do que 1,5 salários mínimos não recebem aumento.

15. Crie um programa que recebe um número e diga se este é par ou ímpar.

16. Crie um programa que receba uma equação de segundo grau e calcule suas raízes. O programa deve tratar o caso de delta negativo, encerrando o mesmo.

17. Crie um programa que receba três notas, calcule a média e trate os seguintes casos: media >= 6.0 -> O aluno passou. media maior que 5 e menor que 6 -> O aluno tem direito a fazer prova extra. media maior que 4.0 ou maior ou igual a 5 -> O aluno tem direito a fazer prova final. media menor que 4 -> O aluno está reprovado.

2.2.2 Desvio condicional múltiplo: case ou switch

18. Codifique um programa que leia um caractere de operação aritmética (+, −, ∗, /), dois números e exiba na tela a operação, seguida do seu resultado.

19. Codifique um programa que leia um número de 1 a 12, indicativo um mês, e imprima o nome do mês correspondente, por extenso.

20. Codifique um programa que pergunte um código de funcionário e, de acordo com o valor digitado, apresente o cargo correspondente, segundo a tabela abaixo. Caso seja digitado um código que não esteja na tabela, deve-se avisar que o código é inválido.

Código	**Cargo**
101	Vendedor
102	Atendente
103	Auxiliar Técnico
104	Assistente
105	Coordenador de Grupo
106	Gerente

21. Codifique um programa que leia a nota de um aluno e exiba o conceito correspondente, segundo as seguintes regras:

    - As notas 10 e 9 equivalem ao conceito A;
    - As notas 8 e 7 equivalem ao conceito B;
    - As notas 6 e 5 equivalem ao conceito C;
    - Notas abaixo de 5 equivalem ao conceito D.

22. Codifique um programa que leia um caractere (C, c, F ou f) indicando a unidade de medida da temperatura de entrada e uma temperatura de entrada. O programa deve realizar a conversão da temperatura inserida, da unidade de entrada para a outra unidade disponível, utilizando a seguinte fórmula de Conversão: $C = (5/9) * (F - 32)$. A saída do programa deve exigir a temperatura após conversão, com 2 casas decimais, e a unidade da temperatura após conversão.

23. Crie um programa utilizando switch_case que permita ao usuário utilizar uma espécie de calculadora, contendo as funções de somar e subtrair. O usuário deve digitar um carácter para sair do programa. O programa deve também, tratar opções inválidas.

2.2.3 Estruturas de Repetição

24. Escreva um programa que leia 6 valores e encontre o maior e o menor deles. Mostre o resultado.

25. Faça um programa que lê um valor N inteiro e positivo e que calcula e escreve o fatorial de N (N!).

26. Escrever um programa que leia um número inteiro n e calcule a tabuada de n. Mostre a tabuada na forma:

$$\begin{aligned} 1 * n &= \_\_\_ \\ 2 * n &= \_\_\_ \\ &\dots \\ n * n &= \_\_\_ \end{aligned}$$

27. Escrever um programa que lê 10 valores, um de cada vez, e conte quantos deles estão no intervalo [10,20] e quantos deles estão fora do intervalo, escrevendo estas informações.

28. Elabore um programa em C para calcular a raiz quadrada de um número positivo, usando o roteiro abaixo, baseado no método de aproximações sucessivas de Newton. O programa deverá prover 25 aproximações.

    Seja Y o número do qual se deseja a raiz quadrada. Obtemos:

    - a primeira aproximação para a raiz quadrada de Y por $X_1 = \frac{Y}{2}$;
    - as demais aproximações para a raiz quadrada de Y por $X_{n+1} = \frac{X_n^2+Y}{2X_n}$

29. Escreva instruções for que imprimam as seguintes series de valores:

    a) 1,2,3,4,5,6,7

    b) 3,8,13,18,23

    c) 20,14,8,2,-4,-10

    d) 19, 27,35,43,51

2.2.4 Teste no começo (while)

30. Faça um programa que receba salários de funcionários ate que o valor -999 seja inserido e apos o termino exiba o maior salario.

31. Algoritmo que calcule a média aritmética de vários valores inteiros positivos que serão lidos ate que o usuário digite um numero negativo.

32. Escrever um programa que leia um número não determinado de valores e calcule a média aritmética dos valores lidos, a quantidade de valores positivos, a quantidade de valores negativos e o percentual de valores negativos e positivos. Mostre os resultados. O número que encerrará a leitura será zero.

33. Efetue um programa que some o peso de pessoas que tenham mais de 30 anos. O usuário deve poder digitar o caracter 's' enquanto desejar continuar entrando com pessoas e digitar 'n' para sair.

34. Escrever um programa que gere e escreva os 4 primeiros números perfeitos. Um número perfeito é aquele que é igual a soma dos seus divisores exceto o próprio número. (Ex.: 6 = 1 + 2 + 3; 28 = 1+2+4+7+14 etc).

35. Escreva um algoritmo que imprima todos os divisores de 3 que existem entre os números 1 e 3000000.

36. A prefeitura de uma cidade fez uma pesquisa entre seus habitantes, coletando dados sobre o salário e número de filhos. A prefeitura deseja saber:

    - média do salário da população;

- média do número de filhos;
- maior salário;
- percentual de pessoas com salário até R$ 100,00;

O final da leitura de dados se dará com a entrada de um salário negativo.

37. Faça um programa que leia uma quantidade não determinada de números positivos. Calcule a quantidade de números pares e ímpares, a média de valores pares e a média geral dos números lidos. O número que encerrará a leitura será zero.

38. Crie para realizar somas enquanto o usuário desejar e no fim, exibir o resultado total.

39. Você irá criar um jogo, chamado jogo da adivinhação. O programa irá sortear um número aleatório entre 0 e 1000. Alternadamente, dois jogadores tentaram adivinhar o valor desse número. A cada jogada, o programa deve informar se o número é maior ou menor que o digitado pelo jogador 1 ou 2. O jogo acaba e para imediatamente quando e somente quando, um jogador acertar o número.

2.2.5 Contador de passo (for)

40. Diga que valores da variável de controle x são impressos por cada uma das seguintes instruções:

a)
```
for(x = 2; x <= 13; x+= 2)
printf("%d\n", x);
```

```
for(x = 5; x <= 22; x+= 7)
printf("%d\n", x);
```

b)
```
for(x = 3; x <= 15; x+= 3)
printf("%d\n", x);
```

```
for(x = 1; x <= 5; x+= 7)
printf("%d\n", x);
```

d)
```
for(x = 12; x >= 2; x -= 3)
printf("%d\n", x);
```

41. Faça um programa que que receba o numero máximo de corredores, receba o tempo de cada um e exiba o maior tempo apos o termino.

42. Faça um programa que calcule os n primeiros termos da sequencia de Fibonacci.

43. Faça um programa que calcule e imprima os 20 primeiros números primos.

44. Escreva um programa que calcule o maior de 10 números inteiros positivos inseridos pelo usuário.

45. Escreva um programa efetue a soma de números de 0 até n, onde n é um número digitado pelo usuário.

46. Construa um programa que imprima um triângulo retângulo invertido abaixo, com o tamanho máximo da base sendo indicado pelo usuário.

```
        *
      * *
    * * *
  * * * *
...
```

47. Construa um programa que imprima um triângulo retângulo invertido abaixo, com o tamanho máximo da base sendo indicado pelo usuário.

```
...
* * * *
* * *
* *
*
```

48. Construa um programa que imprima um triângulo isósceles com base ímpar abaixo, com o tamanho máximo da base sendo indicado pelo usuário.

```
        *
      * * *
    * * * * *
  * * * * * * *
...
```

49. Construa um programa que imprima a menor região complementar de um triângulo isósceles com base ímpar abaixo, com o tamanho máximo da base sendo indicado pelo usuário (no exemplo, a base vale 7).

```
+ + + + + + +
+ + +   + + +
+ +       + +
+           +
```

50. Construa um programa que imprima todas as possíveis jogadas para quatro dados normais, com faces de 1 a 6.

51. Construa um programa que imprima todas as possíveis jogadas para quatro dados normais, com faces de 1 a 6, sem que haja, em cada jogada, repetição de valores para dados distintos.

52. Construa um programa que leia um número n e imprima n linhas no formato a seguir ($n = 6$, no exemplo):

```
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
1 2 3 4 5 6
```

53. Construa um programa que leia um número n e imprima n linhas no formato a seguir ($n = 6$, no exemplo):

```
_ * * * * *
* _ * * * *
* * _ * * *
* * * _ * *
* * * * _ *
* * * * * _
```

54. Implemente um programa que retorne uma aproximação do valor de pi, de acordo com a Formula de Leibniz.

3 Funções

3.1 Exercícios Resolvidos

1. Codifique uma função que receba por parâmetro a idade de uma pessoa, expressa em anos, meses e dias, e retorne essa idade expressa em dias. Desconsidere anos bissextos.

   Resposta:

```c
#include <stdio.h>

int idade(int ano, int meses, int dias)
{
    int ano_dias = ano * 365;
    int meses_dias = meses * 30;

    return ( ano_dias + meses_dias + dias );

}
int main()
{
    int ano, meses, dias, idade_dias;

    printf("Digite sua idade em ano , meses e dias \n");
    printf("Ano: ");
    scanf("%d", &ano);
    printf("Meses: ");
    scanf("%d", &meses);
    printf("Dias : ");
    scanf("%d", &dias);

    idade_dias = idade(ano, meses, dias);

    printf("Idade em dias : %d \n" , idade_dias );

    return 0;
}
```

Código 3.1: codigos_CEA030/Lista1/converte_anos.c

3.2 Exercícios Propostos

3.2.1 Funções simples

1. Para o programa a seguir:

```c
#include <stdio.h>
```

```
int soma1(int q, int c);
int soma2(int ra);

int i = 10;
int j = 20;

int main()
{
    int i,k,ra,p;

    p = 10;
    ra = 5;

    for(i = 0; i< 3; i++)
    {
        k = soma1(ra, p);
        ra = soma2(k);
        printf("%d, %d\n",ra, k);
    }
    return 0;
}
int soma1(int q, int c)
{
    int soma = q+i+c;
    return soma;
}
int soma2(int ra)
{
    int k = j;
    ra = ra + k;
    return ra;
}
```

a) Determine quais são as variáveis locais e globais deste programa, identificando a que função pertence cada variável local.

b) Mostre o que será impresso na tela do computador quando for executado este programa.

2. Crie um programa em C que peça um número inteiro ao usuário e retorne a soma

de todos os números de 1 até o número que o usuário introduziu ou seja: 1 + 2 + 3 + ... + n. Utilize recursividade.

3. Crie uma função em linguagem C chamado Dado() que retorna, através de sorteio, um número de 1 até 6.

4. Use a função da questão 2 e lance o dado 1 milhão de vezes. Conte quantas vezes cada número saiu e exiba a porcentagem de cada um.

5. Crie um programa que calcule o fatorial de um número fornecido pelo usuário através da recursividade. O fatorial de 'n' é representado por n!, onde: n! = n * (n-1) * (n-2)*..1

6. Crie um programa onde o usuário digita três valores e imprima na tela o menor valor, devendo para isso, criar uma função Menor do tipo void que imprime na tela o menor valor.

7. Codifique uma função que receba a média final de um aluno passado por parâmetro e retorne o seu conceito (através de uma variável char), conforme a Tabela 1:

Nota	Conceito
De 0 a 49	D
De 50 a 69	C
De 70 a 89	B
De 90 a 100	A

Tabela 1:

8. Codifique uma função com a assinatura void `estacao`(int `dia`, int `mes`) que exiba no vídeo qual a estação do ano correspondente à data passada por parâmetro. Lembre-se que a primavera começa em 23 de setembro, o verão em 21 de dezembro, o outono em 21 de março e o inverno em 21 de junho.

   Ex:

```
estacao(25,10); /* Deve imprimir a mensagem: 25/10 e ↩
   primavera. */
estacao(29,12); /* Deve imprimir a mensagem: 29/12 e ↩
   verao. */
```

9. Codifique uma função com a assinatura int `containpar`(int `n1`, int `n2`) que retorne o número de inteiros impares que existem entre $n1$ e $n2$ (inclusive ambos, se for o caso). Caso o valor de n2 seja menor que o de n1, a função deve tratar o intervalo como sendo de $n2$ até $n1$ sem que o invocador da função perceba.

   Ex:

```
n = contaimpar(10,19); /* n recebe 5 (referente a: 11,13,15,17,19) */
n = contaimpar(5,1); /* n recebe 3 (referente a: 1,3,5) */
```

10. Codifique uma função com a assinatura `int somaintervalo(int n1, int n2)` que retorne a soma dos números inteiros que existem no intervalo fechado entre $n1$ e $n2$ (ou seja, incluindo $n1$ e $n2$). Caso o valor de n2 seja menor que o de n1, a função deve tratar o intervalo como sendo de $n2$ até $n1$ sem que o invocador da função perceba.

    Ex:

    ```
    n=somaintervalo(3, 6); /* n recebe 18 (referente a: 3 + 4 + 5 + 6) */
    n=somaintervalo(5,5); /* n recebe 5 (referente a: 5) */
    n=somaintervalo(-2,3); /* n recebe 3 (referente a: -2 + -1 + 0 + 1 + 2 + 3) */
    n=somaintervalo(4, 0); /* n recebe 10 (referente a: 4 + 3 + 2 + 1 + 0) */
    ```

11. Codifique uma função com a assinatura `int multiplica_intervalo(int n1, int n2)` que retorne o produto dos números inteiros que existem no intervalo fechado entre $n1$ e $n2$ (ou seja, incluindo $n1$ e $n2$), excluindo-se o número 0, caso esteja no intervalo. Caso o valor de n2 seja menor que o de n1, a função deve tratar o invervalo como sendo de $n2$ até $n1$ sem que o invocador da função perceba.

    Ex:

    ```
    n=multiplica_intervalo(3, 6); /* n recebe 360 (referente a: 3 * 4 * 5 * 6) */
    n=multiplica_intervalo(5,5); /* n recebe 5 (referente a: 5) */
    n=multiplica_intervalo(-2,3); /* n recebe 12 (referente a: -2 * -1 * 1 * 2 * 3) */
    ```

12. Codifique uma função que receba um número inteiro n passado por parâmetro e devolve o primeiro número da série de fibonacci que é maior ou igual a n.

13. Criar uma função que receba uma medida em *cm* e retorne em *polegadas*, sabendo que 1 *polegada* $= 2,54$ *cm*.

14. Codifique uma função com a assinatura `int potencia(int base, int expoente)` que calcule a potência $base^{expoente}$ (**Não utilize a biblioteca mathh ou qualquer**

outra biblioteca). Utilizando esta função, crie um programa que imprima as 11 primeiras potências de 2 (2^0, 2^1, ..., 2^{10}), de 4 (4^0, 4^1, ..., 4^{10}), de 6 (6^0, 6^1, ..., 6^{10}) e de 8 (8^0, 8^1, ..., 8^{10}).

4 Vetores unidimensionais

4.1 Exercícios Propostos

1. Escreva um programa que leia e mostre um vetor de 20 elementos inteiros. A seguir, conte quantos valores pares e ímpares existem no vetor.

2. Faça um programa que leia um vetor de 20 inteiros positivos (permita apenas que valores corretos sejam digitados, pedindo para o usuário repetir cada número negativo até que ele digite um positivo) e imprima todos os valores abaixo da média desses valores.

3. Escrever um programa que gere 10 números sorteados entre 10 e 20 (inclusive) e os armazene em um vetor de 10 posições e o imprima no final.

4. Escreva um programa que leia dois vetores de 10 posições e faça a multiplicação dos elementos de mesmo índice, colocando o resultado em um terceiro vetor. Mostre o vetor resultante.

5. Faça um algoritmo que leia um vetor de 80 posições e encontre o menor e o maior valor. Mostre-os juntamente com seus respectivos índices no vetor.

6. Escreva um programa que leia um vetor de 20 posições e mostre-o. Em seguida, troque o primeiro elemento com o último, o segundo com o penúltimo, o terceiro com o antepenúltimo, e assim sucessivamente. Mostre o novo vetor depois da troca.

7. Escreva um programa que leia um vetor de 10 posições de números inteiros e o imprimi-ma. Logo após, gerar 2 vetores a partir dele, um contendo os elementos de posições ímpares do vetor e o outro os elementos de posições pares. Imprimí-los no final.

8. Codifique um programa que leia um número inteiro e, em seguida, outros 10 números inteiros em um vetor. O programa deve imprimir os 10 números lidos, informando se cada número é, ou não, múltiplo do primeiro número lido. O programa deve criar uma função auxiliar com a assinatura `int eh_multiplo(int a, int b)` que retorne 1 caso a seja múltiplo de b e 0, caso contrário.

9. Codifique um programa que leia e armazene duas notas de 10 alunos e calcule e exiba as respectivas médias. O programa deve utilizar:

    - Um procedimento que receba dois vetores de notas decimais e o tamanho dos vetores e peça ao usuário para digitar as notas, de forma a preencher os vetores.

- Um segundo procedimento deve receber os dois vetores de notas decimais e um vetor de médias decimais, bem como o tamanho dos vetores, e calcular a média das notas, preenchendo o vetor de médias.
- Um terceiro procedimento deve receber os vetores de notas e médias, além do tamanho dos mesmos, e imprimir as informações de cada aluno (utilizando 2 casas decimais onde for preciso) no seguinte formato:

```
Aluno 1
    nota 1: 7.50
    nota 2: 8.50
    media: 8.00
```

10. Codifique um programa que leia e armazene a matrícula (um número inteiro) e o salário de 10 funcionários utilizando vetores. Os funcionários cuja matrícula for um número par devem receber um aumento de 15%; já os funcionários cuja matrícula for um número ímpar, devem receber um aumento de 20%. Utilize procedimentos para ler os salários, aplicar o reajuste salárial e imprimir as informações, no seguinte formato:

```
Funcionario 1
    matricula: 2784
    salario base: R$ 7860.50
    percentual de aumento: 15
    salario corrigido: R$ 9039.57
```

11. Codifique um programa que leia o preço de compra e o preço de venda de 10 produtos e informe quantos produtos proporcionam:

    - lucro inferior a 10%
    - lucro entre 10% e 20%
    - lucro acima de 20%

    O programa deve utilizar um procedimento com assinatura void `le_precos`(int ↩ `precos`[], int `quantidade`, char `opcao`[]) para leitura dos preços de compra e venda, utilizando a string opção para identificar o preço de venda ou compra. Uma função deve calcular o lucro de um produto, dados os seus preços como parâmetro.

12. Codifique um programa que leia uma sequência de até 10 caracteres e a imprima em CAIXA ALTA. Procure pela função `toupper` na biblioteca ctype.h

13. Escreva um programa que leia duas palavras do teclado e determine se a segunda é um anagrama da primeira. Uma palavra é um anagrama de outra se todas as letras

de uma ocorrem na outra, em mesmo número, independente da posição. Exemplos: ROMA, MORA, ORAM, AMOR, RAMO são anagramas entre si.

14. Um palíndromo é uma palavra ou frase, que é igual quando lida da esquerda para a direita ou da direita para a esquerda. Escreva um programa que leia uma string de até 50 caracteres, e imprima "Palíndromo", caso a string seja um palíndromo, e "Não Palíndromo", caso contrário. Assuma que só são usados caracteres minúsculos e sem acentos. Espaços em brancos devem ser descartados. Exemplo de palíndromo: saudável leva duas.

5 Recursão

5.1 Exercícios Propostos

1. Escreva uma função recursiva que retorne o fatorial de uma número que o usuário informar.

2. Escreva uma função recursiva que procure um valor em um vetor e retorne o índice do elemento, caso ele exista no vetor, ou -1 caso, caso contrário.

3. Escreva um procedimento recursivo que receba uma string como parâmetro e a exiba invertida.

4. Codifique um procedimento recursivo com a assinatura void `potencia`(int `base`,↩ int `expoente`, long int * `resultado`) que calcule a potência $base^{expoente}$ (**Não utilize a biblioteca math.h ou qualquer outra biblioteca**). Utilizando este procedimento, crie um programa que imprima as 11 primeiras potências de 3 (3^0, 3^1, ..., 3^{10}), de 5 (5^0, 5^1, ..., 5^{10}), de 7 (7^0, 7^1, ..., 7^{10}) e de 9 (9^0, 9^1, ..., 9^{10}).

6 Laços encaixados

6.1 Exercícios Propostos

1. Criar uma função que receba dois números e calcule uma operação com eles retornando o resultado. A operação deve ser escolhida pelo usuário passando para função e executando, retornando o valor.

2. Codifique um procedimento sem parâmetros que imprima as tabuadas de 1 a 10 utilizando laços encaixados. Apenas este procedimendo deverá ser chamado na função `main`.

3. Codifique uma função com a assinatura void `resulta_em`(int `numero`), em que *numero* pode ser um inteiro positivo ou negativo, que imprima todos os valores x e y (inclusive negativos) tais que $x * y = numero$.

4. Codifique uma função que receba por parâmetro a idade de uma pessoa, expressa em anos, meses e dias, e retorne essa idade expressa em dias. Desconsidere anos bissextos.

5. Codifique uma função com a assinatura int contaimpar(int n1, int n2) que retorne o número de inteiros ímpares que existem entre n1 e n2 (inclusive ambos, se for o caso). Caso o valor de n2 seja menor que o de n1, a função deve tratar o invervalo como sendo de n2 até n1 sem que o invocador da função perceba.

6. Codifique uma função com a assinatura int multiplica_intervalo(int n1, int n2) que retorne o produto dos números inteiros que existem no intervalo fechado entre n1 e n2 (ou seja, incluindo n1 e n2), excluindo-se o número 0, caso esteja no intervalo. Caso o valor de n2 seja menor que o de n1, a função deve tratar o invervalo como sendo de n2 até n1 sem que o invocador da função perceba.

7 Matrizes

7.1 Exercícios Propostos

7.1.1 Matrizes (Vetores multidimensionais)

1. Codifique um programa que leia e armazene o nome e duas notas de 10 alunos. O programa deve calcular e exibir as médias de cada aluno, utilizando as seguintes estruturas:

   - O programa deve utilizar um procedimento que receba uma matriz de caracteres e uma matriz de notas, além das dimensões necessárias, e peça ao usuário para digitar os nomes (apenas o primeiro nome, com até 15 caracteres) e as duas notas de cada aluno, de forma a preencher as matrizes.
   - Um segundo procedimento deve receber uma matriz de notas e um vetor de médias, além das dimensões necessárias, e calcular a média das notas, preenchendo o vetor de médias.
   - Um terceiro procedimento deve receber a matriz de nomes, a matriz de notas e o vetor de médias, além das dimensões necessárias, e imprimir as informações de cada aluno (utilizando 2 casas decimais, onde for preciso) no seguinte formato:

     ```
     Aluno 1
     nome: Nome Sobrenome
     nota 1: 7.50
     nota 2: 8.50
     media: 8.00
     ```

2. Elabore um programa que crie uma matriz 3x4 com valores aleatórios. Ao final o algoritmo deverá:

   - Mostrar os valores da matriz;
   - Mostrar a soma dos valores.

3. Elabore um programa que crie uma matriz 3x6 com valores aleatórios. Ao final o algoritmo deverá:

   - Mostrar os valores da matriz;
   - Pedir um valor para o usuário e multiplicar todos os elementos por esse valor.

   Mostrar a matriz com os novos valores.

4. Elabore um algoritmo que crie uma matriz 4x4 com valores aleatórios. Ao final o algoritmo deverá:

   - Mostrar os valores da matriz
   - Mostrar o valor e a posição do maior elemento
   - Mostrar o valor e a posição do menor elemento

5. Faça um programa que carregue uma matriz F (10x10) com valores aleatórios entre 10 e 20 e imprima a soma de cada linha e de cada coluna.

6. Escreva um algoritmo que leia uma matriz M (5x5). Em seguida calcule e imprima as somas:

   - da linha 4 de M
   - da coluna 2 de M
   - da diagonal principal
   - da diagonal secundária

7. Faça um programa que carregue uma matriz C (20x20) com valores aleatórios entre 0 e 10. Imprimir a posição da linha cujos valores possuem a menor soma e a posição da coluna cujos valores possuem a maior soma.

8. Escreva um algoritmo que leia um vetor de 10 posições e mostre-o ordenado em ordem crescente.

9. Escrever um programa que leia 2 vetores de tamanho 10 e os escreva. Crie, a seguir, um vetor de 20 posições que contenha os elementos dos outros 2 vetores em ordem crescente.

10. Uma matriz quadrada de inteiros é um quadrado mágico se a soma dos elementos de cada linha, a soma dos elementos de cada coluna, a soma dos elementos da diagonal principal e a soma dos elementos da diagonal secundária são todos iguais. A matriz abaixo é um exemplo de quadrado mágico:
    3 4 8
    10 5 0
    2 6 7
    Faça um programa que leia uma matriz quadrada e determine se ela é um quadrado mágico. Utilize funções sempre que possível.

11. Elabore um programa que crie uma matriz 3x4 com valores aleatórios. Ao final o algoritmo deverá:

    - Mostrar os valores da matriz;

- Mostrar a soma dos valores.

12. Elabore um programa que crie uma matriz 3x6 com valores aleatórios. Ao final o algoritmo deverá:

    - Mostrar os valores da matriz;
    - Pedir um valor para o usuário e multiplicar todos os elementos por esse valor.

    Mostrar a matriz com os novos valores.

13. Uma matriz quadrada de inteiros é um quadrado mágico se a soma dos elementos de cada linha, a soma dos elementos de cada coluna, a soma dos elementos da diagonal principal e a soma dos elementos da diagonal secundária são todos iguais. A matriz abaixo é um exemplo de quadrado mágico:

    $$\begin{matrix} 3 & 4 & 8 \\ 10 & 5 & 0 \\ 2 & 6 & 7 \end{matrix}$$

    Faça um programa que leia uma matriz quadrada e determine se ela é um quadrado mágico. Utilize funções sempre que possível.

14. Faça um programa que carregue uma matriz F (10x10) com valores aleatórios entre 10 e 20 e imprima a soma de cada linha e de cada coluna.

15. Faça um programa que carregue uma matriz C (5x5) com valores aleatórios entre 0 e 10. Imprimir a posição da linha cujos valores possuem a menor soma e a posição da coluna cujos valores possuem a maior soma.

7.1.2 Strings

16. Escreva uma função de assinatura `int strchr(char string[], char ch);` que procure a primeira ocorrência do caractere *ch* em *string*. A função retorna um número inteiro indicando o índice do caractere, se for encontrado, ou -1, caso contrário.

17. Escreva um procedimento de assinatura `void strinv(char str[])` que inverta a string recebida como parâmetro.

18. Escreva um procedimento de assinatura `void strupr(char str[]);` que altere todas as letras para letras maiúsculas. Escreva a função principal para testar seu programa.

19. Escreva uma função de assinatura `int strblank(char str[]);` que verifique se `string` é uma string em branco. A função retorna o primeiro caractere não branco ou zero, se a string for formada somente por caracteres brancos.

20. Escreva um programa que leia duas palavras do teclado e determine se a segunda é um anagrama da primeira. Uma palavra é um anagrama de outra se todas as letras de uma ocorrem na outra, em mesmo número, independente da posição. Exemplos: ROMA, MORA, ORAM, AMOR, RAMO são anagramas entre si.

21. Um palíndromo é uma palavra, que é igual quando lida da esquerda para a direita ou da direita para a esquerda. Escreva um programa que leia uma string de até 50 caracteres, e imprima "Palíndromo´´, caso a string seja um palíndromo, e "Não Palíndromo´´, caso contrário. Ex.: radar, arara, reviver...

22. Escreva um procedimento de assinatura void `justify`(char `str`[], int `modo`, int↩ `tamanho`); que:

    - acrescente brancos ao final da `string`, de forma que fique com `tamanho`↩ caracteres se `modo` for igual a 0;
    - insira brancos no início e no final da `string`, de forma que fique com `tamanho` caracteres e os caracteres fiquem centralizados, se `modo` for igual a 1;
    - insira caracteres brancos no início da `string`, de forma que fique com `tamanho`↩ caracteres, se `modo` for igual a 2;

8 Registros

8.1 Exercícios Propostos

1. Implemente um programa que leia o nome, a idade e o endereço de uma pessoa e armazene esses dados em uma estrutura. Em seguida, imprima na tela os dados da estrutura lida.

2. Implemente um programa que leia o nome, a idade e o endereço de uma pessoa e armazene esses dados em uma estrutura. Em seguida, imprima na tela os dados da estrutura lida.

3. Faça uma programa que permita a manipulação de um novo tipo chamado Data, que deverá permitir o armazenamento dos dados de uma data (dia, mês e ano). Solicite ao usuário que digite os dados de uma data e exiba na tela. Obs: Os casos em que ocorrer a entrada de valores inválidos para dia, mês e ano deverão ser verificados e nova oportunidade de digitação deverá ser dada ao usuário.

4. Crie uma estrutura chamada Retângulo. Essa estrutura deverá conter o ponto superior esquerdo e o ponto inferior direito do retângulo. Cada ponto é definido por uma estrutura Ponto, a qual contém as posições X e Y. Faça um programa que declare e leia uma estrutura Retângulo e exiba a área, o comprimento da diagonal e o perímetro desse retângulo.

5. Usando a estrutura Retângulo do exercício anterior, faça um programa que declare e solicite ao usuário que digite os dados de uma estrutura Retângulo e um Ponto. E informe se esse ponto está ou não dentro do retângulo.

6. Crie uma estrutura para representar as coordenadas de um ponto no plano (posições X e Y). Em seguida, declare e leia do teclado um ponto e exiba a distância dele até a origem das coordenadas, isto é, a posição (0,0).

7. Faça uma programa que permita a manipulação de um novo tipo chamado Data, que deverá permitir o armazenamento dos dados de uma data (dia, mês e ano). Solicite ao usuário que digite os dados de uma data e exiba na tela. Obs.: Os casos em que ocorrer a entrada de valores inválidos para dia, mês e ano deverão ser verificados e nova oportunidade de digitação deverá ser dada ao usuário.

8. Faça um programa que permita ao usuário cadastrar os dados de um aluno. Os dados deverão ser armazenados em uma variável do tipo Aluno. Para isso você deverá criar um registro chamado Aluno que conterá os seguintes dados: nome, curso, idade, data

de nascimento (estrutura criada no exercício anterior), notas (armazenadas em um vetor de double contendo 3 elementos). Depois de realizado o cadastro, os dados do estudante deverão ser exibidos de maneira organizada na tela. Antes de exibir os dados do aluno cadastrado, calcule o coeficiente dele fazendo uma média ponderada das notas de cada uma das provas, sendo o peso delas: 2 para a primeira prova, 3 para a segunda prova e 4 para a terceira prova.

9. Adapte o exercício anterior para que ele permita ao usuário cadastrar quantos alunos forem necessários. O programa criado também deverá exibir os dados dos alunos de maneira organizada.

10. Suponha que criamos uma estrutura para armazenar produtos de um supermercado:

```
struct Produto{
  char nome[80];
  double preco;
  int quantidade;
};
```

Implemente dois procedimentos:

- a primeira subrotina, com assinatura `void levantaPrecos(struct Produto * vetProd, int tamVetProd, double * vetPrecos)`, deve retornar um vetor contendo os preços dos produtos;
- a segunda subrotina, com assinatura `void levantaQuantidade(struct Produto * vet, int tamVetProd,)`, deve devolver um vetor contendo as quantidades dos produtos.

Utilize procedimentos para ler e imprimir os registros e, também, para imprimir os vetores.

11. Crie uma estrutura ponto que representará um ponto no espaço. Essa estrutura conterá três números reais (x, y e z). Faça um programa que crie um vetor de estruturas com n pontos (especificados pelo usuário). Crie funções para:

    a) Ler os n Pontos;

    b) Imprimir os pontos lidos;

    c) Calcular a distância de 2 pontos pelo seus índices;

    d) Calcular a distância de todos os pontos consecutivos (primeiro com o segundo, segundo com o terceiro etc);

Ao final, crie um menu que permita selecionar cada uma das funções acima.

12. O que será impresso pelo programa abaixo:

```c
#include <stdio.h>
struct T{
  int x;
  int y;
};
typedef struct T  T;
void f1(T *a);
void f2(int *b);
int main(){
  T a, b, *c, *d;
  c = &a;
  a.x = 2;
  a.y = 4;
  b.x = 2;
  b.y = 2;
  d = c;
  f1(d);
  b = *d;
  printf("x: %d --- y: %d\n",b.x,b.y);
}
void f1(T *a){
  f2(&(a->x));
  f2(&(a->y));
}
void f2(int *b){
  *b = 2*(*b);
}
```

9 Ponteiros e Alocação dinâmica de memória

9.1 Ponteiros

9.1.1 Exercícios Propostos

1. Seja `p` um ponteiro para um tipo de dado qualquer, explique a diferença entre `p++`, `(*p)++` e `*(p++)`.

2. Explique o funcionamento do programa a seguir.

```
int main(){
    int y, *p, x;
    y = 0;
    p = &y;
    x = *p;
    x = 4;
    (*p)++;
     x  ;
    (*p) += x;
    printf ("y = %d\n", y);
    return(0);
}
```

9.2 Alocação dinâmica de memória

9.2.1 Exercícios Propostos

3. Codifique um programa que leia um número inteiro n e, em seguida, leia e armazene o nome e duas notas de n alunos. O programa deve calcular e exibir as médias de cada aluno, utilizando as seguintes estruturas:

   - O programa deve utilizar um procedimento que receba uma matriz de caracteres e uma matriz de notas, além das dimensões necessárias, e peça ao usuário para digitar os nomes (apenas o primeiro nome, com até 15 caracteres) e as duas notas de cada aluno, de forma a preencher as matrizes.
   - Um segundo procedimento deve receber uma matriz de notas e um vetor de médias, além das dimensões necessárias, e calcular a média das notas, preenchendo o vetor de médias.

- Um terceiro procedimento deve receber a matriz de nomes, a matriz de notas e o vetor de médias, além das dimensões necessárias, e imprimir as informações de cada aluno (utilizando 2 casas decimais, onde for preciso) no seguinte formato:

```
Aluno 1
nome: Nome Sobrenome
nota 1: 7.50
nota 2: 8.50
media: 8.00
```

4. Codifique um programa que leia um número inteiro `x` e um número inteiro `y`. Em seguida, o programa deve ler `y` números inteiros em um vetor `vet` alocado dinamicamente. O programa deve imprimir os `y` números lidos, informando se cada número é, ou não, múltiplo do número `x`. Deve-se utilizar um procedimento auxiliar com a assinatura `void eh_multiplo(int a, int b, int * resp)` que armazene em (`*resp`) o valor 1, caso `a` seja múltiplo de `b`, e 0, caso contrário.

5. Codifique um programa que crie dinamicamente 2 matrizes quadradas de inteiros com tamanho informado pelo usuário e calcule o produto matricial entre as matrizes. Utilize procedimentos para leitura das matrizes, cálculo do produto matricial e impressão das matrizes.

6. Codifique um programa que crie dinamicamente uma matriz quadrada de inteiros com tamanho informado pelo usuário e calcule a matriz transposta da matriz lida, salvando-a em uma nova matriz. Utilize procedimentos para leitura da matriz, cálculo da matriz transposta e impressão das matrizes. (Lembre-se: a matriz transposta T de uma matriz M é aquela em que as linhas de M viram colunas de T.)

7. Codifique um programa que crie dinamicamente uma matriz quadrada de inteiros com tamanho informado pelo usuário e leia um inteiro informado pelo usuário. Seu programa deve calcular o produto escalar entre o número e a matriz informada, em uma nova matriz, e imprimir o resultado. Utilize procedimentos para leitura da matriz, cálculo do produto escalar e impressão das matrizes.

8. Escreva um programa que leia um número n e crie um vetor dinamicamente com n elementos. Em seguida, utilize um procedimento para preencher o vetor. Por fim, crie um procedimento recursivo que exiba na tela os elementos do vetor criado, ou seja, que exiba os elementos do vetor cujos índices estão no intervalo $[0, n-1]$.

9. Escreva uma função recursiva que retorne o índice do maior elemento de um vetor. O programa principal deve perguntar ao usuário o tamanho do vetor e alocar dinamicamente este vetor. Utilize um procedimento para preencher o vetor.

10. Escreva uma função recursiva que retorne o índice do menor elemento de um vetor. O programa principal deve perguntar ao usuário o tamanho do vetor e alocar dinamicamente este vetor. Utilize um procedimento para preencher o vetor.

11. Escreva uma função recursiva que retorne a soma dos elementos de um vetor. O programa principal deve perguntar ao usuário o tamanho do vetor e alocar dinamicamente este vetor. Utilize um procedimento para preencher o vetor.